ENVELOPE

Evaluation and validation of connected mobility in real open systems beyond 5GS

Project deliverable D4.1

Deployment of ENVELOPE architecture to the sites and development of data collection tools and internet service repository

HORIZON JU Innovation Actions | 101139048 |
ENVELOPE - HORIZON-JU-SNS-2023

Co-funded by the European Union

6GSNS

Deliverable administrative information

Dissemination level	PU - Public
Type of deliverable	**R — Document, report**
Work package	**WP4**
Title of the deliverable	**D4.1 – Deployment of ENVELOPE architecture to the sites and development of**
Status – version, date	**Final – V1.0 08/04/2026**
Deliverable leader	**ICCS**
Contractual date of delivery	**31/03/2026**
Submission date	**08/04/2026**
Keywords	**Integration, 5G Network Infrastructure, Edge Computing Infrastructure, Vehicles, Data Collection.**

List of contributors

Name	Organisation
Davide Montagno Bozzone	**Hewlett Packard Enterprise**
Nicola di Pietro	**Hewlett Packard Enterprise**
Elvina Gindullina	**Hewlett Packard Enterprise**
Rosario Fria	**Teoresi**
Angela di Fazio	**Teoresi**
Alberto Bertone	**Teoresi**
Gergely Kovács	**CMS**
Edoardo Bonetto	**LINKS**
Dimitris Tsolkas	**FOGUS**

Christos Milarokostas	**FOGUS**
Alex Kakyris	**FOGUS**
Ioannis Kontizas	**NCSRD**
Dimitris Uzunidis	**NCSRD**
Vassilis Pitsilis	**NCSRD**
Harilaos Koumaras	**NCSRD**
Pavlos Basaras	**ICCS**
Dinos Katsaros	**ICCS**
Stratis Myritzis	**ICCS**
Gabriele Scivoletto	**NXW**
Ramon de Souza Schwartz	**TNO**
Geerd Kakes	**KPN**
Guus Botman	**TNO**
Rakshith Kusumakar	**V-TRON**
Nikhil Muthakana	**HAN**
Ján Macháček	**RoboAuto**
Apostolos Siokis	**IQU**

Quality control

	Name	Organization	Date
Peer review 1	Dimitris Uzunidis Ioannis Kontizas	NSCRD	27/3/2026
Peer review 2	Geerd Kakes	KPN	30/3/2026

Version History

Version	Date	Author	Summary of changes
0.1	7/1/2026	Pavlos Basaras	Document structure and ToC discussed and agreed in WP4 meeting.
0.4	11/2/2026	All contributors	First round of contributions.
0.6	16/3/2026	All contributors	Final contributions submitted. Peer review launched.
0.7	30/3/2026	Peer reviewers	Internal review by two reviewers completed.
0.8	6/4/2026	All contributors	Revisions addressed by all contributors.
0.9	8/4/2026	Project coordinator	Project coordinator reviews the final document.
1.0	8/4/2026	Project coordinator	Submits the deliverable to EC.

Legal Disclaimer

Table of Contents

List of Figures

List of Tables

Project executive summary

ENVELOPE aims to advance and open the reference 5G advanced architecture and transform it into a vertical-oriented one. It proposes a novel 5G-advanced architecture to enable a tighter integration of the network and the service information domains by

- exposing network capabilities to verticals,
- providing vertical information to the network; and
- enabling verticals to dynamically request and modify key network aspects,

all performed in an open, transparent, and easy-to-use, semi-automated way.

ENVELOPE will build APIs that act as an intermediate abstraction layer that translates the complicated 5GS interfaces and services into easy to consume services accessible by the vertical domain. The project will deliver an experimentation framework that will facilitate vertical services in accessing a series of innovations developed in the project, namely: edge computing with service continuity support (federation/migration), zero-touch management, multi-connectivity, dynamic slicing and predictive QoS.

ENVELOPE will deliver 3 large scale Beyond 5G (B5G) trial sites in Italy, Netherlands and Greece supporting novel vertical services, with advanced exposure capabilities and new functionalities tailored to the services' needs. Although focused on the Connected and Automation Mobility (CAM) vertical, the developments resulting from the use cases (UC) will be reusable by any vertical. The ENVELOPE architecture will serve as an envelope that can cover, accommodate, and support any type of vertical services. The applicability of ENVELOPE will be demonstrated and validated via the six project CAM UCs and via several 3rd parties that will have the opportunity to conduct funded research and test their innovative solutions over ENVELOPE.

Social Media link:

@envelope-project

For further information please visit www.envelope-project.eu

Deliverable executive summary

This deliverable presents the deployment of the ENVELOPE experimental infrastructure across the three project Living Labs located in Italy, the Netherlands, and Greece. The deployed environments provide the technological foundation required to support the validation of ENVELOPE innovations for Beyond 5G (B5G) networks in Connected, Cooperative and Automated Mobility (CCAM) scenarios.

The document describes the main components of the deployed infrastructures per Living Lab, including the radio access networks, 5G core and edge/cloud computing platforms, vehicles, sensing systems, and supporting devices used for the purposes of the project. In addition, it defines the framework for experimental data collection and monitoring, identifying the logging mechanisms and measurement points that enable the evaluation of ENVELOPE functionalities and Key Performance Indicators (KPIs).

By establishing the experimental environments and the associated data management mechanisms, this deliverable prepares the ENVELOPE trial sites for the upcoming experimentation campaign. The deployed infrastructure and monitoring framework constitute a key step toward the integration and pre-evaluation testing of the trial sites, which will be detailed in Deliverable D4.2. More specifically, D4.2 will link the ENVELOPE infrastructure presented in this deliverable with the targeted use cases, technology enablers, and APIs, and will verify their correct operation through testing at use-case, technology, and API level. This will provide the necessary readiness assurance and green light for the full evaluation activities to be carried out in Work Package 6.

List of abbreviations and acronyms

Acronym	Meaning
AF	Application Function
AI	Artificial Intelligence
ATSSS	Access Traffic Steering, Switching and Splitting
AV	Autonomous Vehicle
B5G	Beyond 5G
CAM	Connected and Automated Mobility
CAN	Controller Area Network
CCAM	Connected, Cooperative and Automated Mobility
CN	Core Network
CU	Central Unit
DMR	Dual Modular Redundancy
DRW	Drive-by-Wire
DT	Digital Twin
DU	Distributed Unit
EaaS	Experimentation as a Service
ECC	Error-Correcting Code
ECU	Electronic Control Unit
EMF	ElectroMagnetic Field
ETSI	European Telecommunications Standards Institute
GNSS	Global Navigation Satellite Systems
GPU	Graphics Processing Unit
HPLMN	Home Public Land Mobile Network
HTTP	Hypertext Transfer Protocol
IaC	Infrastructure as Code
K8s	Kubernetes
KPI	Key Performance Indicator

LiDAR	Light Detection and Ranging
LoS	Line of Sight
MEC	Multi-access Edge Computing
ML	Machine Learning
MIMIT	Ministry of Enterprises and Made in Italy
MIMO	Multiple Input and Multiple Output
NEF	Network Exposure Function
NF	Network Function
NLoS	Non-Line of Sight
NR	New Radio
NWDAF	Network Data Analytics Function
OBU	On-Board Unit
OC	Open Call
O-RAN	Open Radio Access Network
PMF	Performance Measurement Function
PoP	Point of Presence
PQoS	Predictive Quality of Service
QoD	Quality on Demand
QoS	Quality of Service
RAN	Radio Access Network
REST	REpresentational State Transfer
RL	Reinforcement Learning
RSU	Road Side Unit
RSRP	Reference Signal Received Power
RTK	Real-Time Kinematic
RTT	Round-Trip Time
SA	StandAlone
SPS	Stationary Perception Set
TDD	Time Division Duplex

UC	Use Case
UE	User Equipment
URI	Uniform Resource Identifier
URSP	UE Route Selection Policy
V2V	Vehicle-to-Vehicle
VPLMN	Visited Public Land Mobile Network
VM	Virtual Machine
VRU	Vulnerable Road User
WP	Work Package
3GPP	3rd Generation Partnership Project

1 Introduction

1.1 Purpose of the deliverable

This deliverable presents the deployment of the ENVELOPE architecture across the project trial sites and describes the mechanisms established for data collection and management to support experimentation and performance evaluation. In particular, it documents the infrastructure and system components deployed in the Italian, Dutch, and Greek Living Labs (LLs), including the radio access, edge and cloud computing resources, vehicles, sensors, and supporting equipment used for the execution of the project use cases and open-call projects.

Furthermore, the deliverable outlines the data collection framework implemented across the trial sites, detailing the logging mechanisms, data sources, and measurement points used to monitor system behavior and evaluate the ENVELOPE Key Performance Indicators (KPIs). The document also provides technical guidelines for data acquisition, transmission, pre-processing, and storage to support consistent and reliable data management throughout the experimentation activities, while serving as guidance for the Living Lab activities.

This deliverable also serves as an input to Deliverable D4.2, which will report on the integration of the ENVELOPE trial sites and particularly the execution of pre-evaluation testing activities. The results of D4.2 will confirm the readiness of the experimental infrastructure and provide the basis for the large-scale experimentation and evaluation activities conducted in Work Package 6.

1.2 Intended audience

This deliverable is primarily intended for the ENVELOPE consortium partners involved in the deployment, integration, and experimentation activities within Work Package 4 and related technical work packages. It provides detailed information on the infrastructure deployed at the trial sites, the configuration of the experimental environments, and the mechanisms used for data collection and monitoring during experimentation activities.

In addition, the document may be useful for researchers, engineers, and practitioners working in the fields of Beyond 5G (B5G) networking, CAM, and edge/cloud-enabled experimentation platforms. As the deliverable is public, it also targets the wider research and innovation community, including standardization bodies, industry stakeholders, and other European research initiatives interested in large-scale experimental infrastructures and validation methodologies for advanced 5G and B5G systems.

1.3 Structure of this deliverable

The remainder of this deliverable is organized as follows. Sections 2, 3, and 4 describe the ENVELOPE experimental infrastructure deployed in the Italian, Dutch, and Greek Living Labs, respectively. Each section presents the Living Lab setup, including the deployed infrastructure (network-edge-cloud), vehicles and end devices, and the data collection and management mechanisms used to support experimentation activities. Section 5 provides technical guidelines for data management across the trial sites, covering aspects related to data acquisition, transmission, pre-processing, and storage. Finally, Section 6 concludes the deliverable and summarizes the main outcomes of the deployment activities.

2 Italian Living Lab

2.1 Living Lab Description

2.1.1 Description of the infrastructure

The Italian Living Lab is located in Torino, in the area of Corso Castelfidardo, close to the premises of LINKS and Politecnico di Torino. Figure 1 provides a geographical overview of the trial area, Figure 2 provides a view of the trial area from the perspective of the antenna, while Figure 4 and Figure 5 highlight the main infrastructure elements involved in the ENVELOPE experimentation activities, and their deployment within the Italian Living Lab.

The Italian Living Lab has been built from scratch as no previous infrastructure was already deployed. Within the ENVELOPE project, the Italian Living Lab deployed the necessary hardware resources that are required for supporting the radio access network, and the edge and core network components. This infrastructure enables the execution of the two main Italian use cases, IT-UC1 and IT-UC2, under controlled and realistic conditions, including also the use cases and potential technology enablers onboarded to the project via the two open calls (OC#1 and OC#2). Details on the use cases, APIs and ENVELOPE technology enablers using the Italian LL infrastructure will be presented in D4.2 ("ENVELOPE trial sites integration and pre-evaluation testing report").

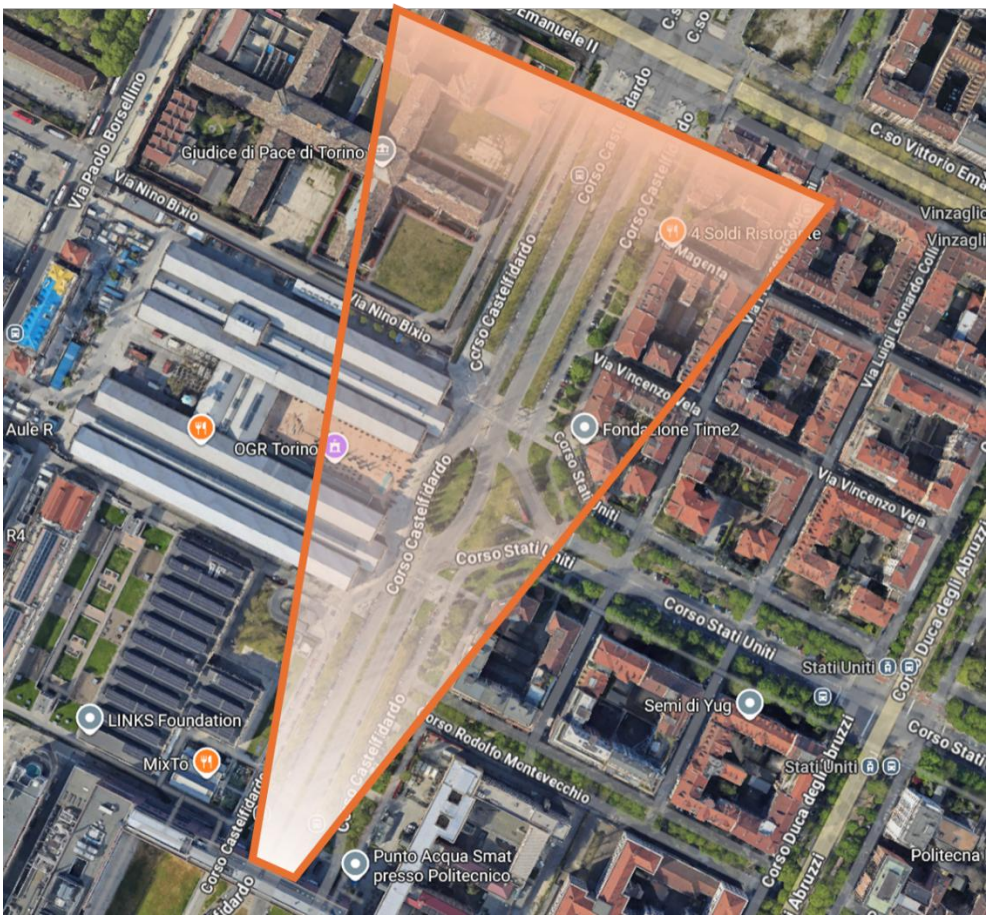


Figure 1: Italian Living Lab – geographical overview.

