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

DELIVERABLE D8.2

PUBLISHABLE FINAL REPORT

Due Date of Delivery:	M38 *Mx* (31/07/2020 *dd/mm/yyyy*)
Actual Date of Delivery:	19/10/2020 *dd/mm/yyyy*
Date of Revision Delivery:	07/12/2020 *dd/mm/yyyy*
Workpackage:	**WP8 – Project Management and Consortium Coordination**
Type of the Deliverable:	R
Dissemination level:	PU
Editors:	CNIT, UBITECH
Version:	2.1

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

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
Franco Davoli, Chiara Lombardo, Riccardo Rapuzzi	
UBITECH	GIOUMPITEK MELETI SCHEDIASMOS YLOPOIISI KAI POLISI ERGON PLIROFORIKIS ETAIREIA PERIORISMENIS EFTHYNIS
Panagiotis Gouvas	

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

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
GUI	Graphical User Interface
IoT	Internet of Things
KPI	Key Performance Indicator
KVM	Kernel-based Virtual Machine
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
PPDR	Public Protection and Disaster Relief
SLA	Service Layer Agreement
UI	User Interface
VAO	Vertical Application Orchestrator
VIM	Virtual Infrastructure Manager
VLD	Virtual Link Descriptor
VNF	Virtual Network Function
WAN	Wide Area Network
ZSM	Zero-touch network and Service Management

1 Introduction and Scope

This public deliverable provides a summary of the MATILDA project final results and outcomes. It is intended for public use and its principal aim is **to highlight the benefits and potential impact of the MATILDA framework toward vertical industries empowered with the 5G infrastructure**. At the same time, it will also summarize the **conceptual, architectural and implementation artefacts** that have been developed within the project's lifespan to achieve such benefits and impact. In order to be acquainted with the novel MATILDA paradigm, the reader is advised to proceed sequentially, by reading first the **MATILDA value proposition** section below, followed immediately by the **success story of the MATILDA framework in the vertical application environment of Industry 4.0.** The description of the other vertical applications that were implemented and demonstrated in MATILDA (**Public Protection and Disaster Relief (PPDR), Smart City Lighting, Intelligent Retail Advertisement and Distributed System Testing**) is reported in Appendix A. Appendix B complements the above-mentioned narration with a more deliverable-oriented description, by providing a short, yet comprehensive, synopsis of the achieved outcomes, organized per Workpackage (WP).

It should be mentioned that the MATILDA paradigm was proposed in 2017 and (in a nutshell) dictated that **cloud-native applications** should be deployed in modern reconfigurable infrastructure **using a clear administrative/domain separation** between **vertical-application-orchestration** and **network-service-orchestration**. The holy grail between these two administrative domains is a sound **OSS (Operations Support System) Northbound Management Application Programming Interface (API)**. We boldly consider that the dominance of cloud-native vertical-application-orchestrators (e.g., Kubernetes) **denotes that this paradigm was correct since its principles are industry-validated.**

The rationale behind MATILDA's separation of concerns also stems from a **business-oriented view of the convergence between the cloud-native applications scenario and its stakeholders and the 5G-enabled network services**. **Vertical industries and cloud application developers should not be obliged to learn the underlying network details** in order to be able to deploy their applications in the 5G (and beyond) networking environment. At the same time, **they should be enabled to fully exploit the capabilities offered by the new networking paradigms, by means of the same abstractions, constructs and operational tools they are used to work with**. By offering these capabilities in a form clearly understandable by application developers, MATILDA fulfils this goal. Likewise, by looking from the **perspective of the network service operators (or even of vertical industries that would like to deploy their own networking facilities) it provides the tools to realize zero-touch network configuration and management** to offer the vertical application access to the full exploitation of a programmable and dynamically (re)-configurable networking platform.

An inherent advantage of this architectural viewpoint is also **the possibility offered to organizations to adopt the whole MATILDA framework, or the parts of it that fall under their interests**, concerning either the (5G empowered) cloud application development and orchestration, or the (application-aware) 5G virtual networking infrastructure.

2 The MATILDA Value Proposition

The advent of 5G networks is predicted as a leading factor in the **digital transformation** by Big Data and Cloud Computing technologies towards a new hyper-connected society. **5G** has been designed to provide **flexible support for radically new and extremely heterogeneous vertical applications**, requiring any custom mix of network and radio services. In particular, applications under the **Industry 4.0** umbrella, with their **zero-perceived latency requirements**, would strongly benefit from the 5G widespread diffusion, because the only alternative satisfying such constraints is to rely on cabled solutions, which are not only far more expensive, but also unadaptable and so less suited to follow the fast, upcoming evolutions of the technologies.

In this context, **5G network slices and edge computing** can provide the required connectivity features by **hosting vertical applications into the 5G infrastructure**, and directly attaching them to the network slice terminations in neighbouring geographical facilities. **Network slices can be customized** to perfectly match the desired network and radio services features and constraints of vertical industries **to interconnect their devices and terminals and directly manage them.**

However, vertical stakeholders lack the basic knowledge to exploit the full potentials of 5G networks. Moreover, Telecom providers owning the physical infrastructure are less than prone to allow third parties independently orchestrating their resources. In such a context, the necessary **integration between the digital systems that enable vertical services and the network layer** remains undefined and represents a big challenge.

MATILDA's main aim has been to bridge the gap between **the vertical application and the network service domains** with a platform able to *i*) transform a cloud application into a 5G-ready one *ii*) provide the vertical stakeholders with a complete 5G network slice, from the radio access to the core, and *iii*) allow them to manage the lifecycle of their applications, regardless of the underneath infrastructure, all in a **Zero-touch network and Service Management (ZSM) fashion.** MATILDA provides verticals with a **familiar, cloud-style interface** in which, by simply adding their **application graph and constraints**, they can obtain a **5G-ready application** instantiated over a wide-area infrastructure that they can **orchestrate in a secure and isolated fashion**.

Therefore, MATILDA comes up with a novel and holistic approach for tackling the overall lifecycle of applications' design, development, deployment and orchestration in a 5G environment. As already hinted in the previous Section, a set of novel concepts have been introduced, including the **design and development of 5G-ready applications – based on cloud-native/microservice development principles**, the **separation of concerns among the orchestration of the developed applications and the required network services that support them**, as well as the **specification and management of network slices that are application-aware and can lead to optimal application execution**.

MATILDA has followed a top-down approach, where application design and development leads to the instantiation of application aware-network slices, over which vertical industries' applications can be optimally served. Different stakeholders are engaged in this process, with clear separation of concerns among them.

MATILDA's main targets are i) **the vertical application stakeholders**, whose developers' work can be substantially eased and enhanced, and ii) the **Telecom Providers (Telcos)**,

including Virtual Network Operators (VNOs), Network as a Service (NaaS) companies and small players with innovative, telecom-centric, business models. With regard to Telcos, MATILDA aims to help them to satisfy their vertical customers' need, by creating a fruitful environment where network-intensive services can be easily prototyped and quickly deployed into production.

2.1 Vertical Industry Domain: Cloud-native approach and Vertical Applications Orchestration

The framework allows **software developers** to create applications following a simple and conventional **microservices-based approach**, where each component can be independently orchestratable. Based on the conceptualization of metamodels (application component and graph metamodels), they can **formally (and easily) declare information and requirements – in the form of descriptors – that can be exploited during the deployment and operation over programmable infrastructure**.

Such information and requirements may regard **capabilities, envisaged functionalities and soft or hard constraints that have to be fulfilled and may be associated with an application component or virtual link interconnecting two components within an application graph**. The produced application is considered to be **"5G-ready"**.

MATILDA also encourages new business and wide collaboration by providing a **Marketplace**, where not only the created **applications and components** can be published but also **Virtual Network Functions (VNFs)** and **Network Services (NSs)** (in the form of enhanced descriptors).

Application Service Providers are able to adopt the developed 5G-ready applications (published to the Marketplace or created internally) and specify policies and configuration options for their optimal deployment and operation over programmable infrastructure. Based on the provided application descriptor, these service providers are able to design operational policies and formulate a **slice intent**. These operational policies describe how the application components should adapt their execution mode in runtime. On the other hand, the slice intent includes a set of constraints that have to be fulfilled during the placement of the application and a set of envisaged network functionalities that have to be provided. **This information is used by the Vertical Application Orchestrator (VAO) to request the creation of an appropriate application-aware network slice from the Telecommunication Infrastructure Provider**.

While the instantiation and management of the application-aware network slice (including the set of network functions) is realised by the **Network and Computing Slice Deployment Platform (managed by the Telecommunications Infrastructure Provider)**, the deployment and runtime management of an application is realised by the **Vertical Application Orchestrator (managed by the application service provider)**, following a **service-mesh**-oriented approach.

This recently introduced approach is adopted as a software management layer for controlling and monitoring internal traffic in microservices-based applications. It consists of a **data plane and a control plane**. The data plane consists of a set of **intelligent proxies** deployed alongside the application software components dedicated to the provision of support/backing services (e.g., service discovery, load balancing, health checking, telemetry). The control plane manages the set of intelligent proxies based on **distributed management techniques** and provides **policy and configuration guidance** for all the running support/backing services. Policies'

definition for the activation and management of the set of required support/backing services is realised based on a policies' editor, while policies' enforcement is realised upon a rule-based management system. **Advanced monitoring and analysis techniques** are also applied for extracting insights that can be proven useful for service providers.

2.2 Telco Domain: Virtual Network Deployment and Orchestration

In order to instantiate and manage the application-aware network slice during the overall lifecycle of the 5G-ready application, Telecommunication Infrastructure Providers rely on the concept of network slice to fulfil the vertical applications' needs. **A network slice is a logical infrastructure partitioning allocated resources and optimized topology with appropriate isolation, to serve the particular purpose of an application graph**.

The Network and Computing Slice Deployment Platform includes an OSS/BSS system, a Network Functions Virtualization Orchestrator (NFVO) and a resources' manager for managing the set of deployed Wide Area Infrastructure Managers (WIMs) and Virtual Infrastructure Managers (VIMs). Based on the interpretation of the provided slice intent, the required network management mechanisms are activated and dynamically handled.

The **Telecommunication Infrastructure Provider is in charge of realising the instantiation of the slice over the programmable infrastructure**. The reserved resources for this slice combine both network and compute resources. A Telecommunication Infrastructure Provider may deliver all these resources based on its own infrastructure or come into an agreement with a Cloud Infrastructure Provider and acquire access to additional compute resources (e.g., in the edge of the network).

These actions are realized in an agnostic way to application service providers. However, **through a set of open APIs, requests for adaptation of the slice configuration may be provided by the Vertical Applications Orchestrator to the Network and Computing Slice Deployment Platform**.

The materialization of the network slice requires the **instantiation of network services that are composed of VNF chains**. These NSs and VNFs can be imported into the telecommunications infrastructure provider's catalogue from the MATILDA marketplace.

Following the generation of the **slice intent** according to the specifications provided by the application developer, the OSS controls (**in a ZSM, automated way**) all the resources/services at any layer in the Telco domain that allow the entire **lifecycle management of the 5G-enabled application (5G vApp)**: from **planning**, to **first deployment**, down to **in-life management** (the so-called **Day-0, Day-1 and Day-2** operations, respectively), until their termination. Day-2 operations include, among others, **upgrade** and **scaling** mechanisms. **These automated operations are common to all instantiated vertical applications**.

In more detail, almost verbatim quoting from our reference [1]:

- In **Day-0 operations**, the VAO asks the BSS/OSS for 5G network slices and edge computing resources, indicating QoS/locality requirements of the vApp, and the BSS/OSS computes a suitable deployment plan (e.g., it selects the datacenters and the wide-area resources to be applied).
- In **Day-1 operations**, the BSS/OSS requests wide-area interconnectivity from the WIM(s), and triggers, if needed, the NFVO to set up and/or properly configure network

services and edge computing resources. Upon the successful fulfilment of the previous operations, the VAO can start deploying the vApp.

- In **Day-2**, all services are running. The VNFs and the vApps' components are monitored by the NFVO and the VAO, respectively, which might independently scale them to cope with the incoming workload, perform self-healing procedures in case of problems, manage upgrade operations, and so on.

With respect to these operations – **which are general and abstracted from all application environments and use cases** – it is worth noting that Network Services Descriptors (NSDs) have not been provided as static packages in ETSI Open-Source MANO (OSM) [2],[3], but they have been rather generalized into "**network slice blueprints**." Such blueprints are meant to be defined by the Telecom Operator and consist of a metamodel representing how many and which types of network services should be created and deployed, in order to cope with performance and operational requirements of the various base telecom service or slice intent requests. These metamodels obviously include a reference on which types of VNFs should be applied (on the basis of their nature and of the supported features), and of their Day-2 configuration (i.e., to be properly parsed to produce the configuration file of the software processes realizing the network function in a Virtual Machine (VM) / Container). Upon the peculiarity of the incoming network service request, **the NFV Convergence Layer in the OSS automatically selects the most appropriate blueprint**, and on the basis of the metamodel and instantiation requirements, **builds on the fly the NSD, onboards it onto OSM, and triggers its (multi-site) instantiation through OSM**.

Another relevant point to be noted in this context is the following. Given that commercial and open-source 5G access and core devices/software were still in early prototyping during the MATILDA platform development activities, the MATILDA Consortium decided to **use 4G technologies with a number of solutions, studied ad-hoc** within the work package dedicated to Multi-site Resource Management and Execution Mechanisms (WP4) **for emulating network slicing capabilities**. Every physical and virtual network function (P/VNF) has been released as a package to be on-boarded on OSM and provided with a V/PNF Manager (in the form of a Juju charm [4], [5]), exposing homogenous interfaces to the Network Service Manager for Day-2 configurations. In total, sixteen P/VNFs have been produced, to enable the composition of five main types of services including two different 4G Enhanced Packet Core (EPC) implementations (one monolithic with extended capabilities, and one distributed but with a more limited set of supported capabilities). **All tests that will be briefly described in the next sections were conducted under this environment**.

2.3 Overall Lifecycle Management

A summary of the described overall lifecycle of an application created with the MATILDA framework is represented in Figure 1 below, highlighting the interaction among the different stakeholders and the usage of metamodels.

The **MATILDA reference architecture is divided in three distinct layers**: namely, the **5G-ready Applications Layer**, the **Applications' Orchestration Layer** and the **Network and Computing Slice Management Layer**. **Separation of concerns per layer** is a basic principle adhered towards the design of the overall architecture. The Applications Layer is oriented to **software developers**, the 5G-ready Application Orchestration Layer is oriented to **application service providers** and the 5G Infrastructure Slicing and Management Layer is oriented to

telecommunications infrastructure providers. It is worth noting that this clear separation of concerns, along with automatized deployment operations, facilitates, on one hand, the adoption and integration in their systems on the part of the involved stakeholders; on the other hand, it allows the separate installation, if desired, of the different parts of the platform of interest to different stakeholders.

The 5G-ready Applications Layer takes into account the design and development of 5G-ready applications per industry vertical, along with the specification of the associated networking requirements. The associated networking requirements per vertical industry are tightly bound together with their respective 5G-ready applications' graph, which defines the business functions, as well as the service qualities of the individual application.

The Applications' Orchestration Layer supports the dynamic on-the-fly deployment and adaptation of the 5G-ready applications to its service requirements, by using a set of optimisation schemes and intelligent algorithms to provide the needed resources across the available multi-site programmable infrastructure.

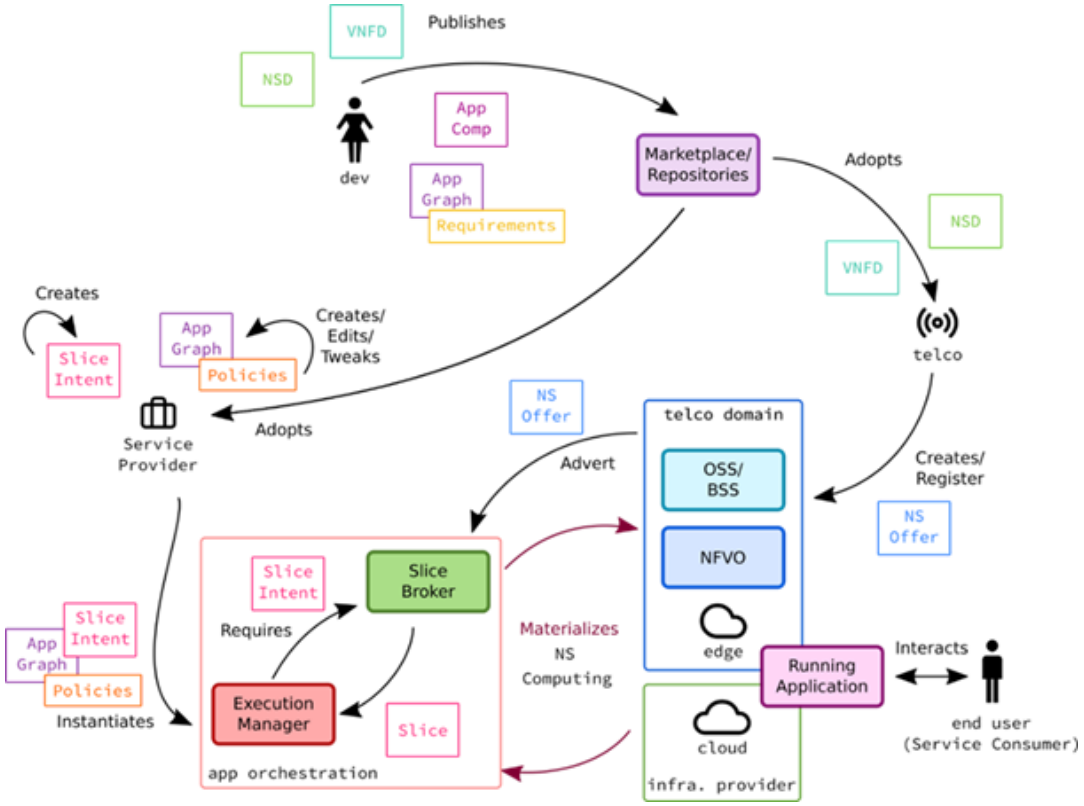

Figure 1: MATILDA workflow highlighting the different stakeholders and metamodels.

Finally, the Programmable 5G Infrastructure Slicing and Management Layer is responsible for setting up and managing the 5G-ready application deployment and operation over an application-aware network slice. Network slice instantiation and management, network services and mechanisms activation and orchestration, as well as monitoring streams management, are realized. Such actions are triggered based on requests provided by the Applications' Orchestration Layer through the specification of Open APIs.

3 Vertical Applications' Success Stories: The MATILDA Platform at Work

3.1 A Paradigmatic MATILDA Vertical: Industry 4.0

3.1.1 A scenario of safe unfenced human-robot interaction in assembly lines

With the development of Industry 4.0, the automation processes in industrial environments increase, posing important requirements for safe collaboration between human beings and machines in the workspace environments. The manufacturing industry has begun to develop the trend of customization, individuation, and flexibility, leading to close contact between robots and workers, while the traditional industrial robot production methods cannot meet the needs of production safety. So, for human-machine collaboration, security aspects have become a top priority.

The implementation of concepts and technical solutions regarding **Human-Robot-Interaction (HRI)** still predominates in the automotive and electronics industry. HRI approaches are currently being tested and implemented in handling and assembly processes by means of collaborative lightweight robots with limited load capacities. However, if heavy-duty industrial robots with load capacities beyond 35 kg are to be used in an HRI-capable assembly process, the applicable safety requirements for collaborative industrial robot systems according to ISO 10218-1, ISO 10218-2 and DIN ISO / TS 15066 (Robots and Robotics – Collaborative Robots) Technical Specifications still represent a significant challenge for safe unfenced human-robot interaction in assembly lines. This is due to the specific robot features (e.g., robot weight, range, speed, …) and resulting human safety hazards. For this reason, industrial end users must be provided with easy-to-integrate functional-safe systems that can enable a higher collaboration level (shorter safety distance between human and heavy payload-robots), **ensuring both human safety and flexible use in existing and future assembly lines**, and that are largely independent of specific robot types.

In this scenario, a significant challenge is to enable an automated process to protect the human factor in human-robot collaborative systems, mainly by co-implementing the following two key safety aspects:

a) detection of possible collisions in real time
b) automated machinery control for **run-time human collision avoidance**.

The key requirement here is to minimize the end-to-end latency between the human movement and the reaction time of the machinery. Taking into consideration the aforementioned aspects, this can further break down to the optimization of both (i) the detection algorithm and (ii) the feedback control process. To optimize the detection algorithm, one has to focus on minimizing the amount of time required to output a valid decision, while feedback control process optimization involves quick transmission, routing, and dispatching of the produced feedback through the underlying network.

The overall implementation concept relies on direct and continuous monitoring of the area around the robotic machinery and the identification of movements in this area. When a movement falls within the coverage area of the robotic machinery's sensor, depending on the speed and direction of the movement, a circle around the moving object is created with a radius proportional to its speed. If this circle overlaps with the set of coordinates of the machinery,

then a control signal is sent to the machinery to halt its motion. The monitoring of the area is done through a video stream from a camera located on top of the monitored machinery.

3.1.2 The Industry 4.0 MATILDA Use Case

This Industry 4.0 application, developed by MATILDA partner BIBA, has been chosen as one of five MATILDA demonstrators and involves a human moving towards or close to the working space of the robotic arm. Before delving into some details of the demonstrator, it is worth highlighting the role and significance of the MATILDA platform in this specific vertical, in the light of the MATILDA characteristic features that have been summarized in Section 2 above. Additional details on this use case and demo can be found in reference [6].

3.1.2.1 Advantages of using the MATILDA platform

A major advantage offered here by the MATILDA platform, with respect to a more "traditional" implementation that would not rely on MATILDA orchestration features, is **the guaranteed reliability and high availability in the interconnection of distributed manufacturing facilities**, which is made possible by the realization of a network slice dedicated to serve the application's needs. Moreover, real-time information collected and transferred from/to production processes via the MATILDA monitoring tools enables a better access to remote process control and process monitoring. This will not only increase the Quality of Experience (QoE)/Quality of Service (QoS), but also the performance and flexibility in the use case scenarios. The presence of 5G New Radio (NR) wireless networking equipment would be necessary to contribute ensuring the satisfaction of the tight delay requirements of this application, by providing the benefits of Ultra-reliable and/or Low Latency Communications (URLLC) services for the wireless segment; alongside this, however, **the MATILDA support of Mobile Edge Computing (MEC) will help ensure the compliance of end-to-end delay requirements**, by deploying intensive data processing tasks in the close proximity of the involved humans and machinery. Additionally, the capability of supporting secure and private end-to-end services is highly relevant in business scenarios where data transfer between interconnected manufacturing facilities is necessary.

Besides these advantages**, the application deployment and orchestration platform offers tools to the application developer to specify the communication needs and constraints of the microservice components and to deploy the application graph and onboard the application components in very short tim**e (2 and 5 min have been demonstrated, respectively, for these operations, as will be shown below). This is **accompanied by the automated generation and deployment of the network slice**, once the slice intent, created on the basis of the developer's specifications, has been passed to the OSS in the network domain.

3.1.2.2 Scenario Architecture and Deployment

The industrial production scenario is depicted in Fig. 2, along with the cloud-native application components that enable its automation. The involved physical devices are the robotic arm, an IP camera to monitor the area around it, and a Programmable Logic Controller (PLC) that implements the control actions upon the arm. The **cloud application** is composed by: i) **a video streaming server**; ii) **a motion detector function** for real-time distance calculation; iii) **a broker that publishes meaningful events** regarding identified objects to the PLC; iv) **a database for storing the produced events** ("Influx DB") along with a data visualization user interface that visualizes the relative positions of the identified objects around the robotic arm (shown as "UI" in Fig. 2).

The IP camera inspects a designated area around the robotic arm and streams the H.264/AVC-encoded[1] video toward the video stream server, which is the first component of the vertical cloud application. Its sole purpose is to transcode the received video frames in order to produce a lightweight equivalent of the video stream. The transcoded video frames are then transmitted to a motion detector (i.e., the second component of the vertical application graph). This component performs two operations: (i) it renders the video frames and (ii) it inputs the rendered frames to a motion detection algorithm. This algorithm employs a minimum enclosing circle scheme[2] to calculate the proximity between the robotic arm and an identified object. Each of the detected objects is represented as a circle along with a set of coordinates that are used to devise its physical distance from the robotic arm. These coordinates are sent to the PLC broker (i.e., the third application graph component) and are in turn communicated to the PLC device through a secure connection channel. Having those coordinates and knowing the exact location of the robotic arm, the PLC knows whether the robotic arm is likely to collide with any of the detected object(s) or not, and thus can act accordingly. The fourth component of the vertical application graph is the Influx database instance connected with a graphical user interface for real-time data visualization.

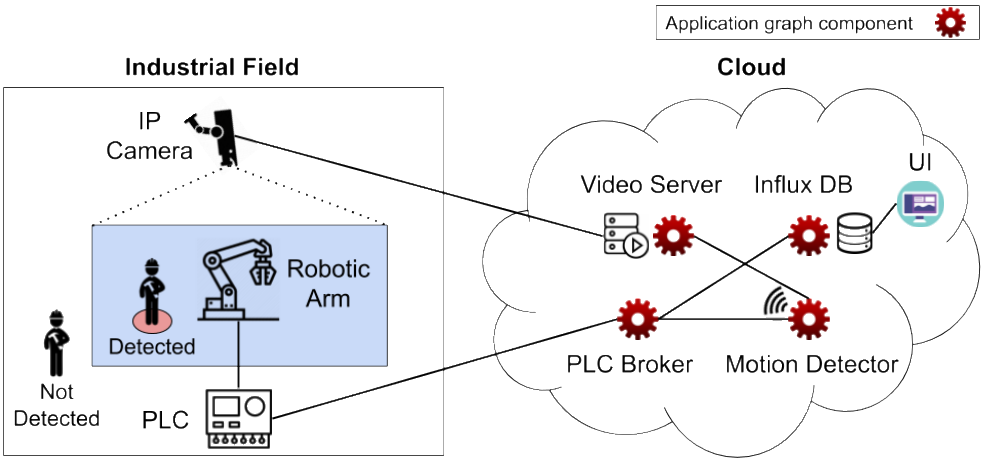

Figure 2. Industrial Production Scenario for the Robotic Arm Control.

The post-deployment application graph is shown in Fig. 3, along with the log of the automated operations triggered by the orchestrator. **In practice, what the developer needs to do to create the application graph and trigger its automated deployment is to fill in the forms that specify the characteristics of the application to be conveyed to the VAO, including the hard or soft constraints requested for, regarding the communication needs of the application components**.

Finally, to complete the description of the paradigmatic use case that is the object of this section, we report some of the results of the demonstrator and performance evaluation test bench that was deployed at Ubitech premises in Athens for this purpose. Though the difficulties created by the outburst of Covid-19 dictated the use of a mock-up robotic arm in lieu of the real industrial device located at the premises of partner BIBA, all significant KPIs of the MATILDA platform related to this use case and the components shown in Fig. 1 could be tested and showcased.

[1] ITU-T H.264 (06/2019) - Advanced video coding for generic audiovisual services.
[2] https://www.geeksforgeeks.org/minimum-enclosing-circle-set-2-welzls-algorithm/

Figure 3. Post-deployment application graph.

The results regarding operational KPIs and network KPIs are shown in Table 1 and Table 2, respectively, along with some of the propagation and processing delays related to individual components (Table 3). As can be seen, **the only unsatisfied network KPI regards end-to-end delay** (in terms of the amount of time it takes for the robotic arm to stop when a human in the operating range is detected), but **this is due to the 4G implementation and would be satisfied upon full 5G compliance without requiring changes to the platform (other than in terms of physical equipment)**.

Table 1: Results of measuring the operational KPIs for the Smart Factory Demonstrator.

KPI	Description	Acceptance Criteria/ Threshold	Result
DLPMA (production scenario) deployment time	Time required to deploy DLPMA (production scenario) application graph	~5 minutes	**~2 minutes** (see log in Figure 1)
DLPMA (production scenario) on-boarding time	Time required for the App developer to onboard the DLPMA App (production scenario) components	~15 minutes	**~5 minutes**
Resource Usage Monitoring	Compute/storage/networking resource usage monitoring	Must be available	**Available**
Scaling time	Time required to trigger the scaling after threshold was reached	~ 30 s	**7 s** **1 minute and 20 s to for the VM to be operational**

KPI	Description	Acceptance Criteria/ Threshold	Result
Availability	Service availability	High >99%	**PASS**
Reliability	Service reliability	High >99%	**PASS**
DLPMA (production scenario) repository	Repository for the on-boarded App	Must be available	**Available**
Locality Awareness	The DLPMA (production scenario) requires locality awareness	Not Mandatory	**{feature supported by MATILDA framework}**
Muti-site management	Functionality at its core is implemented as a distributed application	Not Mandatory	**{feature supported by MATILDA framework}**

Table 2: Results of measuring the network KPIs for the Smart Factory Demonstrator.

KPI	Description	Acceptance Criteria/ Threshold	Result
Availability	Network availability	>99%	**PASS**
Reliability	Network reliability	>99%	**PASS**
Delay/Latency	End to end delay will define the amount of time it takes for the robot arm to stop when a human will be detected	<100 ms	**~500 ms**
Bandwidth	High Bandwidth is required for delivering rich video stream to the Detector component in order to make proximity analysis	~10 Mbps	**~12 Mbps**
Jitter	Time-critical communications should be stable and reliable. Timing variation must be minimal	<1 ms	**PASS**
Packet Loss	Reliability and high availability of the services in HRC needs to be guaranteed. Therefore, packet loss should be made as small as possible	<0.01%	**<0.001%**
Network Slicing Capability	Network Slicing Capability for low latency communication	Must be Available	**PASS**

Table 3: Individual propagation and processing delays.

Action	Delay
Propagation time between the IP camera and the video stream server & transcoder	~5 ms (IP camera is over LTE)
1) Processing time for video transcoding	~380 ms (this time is an aggregate time of actions 1, 2, 3)
2) Propagation time between the video stream server & transcoder and the motion detector	
3) Processing time for rendering the transcoded stream	
Processing time for object identification and distance calculation	~30 ms (measured between the system calls of the Python code)
Propagation time between the motion detector and the PLC broker	~1 ms (intranet delay)
Propagation time between the PLC broker and the PLC which controls the robotic arm	~1 ms (intranet delay)
Response time of the robotic arm after a PLC command has been issued	~20 ms (according to manufacturer)

4 Conclusions

We have examined the structure, architectural organization and significant achievements that specifically characterize the MATILDA project. **We believe that the separation of concerns and the definition of clear interfaces and APIs between the different actors in the 5G applications' value chain (namely, Application Service Providers and Telecommunication Services and Infrastructure Providers) is a very relevant concept for the future development of 5G (and beyond) enhanced services on a sound basis, and that MATILDA has provided a solid contribution in this respect.**

The platform that has been developed by the project is **flexible and user-friendly**. Its installation and usage on the part of application providers requires the **normal knowledge that is expected by cloud application developers, and no specific knowledge of the 5G telecommunication infrastructure**. Once generated, **the application graph is deployed in an automatic fashion on top of the suited network slice** instantiated on the underlying Telco infrastructure of choice by the Telco part of the platform, which conforms to the ETSI MANO standards.

On the other hand, **the Telco part** of the MATILDA platform may prove to be a valuable tool on its own **in the interest of Telco providers (especially small ones)** or even for cloud application developers that want to deploy and manage their own telecommunication services and infrastructure as part of their commercial offer. This capability is provided again by the **separation of concerns** with the cloud application world, by the **specification of clear interfaces and APIs** that allow transparent mutual interaction and by the **availability of VNFs and blueprints that already populate the MATILDA marketplace**.

Along with the synthesis of the above characteristic features, we have provided the **"end-to-end story" behind one among the most significant use cases** that have been addressed and demonstrated by MATILDA, related to Industry 4.0. **Other four significant use cases** are described in a similar fashion in Appendix A of this document.

Appendix A: Other MATILDA Verticals at Work

As we mentioned in the introduction of this document, there are other four vertical application environments where the MATILDA paradigm and platform have been applied and the positive impact of 5G-ready application development and deployment has been demonstrated: namely, **Public Protection and Disaster Relief (PPDR), Smart City Lighting, Intelligent Retail Advertisement and Distributed System Testing**. We describe them and outline the main results in this Appendix.

A.1 5G Emergency Infrastructure with SLA Enforcement (5GPPRD)

This use case, developed by MATILDA partner ININ (Internet Institute), considers a scenario of Public Protection and Disaster Relief (PPDR). It extends the capabilities of a real time intervention monitoring and critical infrastructure protection product suite (iMON), combined with a suite for performance monitoring engines to support Service Level Agreements – SLAs (qMON), thanks to the new capabilities offered by 5G. The detailed description of the latest implementation is contained in reference [7].

The demonstrator has been designed to deliver tailor-made services and applications to support public safety teams in their day-to-day operations as well as during extreme situations requiring large interventions, such as massive natural catastrophes. The implementation is completed on top of a 5G infrastructure and is based on the 5G-enabled emergency response capabilities provided with the iMON product suite for real time intervention monitoring, extended with continuous performance monitoring engines of the qMON solution for supporting active and passive Service Level Specifications (SLS) monitoring and SLA enforcement. The overall architecture of the final demonstrator release (showcased in Ljubljana, Slovenia, in Sept. 2019 and during the MATILDA Webinar of June 2020 – https://www.matilda-5g.eu/index.php/events/39-beyond5g) is shown in Fig. A1. It contains the three main actors involved: namely,

i) **5G PPDR Service Provider** (provisioning and orchestrating the iMON dashboard application through the MATILDA VAO, hosted on the OpenStack-based cloud (IaaS)).

ii) **5G PPDR Network Provider** that also operates the OSS, as well as a fixed and mobile PPDR network, with the latter running on completely virtualized mobile core and thus offering a certain level of remote provisioning and configuration. The network part is segmented into three areas, i.e., Core, Edge and Cell Site, with each of them offering its own IaaS capabilities (Centralized, Edge and Site), which enables considerable flexibility with regard to the deployment and operation of the distributed iMON Dashboard. The deployment configuration is done through the MATILDA OSS that acts as an interface between the MATILDA VAO requesting a slice for the deployment of the iMON Dashboard application, and the OSM which is then responsible for configuring the network part including the mobile network access. Additionally, to provide network monitoring capabilities, OSM is also responsible for the deployment of the qMON NFV-based components (e.g., qMON Server VNF running in the same IaaS where the iMON Dashboard application is hosted, qMON Agent VNF running at the Site IaaS). The implementation of the fixed and mobile PPDR network with virtualized mobile core capabilities, MEC and other network and application related functions were deployed on

the integrated MATILDA testbed from CNIT (Genoa, Italy) and ININ (Ljubljana, Slovenia). ININ made available its experimentation platform PPDR ONE, which was implemented also as part of the 5GINFIRE project

iii) **5G PPDR Users**, representing the users that operate the iMON Dashboard application provided by the **5G PPDR Service Provider** and are connected through the network hosted and operated by **5G PPDR Network Provider.** Users can operate at the disaster site (e.g., field operatives using iMON application on their Android smartphones) or even deploy a mobile Base of Operations (BoO) with a mobile router gateway. In both cases, the qMON Agent can also be deployed to provide end-to-end network monitoring capabilities, either as an Android application running in the background or as standalone program running on the mobile router gateway in the BoO.

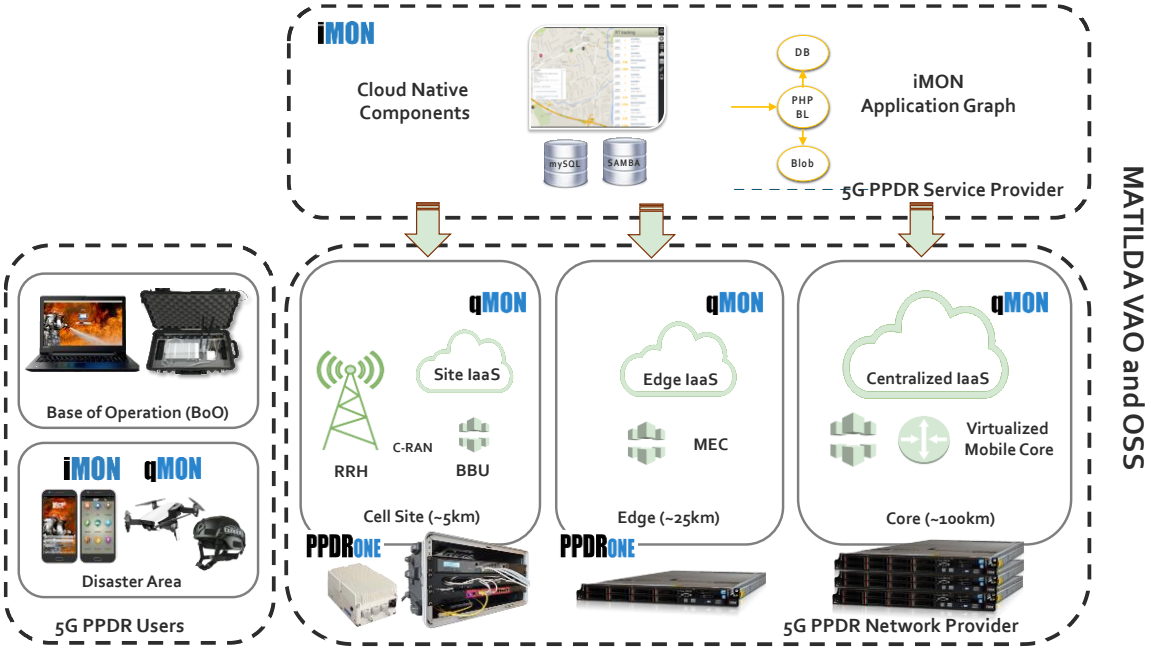

Figure A1. Evolved architecture of the final demonstrator release.

The testbeds' integration is shown in Fig. A2. During the demonstration, the following objectives were achieved and evaluated:

- **PPDR Network Provider infrastructure provisioning and deployment in a distributed environment** (virtualized mobile network (vEPC), Bypass VNF network component, Provisioning of base station (C-RAN))
- **Automated deployment of the PPDR application (iMON Solution) in a distributed environment:** (iMON application graph with components, storage components, scaling of the iMON components)
- **Monitoring of the PPDR Network Infrastructure – QoS related metrics** (deployment of the qMON VNF based on VAO/OSS/OSM, qMON VNF components placed on edge and core network locations)
- **e2e monitoring of the PPDR Applications – QoE (App) related metrics** (automated deployment of the qMON VNF (server) based on VAO/OSS/OSM, manual deployment of the qMON Agent on the PPDR UE device, qMON VNF server components placed on cell site, edge and core network locations).

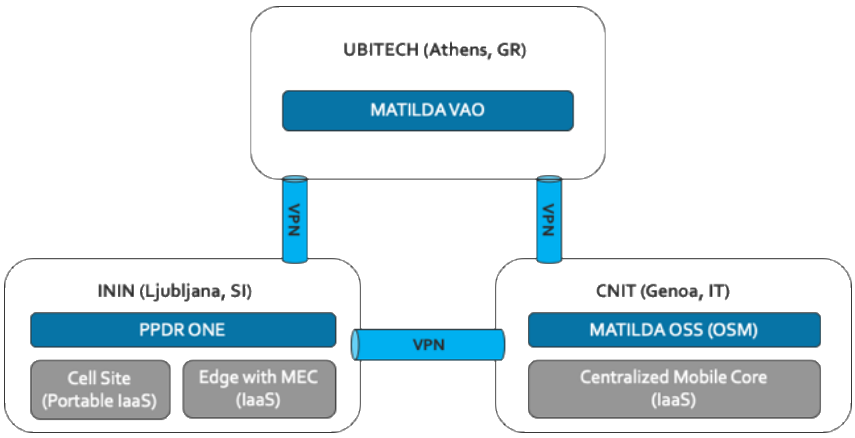

Figure A2. Testbed integration.

The application graph is shown in Fig. A3, and Fig. A4 illustrates horizontal scaling, triggered by measured traffic in pkts/s.

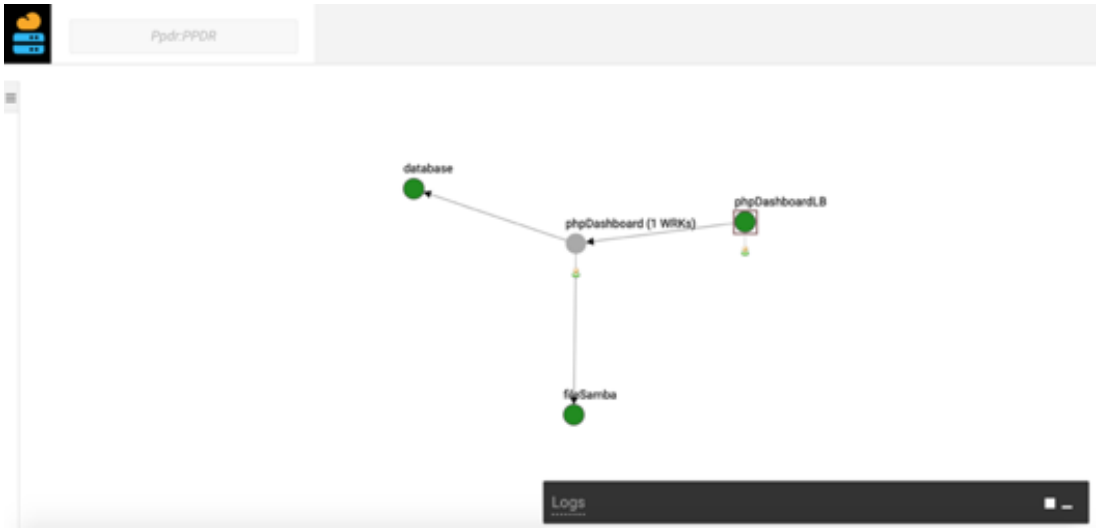

Figure A3. Application graph that will trigger the slice deployment.

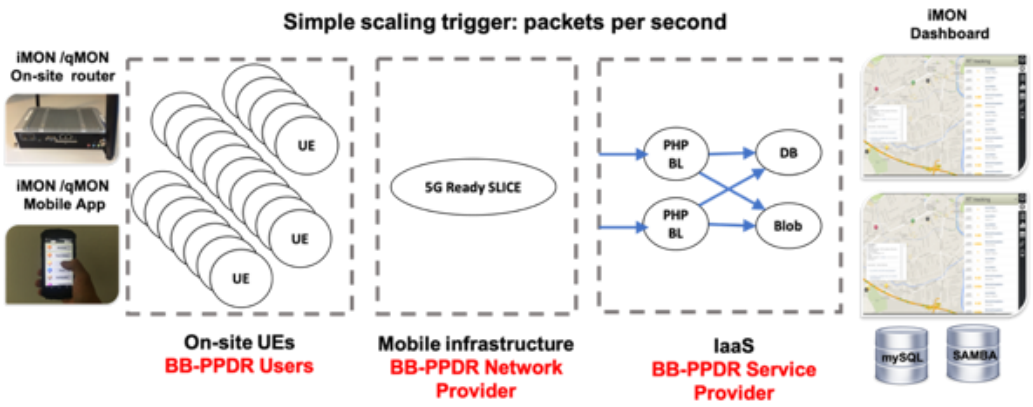

Figure A4. Horizontal scaling.

Measured operational and network KPIs are shown in Tables A1 and A2, respectively.

Table A1. Results of measuring the operational KPIs for the PPDR Demonstrator.

KPI	Description	Acceptance Criteria/ Threshold	Result
iMON Dashboard components on-boarding time	Time required for the App developer to on-board the iMON Dashboard components	~ 15 minutes	PASS
iMON Dashboard component deployment time	Time needed to deploy an individual component of the application graph	~ 3 minutes	PASS
iMON Dashboard application graph deployment time	Time to deploy the iMON Dashboard application graph	~ 5 minutes	PASS
Resource Usage Monitoring	Compute/storage/networking resource usage monitoring	Must be available	PASS
iMON Dashboard component scalability	PHP BL components of the iMON dashboard must support horizontal scaling	Must be available	PASS
Scaling time	Time required to trigger the scaling after certain threshold was reached	~ 30s	PASS
Availability	Service availability	High > 99.99%	PASS
Reliability	Service reliability	High 99.99%	PASS

Table A2. Results of measuring the network KPIs for the PPDR Demonstrator.

KPI	Description	Acceptance Criteria/ Threshold	Result
Availability	Network availability	> 99.999 %	PASS
Reliability	Network reliability	> 99.999 %	PASS
Network Slicing Capability	Network Slice Management	Must be available	PASS
End-to-end Latency for interactive applications	Connected devices should be able to communicate without significant delay/ latency. Example application: Real-time queries in transactional databases	< 20 ms	FAIL[3]

[3] Due to unavailability of 5G terminal equipment (i.e., modem/smartphone) and 5G radio access network (URLLC), the tests were performed using LTE-A technology and thus such KPIs could not be reached.

KPI	Description	Acceptance Criteria/ Threshold	Result
End-to-end Latency for mission critical applications	Connected devices should be able to communicate without significant delay/ latency. Example application: Remote control of drones and robots	< 1 ms	FAIL[4]
Bandwidth	High bandwidth required for: ▪ Data intensive applications for PPDR use ▪ Ultra HD video streaming from disaster site (land and aerial based)	~ 20 Mbps/user	PASS
Jitter	Time-critical communications should be stable and reliable. Timing variation must be minimal	< 1ms	PASS
Packet Loss	Reliability and high availability of the services in extreme conditions is essential for emergency systems. Therefore, packet loss should be made as small as possible	< 0.01%	PASS

A.2 High Resolution Media on Demand with Smart Retail venue integration (5GPACE)

The 5GPACE App developed, deployed and tested in this use case consists of two main parts, one provided by MATILDA partner Incelligent – INC (Retail Recommendation - Machine Learning Powered), the other by MATILDA partner Italtel – ITL (User management, localization data management and storage, video processing functionalities).

While each one of the parts is by itself a modular/microservice-based application, they can be integrated together; thus, forming a larger application graph that can be deployed in any orchestrator that follows the microservice/service mesh paradigm. However, the MATILDA orchestrator is able to provide important aspects of 5G orchestration, which are critical for guaranteeing network performance and deployment infrastructure.

To this end, 5GPACE has been transformed into a collection of components, which run in the MATILDA virtualization infrastructure as containers. The 5G-ready application has been created by following a guided interactive procedure through the MATILDA Graphical User Interface. Details of the final implementation and of the demonstration can be found in [8].

5GPACE (5G Personal Assistant in Crowded Events) aims to offer high valued services to consumers participating in sporadic Crowded Event (CE). In a CE, a high number of end users concentrate in a small area for a relatively short time, ranging from few hours to a week. Well-known examples of CEs are sport events in stadiums or exhibitions in dedicated venues, but also touristic locations during seasonal peak-periods or malls at peak-hours. During CEs, beside an impressive data traffic growth, one can clearly observe a shift of consumers' behaviour, with more video-related activities and social networking, and less voice calls and text messages.

[4] Due to unavailability of 5G terminal equipment (i.e., modem/smartphone) and 5G radio access network (URLLC), the tests were performed using LTE-A technology and thus such KPIs could not be reached.

However, offering innovative services during CEs poses demanding requirements, which have highlighted the weaknesses of the present telecommunication infrastructure, and have thus played a key role in the definition of the new 5G architecture.

Leveraging NFV and Software Defined Networking (SDN), 5GPACE combines the Italtel i-EVS system for high quality, immersive video services and geo-localization functionalities, with the Incelligent's Artificial Intelligence-based framework which, by analysing historical and mobility data, can provide personalized recommendations to each single user in real-time. The proposed system has been specifically designed to operate in a smart retail scenario and can be deployed in a mall or in a touristic location, so that users can benefit from a bundle of innovative services in a straightforward and immediate way. An example scenario is shown in Fig. A5.

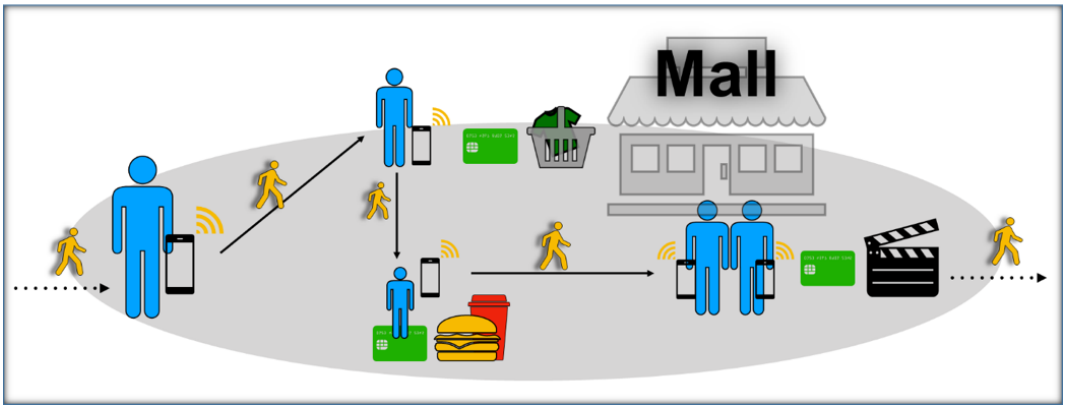

Figure A5. 5GPACE - User journey through a mall.

The user, through his/her mobile device, can discover personalized recommendations and offers, based on the user's exact current location and preferences to consume certain types of products or services (possibly in sequences or bundles). Such data-driven retail recommendations come in the form of a "Move to next shop that offers an A% discount just for you" visual aid on the user's mobile screen and are created by applying advanced machine learning methods on the user's purchase history and mobility data, in user-perceived real time.

The application components are shown in Fig. A6. **i-EVS** provides video processing (VTU: Video Transcoding Unit) and data storage functionalities (sTORE: storage services) and manages information related to the users (organized in groups) accessing i-EVS services, by storing and maintaining updated the related data in the i-EVS user and group database (UGDM: User and Group Database Management). **5GPACE APP** is the Android user application, whose main screen and functionalities are shown in Figure A7. The **Smart Retail Recommendation Service (SRRS)** aims to provide context- and location- accurate recommendations of new shops, purchase opportunities, marketing promotions to a user moving around a sparsely crowded area such as a shopping centre/mall. It includes a Venue Management Service (VMS), a Retail Recommendation Service (RRS), a Load Balancer (LD) and Recommendations Delivery (RD).

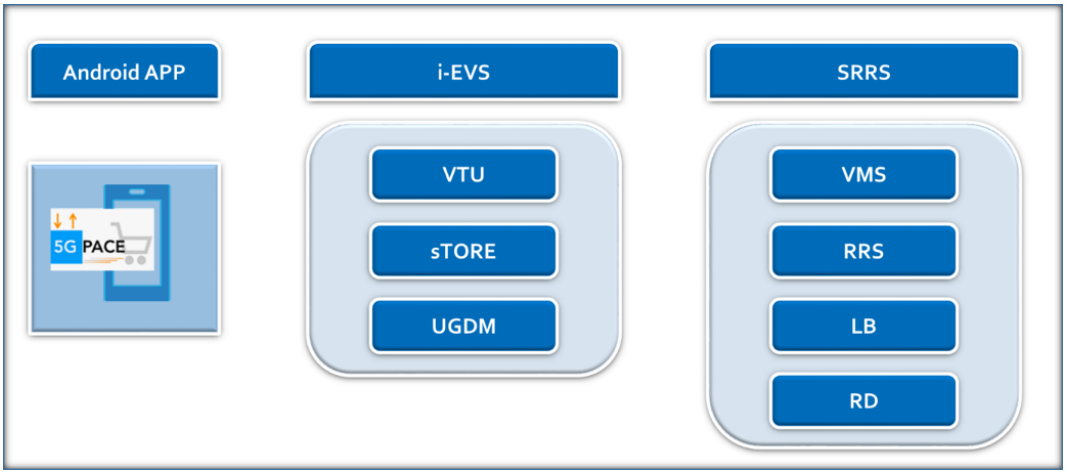

Figure A6. 5GPACE application components.