Figure 2: Italian Living Lab – trial site area.

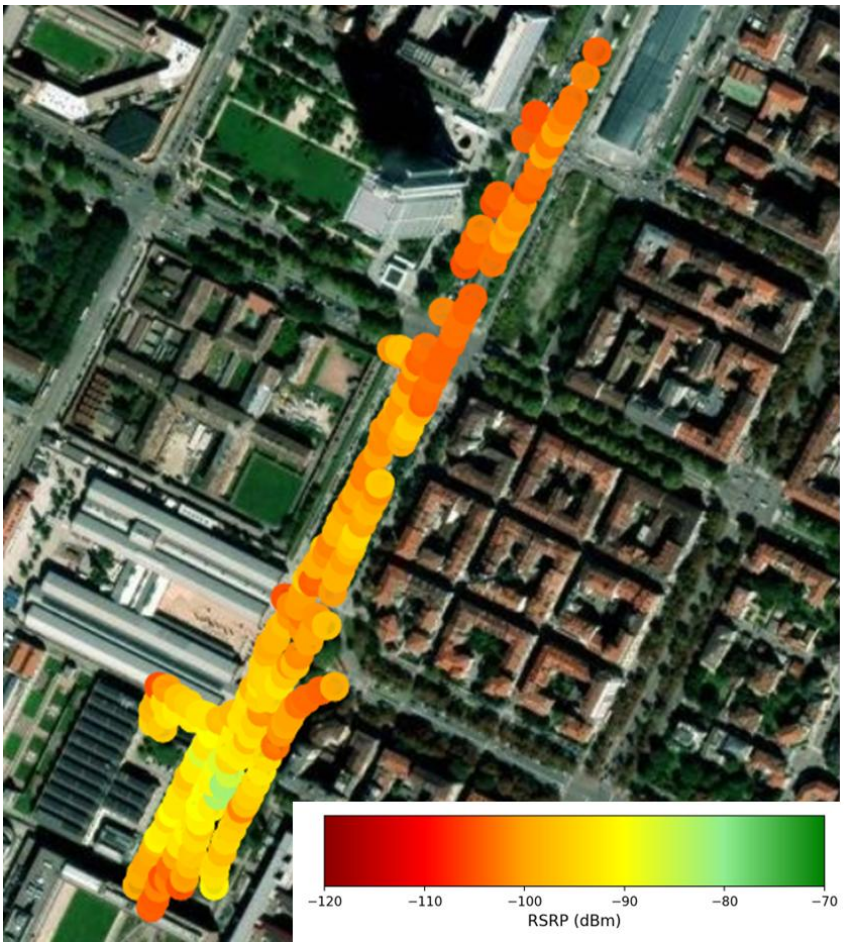


Figure 3: Italian Living Lab – Signal strength (RSRP) measured in the trial site area.

Trial area and coverage

The trial area is highlighted in Figure 1 and covers an area of approximately 0.2 km². It is composed of the main arterial road, consisting of two physically separate lanes per direction with speed limit of 50 km/h, two service roads with speed limit of 30 km/h, pedestrian sidewalks, and an inner courtyard. The area also includes a signalized intersection with a roundabout connecting the main road and the surrounding urban network.

This area provides a typical urban scenario providing a connectivity scenario characterized by the presence of several obstacles making the radio conditions on the UE side depending on having line-of-sight or non-line-of-sight conditions. In this area, heavy traffic of a crowded urban city is present. These conditions match the scenarios of the ENVELOPE use cases implemented in the Italian trial site.

The measured signal strength (RSRP) of the trial area is depicted in Figure 3. The current measurements are performed considering very low power setting of the gNB and a non-optimized setting of the antenna. The highest RSRP values (i.e., light yellow, green dots) are on the main arterial road and on the two service roads. In some specific points (i.e., dark orange, red dots), there is a degradation of the signal strength due to the presence of large and high metallic poles. In the integration activities of the Italian trial site, an improved setup of the antenna will be used to enhance the signal strength in the trial area of interest. Updates will be presented in Deliverable 4.2.

Trial site infrastructure

The following hardware infrastructure of the trial site can be identified:

- The Radio Access Network (RAN) infrastructure: it comprises the gNB, the Radio Unit, and the antenna.
- The edge and cloud infrastructure: it consists of a physical server that is used to deploy the Virtual Machines (VMs) providing the edge layer and the 5G core network

The RAN is located in a building of the Department of Control and Computer Engineering of Politecnico di Torino (Corso Castelfidardo 34/D). On the roof of the building, as shown in Figure 5, there are installed the Radio Unit, the antenna, and the GNSS antenna required for ensuring synchronization to the gNB. The gNB physical server is located on the third floor of the same building in a local datacenter room. A dedicated optical fiber connection has been installed for connecting the Radio Unit to the gNB. Similarly, dedicated cabling has been installed for connecting the GNSS antenna to the gNB.

The physical server for the edge and cloud infrastructure is installed in the main datacenter of Politecnico di Torino. It is located in the same campus as shown in Figure 4. The connection between the gNB and the edge/cloud physical server is provided by using the Local Area Network of the Politecnico di Torino.

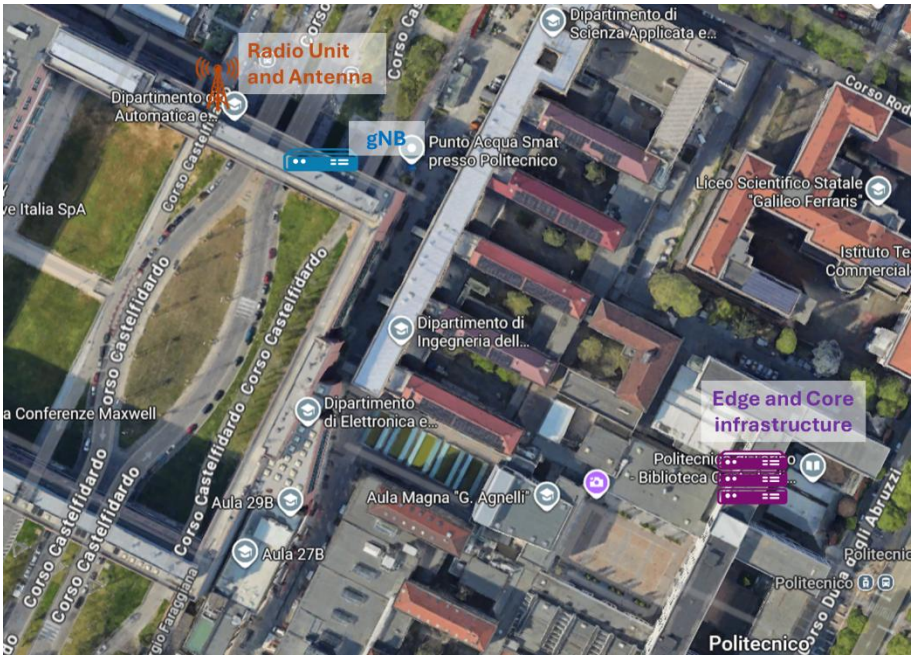


Figure 4: Italian Living Lab – infrastructure elements overview.

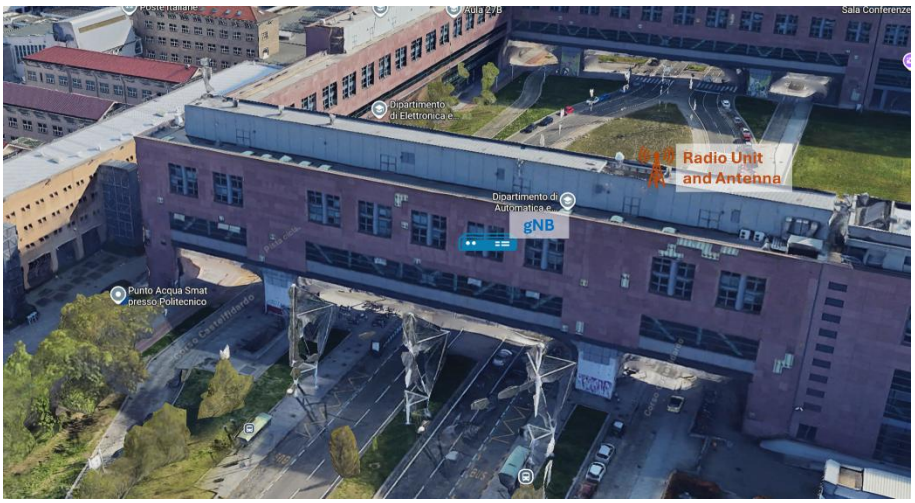


Figure 5: Italian Living Lab – Radio Unit, antenna, and gNB deployment details.

Radio Access Network (RAN) infrastructure

The RAN infrastructure comprises a gNB with associated a Radio Unit (RU) and an antenna. A telemetry and control service for the gNB is included in the system. The RAN infrastructure has been provided by Sma-RTy Italia SRL.

The gNB processing node is used for private 5G Standalone (SA) operation. It provides wireless NR connectivity to UEs and handles both control-plane and user-plane processing. The node is O-RAN compliant and implements a CU/DU split with CU and DU running on the same physical machine. The gNB supports 3GPP Release 16 and operates in Frequency Range 1 (FR1), including NR bands n77 and n78 in TDD. In the current configuration, it is deployed as a single NR

cell, supporting 4×4 MIMO and channel bandwidths of 20/40 MHz with central frequency of 4045 MHz (i.e., n77 band). The gNB interfaces with a 5G Core network through standard SA connectivity (N2/N3) and provides dedicated management access for configuration, monitoring, telemetry/KPIs and software maintenance. External timing synchronization (i.e., GNSS) is used. The deployed gNB is shown in Figure 6.

Figure 6: Italian Living Lab – deployed gNB.

The outdoor RU, that is also O-RAN compliant, provides the RF front-end functions and it is connected to the gNB through a dedicated single-mode optical fiber cable for ensuring a high-bandwidth fronthaul. In this specific case, the RU is O-RAN 7.2x split compliant, and it supports up to 4 TX / 4 RX chains (4×4 MIMO) and up to 20 W total output power (5 W per channel). Figure 7 illustrates the RU and the GNSS antenna for gNB synchronization.

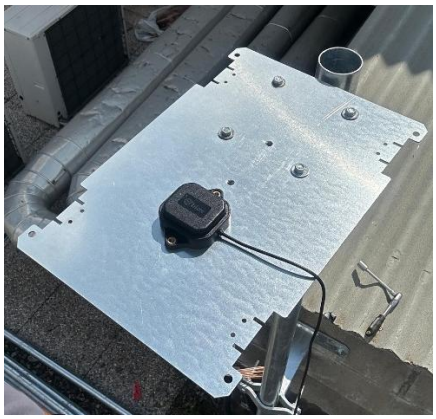


Figure 7: Italian Living Lab – deployed RU and GNSS antenna.

The deployed antenna is HUBER+SUHNER SENCITY® Urban 200, P/N 1399.17.0272. It is a compact outdoor directional panel antenna that is typically used for mid-band 5G deployments. It covers 3300-3800 MHz and 3800-4200 MHz, provides dual-slant ±45° polarization, and supports multi-port MIMO with N-female connectors. The key RF characteristics are 11 dBi gain and a half-power beam-width of 70° (horizontal) and 35° (vertical). The installed antenna is visible in Figure 8.

Figure 8: Italian Living Lab – deployed antenna.

A summary of the RAN configuration is introduced in Table 1.

Table 1: Italian trial site - summary of RAN configuration.

Parameter	
Cell Name	LINKS_TURIN_GNB1
nrPCI	0
rachRootSequence	1
nrTAC	1
arfcnDL/UL	669696
bSChannelBwDL/UL (MHz)	20/40
configuredMaxTxPower	4x5W
ssbFrequency	669696
tddSpecialSlotPattern	[6:3:5]
tddUlDlPattern	DDSUU
NR Numerology	μ = 1, 30 kHz subcarrier spacing
Frequency Band	N77

Central cloud and edge computing infrastructure

The central cloud and edge computing infrastructure is based on a physical machine server with the following characteristics. The server is a 2U rackmount system based on an AMD EPYC 9534

CPU (64 cores / 128 threads), paired with 256 GB of ECC registered DDR5 memory, for supporting virtualization and data processing workloads. A key element of the machine is the availability of an NVIDIA L4 GPU (24 GB), enabling hardware-accelerated workloads (e.g., AI/ML inference or other GPU-optimized processing) required for the experimentation activities. The network connectivity is provided through dual-port 10GbE, supporting high-throughput data exchange within the site.

The physical machine is located in the central datacenter of Politecnico di Torino, as shown in Figure 4. The server is integrated into an OpenStack environment that orchestrates virtual machines and manages the associated networking (virtual networks and connectivity), enabling controlled deployment, isolation, and operation of the cloud and edge components

Edge computing infrastructure

The edge infrastructure is made by two virtual edge servers exploiting the physical machine described above. The edge servers are configured to have direct connection to the 5G CN to ensure lowest latencies. An edge server exploits the NVIDIA L4 GPU. The second edge server is configured to use a NVIDIA A30 GPU that was already available in the computing infrastructure of LINKS and it was not acquired in the context of the ENVELOPE project. The two edge servers are configured to be used within the Kubernetes environment of the Italian trial site.