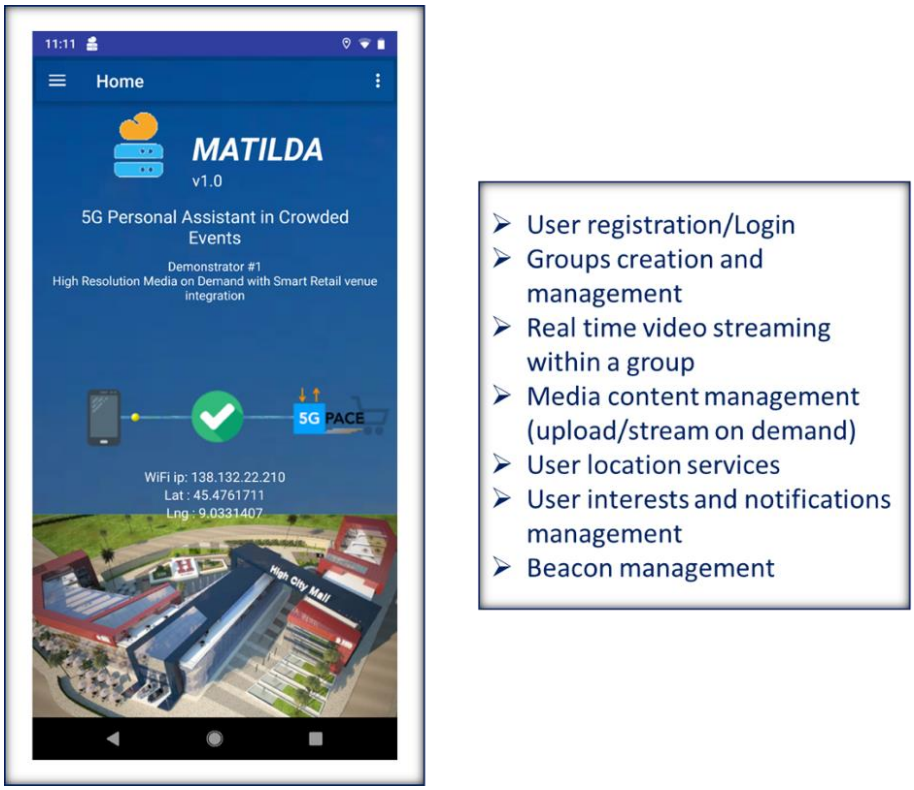

Figure A7. 5G PACE Android App.

Figure A8 shows the functional workflow of the system, as implemented in the demonstration.

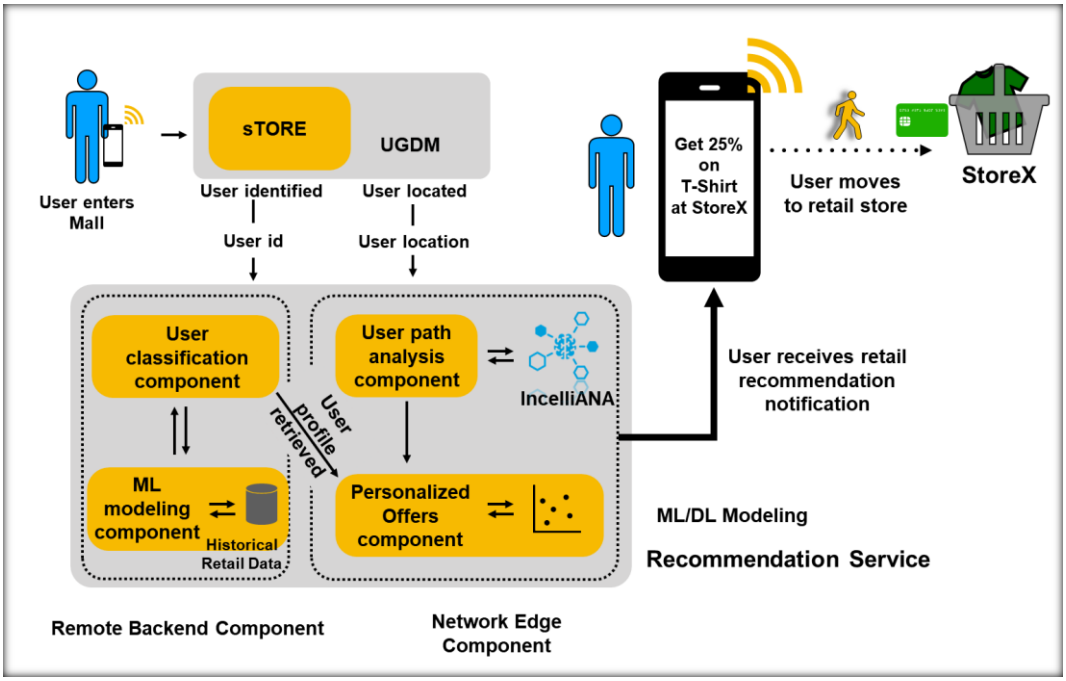

Figure A2: Functional workflow of the system.

The deployed Application Graph is shown in Fig. A9, along with the related logs.

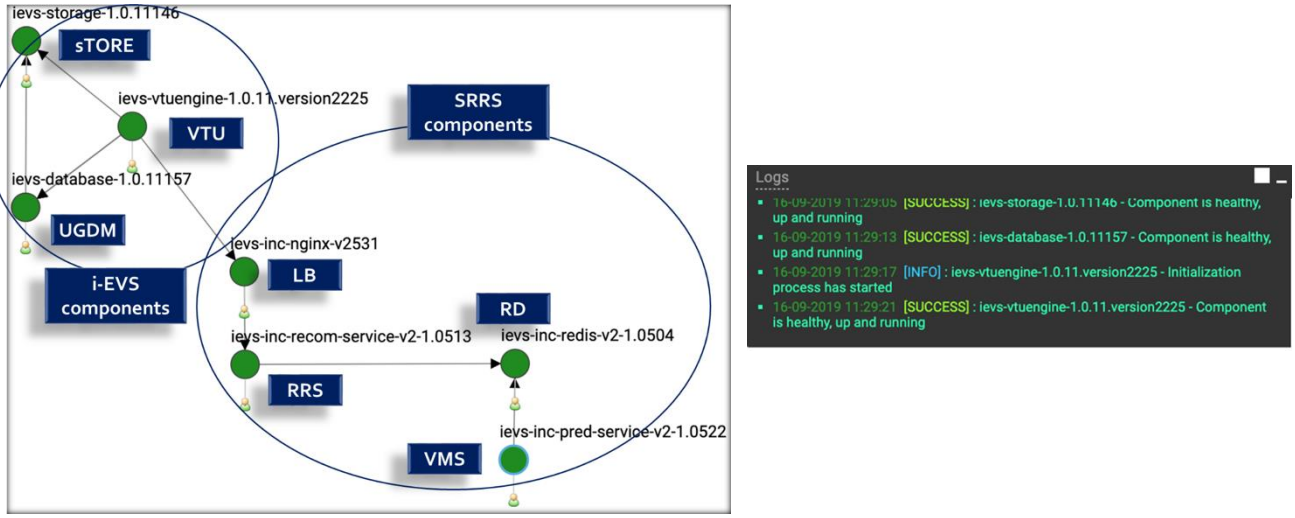

Figure A9: 5GPACE application Graph.

Figure shows a simplified view of the demo components' deployment considering different solutions regarding the physical locations of the MATILDA testbeds. The final demo was executed in the Genoa Edge infrastructure.

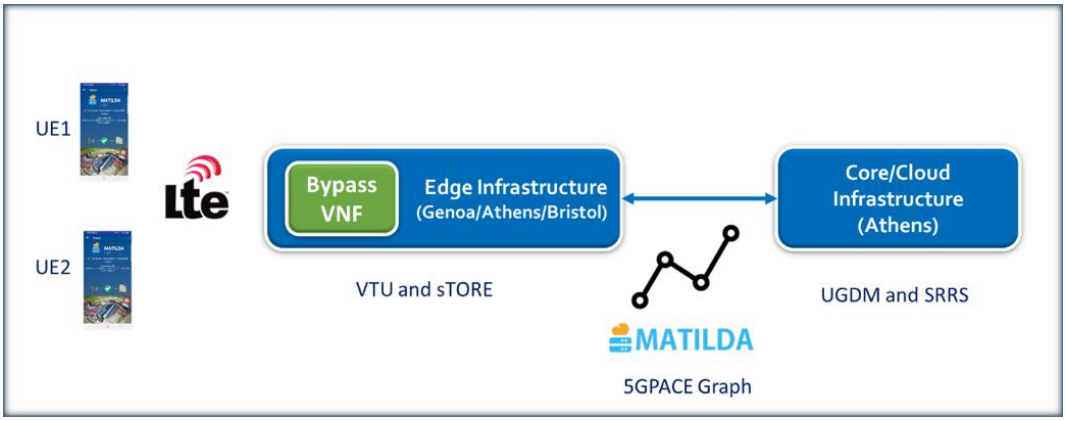

Figure A10: 5GPACE Graph deployment.

The 5GPACE demo aims at showing the main functionalities provided by the application considering the HD video sharing capabilities at the edge and the service scenario that has been described. The demo setup context is depicted in Fig. A11 and includes two scenarios:

- HD video streaming,
- Retail Recommendations process.

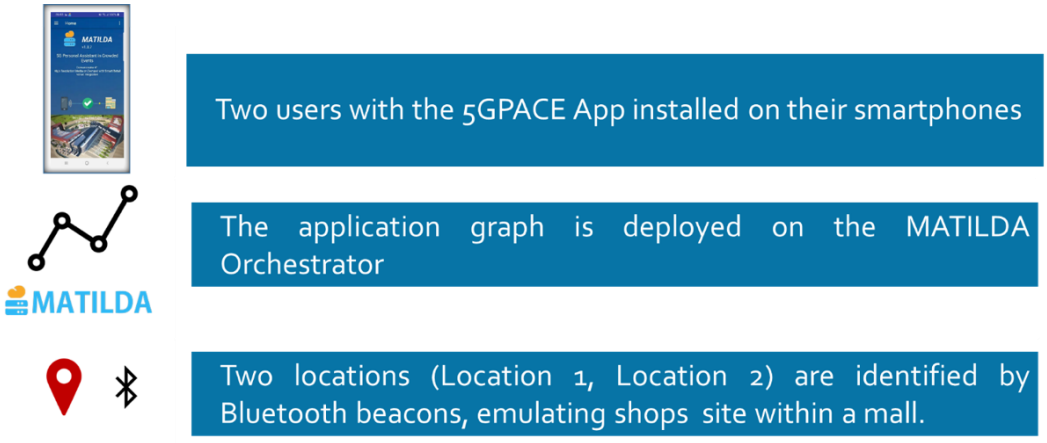

Figure A11: Final demo setup.

A.2.1 HD video streaming

In the case of HD video streaming (Fig. A12), the demo storyline is the following:

Step 1: Registration to 5GPACE services

- User A and user B perform a registration procedure to the 5GPACE application using their own smartphones, where the 5GPACE Android app is installed;
- User A creates the users group G inviting user B;
- User B will join the group upon receiving the notification sent by 5GPACE.

Step 2: HD Video Sharing

- User A moves to "Location 1";
- User A starts shooting a video with the user's smartphone and shares it within the user group G;
- User B, being in "Location 2" decides to watch in real time the HD video that User A is sending.

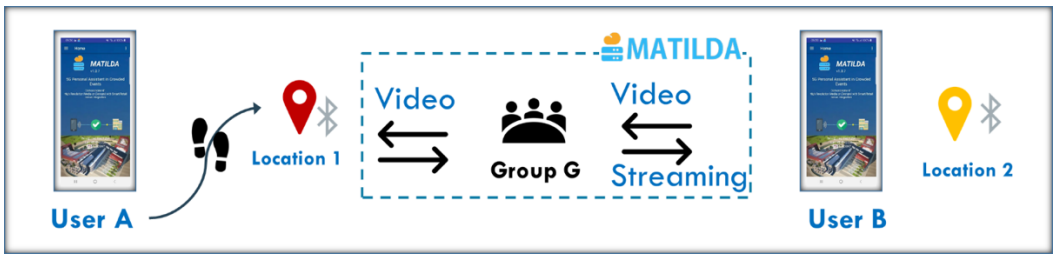

Figure A12: Scenario 1 - HD video streaming.

A.2.2 Retail Recommendation

In the case of retail recommendation (Fig. A13), the demo storyline is the following:

Steps:

- "Location 1" and "Location 2" are identified and associated to Bluetooth beacons that will be activated by hand in due time;
- User A moves to the designated "Location 1", the associated beacon is activated;
- User A gets a recommendation for the next shop, 5GPACE recommends the user to visit in the form of a video, which is streamed to the user's device;
- "Location 1" has two videos to be used as a recommendation:
  - one related to a clothing store,
  - one related to a shoes' store;
- the video that will be streamed will depend on the shopping preferences and interests associated to user A;
- User A then visits "Location 2";
- "Location 2" has two videos to be used as a recommendation:
  - one related to a grocery store,
  - one related to a coffee shop;
- also in this case the video that will be streamed will depend on the shopping preferences and interests associated to the user.

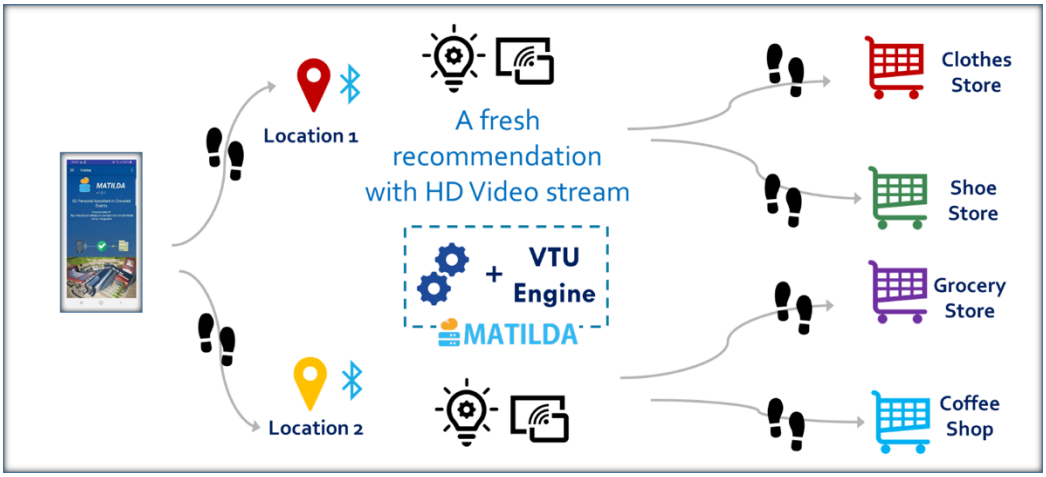

Figure 3: Scenario 2 - Retail recommendation.

In order to develop an efficient service, it is important to fulfill the QoS requirements needed in case of HD video sharing in a crowded event scenario, as well as to ensure the right response time of the system for providing smart retail recommendation through HD video content distribution to users leveraging information about their location and profiles.

To this aim, a set of networking and application KPIs and other acceptance criteria has been defined and evaluated during the different phases of the MATILDA project.

Table A3 and Table A4 report the network and operational KPIs of interest for 5GPACE.

Table A3: Network KPIs

KPI	Description	Measured (Where/ How)	Acceptance Criteria/ Threshold	Results
Network KPIs				
Device Density	The application expects a big number of connected devices	The number of users that can be managed by 5GPACE is influenced by: • the capacity of the radio access that depends on the technology used, Wi-Fi, 4G or 5G on the telco infrastructure; • the capacity offered by the VTU for transcoding and real time video streaming in	~32 per Small Cell ~50 per Wi-Fi Hot Spot	The radio access and bandwidth capacity are under the control of the infrastructure owner. The VTU (leveraging 4 vCPU, 4GB RAM and 20 GB disk) can manage 310 contemporary user sessions (CPU @90%), guaranteeing 8Mb/s per user. The VTU provides the scalability feature that can be

KPI	Description	Measured (Where/ How)	Acceptance Criteria/ Threshold	Results
Network KPIs				
		terms of available performance (i.e., number of users managed by a single instance of the component); • the overall bandwidth capacity at the edge that is offered by the telco infrastructure.		enabled using the MATILDA dashboard.
Mobility	End-User mobility	No measurement needed. The analysis was done analysing the service scenario offered by 5GPACE.	Static users/low mobility (0-3m/s)	The service scenario refers to a static /low mobility user.
Availability	Network availability	To be measured at the telco infrastructure.	>99%	Under the control of the network operator.
Reliability	Network reliability	To be measured at the telco infrastructure.	>99%	Under the control of the network operator.
User Data Rate	As a video application, high data rates per user are required.	The VTU capacity in term of video streaming per user (measured as described in [8].	~10 Mbps/user, depending on quality	HD video stream requires 8 Mb/s per user.
End-to-end Latency	For real-time video sharing, small delays are expected	The 5GPAGE delay introduced during a video streaming (measured as described in [8]).	Maximum 1 s	See Table A4..
Access Interoperability	Interoperability with various access technologies (Wi-Fi, 4G, 5G)	Measured during different steps of the evaluation process.	Must be available	4G and Wi-Fi interoperability is available. 5G not tested due to lack of equipment based on 5G radio access at the time of writing.
Edge Computing	Edge computing capabilities for network offloading	On the edge testbed located in Genoa.	Must be available	Available (VTU component located at the edge NFVI).

KPI	Description	Measured (Where/ How)	Acceptance Criteria/ Threshold	Results
Network KPIs				
Storage at the Edge	Storage capabilities to save multimedia contents at the network edge	On the edge testbed located in Genoa.	Must be available	Available (sTORE component located at the edge NFVI).
Computing acceleration at the edge	High resolution video processing requires HW acceleration	Testbed used for the final review.	Must be available	GPU was not available in the testbed. It was not possible to install it due to the COVID-19 outbreak. However, the 5GPACE application as well as the MATILDA framework are able to support this feature to be associated to the VTU component.
Network Slicing Capability	Network Slice Management	Testbed used for the final review.	Must be available	Available.

Table A4: Operational KPIs

KPI	Description	Measured (Where/ How)	Acceptance Criteria/ Threshold	Results
Operational KPIs				
5GPACE App deployment time	Time to on-board and deploy for the first time the 5GPACE App	Measured for the entire graph in the testbed deployed for the final review, tested already during continuous deployment between ITL (Milan) – INC (Athens) and the development testbeds (Athens/Genoa)	~90 minutes	~15 minutes
5GPACE App on-boarding time	Time required of the App developer to on-board the 5GPACE App	Measured for the entire graph in the testbed deployed for the final review, tested already during continuous deployment	~15 minutes	~5 minutes

		between ITL (Milan) – INC (Athens) and the development testbeds (Athens/Genoa)		
Resource Usage Monitoring	Compute/storage/ networking resource usage monitoring	Measured for the entire graph in the testbed deployed for the final review, tested already during continuous deployment between ITL (Milan) – INC (Athens) and the development testbeds (Athens/Genoa)	Must be available	Available
5GPACE App component scalability	Specific components of the 5GPACE App must be able to scale horizontally	Measured for the entire graph in the testbed deployed for the final review, tested already during continuous deployment between ITL (Milan) – INC (Athens) and the development testbeds (Athens/Genoa	Must be available	Available (VTU)
Scaling time	Time required to start/stop a component once a pre-defined parameter crosses the corresponding threshold	Measured for the entire graph in the testbed deployed for the final review, tested already during continuous deployment between ITL (Milan) – INC (Athens) and the development testbeds (Athens/Genoa)	~ 20s	~ 15s
Availability	Service availability	Measured in the testbed deployed for the final review.	High >99%	Assessed at production network deployments, thus will be assessed at phases of adoption of MATILDA framework in normal operation.

				5GPACE will contribute to reach the KPI target of the "Service Availability" because there are no access limitations to its services (e.g., statistical access to the service) and it is scalable when the number of user increases.
Reliability	Service reliability	Measured in the testbed deployed for the final review.	High 99%	Reached through network slicing and the QoS guarantees offered by the new Telecom layer (OSS).
5GPACE App repository	Repository for the on-boarded App	Measured in the testbed deployed for the final review	Must be available	Available
Locality Awareness	The 5GPACE App requires locality awareness	Measured in the testbed deployed for the final review.	Must be available	Available
HW video acceleration management	Management of HW acceleration resources in the infrastructure	Capability available in MATILDA testbeds (Athens/Genoa), PCI pass-through usable	Must be available	Available, but not tested (a GPU was not installed on the testbed).
Multi-site management	The 5GPACE App can be composed of components instantiated in different sites	Measured for the entire graph in the testbed deployed for the final review, tested already during continuous deployment between ITL (Milan) – INC (Athens) the development testbed (Athens/Genoa).	Must be available	Available

A.3 Smart City Intelligent Lighting System

The **Smart City Intelligent Lighting System** has the goal of increasing the street lighting's efficiency by addressing the connectivity from the lighting sensors installed on poles to the application server used for command, control and maintenance.

This demonstrator has been built upon successful collaboration with the SLICENET project, where part of the programmable infrastructure for MATILDA partner ORO (Orange Romania) test bed was built. While MATILDA is focused more on the end-to-end framework and on the upper layers related to marketplace and application orchestration, SLICENET is more focused on virtualization/slicing concepts. The ORO demonstrator shows how the two frameworks could be integrated in order to automate the deployment and in-life management and bringing key benefits in terms of cost and time to market. Fig. A14 shows the integration of the two testbed platforms, one hosted in CNIT Italy and the other one in ORO Bucharest premises, that supported the final demonstrator.

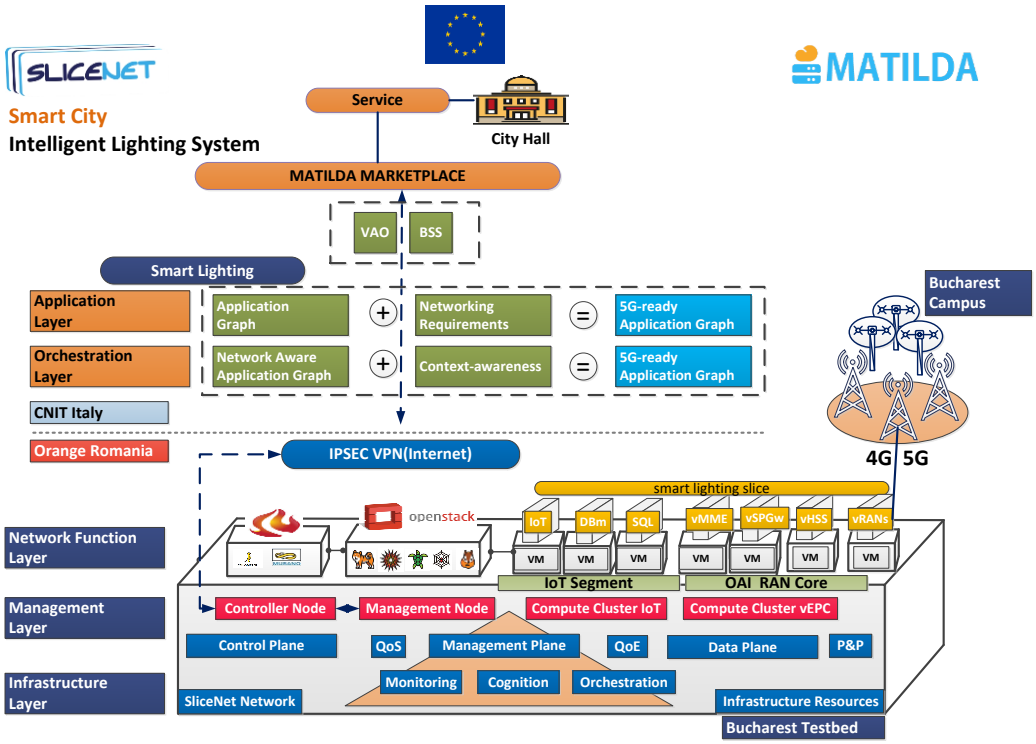

Figure A14. MATILDA – SLICENET Smart City Intelligent Lighting System testbed architecture.

The ORO testbed infrastructure is composed of four main components:

- RAN and Core network part – with the lighting poles/smart lamps and the OpenAirInterface solution for the RAN (USRP x310, vRAN) and the Core part (vMME, vEPG, vHSS), providing the connectivity of the lighting sensors through RAN components to the IoT Middleware where the ThingsBoard component and other applications are deployed.
- IoT Aggregator (IoT connector) platform – containing the BigIP Load balancer and IoT connector platform with the role of secure transmission from the lighting poles to the IoT Middleware component.

- IoT Middleware (Enterprise) component - composed by several virtual servers hosting the demonstrator components acting as:
  - orchestration and management node (h2020-server1) - OpenStack Management node containing VIM (Nova – Neutron – Keystone – Glance)
  - controller node (h2020-server2) - containing OSMv5, metrics collector Ceilometer and Prometheus server
  - h2020-server 3-5 – several computing nodes using KVM virtualization capacity to instantiate several virtual machines for all Smart City application components, having also the role of processing and storage of the information provided by Smart City sensors and actuators
  - h2020-server 4 – hosting computer cluster vEPC.

According to the MATILDA vision, all applications components depicted in Fig. A15 follow a microservice architecture, the on-boarding process of the component requiring a translation of all Smart City application components into a Docker containerized version.

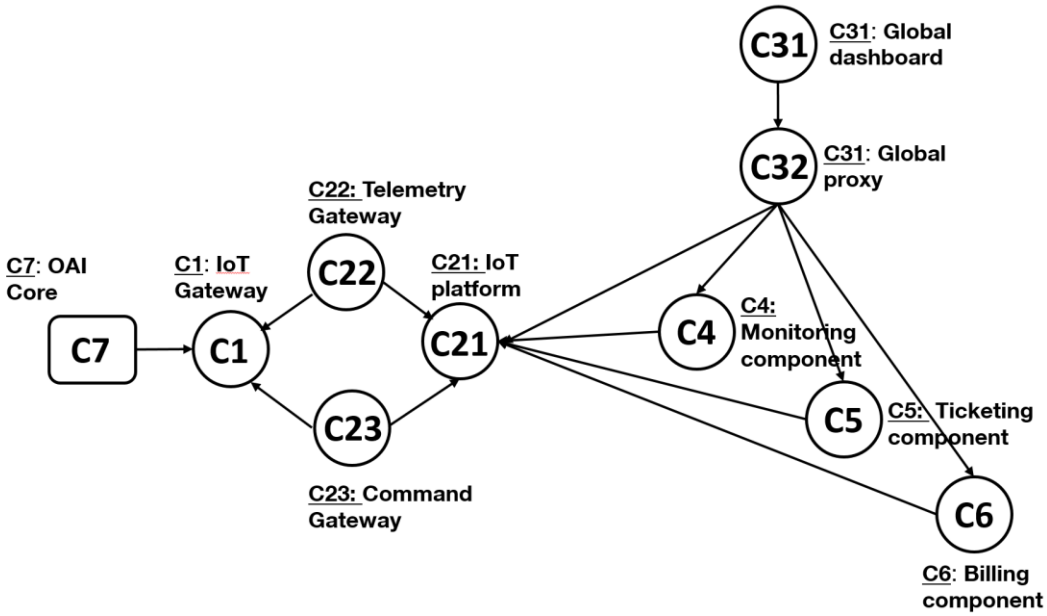

Figure A15. MATILDA – SLICENET Smart City Intelligent Lighting System application components.

The application graph includes:

- **C1 (IoT Gateway) -** a proxy between the smart lighting lamps and C22, it aggregates all the messages from the device and passes them forward towards C22, using the MQTT protocol.
- **C22 (Telemetry Gateway) -** enables the storage of telemetry data received from the smart lighting lamps. It is configured to listen over the MQTT protocol, or to a specific type of message received from C1 for each of the lamps. The messages are processed and stored in the PostgreSQL database.
- **C23 (Command Gateway) -** this module sends specific commands to C1 at every 1-10 minutes (configurable parameter) requesting for telemetry data, which are retrieved by the telemetry application.

- **C21 (IoT platform) -** enables functionalities as 'on', 'off' or value for dimming. These commands are communicated from the broker component of ThingsBoard towards C1 over the MQTT protocol.
- **C31 (Global dashboard) -** is a centralized front-end application with multiple functions that can offer the user/ admin an overall view and general control of the entire system.
- **C4 (Monitoring component) -** checks the status (including alarms) of the infrastructure and services. It has subcomponents developed for specific tasks: agent, monitoring server, database, frontend.
- **C5 (Ticketing component) -** has the role of tracking, it can offer information regarding the alarms/incidents in the smart lighting system (infrastructure or services), root cause and tenants involved.
- **C6 (Billing component) -** is connected to C21 and collects all information about tenants and users (e.g., number of devices connected, number of telemetry messages, type of devices) of the Smart Lighting System.
- **C7 (OAI Core) -** Core Network All in One container, it connects the RAN part of the network with IoT Gateway of the service.

One of the key benefits of the Smart City Intelligent Lighting System provisioned over the MATILDA platform is the zero-touch network operation for both deployment and in-life management to reduce the operational cost of both parts, the infrastructure and the service providers. The in-life management process is a contribution from the SLICENET project, addressing the automation of virtual infrastructure resource scaling (VMs) and radio resource scheduling.

To this purpose, a set of monitoring applications were developed at the infrastructure layer, to monitor the performance of VMs and the performance of radio access. The Smart Lighting service is composed of a set of network elements/VNFs and application components designed to ensure the simultaneous connection of a high number of devices over the slice. Therefore, it is crucial to have an in-life management mechanism in place to guarantee the service QoS in the probable scenario of multiple slices deployed over the same programmable infrastructure. In order to demonstrate the efficiency of the in-life management mechanism, a noisy neighbor scenario was defined:

- One slice (slice 1) was dedicated to the smart lamps – an IoT simulator was developed in order to emulate the traffic of 1000 smart lamps.
- One slice dedicated to video streaming (the "noisy neighbor") monopolizing bandwidth, disk I/O, CPUs, memory or other infrastructure resources that can negatively impact the application performance deployed on slice 1.

In order to guarantee the QoS parameters, per-slice monitoring agents are installed on the VMs and compute nodes. The main goal of these agents is to collect logs and metrics from the infrastructure and feed the information towards trained Machine Learning / Artificial Intelligence (ML/AI) algorithms able to predict and correct the behavior of the VNFs on each of the slices. The 'noisy neighbor' mechanism deployment (Figs. A16 and A17) includes several steps:

- QoE application deployment: developed in order to evaluate in real time the end-to-end metrics status of the Smart Lighting service using a Prometheus monitoring mechanism installed on a bare metal server to supervise all hypervisors of the OpenStack infrastructure and VMs of the smart city slice. The end-to-end network slice is defined by

the communication network from Lighting Pole, Radio, dedicated Core Network elements and the IoT platform. The testing packets are handled to pass the specific NFV/VNFs within the slice up to the end-device lighting pole, exploiting the capability of the end device to respond to such packets.

- Noisy neighbor simulation: the model retrieves regularly (every 15s) the metrics of VNFs to supervise and predict their status; once the noise or overload status is predicted, three consecutive predictions data sets of the same class are collected before updating the state of the VNFs.
- Resource scaling: the corresponding action can be taken according to the VNF status:
  - Scale up the VM hosting the VNF by adding more CPUs or RAM. A differentiation between the overload and noise scenario should be performed in advance.
  - Migrate the VNF to another host not impacted by the noisy neighbor.

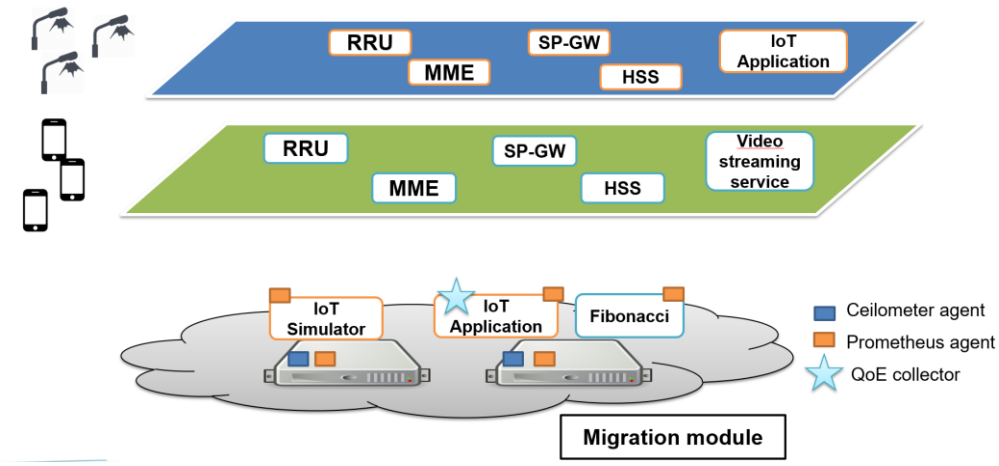

Figure A16. Smart Lighting use case - noisy neighbour scenario setup.

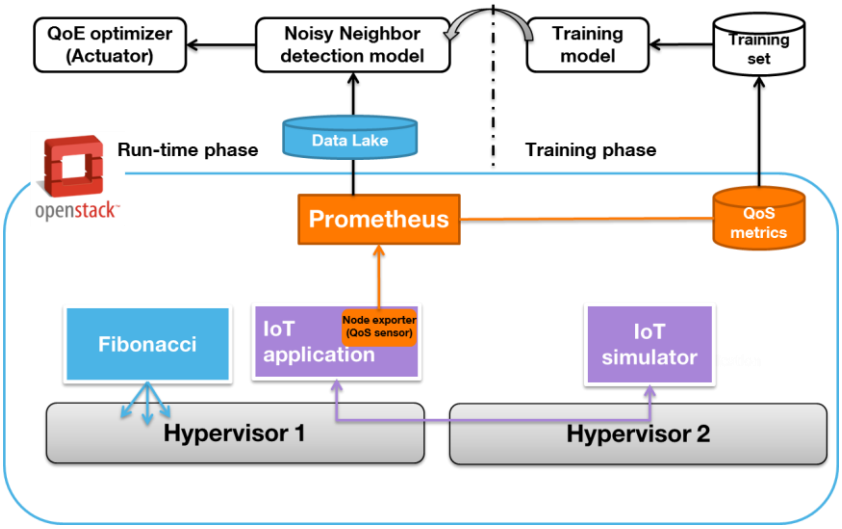

Figure A17. Noisy Neighbour ML/AI training model.

Performance KPIs are shown in Table A5, and the detailed description of the use case and demo is provided in [9].

Table A5. Smart City Lighting KPIs.

KPI	Description	Measured (Where/How)	Acceptance Threshold
Network KPIs			
Availability	Calculated as network up time / total time, reflects in percentage the availability / stability performance of Smart City demo platform	Infrastructure availability; measures how long the allocated compute and network resources are up; data retrieved with Prometeus and graphs displayed using Grafana.	> 99.99%
Operational KPIs			
Device Status	Evaluates the number of smart light sensors deployed on testbed platform.	Viewed on application's dashboard	56 Smart Light sensors
Service Availability	Calculated as service up time / total time, reflects in percentage the availability / stability performance of Smart City service	Measured with 2 scripts. One script measures the availability of service offered to the service consumer -emulates a user login on dashboard. The second script measures the availability of service provided by application: emulates command send or received to / from a sensor. The graphs are displayed using Grafana.	> 99.99%
End-to-end Latency	Measures packet round trip time from IoT platform to device sensor.	Ping measurements initiated from one agent of the IoT platform (component C2); the graph is displayed using Grafana.	< 300 msec
Jitter	Evaluates packet delay variation in latency between IoT platform and device sensor.	Ping measurements initiated from one agent of the IoT platform (component C2); the graph is displayed using Grafana.	~ 100 msec
Packet Loss	Shows the percentage of packets lost during transfer between sensors and IoT platform. The Smart Lighting service is not critical, therefore retransmission is being allowed, without affecting end-to-end application functionality.	Ping measurements initiated from one agent of the IoT platform (component C2); the graph is displayed using Grafana.	< 0.1%

KPI	Description	Measured (Where/How)	Acceptance Threshold
Operational KPIs			
Device Bandwidth Capacity	Evaluates the transfer capacity volume of information collected from sensors to IoT platform.	OAI-RAN resource allocation (component C7)	~ 0.1 Mbps
Total Slice Bandwidth	Evaluates the transfer capacity volume of aggregated information from sensors to IoT platform. Calculated as (device number) x (bandwidth/ device) (helpful for VNFs system parametrization)	OAI-RAN resource allocation (component C7)	~ 100 Mbps

A.4 Automobile Electrical Systems Remote Control

The aim of MATILDA partner ExxpertSystems that developed this use case is to provide a license-based added value service to end-customers by adopting the role of telecommunication equipment vendor (VNF/PNF Developer) and providing a Physical Network Function (PNF): namely, the ExxpertSystems FastWAN solution, based on the MATILDA Network.

The explored scenario demonstrates the data acquisition and remote testing from the automobile control system (System Under Test – SUT) units using the FastWAN solution by ExxpertSystems over the 5G network (Fig. A18) and supports the network slice isolation and dynamic management aspects with the aim to provide guarantees for isolated access to the reserved resources and advanced security levels. The enforcement of security policies during runtime is based on the specification of a set of security rules for identifying and tackling vulnerabilities and threats over the deployed application graph.

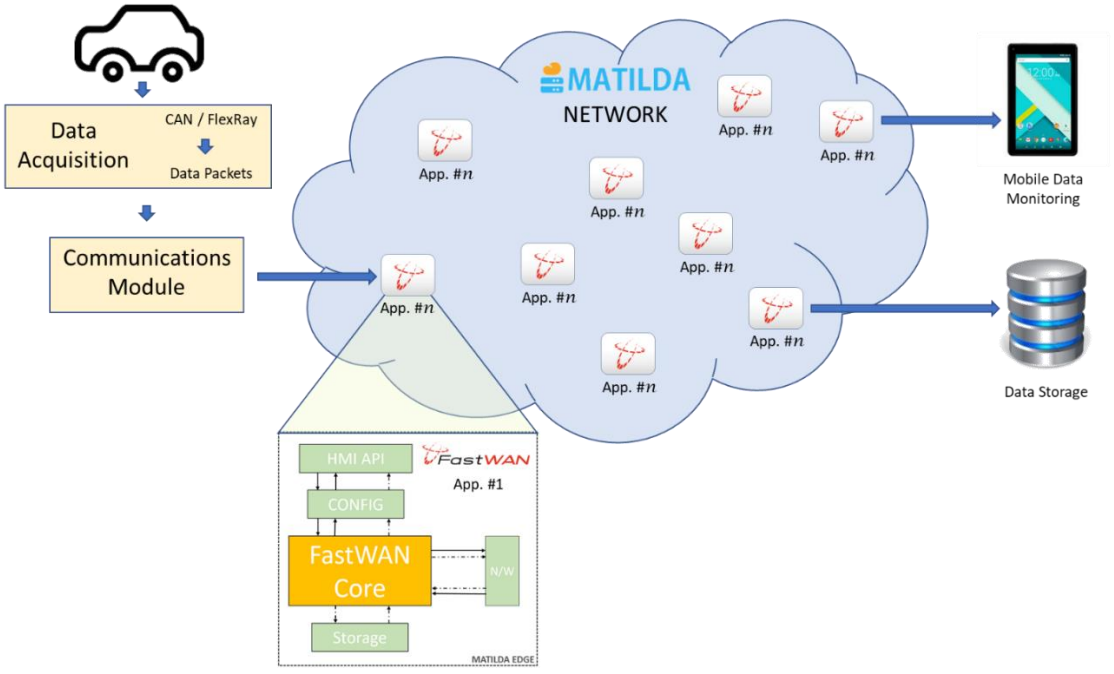

Figure A18. Distributed System Testing.

The MATILDA framework allows ExxpertSystems to create its applications following a microservices approach where each component can be independently orchestrated.

FastWAN is an experimental communication technology that was developed as a solution for the enablement of geographically separated real time industrial test benches. FastWAN presently provides the following features:

- Support and provision of virtual extensions over the Internet for a variety of industry BUS Standards such as AFDX, ARINC 429, Digital, Analog, etc.
- Assurance of time and quality delivery for all signal data with efficient data acquisition and packing.
- Deterministic and reliable reconstruction of signals at the destination by using accurate GPS signal time stamping, configurable fixed delays, and jitter compensation.
- Integration of fast standard data "container" transmission services such as Ethernet UDP.
- Assurance of reliable data transmission integrity by way of failure detection (detection of lost or incorrectly ordered packets), as well as recovery mechanisms (packet re-ordering and re-transmission).
- Efficient bandwidth usage with customized proprietary packing methods based on signal priorities and destination requirements.
- Comprehensive system control and monitoring GUI to monitor the data logistics network and virtual connection statistics.

Within the automotive Industry, FastWAN, based on the MATILDA 5G Framework, can enable a superior interconnected integration and functional testing of these end product systems over wireless WAN infrastructures. This will further improve the reduction in system development life cycle times and costs in an environment where system complexity, competition and certification requirements are dramatically increasing. More particularly, only by means of 5G technology and the MATILDA architecture it is possible to achieve greater quality of service whilst dramatically increasing flexibility. This flexibility refers to the fact that test campaign strategies can be revolutionized by eliminating geographical constraints. This 5G FastWAN solution will allow test campaigns responsible to interconnect a much greater number of devices, over a more diverse range of interface means (e.g., 4G/5G, WLAN, LAN, etc.), whilst being much less restricted by geographic location. This can be achieved while maintaining, for the entire time, the demanding requirements of the closed-loop system and equipment tests: namely, QoS, latency and flexibility. In summary, the four main goals for this demonstration are as follows:

- Unifying the interfaces to Industrial Bus Signal Protocols.
- Providing interoperability with various Access Networks (Ethernet, LTE, WLAN, etc.).
- Reducing / compressing the amount of Industrial Bus Data sent over the network.
- Increasing the QoS, Speed and Integrity of the Network.

The enforcement of security policies during runtime is based on the specification of a set of security rules for identifying and tackling vulnerabilities and threats over the deployed application graph, shown in Fig. A19.

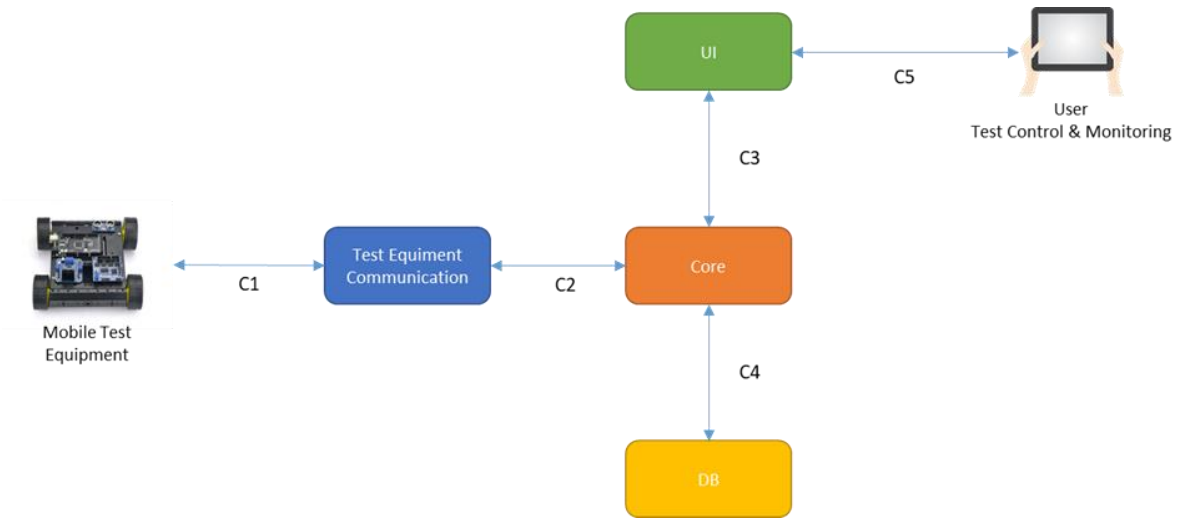

Figure A19. MATILDA - Testing 4.0 – Distributed System Testing application components.

- C1 – Bidirectional communication interface between Mobile Test Equipment and Test Equipment Communication module
- C2 – Bidirectional communication interface between Test Equipment Communication module and onboarded Core component instance
- C3 – Bidirectional communication interface between onboarded Core component instance and UI component instance
- C4 – Bidirectional communication interface between onboarded Core component instance and DB component instance
- C5 – Bidirectional communication interface between onboarded UI component instance and User Test Control & Monitoring device
- Database Component
    - Stores user access information
    - Stores test bench information such as availability and state
    - Stores test data
    - Interfaces:
        - Port (TCP) listening to serve connection requests from core components
        - Interface exposed to store data and retrieve data on request
- Core Component
    - Orchestrates information exchange between UI Component, Communication Component and DB Component
    - Forwards data from Communication Component to DB or UI component depending on user and test execution configuration
    - Forwards data from DB Component to UI component based on user requests

    - Monitors the QoS requirements for communication related to test execution
    - Interfaces:
        - Client interface (TCP) to DB Component, realized with Net Receiver (NR) subcomponent
        - Server interface to Communication and UI components, realized with Net Sender (NS) subcomponent
- UI Component
    - Application component executed on user devices for Configuration, Control and Monitoring
    - User interface provides the following:
        - Access control interface to the system
        - Status of test equipment
        - Database interface
        - Test execution report archives
        - Real-time statistics of test execution
    - Interfaces
        - Client interface (TCP) to Core component
- Test Equipment Communication Component
    - Application component loaded on communication device interfaced with test equipment
    - Advertises the availability and status of test equipment
    - Encodes, encrypts and transmits data from test equipment to UI component and DB component
    - Interfaces
        - Client (C2) interface to core component
        - Server (C1) interface to test equipment

The finalized Testing 4.0 use case has been remotely performed with the application components deployed on the CNIT testbed and edge components connecting from ExxpertSystems' premises. Certain additional steps were performed for on-boarding deployment:

- The FastWAN architecture relies on multicast communication between edge devices and instantiation application components. MATILDA is able to support multicast within the cloud; OpenVPN has been utilized as an additional component to facilitate the connection of edge devices to the MATILDA framework.
- OpenVPN was configured with client-to-client connection functionalities, added to application nodes on the cloud, along with configuration of OpenVPN's UDP interface.

- The Database and the UI application component were combined to a single node as the UI makes requests to an nGinx server on the Database node directly.
- For intrusion detection and prevention, a Snort-based (https://www.snort.org/) intrusion detection and prevention system (IDS/IPS) has been used. The use case of network slice isolation and Snort rule application has been evaluated with the OpenVPN server node component as primary target.
- A "rogue" application component was introduced into the graph to prove the intrusion detection services. The component is configured to sniff FastWAN application traffic and inject rogue packets masquerading as normal packets to bypass system security and compress design.

The App instantiation is shown in Fig. A20.

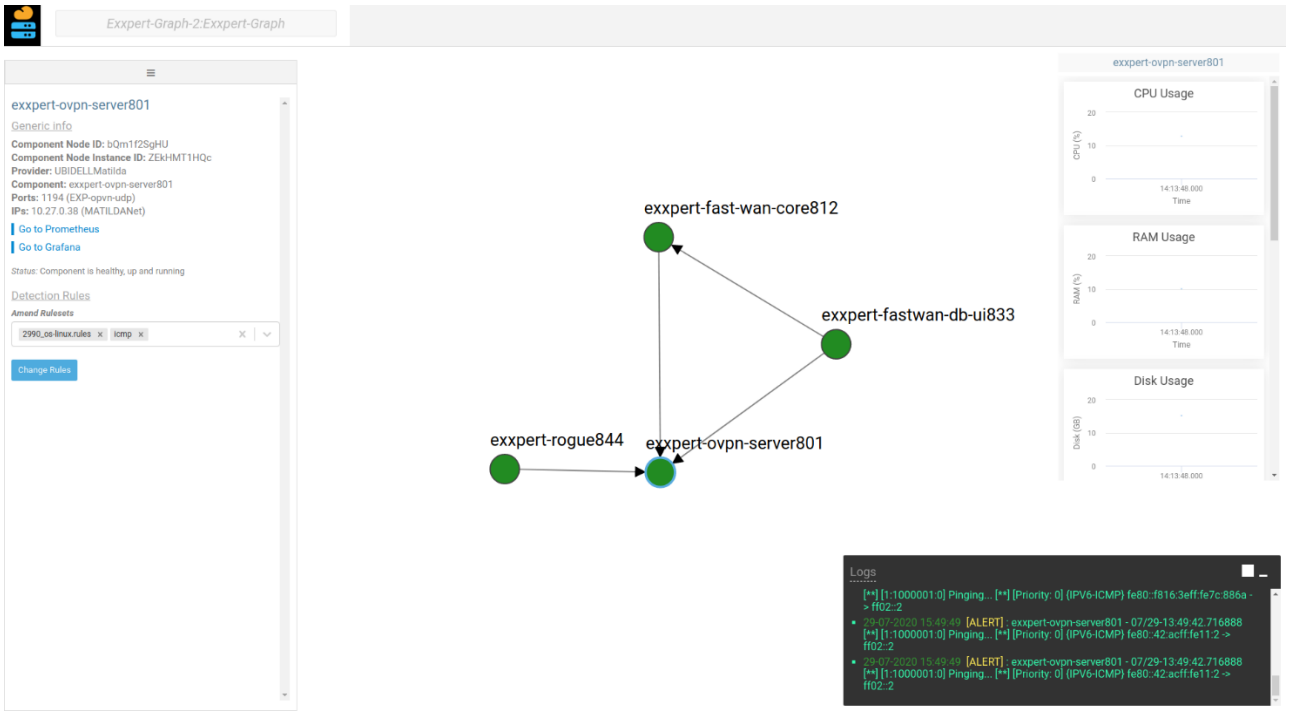

Figure A20. Testing 4.0 app instantiation.

The setup of the demonstration, shown in Fig. A21, consists of the following edge devices communicating with the instantiated application in the MATILDA network:

1. Signal Generator Customer A.
2. Test Equipment Communication Customer A.
3. Database Customer A.
4. Tablet for Customer A edge-device data access and visualisation.
5. Signal Generator Customer B.
6. Test Equipment Communication Customer B.
7. Tablet for Customer B edge-device data access and visualisation.

The demonstration has been broken into three parts: namely, *i*) network slice isolation was demonstrated by subnet intersections and multicast membership, *ii*) Snort rules application was demonstrated to prove the functionality of IDS, and *iii*) blacklisting IP addresses is proved, i.e., the IPS, intrusion prevention system part, which is made available on the MATILDA platform's back end via BPF's, Berkeley Packet Filters, applied in real-time in the background.

Details can be found in reference [10].

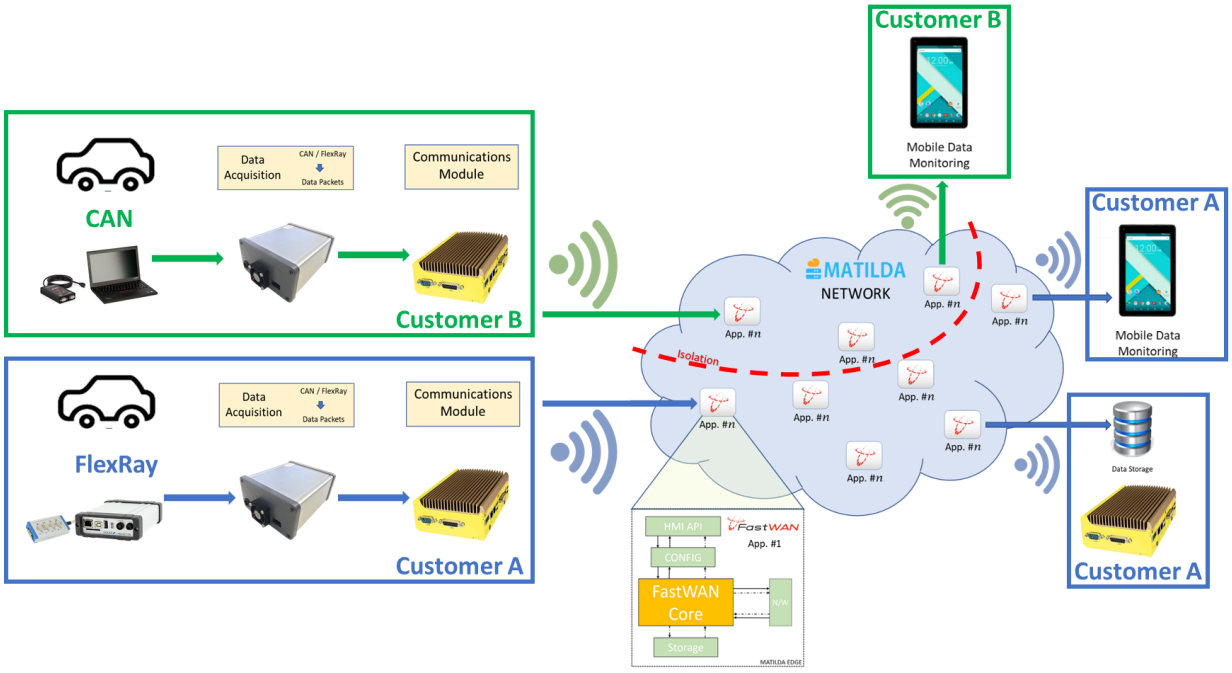

Figure A20. Representation of the use case demonstration.

Network and operational KPIs are reported in Table A6.

Table A6. Network and operational KPIs.