Cloud computing infrastructure

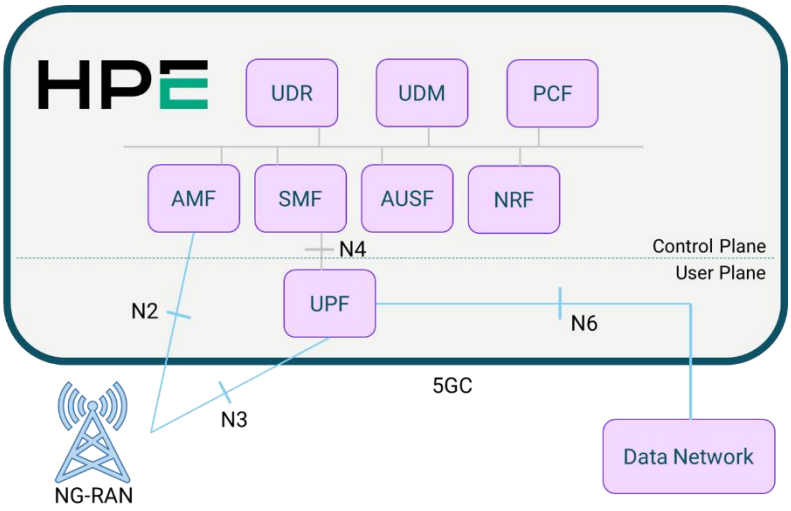
The cloud infrastructure is based on the physical machine described above and it is constituted by a virtual machine that is reachable from the 5G CN. This cloud server is used to host the cloud applications of the experiments. It is also located in the Kubernetes environment of the Italian trial site.

Core Network (CN) infrastructure

The 5G CN is deployed on dedicated server infrastructure in the central datacenter of Politecnico di Torino and is connected to the gNB via the Local Area Network of the campus (10 Gbit connection). The CN hosts the required 5G network functions and is configured in accordance with the ENVELOPE architecture defined in WP2 supporting the ENVELOPE innovations developed in WP3.

HPE is contributing to the project with its Private 5G Core Network, a fully virtualized Standalone system that follows 3GPP (starting from Rel. 16) and ETSI standards. It manages authentication, control signaling, QoS rules, mobility and traffic forwarding between the base stations and external networks. It includes all main network functions such as AMF, SMF, UPF, UDM, AUSF, UDR, PCF, NRF and NSSF. It also has an Equipment Identity Register that improves security by checking the IMEI or PEI of each device to decide if it can join the network. The system supports all standard interfaces from N1 to N6. Through the N5 interface, trusted third party applications can change network parameters and control end to end QoS for important and sensitive services.

The HPE 5G Core also provides REST APIs that follow the OpenAPI 3.0 specification. These APIs allow external OAM systems to configure network functions, deploy new settings, manage users, and adjust the system behavior automatically. They also allow the collection of real time and historical data, including resource usage, 3GPP signaling and traffic information, and metrics from virtual machines such as CPU, memory, disk and network. Monitoring data is exposed through Prometheus. All APIs are protected by TLS certificates, which ensure secure access, prevent data interception and confirm the identity of the server.



Spectrum Authorization and EMF Regulatory Compliance

The Italian trial site must obtain authorization from the Ministry of Enterprises and Made in Italy (MIMIT) to use a specific radio frequency. The requested frequency is within the FR1 range, specifically in the n77 band. The authorization procedure requires the submission of technical information regarding the configuration of the base station, as well as a description of the scope of the experimental activities. The use of the frequency is subject to a fee, and the amount depends on the bandwidth requested (e.g., for a bandwidth of 40 MHz, the fee is approximately EUR 3,600 per month) plus an additional amount of about EUR 1,000 to cover taxes and administrative costs related to the submission of the request.

Another activity is related to the request for ElectroMagnetic Field (EMF) regulatory compliance of the gNB. A formal request must be submitted to the regional agency responsible for EMF regulation. The request must include detailed information on the EMF characteristics and other technical parameters of the antenna, together with the technical description of the entire gNB installation. Additional documentation must be provided as well, including a map of the deployment area showing the surrounding buildings and their number of floors, as well as the antenna radiation diagrams and other relevant technical documentation required for the EMF assessment. The request must be submitted by an authorized technician, as only accredited professionals are allowed to access the online platform used to file the application.

2.1.2 Description of vehicles and end devices

This subsection describes the vehicles and end devices used at the Italian Living Lab, including the onboard hardware, sensing equipment, and user devices involved in the ENVELOPE trials. The focus is on the physical and logical integration of equipment on the vehicles, as well as the connectivity between sensors, onboard computing platforms, and network interfaces.

2.1.2.1 Vehicles used in the Italian Living Lab

The Italian Living Lab includes two experimental vehicles provided by Teoresi S.p.A. (Figure 9), used as the primary platforms for the execution of IT-UC1 and IT-UC2 trials. These vehicles act as mobile experimental nodes enabling testing and validation of connected, cooperative and automated driving functionalities under real urban mobility conditions.

Both vehicles are XeV Yoyo, model year My21 and My24 Pro respectively, as part of Teoresi technological background, with their onboard sensing and computing equipment that represent a consolidated experimental platform, used to validate cooperative, connected and automated driving use cases, with particular focus on vehicle connectivity, perception and data offloading to edge infrastructure. Below we present some pictures of the prototypes and the related components integrated in ENVELOPE.

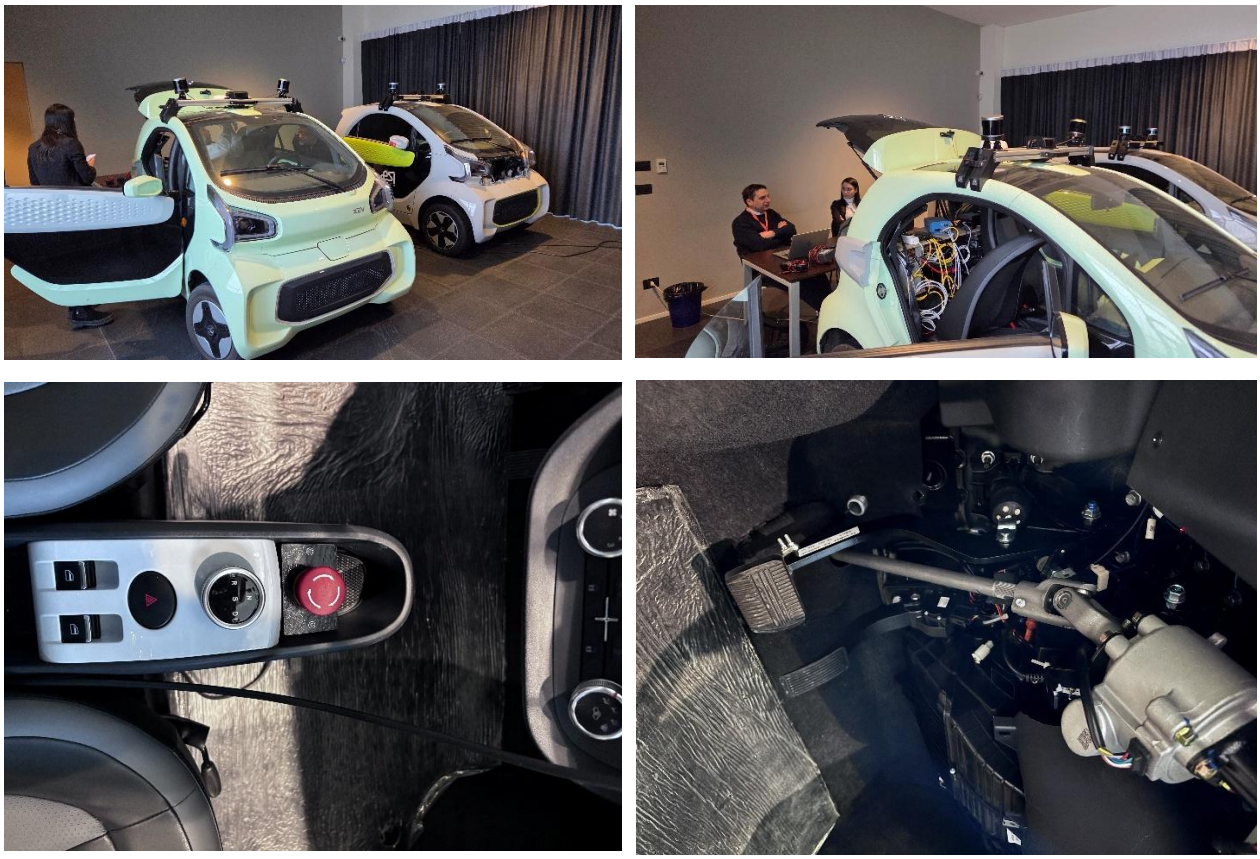


Figure 9: Pictures of the XEV prototypes.

LINKS OnBoard Unit

Each vehicle is equipped with an Onboard Unit (OBU) provided by LINKS. The OBU provides 5G connectivity and supports the execution of ENVELOPE-related software components on the vehicle side. The OBU interfaces with the 5G Radio Access Network and enables communication between the vehicle and the ENVELOPE network and edge infrastructure. The OBU is connected to the vehicle network via an Ethernet interface.

The OBU is a research-driven proof-of-concept device designed to support experimentation in CCAM-related use cases, based on COTS components integrated by LINKS. It is built around embedded devices using the NVIDIA® Jetson Xavier™ NX platform, which provides an ARM64

architecture with an integrated GPU and native 5G connectivity to enable edge processing and low-latency V2X-oriented services. The OBU integrates a Telit FN990 5G modem for cellular communications, a u-blox F9P GNSS receiver with RTK correction support for high-accuracy positioning, one Ethernet interface for connection to the vehicle network and/or external sensors, and a Wi-Fi hotspot to interact with smartphones or wireless sensors. Figure 10 shows the standalone LINKS OBU (highlighted in red) deployed on TEO's prototype vehicle.

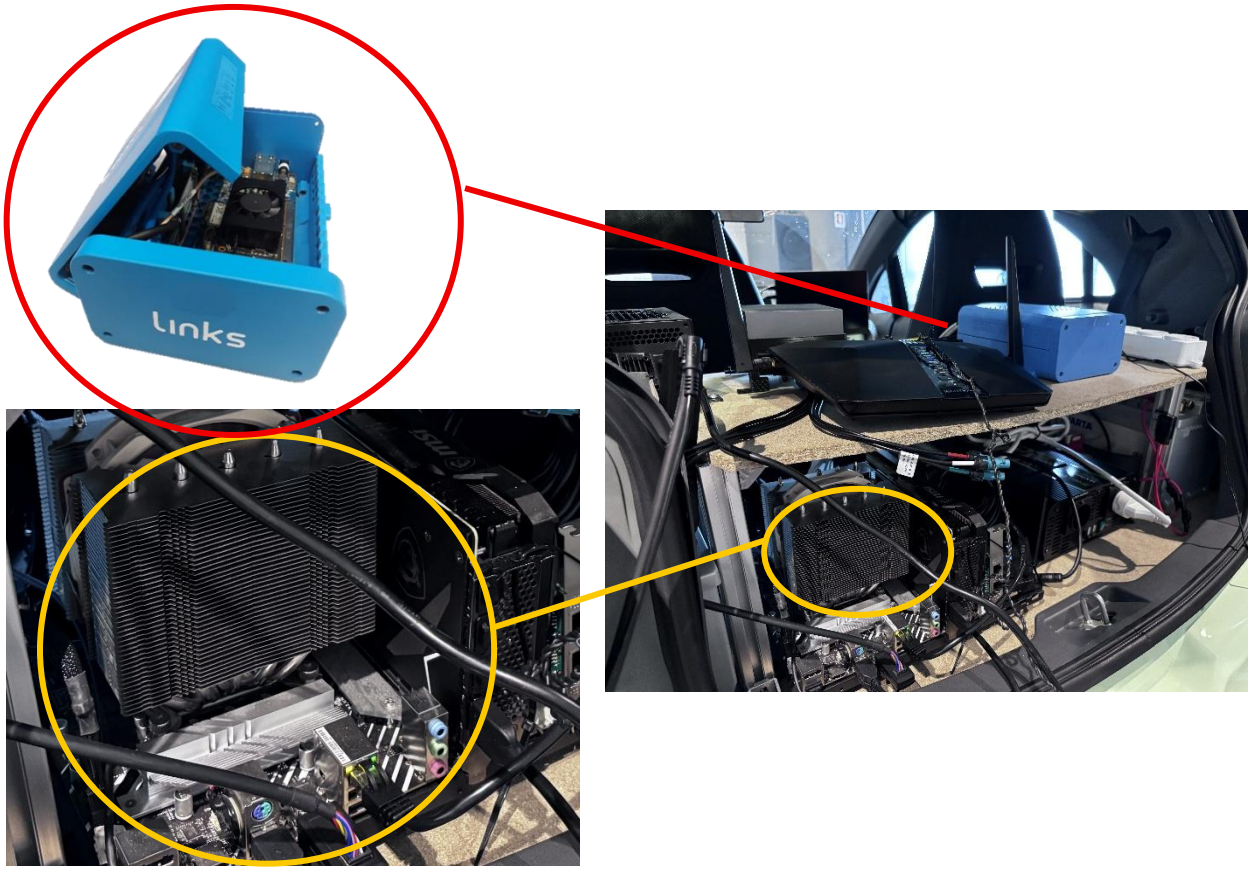


Figure 10: Italian Living Lab – LINKS OBU deployed on the TEO's vehicle (red circle) and the placement of the on-board computing platform (yellow circle).

Onboard computing platforms

The vehicles are also equipped with onboard computing platforms used to host vehicle-side applications, data processing components, and, where applicable, ENVELOPE enablers requiring local execution.

The work carried out on the vehicles inside ENVELOPE was structured into two distinct abstraction layers:

- The low-level layer encompasses mechanical and electrical integration, together with the development of the drive-by-wire control system responsible for direct actuation of the vehicle's throttle, steering, and braking subsystems.
- The high-level layer focuses on the configuration of the perception suite and the implementation of navigation and decision-making algorithms.

To enable reliable communication between these layers, a dedicated software interface was developed. This interface operates over a serial link and handles message translation, serialization and routing between the two endpoints, ensuring decoupling and interoperability between low- and high-level components. Despite differences in the sensing layout, both vehicles share the same onboard computation architecture: each platform is equipped with a high-performance onboard computer with a dedicated GPU and an NVIDIA Jetson AGX Orin, used for real-time perception and object detection, and a custom PCB, that was designed for drive-by-wire control.

In more detail, the characteristics of the hardware are as follows:

Jetson AGX Orin 64GB

- 2048-core NVIDIA Ampere architecture GPU with 64 tensor cores
- 2x NVDIA v2.0
- 12-core Arm® Cortex®-A78AE v8.2 64-bit CPU
- 64GB 256-bit LPDDR5
- 64GB eMMC 5.1
- PVA v2.0

Power

- Voltage input 5V, 7V-20V
- Module Power: 15W - 60W

From a software perspective, the platforms run the Ubuntu operating system and adopt the Autoware autonomous driving framework, built on top of ROS2, which enables distributed communication, real-time data handling and the integration of perception, localization, planning and control modules.

Sensors and data sources

The vehicles are equipped (TEO background) with a set of sensors (Figure 11 and Figure 12) and data sources supporting the ENVELOPE use cases. These may include, but are not limited to, cameras, GNSS receivers, inertial sensors, and vehicle internal data sources.

Two different perception suites have been integrated into the ENVELOPE vehicles. For the My21 model, two Robosense RS-LiDAR-16 sensors together with a Robosense M1 were deployed to capture the point-cloud representation of the navigation environment. In addition, three NileCAM25 cameras were installed at the front of the vehicle. For the My24 model, an upgraded sensor suite was considered. The LiDAR configuration was enhanced with two Robosense Helios32P sensors alongside a Robosense M1. The camera setup was also redesigned to improve coverage and safety, consisting of a front-facing stereo camera, two side cameras, and a rear camera to provide broader environmental visibility around the vehicle.

Within the ENVELOPE project, TEO integrated both sensor suites into the vehicles via GMSL2 and Ethernet communication protocols. The cameras are managed by a Jetson Orin computing platform, whereas the LiDAR sensors are interfaced and controlled through an onboard PC.

This way, we can have greater control over the computational resources without having bottlenecks in data transfer from one board to the other.

To provide the vehicle position, a GNSS with IMU and RTK is used.

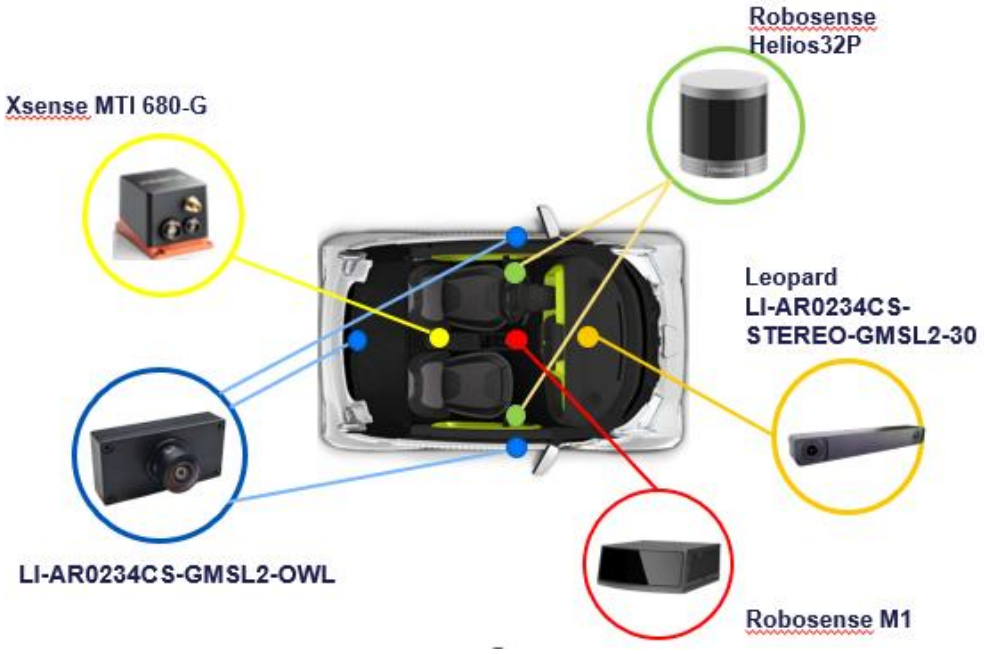


Figure 11: Sensors' placement.

Figure 12 shows part of the above depicted sensors (i.e., Robosense M1 and Robosense Heleios32P) and installed equipment for sensing on the real vehicle. The rotating LiDARs, the solid-state LiDAR, the stereo camera, and the rear camera can be observed.

Figure 12: Mounted sensors.

Figure 13 showcases an example output of the perception system and the respective vicinity detections based solely on LiDARs.

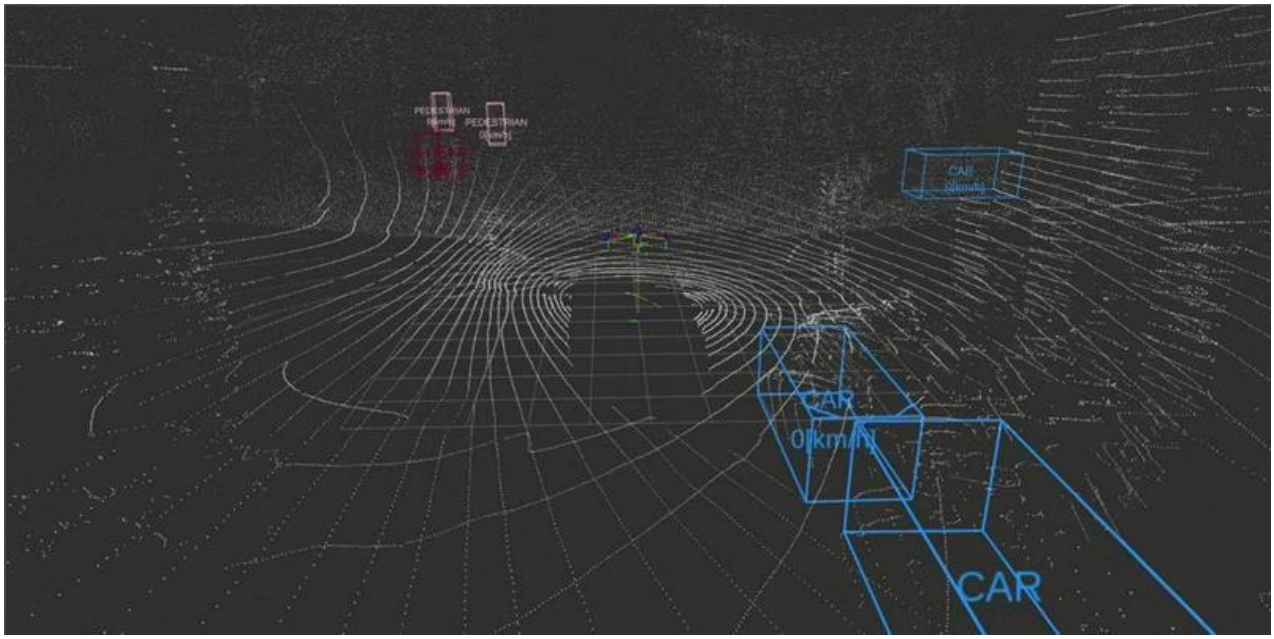


Figure 13: Perception system and detection from LIDARs.

In-vehicle integration and interconnection

All onboard components are interconnected through the vehicle's internal communication infrastructure. The overall architecture follows a modular design where perception, decision-making, and actuation are logically separated while remaining tightly synchronized through standardized communication interfaces.

A key element of the in-vehicle integration is the Drive-by-Wire Electronic Control Unit (DBW-ECU), implemented as a custom-designed controller board.

The DBW-ECU acts as the bridge between the high-level autonomous driving stack and the physical vehicle actuators, translating motion commands generated by the onboard controller PC into electrical signals that actuate.

In addition to actuation control, the DBW-ECU serves as a gateway to the vehicle's CAN bus, enabling acquisition of vehicle status signals (speed, steering angle, battery status). This gateway functionality allows higher-level software components and the OBU to access vehicle telemetry without directly interfacing with safety-critical subsystems, improving modularity and security.

2.1.2.2 Other UEs and end devices

An additional UE deployed at the Italian trial site is a Google Pixel 9. This smartphone is used to test the 5G network with a commercial off-the-shelf device, in order to assess performance differences with respect to the OBU. Figure 14 shows the Google Pixel 9 available in the Italian trial site.

Figure 14: Pixel 9 smartphone of the Italian trial site.

2.2 Data collection & management

The Italian trial site architecture is introduced in Figure 15, in which position of data loggers is identified. Data Logger 1 (DL1) is placed at the Experimentation as a Service (EaaS) module and it is used for collecting and providing data that are used for computing the metrics related to EaaS KPIs. A second data logger (DL2) is located within the exposure framework for collecting the metrics related to the ENVELOPE enablers (i.e., QoD, Edge Cloud, and Devices in Area enablers). The data for metrics about service deployment and edge handover are collected by Data Logger 3 (DL3) that is at the Management and Orchestration layer.

In the Italian trial site, a common, software-integrated approach is used for the collection of data. Instead of relying solely on external probes, the metrics required for KPI evaluation are computed directly within the software code of the respective architectural components (such as the EaaS modules or the enablers).

Once computed, these components expose the data via dedicated REST endpoints. A Prometheus server instance is then configured to periodically scrape these endpoints to retrieve the time-series metrics. The final KPIs for evaluation are derived directly from the data aggregated by this Prometheus server instance.

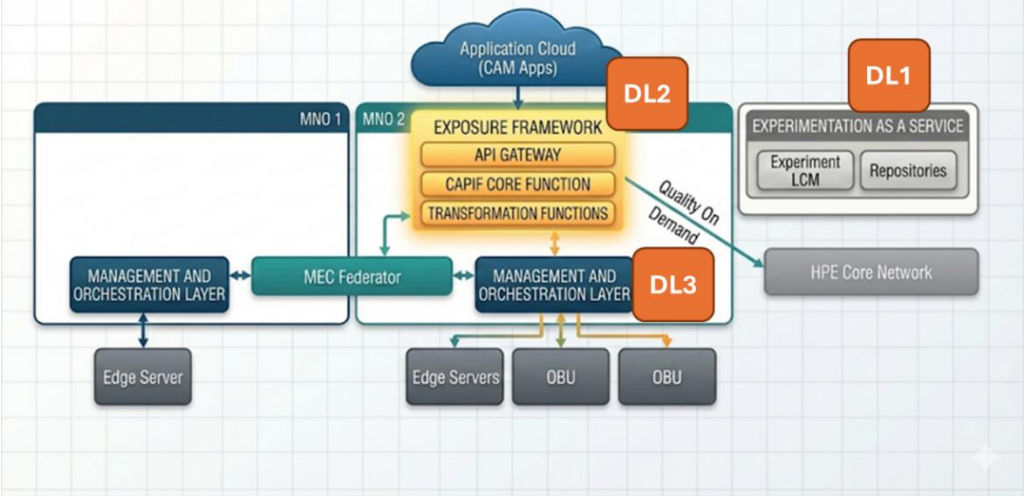


Figure 15: Italian trial site architecture

The data collection performed in the Italian trial is described in Table 2 to Table 6. Each table provides an overview of the data collection per each component, respectively, the EaaS module, the Edge Cloud enabler, the Devices in Area enabler, the QoD enabler, and the Zero Touch Management service.

Table 2: Italian trial site – EaaS data collection.

Item	Description
KPI ID	KPI_EaaS_Availability, KPI_EaaS_AppOnboardingTime, KPI_EaaS_ExperimentStartupTime, KPI_EaaS_ExperimentStopTime
Data logger / PoP (location in architecture)	EaaS data logger – DL1
Log data type	EaaS components availability, application package onboarding time, experiment start and stop duration times
Log format	Time-series Prometheus metrics
Logging event	Periodic logging, application package onboarding, experiment start and stop.
Sampling policy	Periodic (1 s), and event driven
Logging storage	Prometheus server
Log data fields	PortalAvailable, AppRepositoryAvailable, ExperimentLcmAvailable, MonitoringAvailable AppUploadingTimeDuration, AppProcessingTimeDuration, AppSouthboundTimeDuration, ExperimentStartupTimeDuration, ExperimentStopTimeDuration

The EaaS related metrics are computed as follows:

- **PortalAvailable, AppRepositoryAvailable, ExperimentLcmAvailable, MonitoringAvailable (binary):** Computed via background health-check functions.
- **AppUploadingTimeDuration (ms):** Computed by recording a start timestamp when the HTTP upload request begins and an end timestamp when the file is fully written to the database. The code exposes the difference (End - Start).
- **AppProcessingTimeDuration (ms):** Computed by measuring the duration from when the application repository starts downloading the app package from the database, validating its content, and uploading each app package artifact to the database.
- **AppSouthboundTimeDuration (ms):** Computed by measuring the duration from when the application repository makes the HTTP request to onboard the application to the southbound plugin until it is complete.

- **ExperimentStartupTimeDuration (ms):** The code records a timestamp when the "start experiment" API call is received and a final timestamp when all underlying resources (through the southbound plugin) report a "running" state.
- **ExperimentStopTimeDuration (ms):** Computed as the time difference between receiving the "stop" command and the successful termination and cleanup confirmation of all experiment resources from the southbound plugin.

Table 3: Italian trial site – Edge data collection.