KPI	Description	Measured (Where/ How)	Acceptance Criteria/ Threshold
Network KPIs			
Flexible Bandwidth Allocation at various interfaces	Flexible bandwidth allocation is needed between geographically distributed systems / sub-systems under test to ensure the integrity and required performance of distributed functional and integration testing. Bandwidth allocation based on the demand by the end clients by the means of slice re-negotiations by application graph components.	C2, C3, C4, C5 Measured by the components of application graph. Additionally, for standardized results, iperf/uperf tools are used to test the communication bridges between components. The configuration for tests is defined using XML configuration files.	10 Mbit/s (Mbps) between interfaces

KPI	Description	Measured (Where/ How)	Acceptance Criteria/ Threshold
Network KPIs			
Low Delay/Latency	Low Delay is required between geographically distributed systems / sub-systems under test to ensure the integrity and required performance of functional and integration testing.	C2, C3, C4, C5 Measured by the components of application graph. Measured by MATILDA framework and statistics obtained from PROMETHEUS tool.	Inside Germany - Approximately 50 ms Latency Inside Europe - Approximately 100 ms Latency Worldwide - Approximately 200 ms Latency
Interoperability with Various Access Networks (WLAN, LTE) **Only LTE and possibly WLAN is used by the devices in use case**	The infrastructure/services for deploying FastWAN Test Systems shall be supported seamlessly over various Access Networks.	WLAN LTE	Operation and seamless handover of the communication session across the networks
Density of Connections (Not relevant)	Number of connections that can be simultaneously made by edge clients without drop in performance.	C2, C5 Metric measured on the MATILDA infrastructure and on edge clients.	10s to 100s of test equipment, edge databases and HMI devices
Jitter	Variation in the latency of packets on communication channels. This latency shall be minimal especially when doing real time monitoring.	C2, C3, C5 Measured by the edge clients, application components as well as on the MATILDA infrastructure.	Less than 1ms
Mobility	Units under test and remote test monitoring units should maintain established communication session in mobility.	C2, C5 (cannot be tested in the Testbed infrastructure).	Up to 130 km/h
Memory allocation for components of application graph	Mechanisms for demand-based memory allocation as well as scaling in established sessions should be supported.	UI, CORE and DB Components. Measured in platform on application instantiation as well as monitoring in session using Prometheus.	Up to 8GB

KPI	Description	Measured (Where/ How)	Acceptance Criteria/ Threshold
Operational KPIs			
High Availability	The test systems interconnection infrastructure/services shall be always available.	Whole graph. Uptime and Prometheus metrics.	99.99% of operational time
Resource Usage Monitoring	To allow preparation of dynamic scaling the monitoring of the current resource usage is needed.	Whole graph. Prometheus metrics for CPU, Memory, and Disk Usage.	Must be available
Component scalability	Dynamic scaling is needed to fulfil actual user communication requests.	Whole graph. Prometheus metrics.	Must be available
Time to scale	Time required for launching additional instances of application graph components on demand.	Core, UI.	~ 1 minute
Deployment time	Time needed for first installation and onboarding.	All components. MATILDA orchestration and onboarding tool.	~90 minutes
Single component deployment time	Time needed for deployment of individual components of application graph.	All components. MATILDA orchestration and onboarding tool.	~30 minutes
Network Service Deployment time	Time need for end-to-end service deployment.	All components. Observed in docker-compose logs.	~90 minutes
Onboarding time	Time needed for update and onboarding during development.	Single component. Observed on MATILDA orchestrator.	~15 minutes
Onboarding time	Time needed for update and onboarding during development.	Whole graph. Observed on MATILDA orchestrator.	~30 min
Locality Awareness	Locality awareness needed for optimized scalability and routing of data communications over distributed application graph component instances communication.	Whole graph.	Expected to be provided by service host Must be available
Multi-site management	Functionality at its core is implemented as a distributed application. The edge application components as well as core components require network monitoring services and management provided by infrastructure to dynamically adapt to demands.	Whole graph.	Must be available

KPI	Description	Measured (Where/ How)	Acceptance Criteria/ Threshold
Operational KPIs			
Recovery on session loss	Test context and session are managed by the components of the application. In case of communication losses, the session should be re-established, if demanded by the client, based on the new slice information. Data shall be cached local to the instance until certain limit and time for test session recovery.	Whole graph. Measured at two points. Firstly, the recovery time in the orchestrator, in case a component fails (purposeful suspension of component). Secondly, the recovery time using FastWAN service listener.	Capability to associate between application instance session and slice must be available
Comparison between standalone FastWAN vs FastWAN as microservice on 5G.	Bandwidth allocation, latency and jitter are compared between the non-5G deployment and 5G deployment of FastWAN.	Whole graph. Standalone FastWAN statistics obtained from FastWAN internal statistics measurement tools. On MATILDA framework using Prometheus tools.	Above mentioned thresholds for bandwidth allocation, latency and jitter
Security – Slice Isolation	Multiple instances of application graph should have slice isolation mechanisms to provide data security.	Whole graph. Traffic assessment using two application instances, two instances of OpenVPN, two DynDNS assignments and two port mappings into containers (one configured on 1194 another on 4911).	Must be available - access to two slices which contain isolated streams
Security – Dynamic Snort Rule Management	Protection against access to services by unauthorized edge devices.	MATILDA Infrastructure. A rogue producer spoofing the MAC address sending aberrant data. Verifying the capability to identify aberrant behaviour and update Snort rules for application graph VMs. Comparative observation of traffic statistics in Prometheus when applying and removing security policies to instantiated application graph.	Must be available – aberrant data not present in: 1 GUI 2 DB 3 Processing Nodes

Appendix B: MATILDA Achievements per WP

B.1 WP1: MATILDA Reference Architecture, Conceptualization and Use Cases

The purpose of WP1 was to analyse the requirements of cloud native applications in the context of modern reprogrammable 5G networks. The ultimate goals were the **formulation of the high-level architecture**, the **definition of roles**, the **definition of formal normative meta-models** for the component's **signalling**, along with the definition of **functional and non-functional** requirements; accompanied by respective Key Performance Indicators (KPIs). The workpackage produced 6 deliverables from M1 to M15.

Deliverable D1.1-MATILDA Framework and Reference Architecture [11] (delivered on M6) provided a thorough requirements analysis and an overall architecture describing the main components and artefacts of MATILDA. The definition process was iterative in the sense that several component's layers have been identified, such as application, orchestration, network configuration and core-infrastructure (see Figure B1).

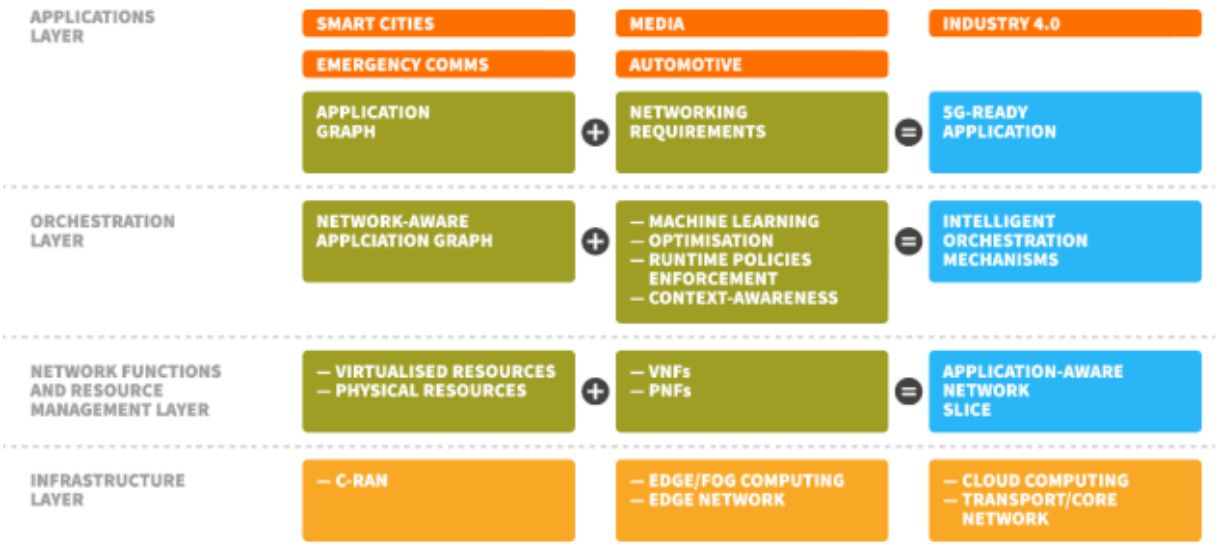

Figure B1: Architectural layers & functional building blocks per use-case.

The five families of use-cases, i.e. smart cities, emergency communications, industry 4.0, automotive and high immersive media communications have been mapped to the different layers along the functional "building blocks" that they, mostly, rely on. Upon this exercise, initial building blocks of the overall architecture have been sketched. The difficulties at this point of time were two:

i) The signalling of the components had to rely on a **completely new paradigm** of reconfigurable infrastructure according to which a traditional communication service provider acts simultaneously as a networking and cloud provider on both the backhaul and the edge part;

ii) The paradigm could not fit 100% a standardization artefact, since the 5G componentization is (as reviewers also pointed out during the 1st Review) **a moving target problem**.

Taking under consideration the above problems and the functional requirements of the use-cases, the MATILDA high-level architecture has been formulated (see Figure B2). During the formulation of the architecture special emphasis was provided in the definition of the different **administrative domains** that each component belongs to.

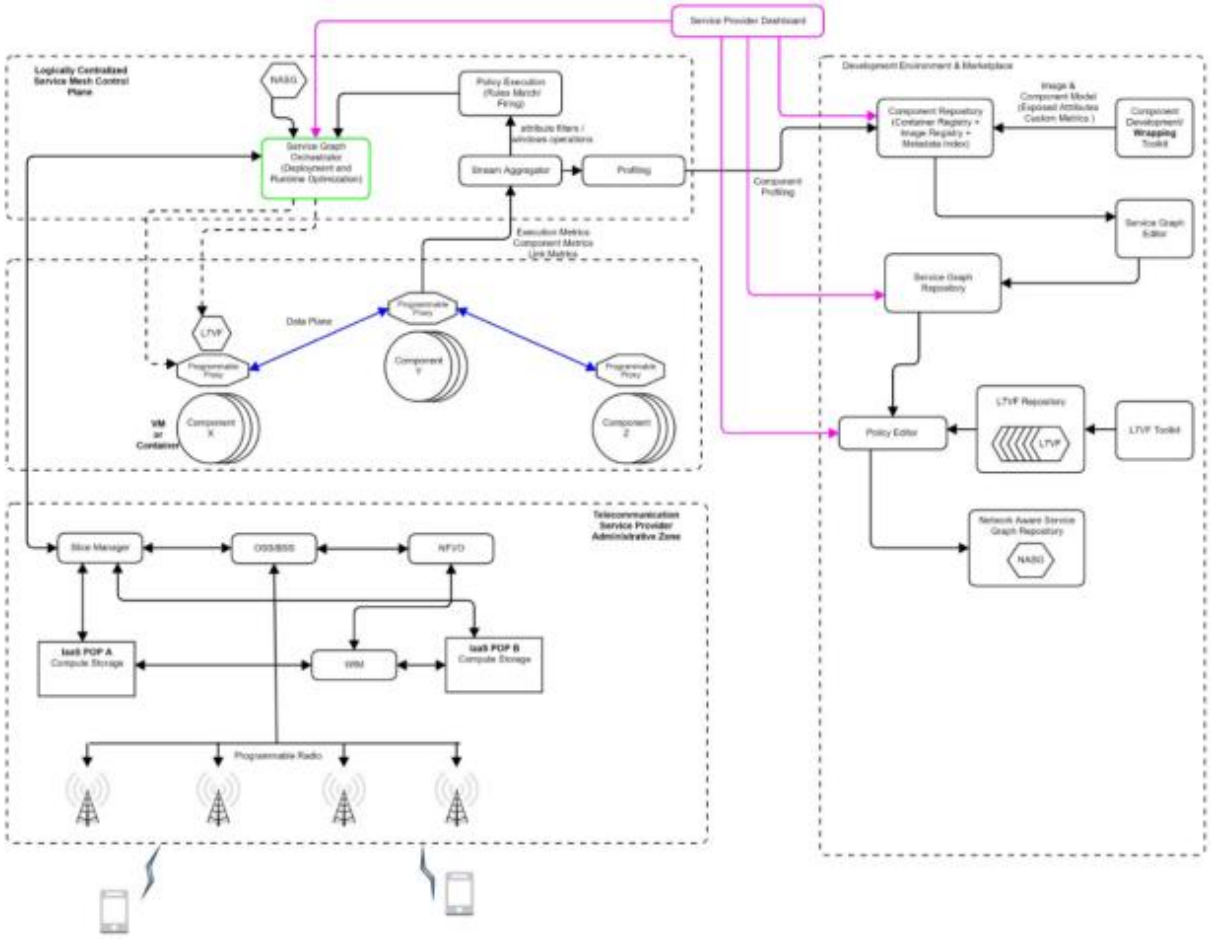

Figure B2: MATILDA Architecture.

More specifically, all components have been grouped in **two layers**; namely:

- The **Vertical Application Orchestration** layer (VAO) running under the **administrative domain of the vertical-application service provider** and responsible for managing the physical onboarding of the application and defining the runtime policies.
- The **OSS layer** running under the **administrative domain of the telecommunication service provider** that is responsible for a) **interpreting** the requested policy to a proper slice definition and b) **materializing** this slice definition to an actual slice that is usable by the VAO.

This interplay between the VAO and the OSS has been formalized in the frame of a "Slice-Negotiation" pattern. It should be mentioned that this formal model has been proposed for adoption on several projects running simultaneously to MATILDA (e.g., SliceNet).

Deliverable D1.2-Chainable Application Component & 5G-ready Application Graph Metamodel [12] (delivered on M9) aimed to conceptualize the metamodel of any application that can be deployed to a MATILDA-enabled infrastructure. Since applications that can be supported are cloud-native components that can be chained to each other, the model is formally addressed as **chainable application component metamodel**. This model was defined in a normative format using XSD language (see Figure B3).

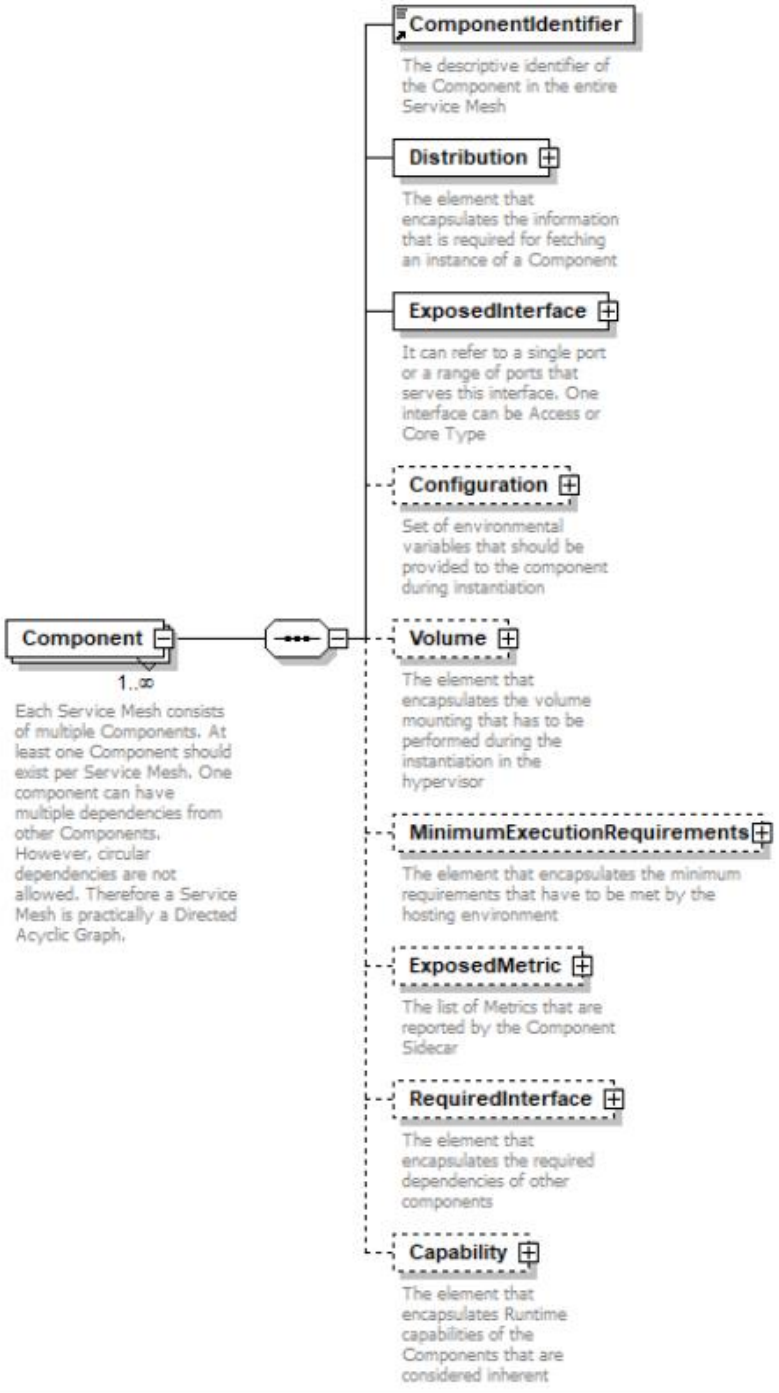

Figure B3: MATILDA component element.

The model covered aspects of component's **metadata such as operational requirements, exposed metrics, mapped volumes, chaining information**, etc. It should be noted that during the formulation of these models, the pursuit of a common acceptable format of cloud-native application graphs was still open. During 2020, it could be argued that the dominance of Kubernetes imposed a **graph-metamodel that is de-facto dominant**. It should be noted that the **MATILDA model is totally forward-compatible/mappable to this model.**

Deliverable D1.3-VNF/PNF & VNF Forwarding Graph Metamodel [13] (delivered on M9) provided a conceptualization of the VNF/PNF and the VNF-FG metamodel that support the OSS layer of the architecture. While the metamodel described in D1.2 lives in the "boundaries" of the VAO, the OSS requires its own internal operational models. The heart of the OSS is the OSM orchestrator that undertakes the task of deploying the network forwarding graphs that have been selected in the frame of a slice creation, as depicted in Figure B4.

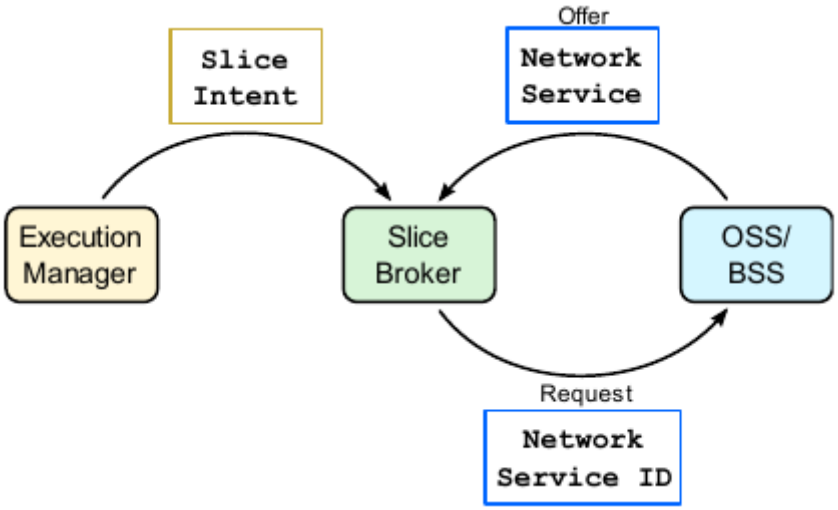

Figure B4: Transition from Slice Intent to the OSS.

A detailed view of the employed model has been provided, as depicted in Figure B5. The model covers several operational aspects of the VNFs/PNFs that are deployed in the frame of a slice instance.

The employed descriptors have been changed during the last part of the project, since in the first version of the MATILDA release OSM version 4 was used, while in the latest release an upgrade to version 6 was performed.

Deliverable D1.4-Network-aware Application Graph Metamodel [14] (delivered on M9) provided a conceptualization of the **MATILDA slicing model**. The previous two models corresponded to the VAO and to the OSS, respectively. The two 'worlds' are combined by a request/response pattern which is addressed as **Slice Negotiation**, as depicted in Figure B6. In a nutshell, the VAO is asking for a slice and the OSS is, at first, trying to check if this slice can be supported and, in a second stage, it materializes this slice.

Field	Type	Description
description	**string**	Text describing the VLD
id	**string**	Identifier for the VLD
internal-connection-point	**list[1...N]**	List of internal connection points in this VLD See: vnfd-base:internal-connection-point
leaf-bandwidth	**uint64**	For ELAN this is the bandwidth of branches.
name	**string**	Name of the internal VLD
provider-network		Container for the provider network. See: manotypes:provider-network
root-bandwidth	**uint64**	For ELAN this is the aggregate bandwidth.
short-name	**string**	Short name to appear as label in the UI
type	**virtual-link-type**	Type of virtual link. Possible values: • ELAN: A multipoint service connecting a set of VNFs • ELINE: For a simple point to point connection between a VNF and the existing network.

Figure B5: Part of the network forwarding graph model (VLD: Virtual Link Descriptor).

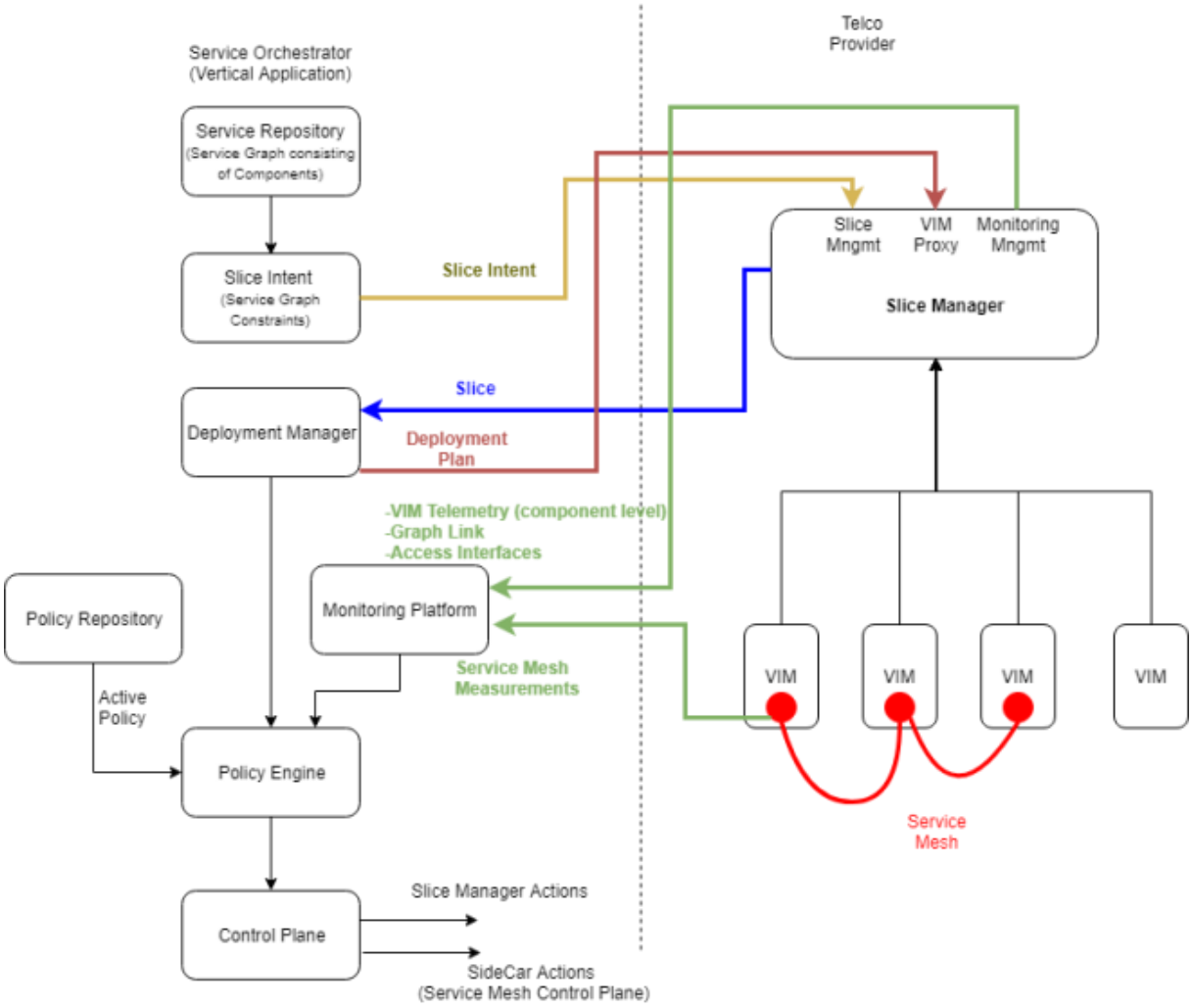

Figure B6: Slice Request/Response pattern.

This "two-phase" protocol was formally designed and created in XSD language as depicted in Figure B7. The language contains several descriptors that conceptualize the constraints at the component level and at the link level. Issues like initial quotation of DC/edge resources, acceptable link characteristics (jitter, packet drop), required QoS Class Identifiers, are expressed in a formal notation.