Item	Description
KPI ID	KPI_Enab_availability, KPI_Enab_Edge_ServiceDeploymentTime, KPI_Enab_Edge_HandoverTime, KPI_Enab_Edge_HandoverSR
Data logger / PoP (location in architecture)	Exposure framework data logger – DL2 MANO data logger – DL3
Log data type	Edge enabler availability, edge service deployment time, edge application handover time and success rate
Log format	Time-series Prometheus metrics
Logging event	Periodic logging, service deployment request, edge application handover
Sampling policy	Periodic (1 s), and event driven
Logging storage	Prometheus server
Log data fields	EdgeEnablerAvailable, EdgeServiceDeploymentTimeDuration, EdgeHandoverTimeDuration, EdgeHandoverSuccessRate

The Edge Enabler related metrics are computed as follows:

- **EdgeEnablerAvailable (binary):** A binary gauge computed by a continuous heartbeat function that verifies the edge enabler's readiness.
- **EdgeServiceDeploymentTimeDuration (ms):** Computed by capturing a start timestamp upon accepting a deployment payload and an end timestamp when the Kubernetes API confirms the pods are fully active.
- **EdgeHandoverTimeDuration (ms):** Computed within the handover logic by tracking the exact timestamp the source edge node initiates the state transfer, up to the millisecond the target edge node confirms the application is migrated.
- **EdgeHandoverSuccessRate (percentage):** Computed in code by maintaining two counters: total handover attempts and successful handovers. The software divides successful handovers by total attempts over a rolling window and exposes the percentage/ratio.

Table 4: Italian trial site – Devices in Area data collection.

Item	Description
KPI ID	KPI_Enab_availability, KPI_Enab_DL_Effect-Latency, KPI_Enab_DL_DevicesInAreaQuer-ySR
Data logger / PoP (location in architecture)	Exposure framework data logger – DL2
Log data type	Devices in Area enabler availability, Devices in area query response time, Devices in Area success rate
Log format	Time-series Prometheus metrics
Logging event	Periodic logging, Devices in Area Query, De-vices in area query and subscription success rate
Sampling policy	Periodic (1 s), and event driven
Logging storage	Prometheus server
Log data fields	DevicesInAreaAvailable, dl_api_request_dura-tion_seconds, dl_effective_latency_ms, dl_queries_total, dl_queries_successful

The Device in Area Enabler related metrics are computed as follows:

- **DevicesInAreaAvailable (binary):** A gauge set to 1 or 0 based on the internal health checks of the enabler
- **dl_api_request_duration_seconds (s):** Computed from the API Gateway that logs the timestamp when an API request arrives and subtracts it from the timestamp when the HTTP response is sent back to the client.
- **dl_effective_latency_ms (ms):** Computed by tracking the time it takes for the enabler to query the underlying 5G core network for location data and receive the answer, excluding internal API overhead.
- **dl_queries_total (number):** An integer counter that increments by +1 every time a device location query hits the endpoint.
- **dl_queries_successful (number):** An integer counter that increments by +1 only when a query successfully resolves and returns valid location data (HTTP 2xx response).

Table 5: Italian trial site – QoD data collection.

Item	Description
KPI ID	KPI_Enab_availability, KPI_Enab_QoD_SuccessRate, KPI_Enab_QoD_QoSProfileEstablishmentTime, KPI_Enab_QoD_QoSSessionRetainability, KPI_Enab_QoD_QoSAccuracy
Data logger / PoP (location in architecture)	Exposure framework data logger – DL2
Log data type	QoD enabler availability, QoD enabler Success Rate, QoD Enabler QoS Profile Establishment Time, QoD Enabler QoS Session Retainability, QoD QoS Accuracy
Log format	Time-series Prometheus metrics
Logging event	Periodic logging, QoD request
Sampling policy	Periodic (1 s), and event driven
Logging storage	Prometheus server
Log data fields	QoDAvailable, QoDSuccessRate, QoDQoSProfileEstablishmentTime, QoDQoSessionRetainability, QoDQoSAccuracy

The QoD Enabler related metrics are computed as follows:

- **QoDAvailable (binary):** Computed via a background health-check routine to the QoD Enabler
- **QoDSuccessRate (percentage):** Calculated as the ratio of successful QoS profile applications to the total number of QoD requests received.
- **QoDQoSProfileEstablishmentTime (ms):** The duration (in milliseconds) between the QoD API receiving a request and the 5G core acknowledging the applied QoS policy.
- **QoDQoSSessionRetainability (s):** The total continuous uptime of a QoS session, tracked from its activation timestamp to its termination.
- **QoDQoSAccuracy (percentage):** An accuracy score or percentage calculated by measuring the deviation between the requested QoS parameters (e.g., guaranteed bit rate, latency) and the actual performance telemetry delivered by the network.

Table 6: Italian trial site – Zero touch management data collection.

Item	Description
KPI ID	KPI_ZT_NativeAIframeworkServiceTime, KPI_ZT_NativeAIframeworkServiceAccuracy, KPI_ZT_Decision&ProvisioningTime
Data logger / PoP (location in architecture)	MANO data logger – DL3

Log data type	ZTM service, decision, and provisioning time duration and ZTM service accuracy
Log format	Time-series Prometheus metrics
Logging event	Zero touch request from a user
Sampling policy	Event driven
Logging storage	Prometheus server
Log data fields	NativeAIframeworkServiceTime, NativeAIframeworkServiceAccuracy, Decision&ProvisioningTime

The Zero Touch Management related metrics are computed as follows:

- **KPI_ZT_NativeAIframeworkServiceTime:** The Zero-touch action service time includes the time to translate the high level natural language to TMF representation, the time to call appropriate APIs reconfiguring the app (Decision engine as part of T3.5) as well as the time for the actual reconfiguration. It will be calculated by using timestamps in the Intent Engine.
- **KPI_ZT_NativeAIframeworkServiceAccuracy:** This KPIs checks whether a specific request gets translated correctly (from high level natural language to specific values) and served appropriately. It will be measured by identifying which functions are called in the back-end of the Intent Engine as a response to specific intents (and verify that the correct functions are called).
- **KPI_ZT_Decision&ProvisioningTime:** This KPI is related to the time it takes to perform a Zero-Touch decision (not including the front-end elements). It can be calculated using timestamps in the Intent Engine in order to isolate only the time needed for the actual reconfiguration.

3 Dutch Living Lab

3.1 Living Lab Description

3.1.1 Description of the infrastructure

The Dutch Living Lab is located at the Automotive Campus in Helmond, The Netherlands. This infrastructure will enable the execution of the three main Dutch use cases, DT-UC3, DT-UC4, DT-UC5 and the projects of OC#1 and OC#2, under controlled and realistic conditions. All radio and server infrastructures have been newly acquired and deployed in the context of the ENVELOPE project. Details on the use cases, ENVELOPE APIs, and technology enablers use of the infrastructure will be detailed in D4.2 ("ENVELOPE trial sites integration and pre-evaluation testing report").

Trial area and coverage

The trial area is highlighted in Figure 16. The radio coverage configured covers the whole Automotive Campus area of approximately 1000x250 m². It is composed of multiple road segments within and around the campus, including intersections, and access roads.

This area provides a wide variety of mobility and connectivity scenarios with speed limit ranging from 15-30 Km/h in its urban and local access roads. It includes the possibility to reserve and close certain areas to support use cases that require automated and tele-operated vehicle maneuvers. Regular traffic with cars, trucks, and buses are present on a daily basis that can be useful for testing use cases under different patterns of road traffic mobility.

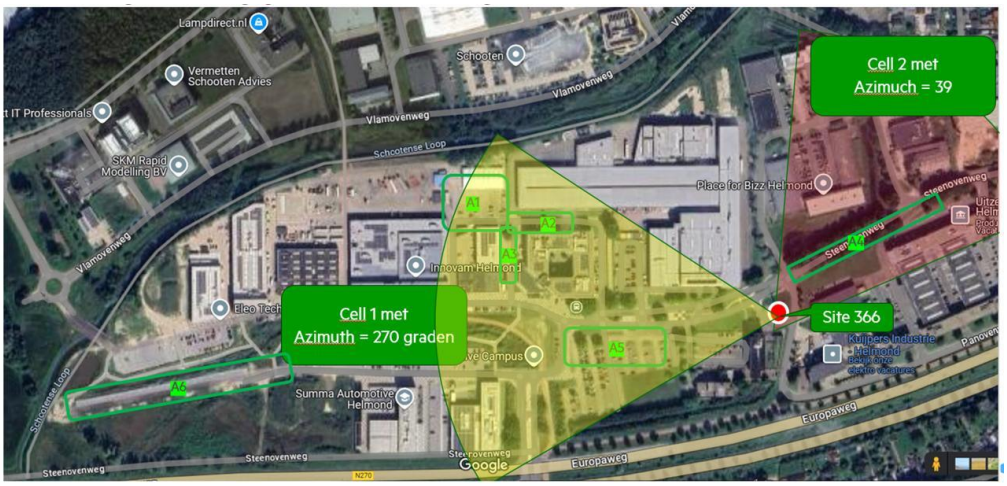


Figure 16: Radio site at Helmond automotive campus with two cells in the lowest position in the tower.

Figure 17 shows a drive-test map with measured RSRP over two adjacent 5G cells: Cell 1 (West) and Cell 2 (East). These drive-test results indicate strong signal strength level across the measured route, with most RSRP samples falling within the ≥ −80 dBm to −50 dBm range (green), which corresponds to good to excellent signal strength. Only limited sections exhibit weaker levels approaching −110 dBm (yellow) in areas without direct light-of-sight (NLOS), e.g., behind obstacles

(buildings). Overall, the measurements suggest that both cells provide adequate coverage along the driven path making it suitable for performing a wide range of use cases.

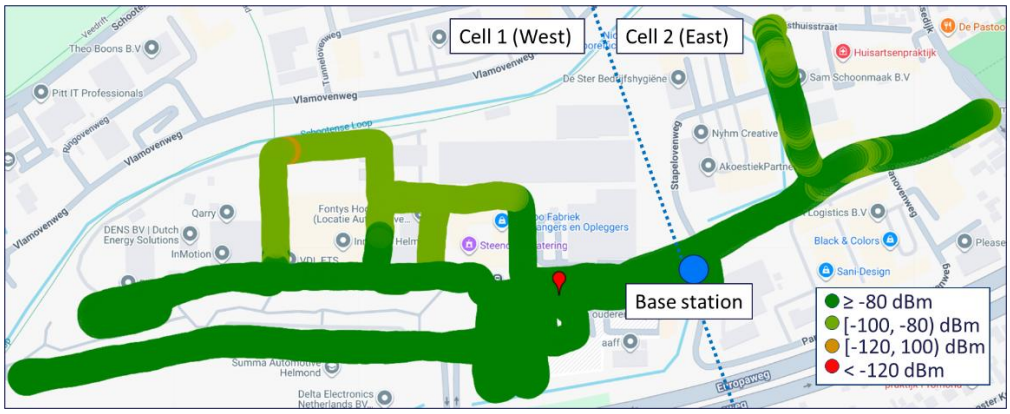


Figure 17: Signal strength (RSRP) measured in the trial site area.

Radio Access Network (RAN) infrastructure

The Dutch Living Lab RAN consists of a single 5G gNB operating in a standalone configuration as shown in Figure 18. The gNB is a Nokia AirScale solution. The base station model is identified as MRBTS-3002, utilizing the ASOFA AirScale Core (SMOD) for control and baseband processing. The system runs on software version SBTS24R3. This Nokia AirScale release is typically aligned with 3GPP Release 17 features and enhancements.

The gNB operates in TDD (Time Division Duplex) mode. The TDD frame configuration uses a 2.5ms periodicity with a pattern of 3 Downlink slots, 1 Special slot, and 1 Uplink slot (DDDSU), which is optimized for high-capacity downlink traffic. The setup consists of two NR (New Radio) cells. Both operating in the N77 band, with the center frequency at 3950 MHz. Each cell is configured with a channel bandwidth of 100 MHz.

The radios consist of two AWHQ3 AirScale Micro 4T4R n77 80W systems. Both are mounted at 20 meters height with a panel antenna mounted in front of the radio. The cells have an azimuth of respectively 270 and 39 degrees. Downward tilt is set at 8 degrees.

For Synchronization the gNB uses a multi-constellation GNSS receiver supporting Galileo, GLONASS, and BeiDou.

Table 7: summary of RAN configuration.

Parameter	West cell	East cell
Cell Name	NRCELL-152 (HEL-MOND_GN1_NR37_S2)	NRCELL-151 (HEL-MOND_GN1_NR37_S1)
nrPCI	551	552
rachRootSequence	60	30
nrTAC	491	491

arfcnDL/UL	663334	663334
bSChannelBwDL/UL (KHz)	40.000	40.000
configuredMaxTxPower	43.0 dBm	43.0 dBm
ssbFrequency	663360	663360
tddSpecialSlotPattern	SSP6	SSP6
tddUlDlPattern	3D1S1U	3D1S1U
NR Numerology	μ = 1, 30 kHz subcarrier spacing	μ = 1, 30 kHz subcarrier spacing
Frequency Band	N77	N77

Figure 18: gNodeB in the Dutch trial site.

Central cloud and edge computing infrastructures

Edge computing infrastructure:

The edge is deployed at the KPN Metro core in Helmond. This serves as an aggregation point for all connections in the KPN network. The edge infrastructure is built upon a high-availability architecture consisting of two identical, enterprise-grade HPE ProLiant DL360 Gen9 rack servers. These units are optimized for high-density virtualization and serve as the physical compute nodes for the OpenNebula cloud management platform.

- **Compute Performance:** The dual Intel Xeon E5-2637 v3 processors provide a high base clock frequency of 3.50GHz.
- **Memory Architecture:** Each server is equipped with 256GB of DDR4 Error-Correcting Code (ECC) memory. The configuration utilizes 8 out of 24 available DIMM slots, allowing for future scalability (up to 1.5TB per node).

- **Storage Reliability:** Redundancy is managed at the hardware level via the Smart Array Gen9 P440ar controller, presenting a unified 2.4TB logical volume to the Ubuntu OS.

The edge computing setup (Figure 19) was established by repurposing existing hardware, ensuring a zero-cost capital investment for the ENVELOPE project while achieving a high-performance deployment.