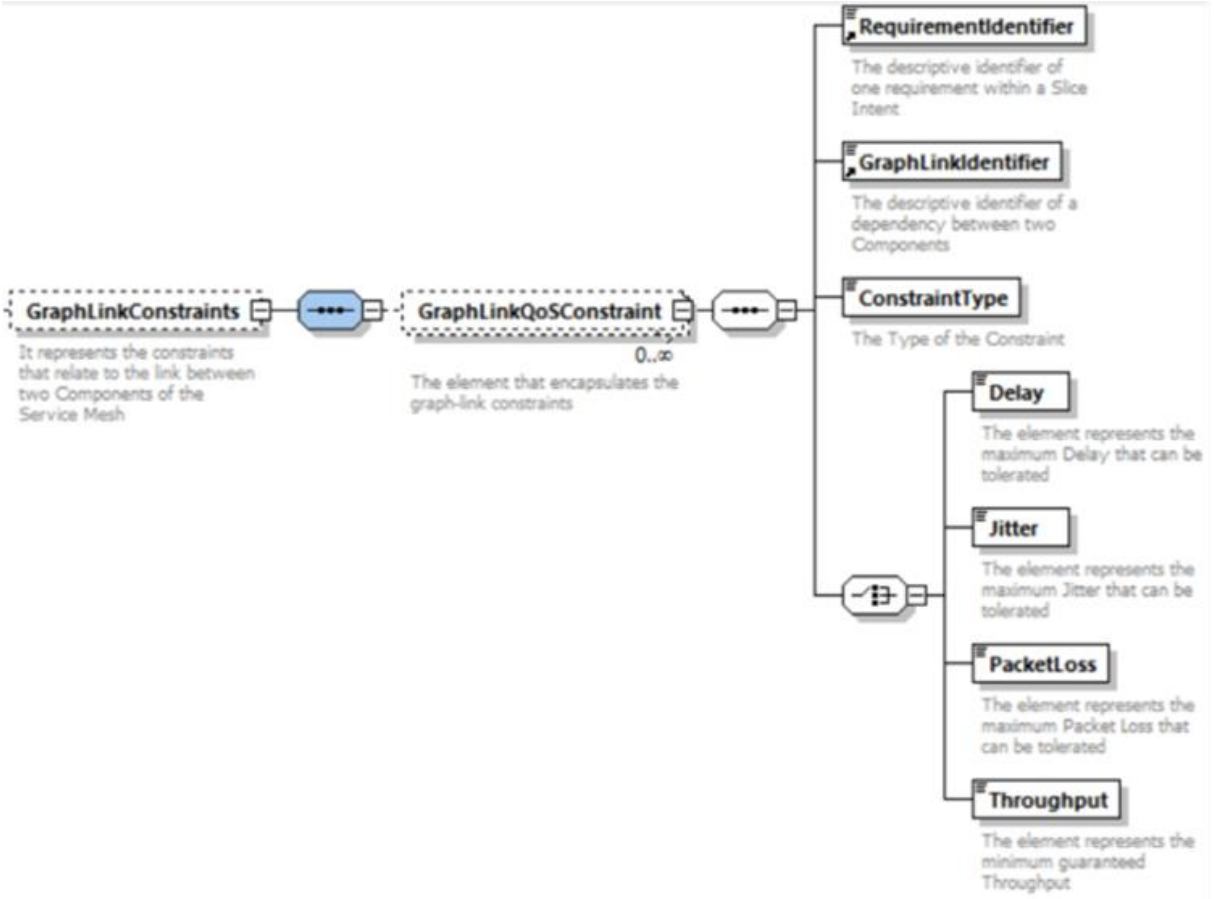

Figure B7 Part of the Slice Request model.

The successful or not-successful materialization of the slice was also modelled, since the **formal-**response of OSS is interpreted by the VAO in order either to perform the actual deployment or to handle the slice-creation-failure in a graceful manner. It should be mentioned that a successful response contains all details that are required by the VAO in order to perform interaction with the created VIMs. These include location-identifiers, credentials, etc.

Deliverable D1.5-Deployment and Runtime Policy Metamodel [15] (delivered on M9) provided a formal model of the design-time and runtime policies that accompany each deployment. After the slice negotiation finishes, the VAO is responsible to deploy the entire service graph on top of the MATILDA-enabled telco provider, i.e. the provider that operates the MATILDA OSS. Upon finalization of the deployment the service graph **is not static**, in the sense that many aspects of the deployment **may alter during runtime**. Such aspects include the amount of allocated quotas, the amount of instances per application (scaling), the alternation of the QoS parameters, etc. Hence, a crucial question is: **how are these 'alternations' expressed** and **which component is performing them**?

The answer to this question derives from the Runtime Policy Metamodel. Such a metamodel describes the full set of conditions that can be used by triggering rules (see Figure B8).

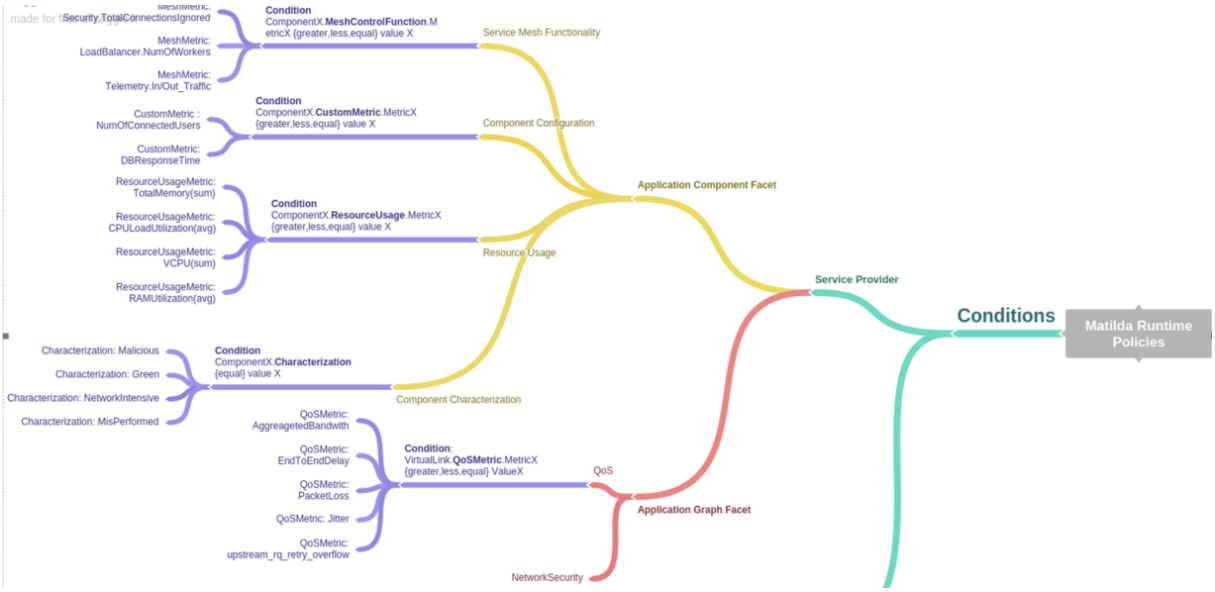

Figure B8: Modelling of conditions.

It should be clarified that the rules are fed by monitoring streams that are propagated by the two instrumentation engines – the one that lives under the administrative domain of the VAO and the other that operates under the domain of the OSS. Technically, these two engines could be one; yet there are political reasons (i.e., privacy issues) that prevent the unification of these engines. Conclusively, it is inferred that there are two layers of enactment regarding the dictated actions (e.g., scale up/down, extend allocations): a) the VAO and b) the OSS. The respective mechanisms that materialize these actions have been provided in WP3 and WP4, respectively.

Deliverable D1.6-Supported Verticals, Use Cases and Acceptance Criteria [16] (delivered on M15) documented the vertical use cases implementation scenarios. As already mentioned, the artefacts of MATILDA were battle-tested in five thematic areas; namely in High Resolution Media Communications, Industry 4.0, Smart Cites, Public Protection and Disaster Recovery and Distributed System Testing. The actual goal of this deliverable in the first phase of the project was to **verify** that **all modelling** artefacts that were formulated within WP1 were **meaningful and expressive enough** in order to capture the specificities of each of the thematic areas. Each use-case created **indicative instances** of **all developed models** such as a service graph, a slice intent and a runtime policy (see indicatively Figure B9).

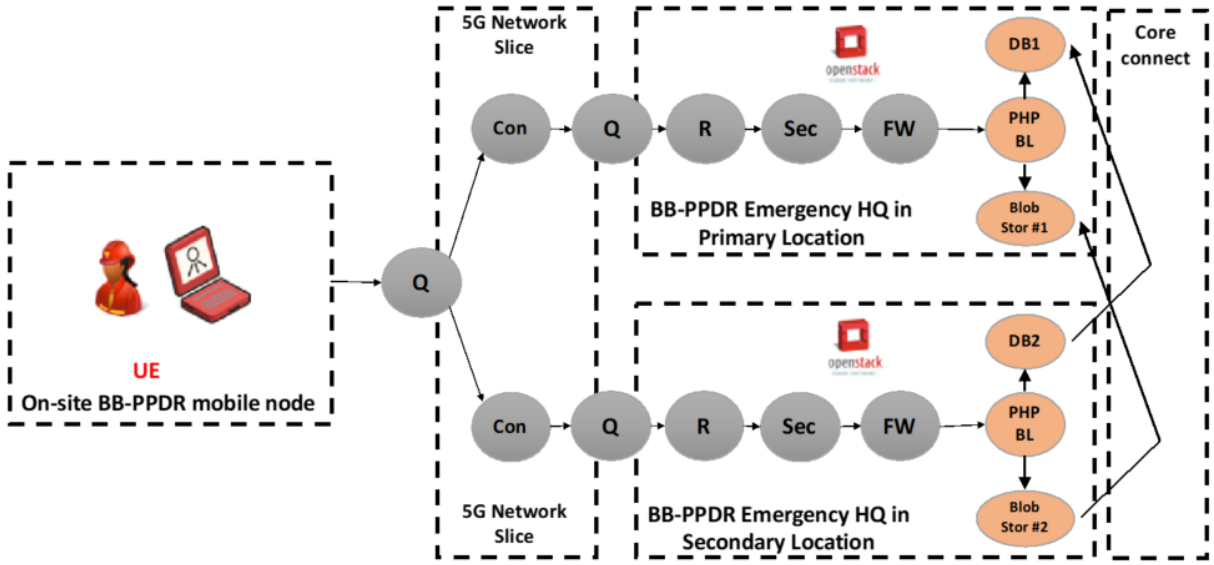

Figure B9: Indicative Visual representation of the PPDR Service Graph.

This exercise turned out to be extremely helpful, since it was conducted prior to the implementation phase of the VAO and the OSS.

B.2 WP2: 5G-Ready Applications and Network Services Development Environment and Marketplace

The main objective of WP2 was the design and development of service graphs that aim to be deployed on top of reprogrammable infrastructure. As already mentioned, the general purpose of all meta-models was defined in the frame of WP1-Deliverable D1.2. This modelling framework captures all aspects regarding **chain-ability, component's hosting requirements, scaling capabilities, minimum quotas, exposed metrics, link-characteristics, and radio requirements.**

Although the **developed models are capable enough** to capture all semantics of the developed applications and their policies, the **creation of model-instances** that comply with these metamodels **is a difficult and error-prone task**. That was the reason why a graphical environment was created that may assist the "adopter" and lower the entry-barrier cost.

D2.1/D2.2-5G-Ready Applications and Network Services Development [17][18] (first version delivered on M15 and final version on M24) documented the **implemented environment** that assists end-users embrace the MATILDA paradigm. It could be argued that the environment is logically separated in four distinct sub-environments:

- One sub-environment that assists developers to onboard their cloud-native components,
- One sub-environment to construct valid service graphs,
- One sub-environment to create and manage valid Slice-Requests, and
- One sub-environment to create and manage runtime policies.

Cloud-native onboarding environment

The cloud-native-onboarding-environment (Figure B10) aims to ease the task of initial registration of individual components. As depicted, the environment guides the end-user in an intuitive way to describe the characteristics of the component such as the build-architecture, the repository that is published, the minimum quotas, the health-checking expressions, the required cpu-extensions, etc.

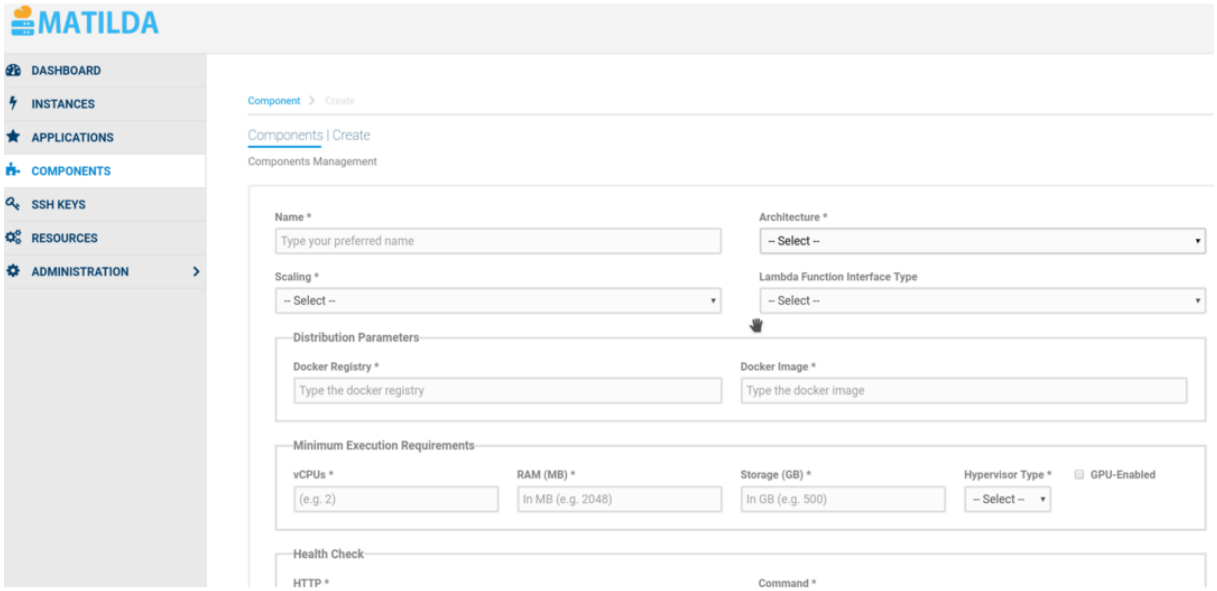

Figure B10: Component Registration.

It should be noted that the component is able to host any software entity that is built using the 12-factor paradigm[5].

Service Graph Creation environment

The specific environment is dedicated to assisting the creation of direct acyclic graphs (DAGs) which are practically the serialization format of the deployable service graphs. Heavy emphasis is given to some complex requirements such as:

- assist the identification of components that can be chained by-design with an existing component,
- validate that all upward and downward dependencies are satisfied in the frame of one DAG, and
- Assist in the enrichment of the DAG with dedicated components that are used for scaling purposes (a.k.a. balancing elements).

A valid service graph can be instantiated many times. Each time there might be different slice requirements. In order to assist this process, a dedicated environment was developed (Figure B11).

Slice Management environment

The specific environment allows the graphical definition of formal slice requests. In order to perform such requests, the service provider has to denote what are the desired requirements at the component level and at the link level (see Figure B12).

[5] https://12factor.net/

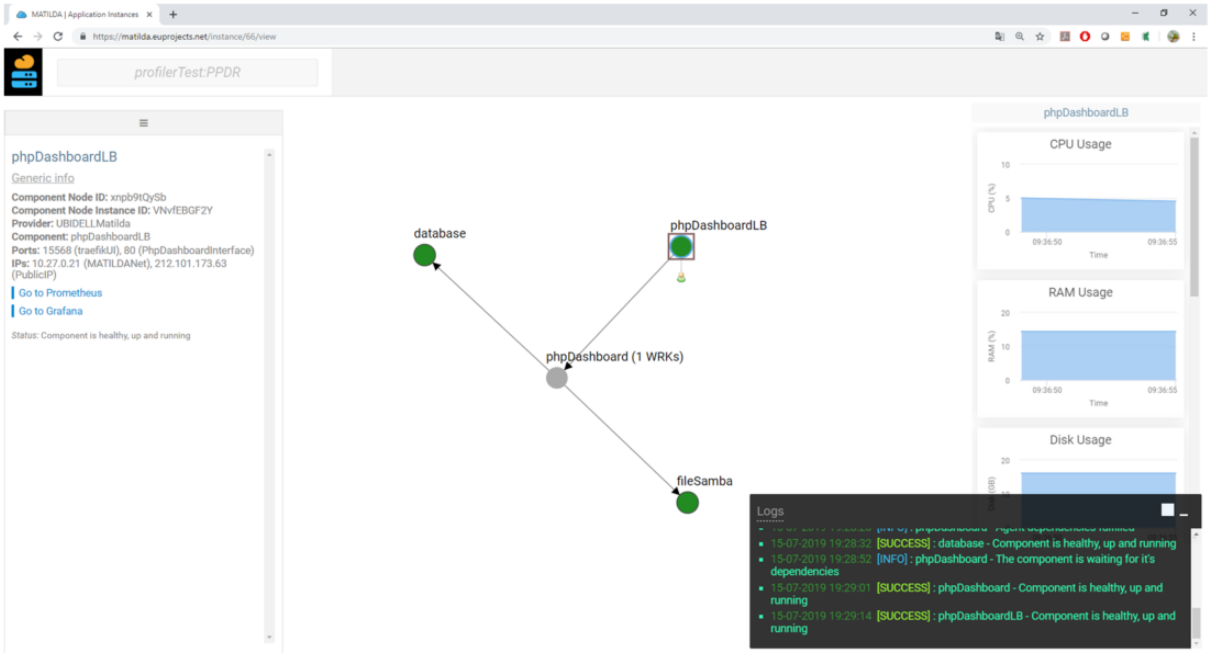

Figure B11: Service Graph creation.

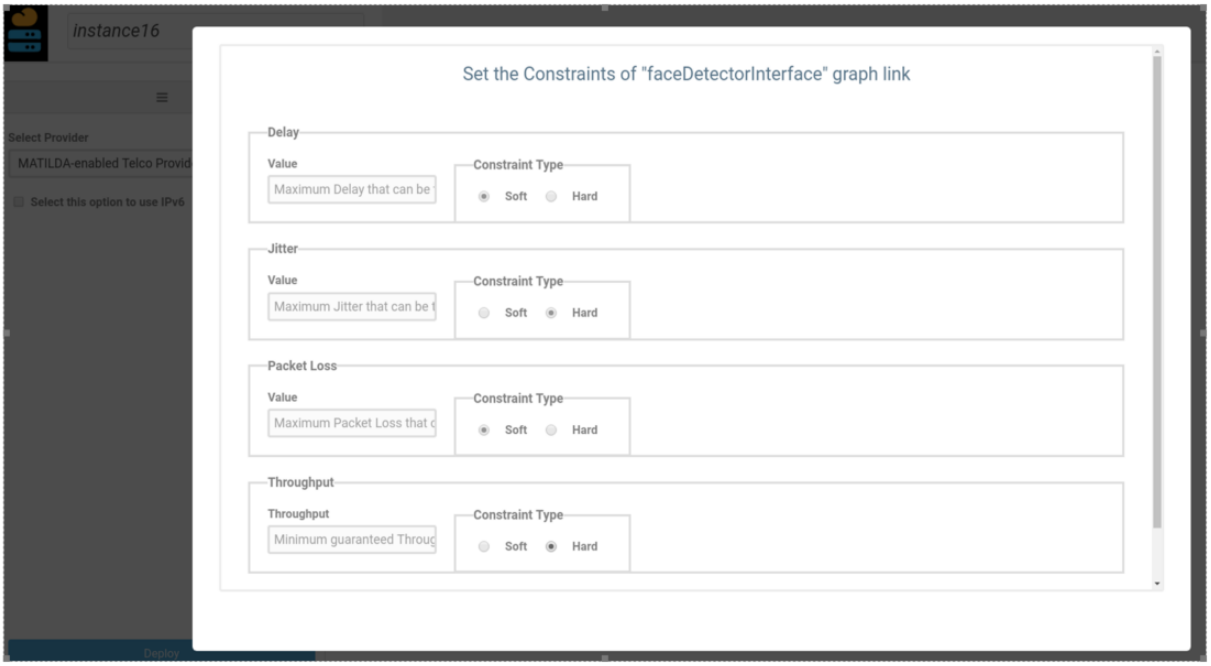

Figure B12: Slice Request Management.

As already mentioned, upon deployment a running instance is **not static**, i.e. several runtime aspects can be altered. The **attached policies** to the runtime graphs practically affect these alterations.

Runtime policy management

It could be argued that the definition of the runtime policies is the most error-prone process, since in order to define such a policy one has to combine several metrics that derive from the monitoring system **using aggregator operands** (i.e., sum/mean) and **logical operands** (i.e., **and/or**). It is **practically impossible** for a plain user to author a valid expression taking under consideration that s/he has to select **valid metric-identifiers**, **valid component-identifiers, acceptable thresholds per metric- identifiers**, etc.

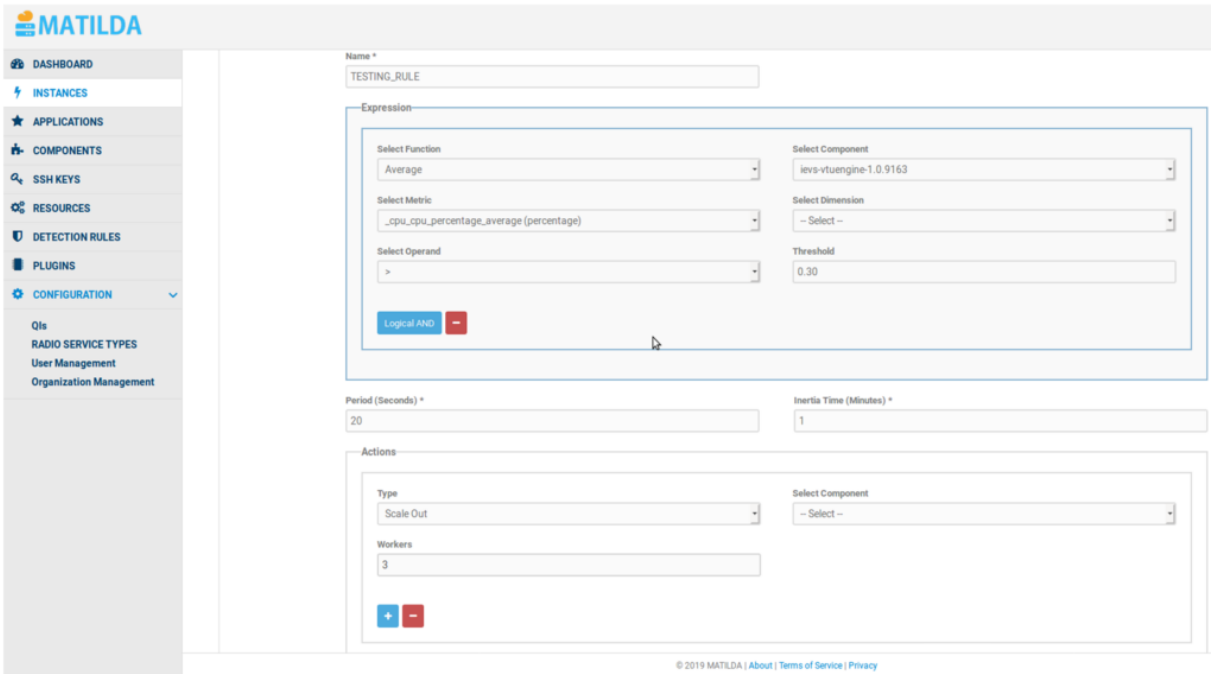

Figure B13: Policy Management.

As is depicted in Figure B13, the task of creating policies is totally guided, since a user has to select a deployed instance and then, per component, s/he can combine graphically several expressions in complex expressions. The same environment can be used for defining triggering rules that may be executed both under the VAO administrative domain or the OSS administrative domain.

B.3 WP3: Intelligent Orchestration Mechanisms

The main objectives of WP3 (Intelligent Orchestration Mechanisms) included the design and development of the VAO. The VAO includes a set of intelligent orchestration mechanisms covering the deployment and runtime management of 5G-ready applications. The main components constituting the MATILDA VAO are: (i) the deployment and execution manager that supports the production of optimal deployment plans, as well as managing the overall execution of the application, (ii) a set of data monitoring mechanisms which collect feeds from network and application-level metrics, (iii) a data fusion, real-time profiling and analytics toolkit that produces advanced insights through machine learning mechanisms and provides real-time profiling of the deployed components, application graphs and VNFs, (iv) the MATILDA Agent and a set of service discovery mechanisms for supporting registration and consumption of application-oriented services following a service mesh approach, (v) a context awareness engine, in order to provide inference over the acquired data and support runtime policies enforcement, and (vi) mechanisms supporting interaction among the VAO and the OSS. The following deliverables have been released during the project lifetime that contain the reporting of the main work done in WP3.

D3.1/D3.2-Intelligent Orchestration Mechanisms [19][20] (first version delivered on M15 and final version on M24) provide **implementation details of each of the VAO components**. Following, we provide short details per component. The operational goal of the **Deployment and Execution Manager** is to facilitate the initial deployment, the execution and the deprovision of the vertical application components. The main components of the "Deployment and Execution Manager" are the following:

- **Slice Interpreter**: The interpreter is responsible to parse the response of the telco provider and infer the proper VIMs that should be used in order for deployment to take place.
- **VIM client**: This component enables the execution of the VIM-related commands that are provided to the MATILDA-enhanced OSS of the telco provider.
- **MATILDA Agent**: The agent is responsible to monitor the boot sequence of the vertical application. Moreover, it is responsible to activate the monitoring probe of the VM.
- **Service Discovery Server (SDS)**: The server keeps track of the operational state of all vertical applications and their components. It acts as a cache memory; thus, keeping track of the state and the configuration of each VM.
- **Execution Manager Control loop**: Each orchestration engine spawns one 'infinite' control loop per deployed application. This control loop is responsible for tracing the execution of all components and react on possible errors.
- **Elasticity Controller**: It is responsible to handle the scale-in/out operations as analysed above.

The Deployment and Execution Manager is built using the Spring Framework.

The **MATILDA Agent**'s main duty is to handle the signalling (at layer 7) between the orchestrator and the core platform. When a vertical application is deployed to a MATILDA enabled provider each component is associated with a VM and each VM is spawned

simultaneously (see also Figure B14). Upon VM spawning there are 7 discrete steps that have to be executed in order for the vertical component to be operational. Moreover, the MATILDA Agent contains functionality to support efficient security policy enforcement at the component level.

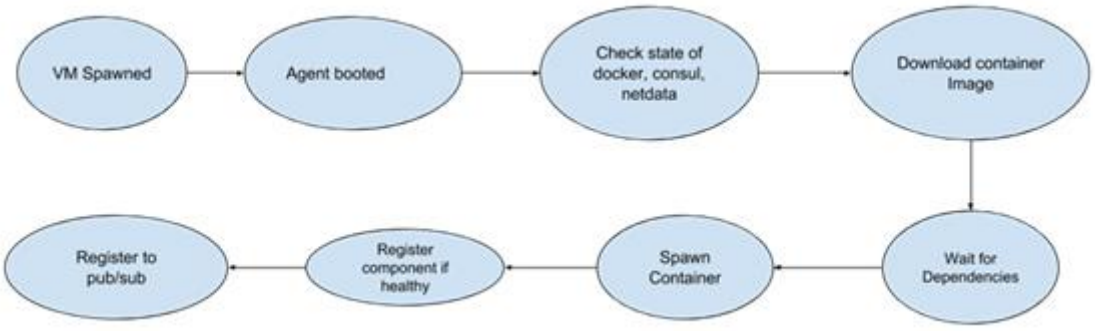

Figure B14: Lifecycle of MATILDA Agent

The **Service Discovery Server** is a rather crucial component of the architecture, since it acts as a Key-Value store which is accessible by all Agents that are booted. In this store, all aspects regarding Agent arguments, vertical component dependencies and docker image location are provided. The technology base of the SDS is Consul. Consul is a distributed service mesh to connect, secure, and configure services across any runtime platform and public or private cloud. The selected SDS provides a full featured control plane with service discovery, configuration, and segmentation functionality. Each of these features can be used individually as needed, or they can be used together to build a full-service mesh.