Figure 19: Edge hardware, located at the KPN Metro Core.

Central cloud computing infrastructure:

The cloud-based component of the ENVELOPE project is deployed within the Amazon Web Services (AWS) ecosystem, utilizing a highly structured Infrastructure as Code (IaC) approach via Pulumi. This environment is designed to host a distributed Kubernetes (K8s) cluster, specifically engineered to support 5G Core Network Functions (NFs) through advanced multi-homed networking.

- **Virtualized Compute Architecture:** The deployment consists of a multi-node Kubernetes cluster, utilizing AWS EC2 t2.medium instances. The architecture is partitioned into dedicated Control Plane and Worker nodes, ensuring a clear separation between cluster orchestration and the execution of high-performance containerized workloads.
- **Multi-Homed Networking & Traffic Isolation:** Each instance is configured with multiple Elastic Network Interfaces (ENIs) across three distinct subnets:
    - **Management Subnet:** Facilitates cluster orchestration, SSH access, and administrative traffic.
    - **N6 Subnet:** Dedicated to providing the gateway to external data networks.
    - **NGAP Subnet:** Specialized for 5G control and user plane signaling (AMF/UPF communication).
- **Dynamic Network Function Mapping:** The infrastructure supports the programmatic assignment of specific 5G roles (e.g., N6, AMF, and UPF) to worker nodes. This is achieved through a dynamic IP mapping logic that injects secondary private IPs into the ENIs, allowing for flexible scaling and re-allocation of network functions without hardware re-configuration.

- **Automated Provisioning & Configuration:** Every node is automatically initialized using **Cloud-init** and **Netplan** templates. This ensures that complex routing tables and interface definitions are applied at boot time, creating a consistent, "zero-touch" deployment environment that is both reproducible and scalable.
- **Storage & Performance:** Each compute node is backed by 64GB gp3 EBS volumes.

Core Network (CN) infrastructure

The TNO Open5GS core extends the 3GPP release 17 compliant version of Open5GS (commercial NextEPC version) with additional components such as NEF, NWDAF, as well as modifications to AMF (to support NGAP Location reporting) and to PCF and other NFs to support UE Route Selection Policy (URSP).

Figure 20 shows the architecture for the edge and central cloud infrastructures to support the seamless deployment of applications across two heterogeneous K8s cluster environments: K8s Cluster 1 at the Helmond edge (with OpenNebula) and K8s Cluster 2 in the central cloud (with Amazon AWS). Each cluster contains a master node and multiple worker nodes used to deploy components of the TNO Open5GS core and end-user experiment applications through the EaaS platform (also see D3.1). In the edge K8s cluster 1, the main TNO Open5GS core components are deployed (AMF, NWDAF, SMF, PCF, UPF, etc.) while in the central cloud K8s cluster 2 another instance of the UPF network function is also present to enable UE traffic re-routing between edge and central cloud networks.

Both clusters expose 5G core interfaces – specifically the NGAP (N2, N3) and N6 data paths – shown as horizontally aligned blue channels connecting edge and cloud components. The two clusters are mutually reachable through Liqo[1], an open-source tool that enables multi-cluster resource sharing and workload federation seamlessly across the edge-cloud continuum. Traffic from the radio access network terminates at the edge cluster, while both clusters maintain external connectivity to the Internet for management and service exposure. By connecting both the K8s data plane and 5G interfaces across the two clusters, this design allows for the offloading of applications and their network traffic to either edge or central cloud clusters.

[1] https://liqo.io/

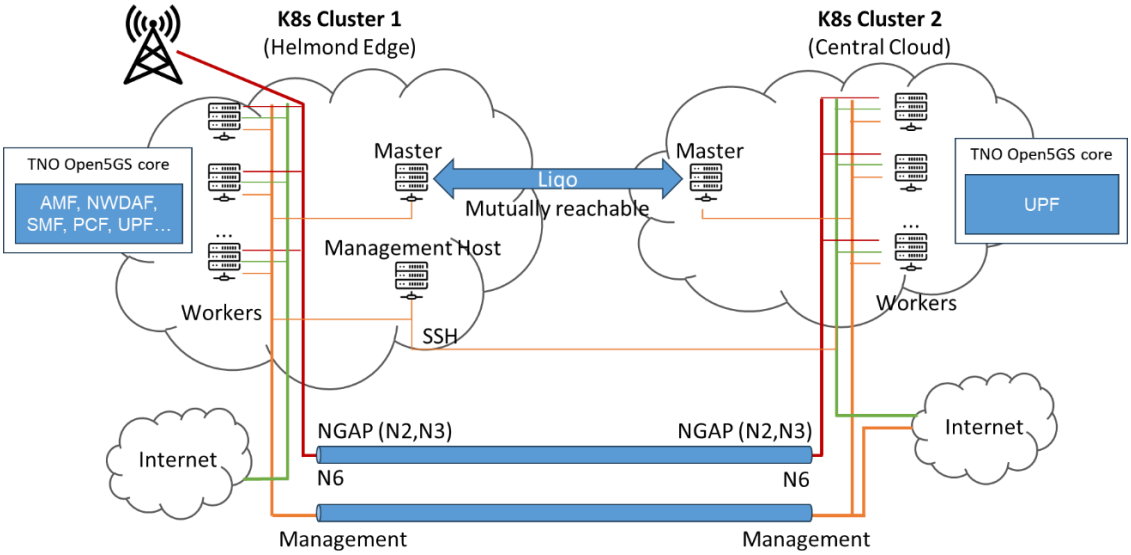


Figure 20: Architecture of edge & central cloud infrastructures.

Other infrastructure equipment

RSU

The Commsignia RSU running the proprietary V2X-stack and other software components enabling the Dutch site use cases is a prototype device based on a RISC architecture. It embeds a Quectel RM520N 5G modem for compatibility with the trial site's use case requirements. Its main role is to integrate smart sensors and allow the distribution of camera detections (e.g., pedestrians) using standardized V2X messages for use in UC4 and UC5. The exact deployment location of the RSU (and camera) will be determined later in the project during the trials. Potential areas to deploy the RSU include A3 or A5 as shown in Figure 16.

Camera

The smart sensor used for the demonstrations is a Bosch Dinon 7100i IR bullet camera. The 4MP wide-angle camera is ideal for traffic monitoring and supporting use cases requiring infrastructure-based passive VRU protection. A setup with the Commsignia RSU is shown in Figure 21.

Figure 21: Commsignia RSU with smart camera.

Stationary Perception Set (SPS) – Teleoperation dedicated equipment

The Dutch Living Lab employs a high-performance, stationary perception set supplied by Roboauto. This system serves as the primary infrastructure-side node for the execution of the Dt-UC5 (teleoperation) test iterations. Its core function is to provide an external environmental model to the HAN Digital Twin (DT), specifically designed to supplement or replace onboard vehicle sensing in scenarios where local perception is degraded or has failed. By detecting and tracking dynamic objects from an infrastructure perspective, the system ensures the teleoperator maintains a reliable "ground truth" of the operational environment. The exact deployment location of the Stationary Perception Set (SPS) will be determined later in the project during the trials. Potential areas to deploy the SPS include the A5 as shown in Figure 16.

Computing unit on SPS:

The processing architecture is centered around an industrial-grade computing platform. This unit serves as the high-level perception engine for the site.

Integration: The computer is interfaced with the sensor suite (cf. next paragraphs) via high-speed Ethernet to ensure low-latency data ingestion.

Data Processing: The unit executes advanced algorithms for point-cloud processing, including filtering, clustering, and object tracking. It transforms raw spatial data into actionable telemetry.

Communication: Following local processing, the system encapsulates geometric object data into MQTT-formatted payloads. These messages are transmitted via a physical Ethernet interface to the Sierra Wireless XR90 5G gateway. This process ensures that processed situational data is formatted into a lightweight telemetry stream optimized for wide-area network transmission.

Sensors on SPS:

Livox HAP (TX): The primary sensing modality for this deployment is the Livox HAP (TX) LiDAR shown in Figure 22, with its specifications depicted in Table 8. This is an automotive-grade sensor designed for wide-area surveillance and high-reliability object detection.

Technology: The Livox HAP utilizes a non-repetitive scanning pattern to provide high-density coverage of the environment.

Data Output: The sensor generates a 3D point cloud of the monitored area. The specific performance metrics and scanning parameters for this unit are detailed in Table 8. In this setup, dynamic actors (vehicles, pedestrians, etc.) appear as distinct clusters of points within the scene.

Connectivity: The sensor is networked via a dedicated Ethernet interface, providing a continuous stream of raw spatial data to the onboard processing unit.

Figure 22: LiDAR sensor for DT-UC5 focused on tele-operation.

Description	Values
Detection range	150m @10%
FOV	120° x 25°
Equivalent point cloud density	144 lines
Angular resolution	0.18° x 0.23°
Point rate	452,000 pts/s
FPS	10 Hz

Table 8: LiDAR technical parameters.

Network Integration

To support the high-bandwidth requirements of raw LiDAR data processing and the low-latency needs of teleoperation, the perception set is integrated into the site's 5G infrastructure.

5G Connectivity: The industrial computing unit interfaces with a Sierra Wireless XR90 5G router (acting as a mobile or stationary node) to establish a high-throughput uplink to the ENVELOPE 5G Radio Access Network.

Uplink Strategy: This connection is dedicated to streaming the processed telemetry, specifically the oriented bounding boxes to the edge-based MQTT broker via a 5G NR Standalone (SA) interface.

By utilizing 5G's Reliable Low-Latency communication capabilities and Quality-on-Demand mechanisms, the system ensures that the environmental "ground truth" is synchronized with the DT in near real-time.

Functional Data Flow and Integration

The perception set operates as a continuous loop to support the teleoperation use case. The Livox LiDAR captures the raw geometry of the scene, which the Industrial Computing Unit processes into geometric object-level data. The engine calculates the precise spatial coordinates and orientation of these point clusters.

The MQTT interface broadcasts these oriented bounding boxes to the DT. This system enables the remote operator to navigate the vehicle safely, even if the vehicle's native sensors are obstructed or inactive. This architecture ensures strict decoupling between the physical sensing hardware and the high-level teleoperation logic used by the end-user.

3.1.2 Description of vehicles and end devices

This section describes the physical assets deployed at the Dutch Living Lab for the execution of use cases Dt-UC3, Dt-UC4, and Dt-UC5. The primary asset is an instrumented experimental vehicle equipped with an OBU, a layered computing architecture, and a dedicated sensor suite. Complementary end devices including 5G-capable smartphones and UE nodes for background traffic generation are deployed alongside the vehicle. A DT platform hosted on-site processes and visualises real-time data from all connected assets. Each component is described in the subsections below.

3.1.2.1 Vehicles used in the Dutch Living Lab

System Overview

Figure 23 provides a top-level overview of all hardware and software components deployed at the Dutch Living Lab and their interconnections. The diagram is organized into four functional layers: off-vehicle systems (remote operator workstation, V-Tron back office, the HAN DT platform, and VRU end devices); the 5G RAN, which serves as the shared communication backbone, the onboard computing stack inside the experimental vehicle, and the physical sensor and actuation hardware.

All off-vehicle systems exchange data with the vehicle exclusively via the 5G radio access network. Within the vehicle, the Commsignia OBU serves as the single 5G gateway, interfacing the external network with the internal computing and sensor infrastructure over Ethernet. The onboard computing stack processes sensor data and teleoperation commands, and issues control outputs to the drive-by-wire actuation subsystems via the BOSCH RC40 CAN gateway. Sensor data originates from three physical sources namely four Liebherr MDC3 industrial cameras, vehicle sensors and a u-blox F9P GNSS module each interfaced directly to the relevant Jetson processing units. The actuation subsystems (steering, braking, and throttle) all implement hardware-level redundancy through Dual Modular Redundancy (DMR) on both the mechanical actuators and their control modules. The infrastructure unit that will be used in UC5 consists of a Livox lidar paired with a processing unit and a Sierra wireless router for providing input to the DT.

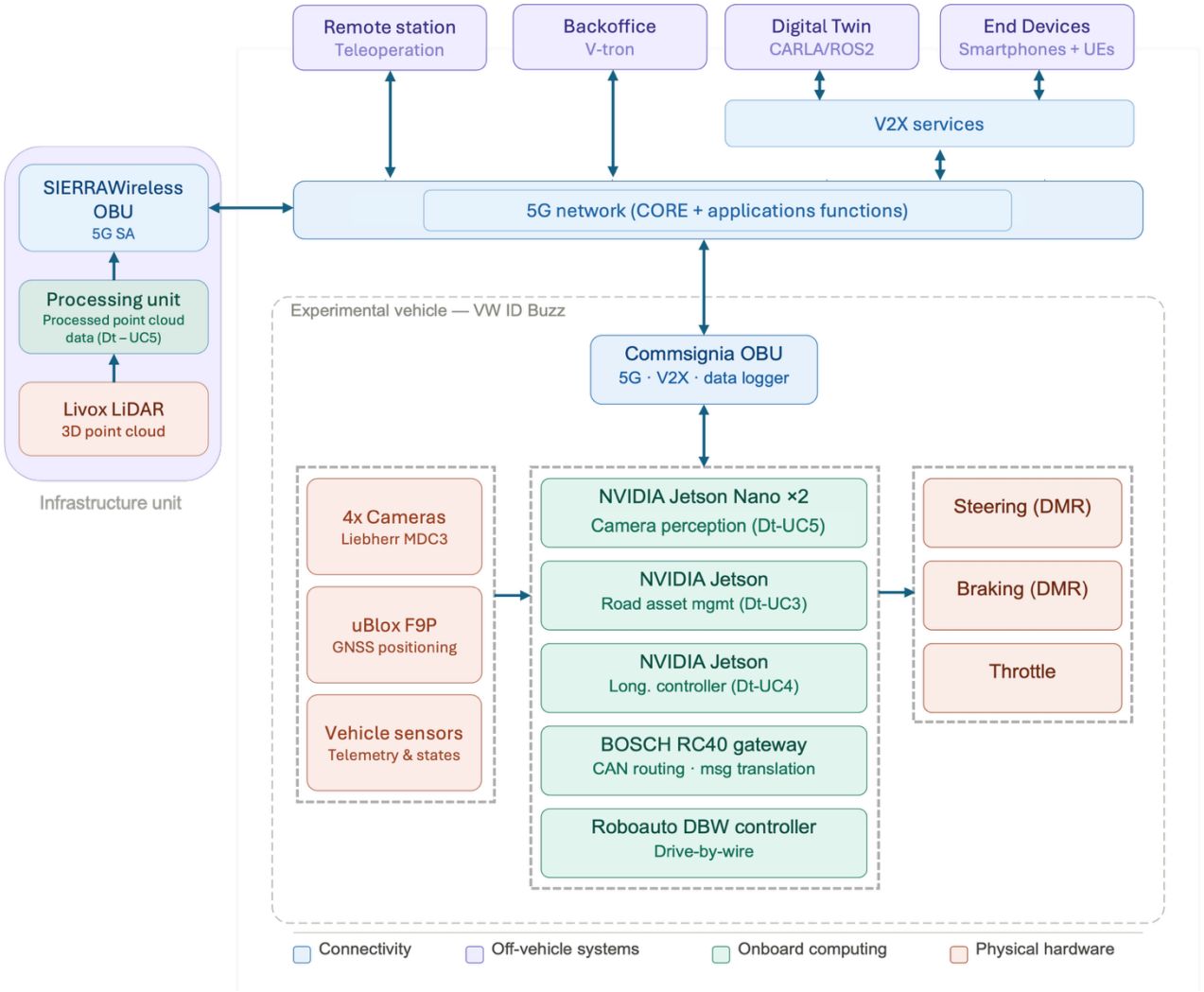


Figure 23: Dutch trial site system overview.

Vehicle Platform

The primary experimental vehicle (Figure 24) used at the Dutch Living Lab is a Model Year 2025 Volkswagen ID Buzz, from V-Tron and part of their development fleet. The vehicle serves as the core mobile node for executing and validating connected and teleoperated driving functionalities in real-world urban road environments. It has been instrumented with an onboard communication unit, layered computing hardware, a drive-by-wire (DBW) actuation system, and a multi-modal sensor payload, each of which is detailed in the subsections that follow.

Figure 24: V-tron test vehicle.

Onboard Communication Unit (OBU)

Network connectivity and V2X communication for the vehicle are managed by a Commsignia OBU (Figure 25), which is a prototype device based on a RISC architecture. The OBU embeds a u-blox GNSS module and a 5G modem compatible with the trial site's use case requirements. It connects to the 5G RAN via the ENVELOPE edge infrastructure, and interfaces with the internal vehicle network through a dedicated Ethernet connection.

The OBU fulfils two primary functions within the experimental setup. First, it provides 5G cellular connectivity for the instrumented vehicle during active use case execution. Second, it operates as one of the time-synchronised data logging endpoints within the overall measurement setup. The embedded Linux environment on the device supports versatile configuration, the deployment of use case-specific client applications, and the execution of ENVELOPE-specific software modules, including the proprietary V2X stack.

Figure 25: Commsignia OBU prototype device.

Onboard Computing Architecture

Beyond the Commsignia OBU, the vehicle integrates a secondary computing platform (Figure 26) running Ubuntu OS to execute local vehicle-side applications, handle data processing, and support specific ENVELOPE enablers. The computing architecture is divided into two operational abstraction layers, a low-level layer and a high-level layer which is interconnected via a central gateway. All hardware components are packaged into a single integrated unit mounted within the vehicle.

Figure 26: Onboard computing unit (packaged).

Low-Level Layer

The low-level layer manages electro-mechanical integration and drive-by-wire (DBW) control, directly operating the throttle, steering, and braking mechanisms. This layer is implemented using a custom DBW controller engineered by Roboauto. The physical actuation systems feature Dual Modular Redundancy (DMR) across both the mechanical actuators and their respective electronic control modules, ensuring fault tolerance during active teleoperation. Each control module performs continuous fault detection on its respective actuator and reports anomalies to the central gateway to trigger functional restrictions and activate system logging immediately.

High-Level Layer

The high-level layer manages the vehicle's perception sensor suite and the core teleoperation logic. Two NVIDIA Jetson Nano modules are dedicated to processing real-time camera perception data. An additional, independent NVIDIA Jetson computing unit powers the road asset management system used in Dt-UC3, enabling real-time road sign detection. For Dt-UC4, a further independent NVIDIA Jetson board houses the local longitudinal controller, which receives virtual object inputs from the DT and computes the appropriate vehicle response.

Central Gateway

Data exchange between the two abstraction layers is mediated by a BOSCH RC40 central gateway. This gateway uses a Controller Area Network (CAN) bus to perform message routing,

serialisation, and translation between the high-level teleoperation logic and the low-level actuation endpoints. During active teleoperation, the Roboauto DBW controller receives Ethernet-based control inputs and forwards them over the CAN bus to the gateway, which translates these generic CAN messages into the specific message formats and IDs required by the redundant actuator control modules. A secondary CAN bus on the gateway continuously retrieves native vehicle sensor data, which is logged locally and transmitted to the V-Tron back office for analysis.

Sensor Suite

The vehicle's sensor payload is configured to support the ENVELOPE use cases through a combination of industrial cameras, a GNSS positioning module, an inertial measurement unit, and a LiDAR unit. Together these sensors provide the environmental perception and spatial localisation data required by the teleoperation and DT systems.

Industrial Cameras

The primary vision system for teleoperation relies on four Liebherr MDC3 industrial cameras (Figure 28) specifications of which are shown in Figure 27. These cameras are networked via Ethernet to the onboard processing units. The cameras are physically distributed around the vehicle to provide a continuous 360-degree environmental feed. Video streams from these cameras are processed in real time by the high-level NVIDIA Jetson Nano modules and transmitted to the remote operator station for teleoperation.

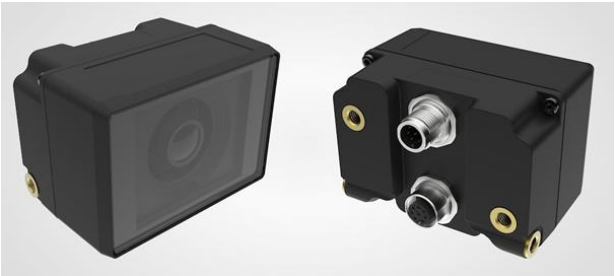


Figure 28: Liebherr MDC3 industrial camera unit.

Description	Values
Resolution	1280 x 960 pixels
Latency	< 75 ms
Frame rate	Max 40 fps
Protection class	IP67, IP6K9K (ISO 20653)

Figure 27: Camera specs.

GNSS Positioning Module

For spatial positioning, the vehicle is equipped with a u-blox F9P GNSS module paired with a dual-band antenna. Localization data is transmitted from the module to the NVIDIA Jetson computing units via a serial communication interface. The u-blox F9P supports multi-band RTK operation, providing centimeter-level positioning accuracy suitable for the precise localization requirements of the teleoperation and DT use cases.

LiDAR

A Livox LiDAR unit is integrated to provide three-dimensional geometric scene data. Raw point cloud data captured by the LiDAR is processed by the Industrial Computing Unit into geometric object-level representations. The processing engine computes the spatial coordinates and orientation of detected point clusters, generating oriented bounding boxes. These bounding boxes are broadcast to the DT platform via the MQTT interface, enabling the remote operator to navigate the vehicle safely even when the vehicle's native sensors are partially obstructed or inactive.

In-Vehicle Network Architecture

Figure 29 shows the teleoperation system architecture, illustrating how the onboard computing modules, communication interfaces, and sensors are interconnected. Figure 30 details the in-vehicle actuation hardware architecture, including the signal routing between the central gateway, the actuator control modules, and the native vehicle systems. Figure 29 shows a detailed view of the teleoperation box components shown in Figure 30.

The central gateway bridges the high-level teleoperation stack with the low-level vehicle actuation systems. On the actuation side, both the steering and braking subsystems implement Dual Modular Redundancy: each subsystem comprises two independent mechanical actuators paired with two redundant electronic control modules. Each module performs continuous self-monitoring and reports faults to the gateway, which immediately triggers functional restrictions and event logging upon detecting any anomaly.

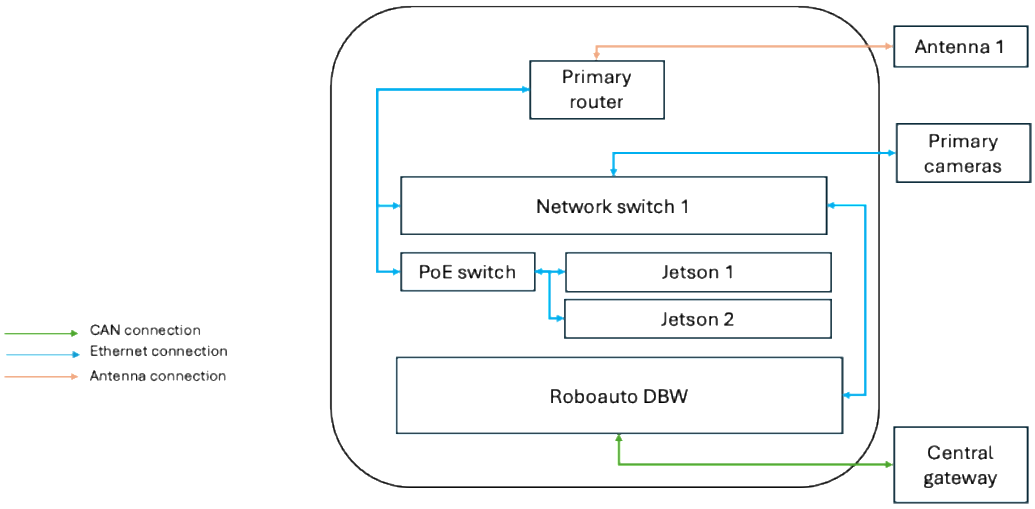


Figure 29: Teleoperation system architecture.

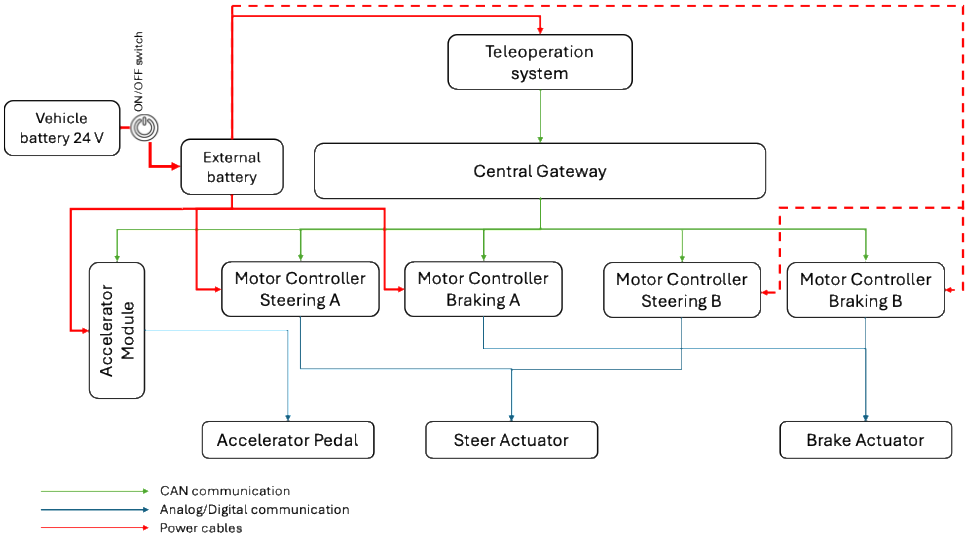


Figure 30: In-vehicle actuation hardware architecture.

Functional Data Flow and System Integration

The perception and teleoperation pipeline operates as a continuous closed loop. The Livox LiDAR captures raw geometric scene data, which the Industrial Computing Unit processes into object-level geometric representations, specifically oriented bounding boxes describing the spatial coordinates and orientation of detected objects in the vehicle's surroundings. These bounding boxes are published over the MQTT interface to the DT platform, where they are consumed by the remote operator interface.

The four Liebherr MDC3 industrial cameras simultaneously provide a continuous 360-degree video feed, which is processed by the onboard Jetson Nano modules and transmitted over the 5G link via the Commsignia OBU to the teleoperation workstation. Vehicle localization data from the u-blox F9P GNSS module is forwarded to the Jetson units for onboard processing and to the DT via UDP.

This architecture enforces strict decoupling between the physical sensing hardware and the high-level teleoperation logic consumed by the remote operator. Actuation commands generated by the remote operator are routed from the teleoperation workstation over the 5G link to the Roboauto DBW controller, which forwards them via Ethernet to the BOSCH RC40 gateway. The gateway translates and distributes these commands over the CAN bus to the appropriate redundant actuator control modules. Native vehicle sensor data retrieved by the secondary CAN bus on the gateway is logged locally and transmitted to the V-Tron back office for post-session analysis.

3.1.2.2 Other UEs and end devices

In addition to the instrumented vehicle, the Dutch Living Lab deploys two categories of end devices to support use case execution: 5G-capable smartphones functioning as VRU (Vulnerable Road User) endpoints, and dedicated UE devices used for background traffic generation.

Smartphones

5G-capable smartphones running the Commsignia VRU protection application serve as additional data inputs, particularly for the teleoperation use case. The primary device is a Google Pixel 9 Pro

(Figure 31), selected for its high-accuracy positioning capabilities. Additional Samsung Galaxy devices are available as supplementary units. These smartphones transmit VRU position data via the Commsignia application to the RSU and the DT, enabling VRU movement tracking in Dt-UC4 and Dt-UC5.