The MATILDA **monitoring solution** addresses multi-site network infrastructure deployments and performs metrics acquisition from a variety of domains. It is responsible for the management of the metrics captured from the various infrastructure components, the management of alerts and events based on these metrics, and the visualization of the available data. The monitoring mechanisms can operate in passive or active manner. Passive monitoring in MATILDA refers to the capture of service and network metrics locally at the application or VNF component level. Examples of such metrics are CPU utilization, RAM usage, etc. On the other hand, the active monitoring provides QoS/QoE measurements based on the injected traffic by the monitoring application itself. The monitoring solution is implemented based on Prometheus monitoring engine, while Netdata and qMon agents are used for collecting resource usage and QoS data.

A set of **profiling mechanisms** are designed and partially implemented, focusing on supporting various profiling aspects at application component and application graph level. Such mechanisms may be applied in benchmarking scenarios, as well as over data collected in an operational environment. To realise profiling, an analytics toolkit has been developed that supports:

- The ease of integration of analysis processes/scripts by data scientists independently of the programming language used;
- The ease of selection of monitoring metrics (resource usage, orchestration, application component specific metrics) and the fetching of the required time-series data from the Monitoring Engine in order to realise analysis over them;

- The production of analysis results in the form of URLs that can be easily viewed and compared by the interested parties (e.g., data scientists, network administrators);
- The design and implementation of a set of APIs for supporting the registration and execution of analysis processes.

The overall architectural approach of the analytics toolkit is depicted in Figure B15. The baseline technologies used for the development of the data fusion, real-time profiling and analytics toolkit include, among others, the OpenCPU framework for scientific computing, the Flask framework, R statistics package and Prometheus.

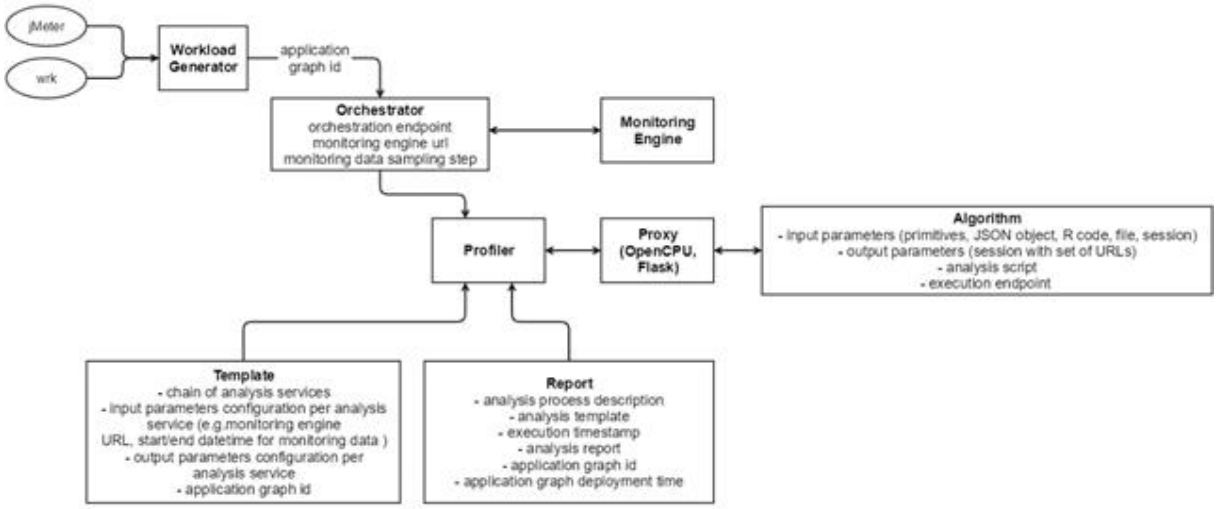

Figure B15: Analytics toolkit architectural approach.

The **Policies Manager** provides policies' enforcement over the deployed application graphs following a continuous match-resolve-act approach. The Policies Manager (following a Drools approach) consists of (i) the working memory (WM); facts based on the provided data, (ii) the production memory (PM); set of defined rules, and (iii) an inference engine (IE) that supports reasoning and conflict resolution over the provided set of facts and rules, as well as triggering of the appropriate actions. Data is fed to the WM through the monitoring mechanism that is in charge of collecting data based on a set of active monitoring probes. The Policies Manager is also fed by policies associated with the deployed application graphs, as provided through the Policies Editor – the editor made available to service providers for policies definition. The policies enforcement mechanisms include also a set of tools for improving performance and scalability aspects. The high-level interaction between the Policy Manager and the Monitoring mechanisms is depicted in Figure B16.

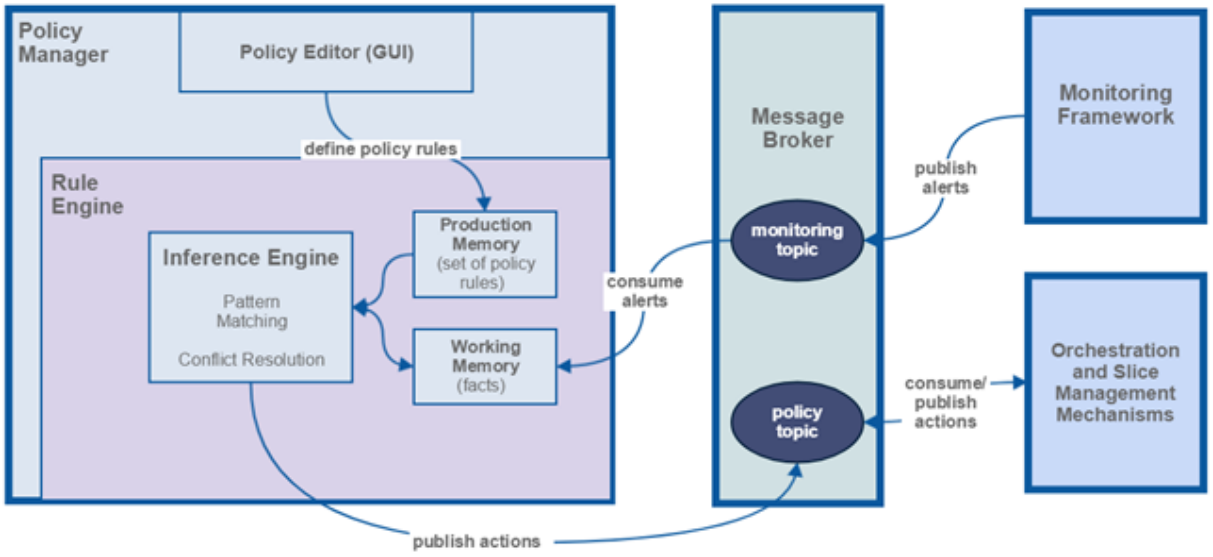

Figure B16: Policies Manager and Monitoring Mechanisms Interaction.

The **Complex Event Processing** mechanism of the MATILDA Intelligent Orchestrator suggests a dynamic engine that, enriched with Machine Learning techniques, can be fully adaptive to its environment. The engine is divided in two major components; namely: the Complex Event Processing engine and the Threshold identifier. The Complex Event Processing (CEP) engine regards a Drools Fusion engine, whose job is to trigger the rules from the Knowledge Base when a condition is met. These rules are created in the first place by a Domain expert or a Service provider. The Complex Event Processing engine, as is, regards a typical engine without any level of intelligence. For this reason, the Threshold identifier service is developed, whose purpose is to identify in real-time the behaviour of the deployed services and update the Knowledge Base (i.e., the rules), in order for the CEP engine to act accordingly.

Finally, a set of APIs have been developed for supporting the interaction among the VAO and the OSS. These APIs support functionalities for:

- Slice Intent Request: through this API, we support the request from the VAO towards the OSS for the creation of a slice based on the slice intent description. Removal of an existing request is also supported.
- Slice Instance Response and Profiler Status: through this API, the OSS provides a response to the VAO with information about the instantiated slice (following a slice intent request), as well as the examination of the status of the profiling tool.
- Declaration of QoS and Locality Constraints (Application Instance Management): through this API, a set of QoS and locality requirements for the under-deployment application are provided.

As far as the APIs' development is concerned, Spring Boot has been used. Spring Boot made it easy to develop the APIs, as it provides a range of non-functional features such as embedded servers, security, externalized configuration, etc. On the other side, Thymeleaf template engine along with Bootstrap are used for the user interface.

B.4 WP4: Network and Computing Slice Deployment Platform

The main objective of WP4 was to implement the MATILDA OSS, i.e., to provide a set of functionalities and building blocks to enable Telecommunication Service Providers to construct and put in operation a slice composed by network and computing resources, tailored on the requirements expressed by the vertical application orchestrator. The OSS is complementing the VAO in order to support the entire set of lifecycle management operations in cloud-native service graphs.

D4.1/4.2-Network and Computing Slice [4], [5] (first version delivered on M16 and final version on M24) delivered a full reference implementation of the OSS. The OSS is a modular, yet highly complex, component. Its internal structuring is depicted in Figure B17.

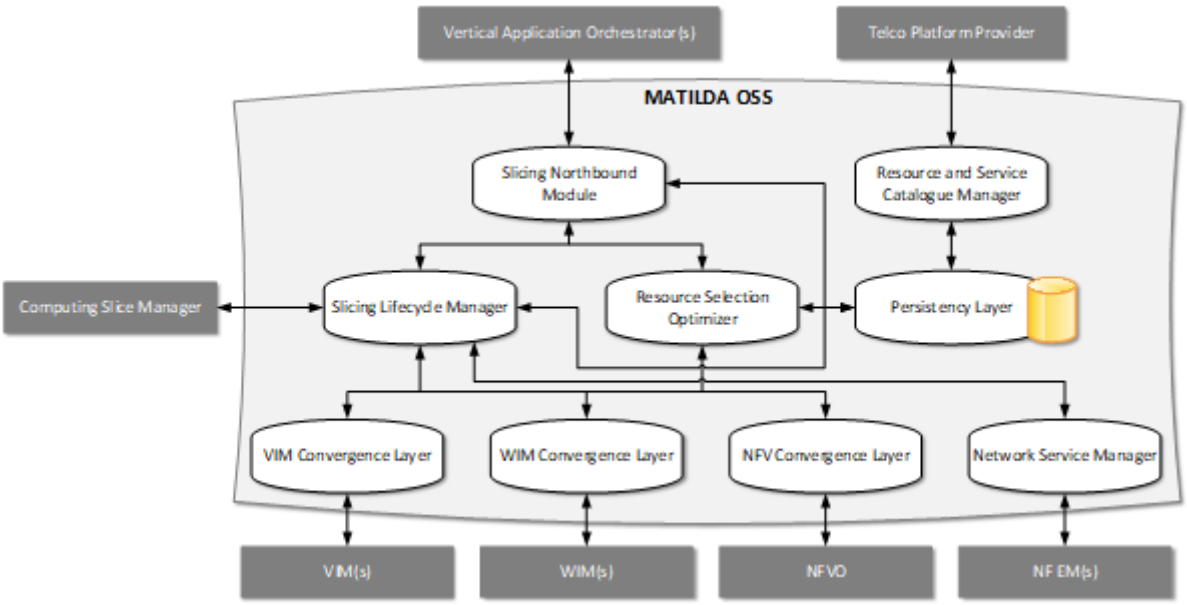

Figure B17: Internal structure of OSS.

Briefly, the flow starts when a slice intent is submitted from the VAO to the **Northbound Interface**. This interface is a REST-based interface that acts as a single point of interaction with the entire OSS, i.e., it proxies the full signalling from and to the VAO.

In case the request is a CREATE_SLICE_REQUEST, it is propagated to the **Slicing Lifecycle Manager**. The Slicing Lifecycle Manager will ask the **Resource Selection Optimizer** to find the best-fit configuration that can match the slice request. The optimizer itself is, practically, a constraint-satisfaction solver and as such it tries to find heuristically the **near-optimal** solution to the problem. Near optimal refers to the fact that, theoretically speaking, the **real optimal** is **computationally intractable**. In order for the optimizer to produce a solution, it must "feed" its internal constraint-satisfaction solver with **facts** and **rules**. **Rules** correspond to the actual constraints (that are expressed in the slice request per se) and facts correspond to the actual **state** of resources, i.e., which resources are registered, which of them are occupied, what is the oversubscription ratio that is configured, etc. All these facts reside in the **Persistency Layer.**

The persistency layer is implemented using NoSQL technology in order to achieve linear scalability.

The Persistency Layer is continuously updated by a dedicated component which is the **Resource and Service Catalogue Manager.** The specific component undertakes the task of extracting and persisting the latest "snapshot" of resources along the supported network services that can be used during the materialization of a slice. If the slice can be materialized the request is propagated to the **NFV Convergence Layer (NFVCL)**. This layer defines the processes that are required to **configure, instantiate and operate the MATILDA Network Functions using Opensource MANO.**

Each slice incorporates several **Virtual Network Function Forwarding Graphs** (VNFFGs). The NFVCL component oversees the NFV orchestration end to end within the slice that creates the necessary platform in terms of network functions, links or specific services required by the VAO to interconnect Application Components fulfilling requirements in terms of QoS, security and accessibility.

Internally, the NFVCL is composed of two modules, as depicted in Figure B18: the NFV Service Mapper (NFVSM) and the NFV Service Setup (NFVSS). The NFVSM is the component in charge of selecting the correct blueprint inside MATILDA's blueprint catalogue and calculating the amount of resources to be used in each node for the blueprint selected, once a bootstrap message is requested by the MATILDA OSS.

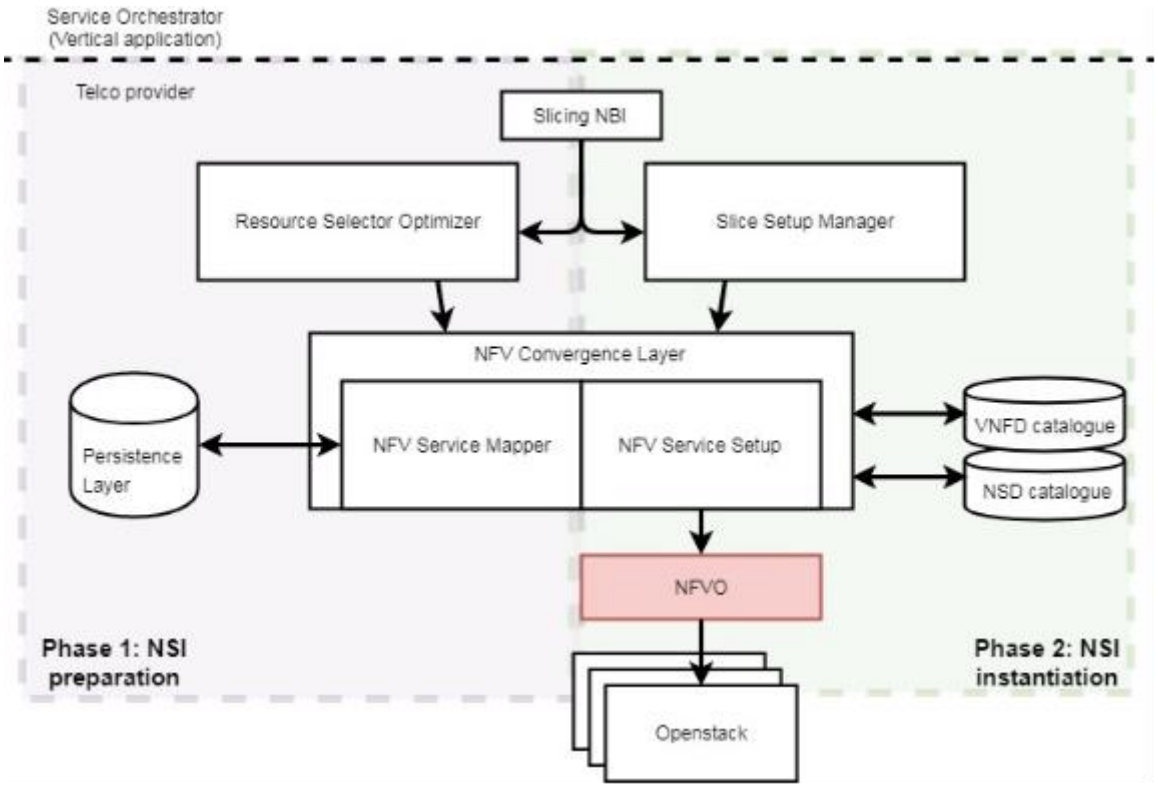

Figure B18: NFVCL internal architecture.

On the other hand, the NFVSS has three different interfaces; namely:

- **Interface-1 - Receives requests from the Slice Setup Manager** to instantiate, modify or delete the slice;
- **Interface-2 - Persists the data** about the status of the slice in the persistency layer;
- **Interface-3 - Interacts with OSM** for the materialization of the slice. The NFVCL implements a client that consumes OSM Northbound Interface to orchestrate the slice components in an ETSI-compliant way.

Upon materialization of the slice, the MATILDA-enabled 5G Telco Provider has:

- A set of configured VIMs that will be given to the VAO for component-deployment,
- Configured virtual and network resources (e.g., PNFs),
- Established paths/connectivity from and to all attachment points,
- Instantiated Network Services that are relevant to the Slice Request as part of the Virtual Network Function Forwarding Graph instantiation.

B.5 WP5: MATILDA Orchestrator, Testing and Refinement

The main objectives of WP5 (MATILDA Orchestrator, Testing and Refinement) included the detailed definition of the technical integrated endpoints and proactive planning of the integration of the different software components and mechanisms developed in WP2, WP3 and WP4, technical testing, refinement and finalization of the MATILDA framework Platform, Orchestrator and development environment. The following deliverables have been released during the project lifetime that contain the reporting of the main work done in WP5.

D5.1-Technical Integration Points and Testing Plan [21] provided the **specification of the development methodology** followed in the project, as well as a development and testing plan, focusing on the main integration points. A guideline was defined for the adequate development of the MATILDA software components, following a microservices-based approach. It includes an integration and testing plan, so each of the components should go through a set of tests (unit testing, stress testing, user acceptance, etc.) and satisfy requirements in terms of interfacing and software quality, in order to be considered ready for the final integration. A continuous integration (CI) methodology has been followed as a fundamental part of the integration and testing plan which has been adopted from the beginning of the project. A set of unit tests targeted to the MATILDA integrated framework have been also defined and presented.

D5.2 [22] and **D5.3** [23] (M18 and M24, respectively) provided **a short summary of the implementation of the MATILDA integrated framework**, along with concise information about the implementation technologies adopted and the relevant pointers to the produced software repositories. Focus is placed on the description of the implementation and integration status per component and per layer of the MATILDA architectural approach. The description of the main interactions among the MATILDA core architectural components is provided, along with a high-level workflow for these interactions and details for the developed APIs for supporting the interaction among the Software Development Kit (SDK), the VAO and the OSS. A set of installation guidelines for the final release of the MATILDA framework is also provided, aiming to facilitate the various adopters (e.g., MATILDA demonstrators, interested parties from other 5G PPP projects). Based on the integration and testing plan defined in D5.1, a set of functional tests have been realised and reported, leading to the validation of the proper functionality of the various MATILDA components and their successful integration. A short walkthrough in the MATILDA platform is provided, aiming to assist potential adopters to identify and use the main functionalities provided through the implemented graphical user interface.

The **main supported interactions** in the integrated MATILDA framework (see Figure B19) are as follows:

- The 5G-ready Applications Layer is interconnected with the Applications' Orchestration Layer for supporting onboarding of 5G-ready applications and application components to the MATILDA VAO. Onboarding is realised by application developers that provide access to the developed software to application service providers. Upon the completion of the software development and validation processes, the developed software is made available into a set of Repositories and is accessible to a set of components within the

VAO. Such components include the application graph composer and the policies' editor. Furthermore, export of the composed application graphs in a YAML format is also supported, making the application descriptors accessible to application developers for further usage.

- The Applications' Orchestration Layer is interconnected with the Network and Computing Slice Management Layer for supporting interaction between telecom operators and application service providers during the application-aware network slice lifecycle management. The main interactions include the request for instantiation of an application-aware network slice, the provision of information concerning the successful (or not) instantiation of the requested slice, and the provision of continuous network monitoring information related to QoS and QoE aspects.

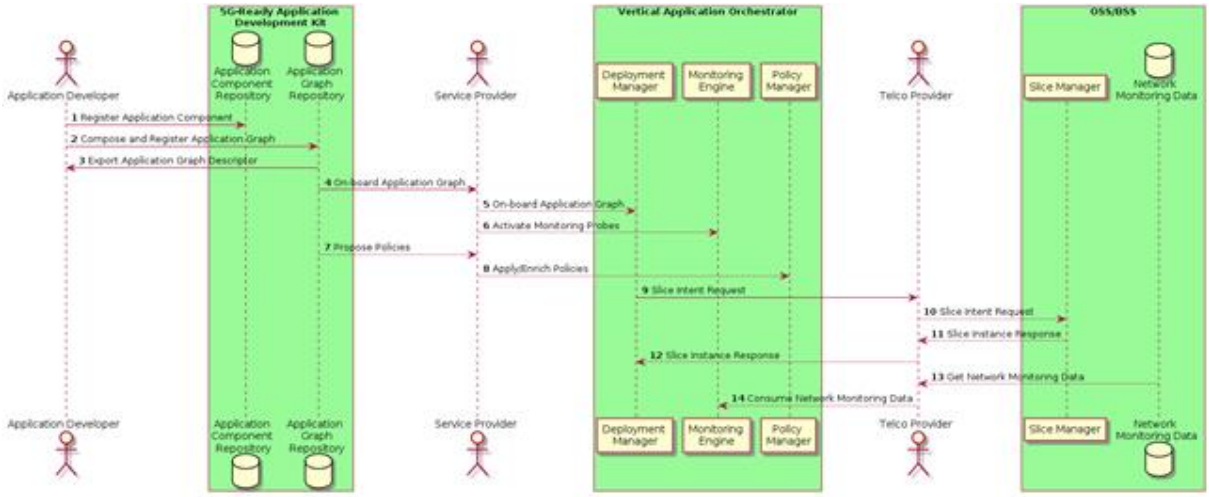

Figure B19: Main MATILDA Components' Interactions.

Short descriptions are provided for the following APIs:

- API#1: Onboarding API - Application Component Registration
- API#2: Onboarding API - Application Graph Registration
- API#3: Slice Intent Request
- API#4: Slice Instance Response and Profiler Status
- API#5: Application Instance Management.

B.6 WP6: MATILDA Industrial Demonstrators and Performance Evaluation

As already described in detail in this document, MATILDA has defined five different demonstrators, which are mapped to five corresponding vertical applications. These vertical applications have been demonstrated on top of MATILDA architectural layers and functional components to prove the applicability, usability, effectiveness and value of the MATILDA framework for vertical industries:

- **5G Emergency Infrastructure: a 5G system for Public Protection and Disaster Recovery (PPDR)**. It extends the capabilities of a real time intervention monitoring and critical infrastructure protection product suite with a suite for performance monitoring engines and advanced Operation, Administration and Management functionalities to support Service Level Agreements.
- **5G Personal Assistant during Crowded Events (5GPACE)**, offering end-users Immersive Media Services combined with Machine Learning-based personal retail recommendations.
- **Testing 4.0 - Distributed System Testing**: based on FastWAN, an experimental communication technology that was developed as a solution for the enablement of geographically separated real-time industrial test benches.
- **Industry 4.0 Smart Factory**: focusing on a logistic application, which offers customers the possibility to track, change and prioritize their orders and also in a use case to evaluate the benefits and limits of 5G within safety critical applications, such as Human-Robot Collaboration.
- **Smart City Intelligent Lighting System**: deployed in Politehnica University of Bucharest, in order to provide an easy replicable solution with fast time to market, automated maintenance and a modular approach enabled by 5G application graphs that will assure better monetization of the already existing intelligent city lighting solution.

The following deliverables have been released during the project lifetime that contain the reporting of the main work done in the following way:

D6.1 [24] reported on the activities part of **Task 6.1 – Demonstrators Planning and Validation Scenarios** and **T6.2 – Evaluation Framework Definition**. The main takeaways of those are detailed below.

- Initially, focus was given on reviewing the use cases defined in the scope of WP1 for the Project demonstrators. In more detail, evaluation steps and associated roles have been defined for the five use case applications, along with the expected releases and the capability/features of the demo together with test-bed requirements to emphasize the evolution among releases in terms of requirements. The acceptance criteria for each of the releases have been set as well, as guidance for final validation. Along with the use case applications, the selected test beds have been described in order not only to highlight their features and capabilities, but also to ensure homogeneity, compatibility with the use cases and feasibility for the integration with the MATILDA framework components. This latter aspect also regarded the possibility of providing for components

that would not be available until later stages of the Project lifetime by means of emulated software. A description of all the project phases has been produced, highlighting the planning to be followed for the test beds' development, each of them with a relevant set of capabilities to be demonstrated during international conferences or project review meetings and demonstrators' deployment strategy over the selected test beds.

- The evaluation framework required to assess the applicability and potential of the innovations brought forth by the MATILDA Project has been defined. A number of tests has been defined, spanning from end-user experienced performance to specific solution/component testing level. A template format has been fixed to homogeneously describe the tests to be performed and to collect the results, including the structuring of the test procedures and the tests' definition template. The tests to be performed have been classified in performance testing, aiming at evaluating the whole MATILDA solution on the basis of specific applications' QoS KPIs, solution components and functionality testing, to evaluate the performance of single or multiple components of the MATILDA framework, and general solution testing to evaluate the solution as a whole.

D6.2 [25] and **D6.8** [7] provide an overview of the completed implementation of MATILDA's public safety demonstrator titled 5G **Emergency Infrastructure with SLA Enforcement** (5G PPDR demonstrator). The demonstrator is designed to deliver tailor-made services and applications to support public safety teams in their day-to-day operations, as well as during extreme situations requiring large interventions, such as massive natural catastrophes.

The existing monolithic-based iMON Dashboard Application was disassembled to the application component level to enable deployment and intelligent orchestration of the iMON Dashboard application. The challenges that were faced during integration and implementation, such as achieving stateless operation and enabling horizontal scaling, were identified and resolved. Application components were prepared as docker images and on-boarded on the MATILDA framework.

The implementation is completed on top of a 5G infrastructure and is based on the implementation of 5G-enabled emergency response capabilities provided with the iMON product suite for real time intervention monitoring, extended with continuous performance monitoring engines of the qMON solution for supporting active and passive SLA monitoring and SLS enforcement. In the first implementation phase, iMON has been successfully deployed, while in the second phase both iMON application and qMON network and application performance monitoring components were automatically deployed with the MATILDA framework over the programmable 5G-ready network slice. The performance evaluation that took place after the second demonstrator release confirmed that the demonstrator is capable of reaching the expected thresholds and KPIs. This use case has been demonstrated in various events both during the first and the second release.

D6.3 [26] and **D6.9** [8] describe the evaluation activities of the demonstrator "**High Resolution Media on Demand with Smart Retail venue integration**" (5GPACE) into the MATILDA framework. 5GPACE is a large application that consists of two parts: one provided by INC (Retail Recommendation - Machine Learning Powered), the other by ITL (User management, localization data management and storage, video processing functionalities).

Specifically, Italtel's i-EVS system provides high quality, immersive video services and geo-localization functionalities; Incelligent's Artificial Intelligence-based framework provides complex analysis on historical and mobility data, and thus it can be used to provide personalized recommendations to each single user in real-time. Combined together the two parts have formed the demonstration scenario where the user, through his/her mobile device, can discover personalized recommendations and offers, based on the user's exact current location, and preference to consume certain types of products or services (possibly in sequences or bundles). Such data-driven retail recommendations come in the form of a "Move to next shop that offers an A% discount just for you" visual aid on the user's mobile screen and are created by applying advanced machine learning methods on the user's purchase history and mobility data, in user-perceived real time. This application has been successfully onboarded in the MATILDA Framework. The final release of 5GPACE confirmed a successful integration and deployment of its graph including all the components, this way obtaining a "5G-ready" application managed within the MATILDA framework. Moreover, 5GPACE has taken advantage of the MATILDA capability to assure the needed computational, networking and storage resources over a 5G telco infrastructure, as well as other service constraints, in order to be able to offer the required QoS to the users. The majority of the testing procedures and KPIs' evaluation have been completed, and the expected performance thresholds have been successfully reached. Due to the impossibility to access the MATILDA project testbeds, caused by the COVID-19 outbreak, it should be highlighted that the evaluation process suffered interruptions and delays. In this context, it was not possible to complete the evaluation process (e.g., for GPU use for demonstrating the hardware acceleration capabilities at the edge). However, this has not had any other significant effect in the proposed demonstration scenarios and it does not take away from the otherwise highly successful results of the testing process.

D6.4 [27] and **D6.10** [9] detail the implementation and evaluation of a **Smart City Intelligent Lighting System** that is a smart city application that fulfils the demand for flexible and fast management of a city's lighting system. The application is targeted on lowering the operational costs while maximizing the energy efficiency of that lighting system. The Smart City Intelligent Lighting platform runs over a LoRaWAN/LTE-M network and provides end to end management and maintenance of all the lighting poles registered. The demonstrator was developed as a joint effort between MATILDA and SLICENET projects and deployed using components from both projects. The physical deployment that contains all the lighting poles takes place inside the Polytechnics campus in Bucharest. The application has been successfully onboarded in the MATILDA Framework and has been deployed and demonstrated over different programmable infrastructures; namely, over ORANGE and over CNIT test beds. Based on the performance evaluation that has been realised, the most part of the operational and network KPIs have been achieved.

D6.5 [28] and **D6.11** [6] detail the implementation and evaluation of the **Industry 4.0 Smart Factory** demonstrator. The demonstrator emphasizes on BIBA's Industrial Technology over the MATILDA 5G Framework and focuses specifically on the production scenario which deals with the provisioning of workspace safety services in industrial environments, thus improving efficiency and flexibility in INDUSTRY 4.0 manufacturing. The demonstrated scenario showcases and evaluates the establishment of safe human–robot coexistence and collaboration through active collision avoidance between robots and human beings entering the robot's workspace. Furthermore, a logistics scenario has been examined that deals with the tracking

and monitoring of sensitive goods, starting from a supplier and moving to a production facility, as well as the tracking of transport vehicles carrying the goods. After a successful onboarding and deployment of the workspace safety application in MATILDA, the performance evaluation showed that all targeted operational KPIs have been successfully met. For the networking KPIs, while the majority of them have been reached, lower end to end delay can be achieved with further optimizations with full-fledged 5G equipment.

D6.6 [29] and **D6.12** [10] detail the implementation and evaluation of the demonstrator "**Testing 4.0 – Distributed System Testing**". In this demonstrator, ExxpertSystems has embedded a distributed (remote) interconnection technology called FastWAN that enables machine to machine communication with the following characteristics: Real Time, Interoperable, High Quality of Service (QoS), Guaranteed Data Delivery with High Data Volume Capabilities, Plug and Play, Support for both Industrial Signals and Network Data. The Distributed System Testing demonstrator exhibits the possibility of using FastWAN technology over the MATILDA 5G Framework for the Automotive Industry. The scenario performs tests on mobile Systems Under Test, such as automobiles and the different tests will be controlled and monitored from a remote location (Command and Control Center) by different users (customers). In the first implementation phase the basic version of the application has been onboarded and deployed in the MATILDA Framework. Several aspects of the application have been validated, like the communication establishment, the service discovery and multicast communication, among the application components once deployed. In the second implementation phase, FastWAN has been extended with QoS policies over the type of data, graphical representation of the information carried inside a data stream has been made available within the UI, and the service discovery between the components has been extended. Now that the testing procedures and the performance evaluation have been completed it is reported that the performance thresholds with regard to certain KPIs have been successfully reached. However, several KPIs could not be evaluated, as they demand physical access to the test bed.

D6.7/D6.13-Validation Results, Performance Evaluation and Adoption Guidelines [30][31] (first and final version) provide details for the MATILDA evaluation framework along with a complete set of validation and performance evaluation results. Initially, the MATILDA Evaluation framework is presented, including a mapping between the objectives and the different project stages. Following, a set of validation and evaluation activities have been realised leading to the measurement of various KPIs and the validation of the successful operation of the various MATILDA components. The following Figure B20 illustrates the generic approach and a mapping between the MATILDA implementation and the validation and evaluation activities. 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.

- **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.

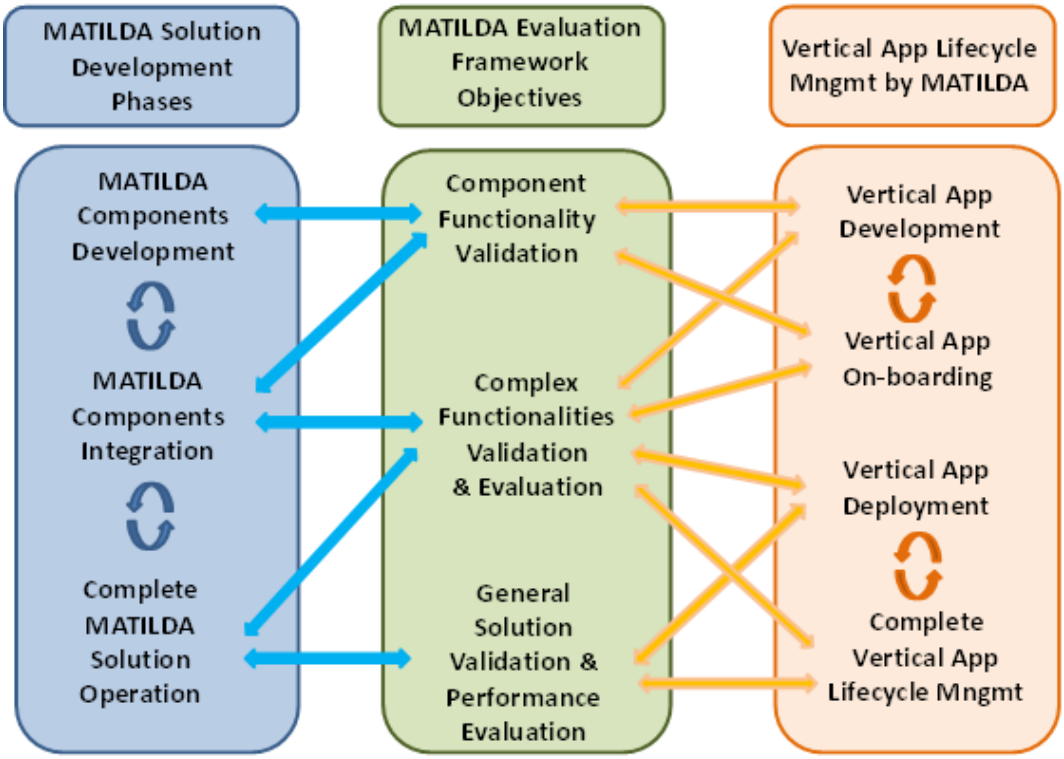

Figure B20: MATILDA Evaluation Framework Overview

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.

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. 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 the VAO in order to provide instantaneous help to the potential end-user.

B.7 WP7: Dissemination, Communication, Exploitation and Business Planning

WP7 has performed several activities with the overall aim of maximising the impact of MATILDA outcomes from the rest of WPs, including promotion and communication activities, scientific dissemination, standardization and industrial clustering, 5G PPP interaction, market analysis and business plan elaboration, and innovation management. The following deliverables have been released during the project lifetime, which contain the reporting of the main work done in the way described.

D7.1 [32] covered the initial communication plan of the project, where all the communication actions to be performed in the project are collected, including website, presence in social media (Twitter, YouTube, LinkedIn, etc.), creation of promotional material, leaflets, videos, newsletters, as well as timelines and responsibilities among the partners.

D7.2 [33] and **D7.6** [37] reported on the activities part of **Task 7.1 – Dissemination, clustering and standardization activities** and **T7.2 – 5G PPP interaction**. The main takeaways of those are detailed below.

- MATILDA partners have performed multiple dissemination activities as presentations in different events (detailed in D7.6 and in the MATILDA website: http://www.matilda-5g.eu/index.php/outcomes), as well as submitted multiple papers acknowledging MATILDA work to technical journals and conferences. The following Table shows the results achieved within the dissemination activities with respect to the initially defined key performance indicators.

KPIs	Overall Target value	Achieved Value
Participation in MWC 2019	Yes	The consortium participated
Contribution to EU booth at MWC 2019 with demo/testbed showcasing MATILDA's results	Yes	Done
White papers and contribution to roadmaps	≥5	3
Journals	≥10	13
Conferences	≥20	32
Posters	≥3	3
Scientific and technical workshops	3	3
Book chapters	0	3

- As part of the **clustering activities**, some of the realised events include the following:
  - A clustering workshop titled "5G-ready Network Applications Development and Orchestration over Network Slices with Mobility Support" was held in conjunction with the fourth IEEE Conference on Network Function Virtualization and Software Defined Networks, in Verona, Italy on the 27th of November 2018.
  - In the context of EuCNC'18 and EUCNC'19, MATILDA has co-organized and participated in various workshops, such as:
    - WS 1: "**From cloud ready to cloud native transformation: What it means and why it matters**", which was co-organized under the umbrella of the software network work group by partners from the MATILDA project and in which other 5G PPP project members participated.
    - WS 2: "**Multi-provider, multi-vendor, multi-player orchestration: from distributed cloud to edge and fog environments in 5G**".
    - WS 3: "**5th International Workshop on programmable networks: Demystifying software networks for Vertical Industries**", organized under the umbrella of the programmable networks work group.
    - WS 4: "**3rd Network Management and QoS for 5G Networks**", organized under the umbrella of the network management work group.
  - MATILDA co-organized a workshop with the SLICENET project between 12-14th of November 2019, where consortium members from both projects had effective discussions to foster better understanding and alignment of both projects and benefit of their mutual results, as well as for other 5G PPP projects.
  - The last clustering workshop titled "**MATILDA – Autonomic deployment and lifecycle management of 5G & beyond services**" was initially organized alongside with the 2nd 6G Wireless Summit 2020, in Levi, Lapland, Finland. However, due to the outbreak of the coronavirus across Europe, the event took place online, seeing the participation of a relevant audience and of numerous external invited speakers (website: https://www.matilda-5g.eu/index.php/events/39-beyond5g).
- As part of the **5G PPP interaction** commitment CNIT has participated in and followed Steering Board activities, as well as UBITECH the Technical Board activities. Besides the collaboration with other projects as explained in the clustering activities, MATILDA has also elaborated on technical synergies with other 5G PPP projects, e.g., SLICENET, 5GTANGO and NGPAAS, and has participated actively with MATILDA perspective in multiple 5G PPP working groups (WGs), as collected in the following table. Furthermore, MATILDA has contributed to various white papers produced within the 5G PPP WGs.

5G PPP Working Group	MATILDA partners involved
Software Networks	ORO (co-organizer), INC
Network Management and QoS	ININ, UBITECH
SME	ININ, SUITE5, INC, UBITECH
Trials	ATOS
Architecture	ININ, UBITECH, CNIT
Communications	ATOS, ERICSSON
Pre-standardization	ATOS
Vision	ATOS, CNIT
Test, Measurement and KPI Validation	UBITECH, CNIT

- In terms of standardization activities, MATILDA partners provided contributions or material to standardization bodies, mainly in groups within ETSI.
  - A contribution to the ETSI NFV IFA group was prepared, concretely linked to MATILDA NFV Convergence Layer, with the objective of influencing activities around NFV release 4.
  - The aforementioned contribution has been implemented and promoted to ETSI OSM NFVO, since it can be implemented with OSM by means of integrating some components of the NFV convergence layer (NFVCL) inside OSM. A demonstration was provided during the 8th Hackfest held in Lucca, Italy, in November 2019, focused on showcasing the MATILDA Telco Layer Platform, and in particular how it exploits OSM to manage the lifecycle of network services to expose the 5G network infrastructure to vertical applications in terms of network and computing slices.
  - Within the ETSI EE TC (Environmental Engineering Technical Committee), the second version of the Green Abstraction Layer (GAL v2.0) has been approved and published at the beginning of 2020, with the name ETSI ES 203 682 (as an evolution of the previous ES 203 237). This interface is relevant to the vision of MATILDA as well, since the second version has been extended explicitly for controlling the energy consumption in the NFV environment.
  - A new model for the integration of external network access to micro-services in a Mobile Edge Computing environment has been proposed and raised potential interest to ETSI MEC WG upon invitation in MEC#22.

D7.3 [34] and **D7.7** [38] have provided the communication plan and the set of communication activities performed in the project related to: project website, presence on social media channels (Twitter, YouTube, LinkedIn), press releases, leaflet elaboration sent to key events such as MWC and EuCNC, newsletters, and posters. They regard the outcome of the activities of **Task 7.3 Communication Activities and Data Management.**

D7.4 [35] and **D7.8** [39] comprise the outputs of the work done within **Task 7.4 Innovation Management** and **Task 7.5 Market analysis and Exploitation, Business and Sustainability Planning**. The following activities have been realised:

- An extensive but general **market research** has taken place to understand the market context where MATILDA aimed to place itself. This context, and the requirements and trends of the market, which have been constantly watched, have influenced MATILDA architecture design, as it was the consortium's objective to ensure that the finished technology was usable and addressed real challenges of the market. 5G market forecasts for key technological areas related to MATILDA (NFV, Network Slicing, MEC, Zero-touch Service Management) are made available.
- In relation to **Market assessment exploitation and business planning** the following activities have been performed: market context analysis, including study of trends, drivers and barriers, and elaboration on how MATILDA can have niche opportunities due to its contribution to mitigate some of those barriers; analysis of the full value chain, but special look at actual ***main MATILDA users, e.g. telecom providers, vertical services providers and application developers***, as well as analysis of the business impact for each of the MATILDA use cases; specification of individual exploitation plans. Reflecting upon the product that MATILDA has to offer to the market, and taking into account the

ownership and licences given to the single MATILDA components, the Consortium has identified not only the end to end solution as an exploitable product, but also some component packages addressed to the different targeted market segments and with different forms of exploitation. Four offerings are identified:

  - Offering (1): Promote an end-to-end solution with the integrated MATILDA platform mainly targeted to small TSPs (proprietary solution),
  - Offering (2): Make available the MATILDA Network Platform as an open-source tool that can be adopted by TSPs,
  - Offering (3): Make available the VAO as a proprietary solution that can be adopted by OTT players and be bound to network slice management solutions,
  - Offering (4): Promote joint usage of the integrated MATILDA platform with specific services integrated (as a proprietary solution in collaboration with use case partners).

- Based on these offerings, Lean Canvas Analysis has been realised for the main MATILDA stakeholders (Telecom Service Providers, Cloud/Edge Computing Application Providers), as well as per Vertical Industry addressed in the MATILDA Demonstrators. Being small TSPs the main potential customer for the MATILDA end-to-end solution, their problems and particularities have been considered to refine MATILDA's value proposition for **small TSPs**. The fact that these emerging small Telco Operators have not implemented their own orchestration systems underlines that **the adoption of the MATILDA framework would be very beneficial, allowing them to easily register their programmable 5G resources and provide their services without the need to buy or implement their own OSS and orchestration systems**.
- In relation to **Innovation Management**, a list of the expected innovative results from MATILDA and the partners contributing to each of them has been elaborated among all the partners, leading to the specification of licensing schemes and IPR management.

D7.5 [36] reported on the activities part of **Task 7.6 -Project's Impact Assessment**. In order to maximise the MATILDA project results' exploitability and sustainability, a specific activity to monitor and assess the project impact has taken place. In this context, an analysis of the research process has been conducted, along with a review of impact assessment classifications, concepts, methodologies and frameworks focusing on R&D projects and activities. The MATILDA impact assessment framework has been defined using a logical model approach applicable to the MATILDA research process, incorporating well defined indicators as criteria for assessment of research impact, emphasizing on activities, output/outcomes, and impact measures. The technical, scientific and business impact of MATILDA is provided.

- the Technological impact to the field of 5G networks, in the worldwide technology/research community, and on specific vertical markets, has been evaluated against KPIs related to the MATILDA technical output on Lifecycle Management of 5G-ready applications (components and graphs), VNFs (including Forwarding-Graphs and orchestration), and Network Services (associated also to Telecom Layer OSS capabilities). As detailed in D7.5, the project achieved all its technical goals against these KPIs and even exceeded the initially defined targets.
- the Scientific impact in the research field of 5G networks associated to the Dissemination output of the project has been evaluated against KPIs related to MATILDA participation

in conferences / workshops / exhibitions with demonstrators, presentations, papers. In addition, the same strategic impact associated to the Communication output of the project to approach stakeholders and the market has been evaluated against KPIs related to establishing social media and corporate communication channels and generating / presenting material in industrial events / forums / tutorials/ webinars / websites, etc. As detailed in D7.5, the project achieved all its dissemination / communication goals against these KPIs and even exceeded the initially defined targets, while additional means / channels of dissemination / communication were utilised (not initially foreseen).

- the Business/Economic impact was very difficult to be assessed at project time, since the business remains to be evolved after the technical development finishes with the project end. However, estimations about the expected business/economic impact on partners as well as on target stakeholders in the 5G value chain in the long run have been analysed as part of D7.8.

D7.9 [40] drafted the data management plan that has been followed during the project.

References

[1] R. Bolla, R. Bruschi, F. Davoli, P. Gouvas, A. Zafeiropoulos, “Mobile edge vertical computing over 5G network sliced infrastructures: an insight into integration approaches,” *IEEE Communications Magazine*, vol. 57, no. 7, pp. 78-84, July 2019.

[2] OSM Rel. 5, https://osm.etsi.org/wikipub/index.php/OSM_Release_FIVE

[3] OSM White Paper, https://osm.etsi.org/images/OSM-Whitepaper-TechContent-ReleaseFIVE-FINAL.pdf.

[4] MATILDA project deliverable, "D4.1 - Network and Computing Slice – First Release", Confidential, available on request online at: https://private.matilda-5g.eu/Documents/DownloadFile/344.

[5] MATILDA project deliverable, "D4.2 - Network and Computing Slice", Public, available online at: https://private.matilda-5g.eu/documents/PublicDownload/487.

[6] MATILDA project deliverable, "D6.11 - Industry 4.0 Smart Factory Implementation Report – Second demonstration phase", Confidential, available on request online at: https://private.matilda-5g.eu/Documents/DownloadFile/585.

[7] MATILDA project deliverable, "D6.8 - Emergency Infrastructure with SLA Enforcement Implementation Report – Second demonstration phase", Confidential, available on request online at: https://private.matilda-5g.eu/Documents/DownloadFile/574.

[8] MATILDA project deliverable, "D6.9 - High Resolution Media on Demand Implementation Report – Second demonstration phase", Confidential, available on request online at: https://private.matilda-5g.eu/Documents/DownloadFile/575.

[9] MATILDA project deliverable, "D6.10 - Smart City Intelligent Lighting System Implementation Report – Second demonstration phase", Confidential, available on request online at: https://private.matilda-5g.eu/Documents/DownloadFile/545.

[10] MATILDA project deliverable, "D6.12 - Automobile Electrical Systems Remote Control Implementation Report – Second demonstration phase", Confidential, available on demand online at: https://private.matilda-5g.eu/Documents/DownloadFile/576.

[11] MATILDA project deliverable, "D1.1 - MATILDA Framework and Reference Architecture", Public, available online at: https://private.matilda-5g.eu/documents/PublicDownload/119

[12] MATILDA project deliverable, "D1.2 - Chainable Application Component & 5G-ready Application Graph Metamodel", Public, available online at: https://private.matilda-5g.eu/documents/PublicDownload/192

[13] MATILDA project deliverable, "D1.3 - VNF/PNF & VNF Forwarding Graph Metamodel", Public, available online at: https://private.matilda-5g.eu/documents/PublicDownload/194

[14] MATILDA project deliverable, "D1.4 - Network-aware Application Graph Metamodel", Public, available online at:
https://private.matilda-5g.eu/documents/PublicDownload/196

[15] MATILDA project deliverable, "D1.5 - Deployment and Runtime Policy Metamodel", Public, available online at:
https://private.matilda-5g.eu/documents/PublicDownload/198

[16] MATILDA project deliverable, "D1.6 - Supported Verticals, Use Cases and Acceptance Criteria", Public, available online at:
https://private.matilda-5g.eu/documents/PublicDownload/333

[17] MATILDA project deliverable, "D2.1 - 5G-Ready Applications and Network Services Development Environment and Marketplace – First Release", Confidential, available online at:
https://private.matilda-5g.eu/Documents/DownloadFile/340

[18] MATILDA project deliverable, "D2.2 - 5G-Ready Applications and Network Services Development Environment and Marketplace", Public, available online at:
https://private.matilda-5g.eu/documents/PublicDownload/483

[19] MATILDA project deliverable, "D3.1 - Intelligent Orchestration Mechanisms – First Release", Confidential, available online at:
https://private.matilda-5g.eu/Documents/DownloadFile/330

[20] MATILDA project deliverable, "D3.2 - Intelligent Orchestration Mechanisms", Public, available online at:
https://private.matilda-5g.eu/documents/PublicDownload/485

[21] MATILDA project deliverable, "D5.1 - Technical Integration Points and Testing Plan", Confidential, available online at:
https://private.matilda-5g.eu/Documents/DownloadFile/250

[22] MATILDA project deliverable, "D5.2 - MATILDA Integrated Framework – First Release", Confidential, available online at:
https://private.matilda-5g.eu/Documents/DownloadFile/392

[23] MATILDA project deliverable, "D5.3 - MATILDA Integrated Framework", Confidential, available online at:
https://private.matilda-5g.eu/Documents/DownloadFile/489

[24] MATILDA project deliverable, "D6.1 - Evaluation Framework and Demonstrators Planning", Public, available online at:
https://private.matilda-5g.eu/documents/PublicDownload/328

[25] MATILDA project deliverable, "D6.2 - Emergency Infrastructure with SLA Enforcement Implementation Report – First demonstration phase", Confidential, available online at:
https://private.matilda-5g.eu/Documents/DownloadFile/409

[26] MATILDA project deliverable, "D6.3 - High Resolution Media on Demand Implementation Report – First demonstration phase", Confidential, available online at:
https://private.matilda-5g.eu/Documents/DownloadFile/443

[27] MATILDA project deliverable, "D6.4 - Smart City Intelligent Lighting System Implementation Report – First demonstration phase", Confidential, available online at:
https://private.matilda-5g.eu/Documents/DownloadFile/403

[28] MATILDA project deliverable, "D6.5 - Industry 4.0 Smart Factory Implementation Report – First demonstration phase", Confidential, available online at: https://private.matilda-5g.eu/Documents/DownloadFile/465

[29] MATILDA project deliverable, "D6.6 - Automobile Electrical Systems Remote Control Implementation Report – First demonstration phase", Confidential, available online at: https://private.matilda-5g.eu/Documents/DownloadFile/441

[30] MATILDA project deliverable, "D6.7 - Validation Results, Performance Evaluation and Adoption Guidelines – First demonstration and evaluation phase", Public, available online at: https://private.matilda-5g.eu/documents/PublicDownload/436

[31] MATILDA project deliverable, "D6.13 - Validation Results, Performance Evaluation and Adoption Guidelines – Second demonstration and evaluation phase", Public, available online at: https://private.matilda-5g.eu/documents/PublicDownload/587

[32] MATILDA project deliverable, "D7.1 - Communication Roadmap", Confidential, available online at: https://private.matilda-5g.eu/Documents/DownloadFile/156

[33] MATILDA project deliverable, "D7.2 - 5G-PPP Interaction, Dissemination, Clustering & Standardisation Activities Report - Halfway", Public, available online at: https://private.matilda-5g.eu/documents/PublicDownload/302

[34] MATILDA project deliverable, "D7.3 - Communication Activities Report - Halfway", Public, available online at: https://private.matilda-5g.eu/documents/PublicDownload/304

[35] MATILDA project deliverable, "D7.4 - Market Analysis, Business Plan, Sustainability Model & Innovation Management", Confidential, available online at: https://private.matilda-5g.eu/Documents/DownloadFile/306

[36] MATILDA project deliverable, "D7.5 - MATILDA Impact Assessment", Public, available online at: https://private.matilda-5g.eu/documents/PublicDownload/577

[37] MATILDA project deliverable, "D7.6 - 5G-PPP Interaction, Dissemination, Clustering & Standardisation Activities Report - Final", Public, available online at: https://private.matilda-5g.eu/documents/PublicDownload/578

[38] MATILDA project deliverable, "D7.7 - Communication Activities Report - Final", Public, available online at: https://private.matilda-5g.eu/documents/PublicDownload/579

[39] MATILDA project deliverable, "D7.8 - Market Analysis, Business Plan, Sustainability Model & Innovation Management - Final", Confidential, available online at: https://private.matilda-5g.eu/Documents/DownloadFile/580

[40] MATILDA project deliverable, "D7.9 - Data Management Plan", Confidential, available online at: https://private.matilda-5g.eu/Documents/DownloadFile/207