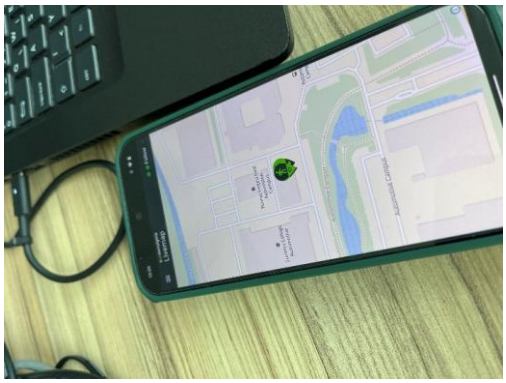


Figure 31: Google Pixel 9 Pro running the Commsignia VRU application.

User Equipment for Background Traffic Generation

For background traffic generation, two Netgear M6 Pro (model MR6450AB) devices are deployed inside the TNO building at the trial site. Each unit is equipped with a Qualcomm Snapdragon X65 5G modem, supporting 5G NR in SA mode to ensure reliable high-uplink 5G connectivity throughout testing. Both devices are positioned at locations with clear line-of-sight (LOS) to the 5G base station, providing stable signal quality and consistent throughput for the background traffic profile.

3.1.3 Digital Twin Platform

The Dutch trial site use cases are supported by HAN's digital twin platform, which provides real-time visualisation and data processing for all connected assets. Specifically, the platform displays: the real-time position of the V-Tron instrumented vehicle in Dt-UC3, Dt-UC4, and Dt-UC5; VRU movements sourced from the Commsignia RSU in Dt-UC4 and Dt-UC5; and surrounding traffic data derived from the Roboauto perception set in Dt-UC5.

The digital twin uses a photorealistic Unreal Engine-based CARLA simulation environment, integrated with a ROS 2 DDS (Data Distribution Service) middleware layer. The virtual models within CARLA are tightly coupled with the real-world sensor data streams. The platform is hosted on a Lenovo ThinkStation P3 Ultra (Figure 32) configured with an Intel Core Ultra 9 processor, 32 GB DDR5 memory, an NVIDIA RTX 4000 20 GB workstation GPU, and a 10 GbE network interface.

The digital twin is configured with multiple input and output interfaces. ROS 2 acts as the internal middleware layer of the digital twin, handling the exchange of data between the CARLA-based environment and the external interfaces; in particular, incoming MQTT and UDP data streams are converted into ROS 2 messages before being processed by the DT applications. Vehicles, the perception set, and VRU-associated UEs transmit real-time positional and contextual data to the platform via MQTT and UDP. Depending on the active use case, the digital twin also transmits data back to the V-Tron vehicle over UDP to support mixed-reality testing scenarios for example,

providing virtual object positions to the Dt-UC4 longitudinal controller. Network connectivity for the hosting PC is provided via a Sierra Wireless XR90 5G dual router.

Figure 32: Lenovo ThinkStation P3 Ultra hosting the Digital Twin platform.

3.2 Data collection & management

At the Dutch site ENVELOPE architecture, the location of the data collection tools for enabler KPIs is illustrated in the figure below (Figure 33).

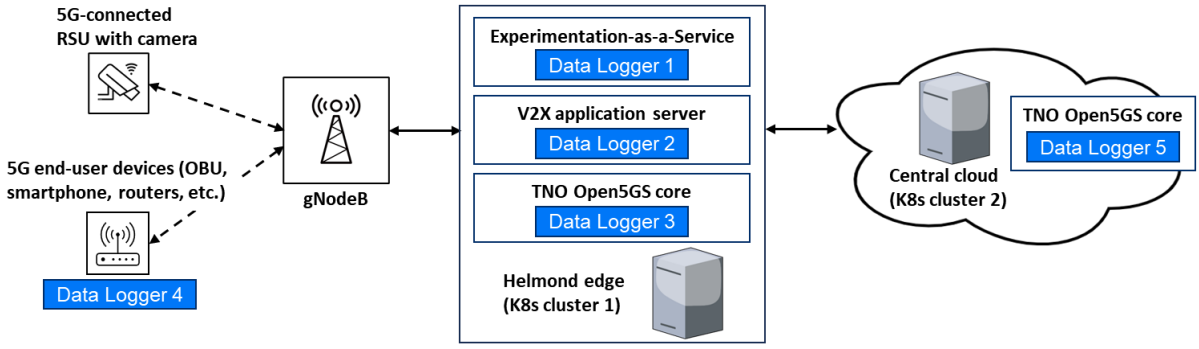


Figure 33: Dutch site architecture with indicated data logger points for enabler KPIs.

Table 9: Data logged in the V2X Application Server.

Item	Description
KPI ID	KPI_Enab_Availability (all enablers) KPI_Enab_PerMetrics_NotificationSR KPI_Enab_PerMetrics_NotificationLatency KPI_Enab_DL_EffectLatency (partially) KPI_Enab_DL_DeviceLocationGeofenceSR (partially)

	KPI_Enab_QoD_SuccessRate KPI_Enab_QoD_QoSProfileEstablish-mentTime KPI_Enab_QoD_QoSSessionRetainability (partially) KPI_Enab_pQoS_SuccessRate KPI_Enab_pQoS_EffectLatency KPI_Enab_pQoS_InferenceAccuracy KPI_Enab_pQoS_InferenceDuration KPI_Enab_pQoS_InferenceConfidence KPI_Enab_Edge_ServiceDeploymentTime KPI_Enab_Edge_HandoverTime KPI_Enab_Edge_HandoverSR KPI_Enab_Edge_TrafficInfluenceUEPoli-cyUpdateTime (partially)
Data logger / PoP (location in architecture)	Data Logger 2(@V2X server)
Log data type	Enabler API events (subscription, notifications)
Log format	**JSON logs** with optionally exposed Prome-theus metrics for “live” evaluation
Logging event	Enabler-related events: - Session/subscription creation and dele-tion - Periodic or event-based notifications from the enabler
Sampling policy	Event-driven
Logging storage	Docker logging (journalctl) + timestamped per-experiment (test run) extract of the logs
Log data fields	Timestamp (UTC), API event details and prop-erties

- **API event details**: The API consumer component of the V2X Application Server collects data on an event-driven basis of session/subscription events. The component determines enabler availability based on HTTP responses from each enabler. It tracks the stages of synchronous calls and digests asynchronous notifications to calculate latency and success rate. A script is used to conveniently extract log entries of a test run to a timestamped file via journalctl.

Table 10: Data logged in TNO Open5GS core.

Item	Description

KPI ID	KPI_Enab_DL_EffectLatency (partially) KPI_Enab_DL_DeviceLocationGeofenceSR (partially) KPI_Enab_pQoS_TrainingDuration KPI_Enab_pQoS_TrainingConvergence KPI_Enab_pQoS_CAMSafeModeTA (partially)
Data logger / PoP (location in architecture)	Data Logger 3
Log data type	Enabler events (API calls, notifications)
Log format	Open5GS log format
Logging event	Enabler-related events
Sampling policy	Event-driven
Logging storage	Docker/K8s logging
Log data fields	Timestamp (local time/UTC), API event details and properties

- **Open5GS API event details**: native 5G core API calls and notification logs that related to the different CAMARA API calls are recorded by an external program which monitors and filters ongoing logs of the Open5GS core. These logs are timestamped in local time or optionally converted to UTC time. Open5GS uses a plain-text, line-oriented log format by default, designed to be human-readable. The format is consistent across all Open5GS network functions (AMF, SMF, UPF, etc.), with optional integration to syslog.

Table 11: Data logged in the Commsignia OBU.

Item	Description
KPI ID	KPI_Enab_QoD_QoSSessionRetainability (partially) KPI_Enab_Edge_TrafficInfluenceUEPoli-cyUpdateTime (partially) KPI_Enab_pQoS_CAMSafeModeTA (partially)
Data logger / PoP (location in architecture)	Data Logger 4
Log data type	MQTT message details (Rx/Tx), aggregated transmission statistics
Log format	JSON
Logging event	Event-driven: MQTT message transmission/re-ception Periodic: statistics output

Sampling policy	Event driven Periodic: 5s (configurable)
Logging storage	Local storage of JSON file
Log data fields	MQTT messages: time (UTC), id, payload_size Statistics: time (UTC), avg_rtt_ms, through-put_mbps, msg_count

- **MQTT message**: The MQTT payload is specifically designed to support the tests with randomly generated round-trip traffic. Each message contains random payload, a unique ID (UUID), and the timestamp of message creation.
- **Statistics**: For KPI evaluation, an iperf-like application generating MQTT traffic and logging statistics is used on the OBU. The application subscribes to the generated data feed to achieve high uplink and downlink throughput. The app periodically calculates a windowed average of the message round-trip time and throughput, which is essential data for evaluating the session retainability of an active QoD session. The throughput data values reported in the Performance Metrics notifications can serve as additional information for this KPI evaluation. Furthermore, the app logs the timestamp, id, and message size all sent and received messages. For received messages, the round-trip latency is also logged.

Data collection & management: Roboauto Teleoperation System

This data logger is located at the central Gateway (Section 3.1.2.1). It is responsible for monitoring the integrity of the remote-control loop and the quality of the 5G network connection during the NL-UC5 test iterations (Figure 34).

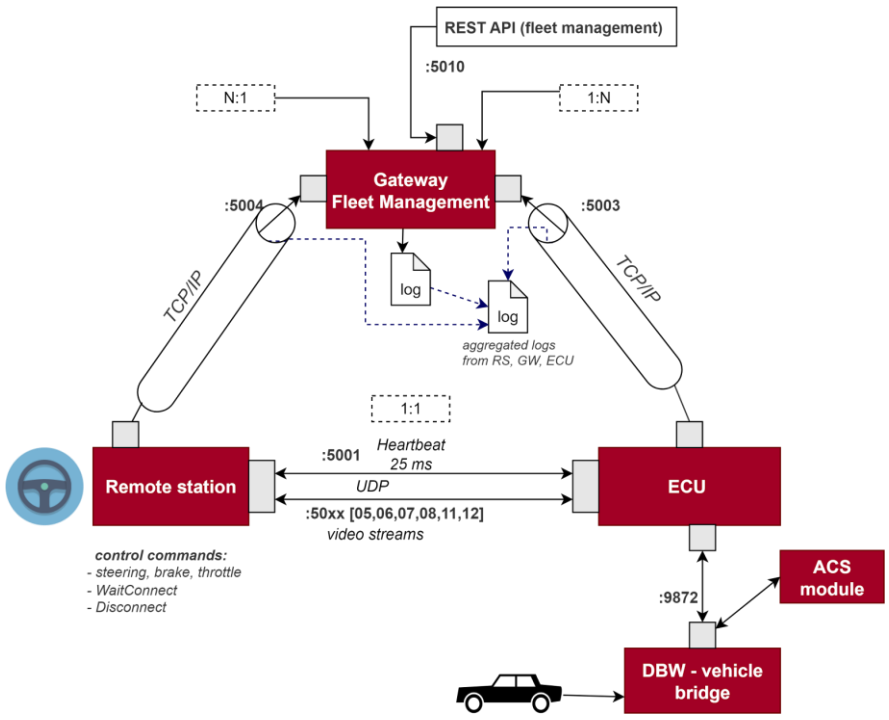


Figure 34: Roboauto teleoperation architecture with indicated data logs aggregation.

Table 12: Data logged in the Roboauto Teleoperation System.

Item	Description
KPI ID	KPI_Teleop_E2E_Latency KPI_Teleop_Service_Reliability KPI_Teleop_Disengagements
Data logger / PoP (location in architecture)	Roboauto Gateway
Log data type	Network stats (latency), System diagnostics
Log format	Structured text / JSON-ready log files
Logging event	Periodic: Latency statistics Event-driven: Control mode changes, warnings, and disengagements.
Sampling policy	Event-driven for control logic; Periodic window (60 samples) for latency aggregation.
Logging storage	Local Gateway; exported as timestamped files post-experiment.
Log data fields	Timestamp, verbosity, file, function, line, current_latency, mean_latency

The Teleoperation related KPIs are computed as follows:

- **Timestamp**: Recorded automatically for every log entry by Gateway to ensure a precise timeline of events.
- **Verbosity, File, Function, Line**: Diagnostic metadata fields are included in each log to identify the source and severity (e.g., Info, Error) for later formatting or post-processing.
- **Current_latency (ms)**: Measured round-trip delay of control commands between the Remote station and vehicle's ECU.
- **Mean_latency (ms)**: Calculated as average over the last 60-samples of current latency, this value is included in every latency-related log message.
- **KPI_Teleop_Service_Reliability**: Computed as percentage of latency log messages which successfully meet the KPI latency goal.
- **KPI_Teleop_Disengagements**: Calculated by counting number of logs with rsConnectionLost status during the active teleoperation session.

Data collection & management: Digital Twin

This data logger is located at the Remote Station used for the Dutch trail use cases. The digital twin running at the remote station PC will log the interface traffic based on the use case. The PC obtains GPS time similar to the use case vehicle and roadside unit perception and hence is synchronized across all use case equipment.

Table 13: KPIs measurement at the Digital Twin.

Item	Description
KPI ID	KPI_DTupdate_E2E_Latency KPI_DTobjects_E2E_Latency KPI_DTobjects_E2E_LatencyJitter KPI_DT_RSU_E2E_Latency
Data logger / PoP (location in architecture)	Digital twin's remote station PC
Log data type	MQTT, UDP messages
Log format	JSON
Logging event	Periodic: Vehicle telemetry and RSU updates during real-time communication with the DT. Event-driven: DT object generation.
Sampling policy	Event-driven: upon message arrival, transmission
Logging storage	Locally on remote station PC
Log data fields	Timestamp, vehicle position, virtual object position, dynamic object positions

The Digital Twin related metrics are computed and logged as follows:

- **Timestamp:** Extracted from the GPS-synchronised clock of the remote station PC to ensure absolute time alignment with the vehicle, RSU, and roadside perception components in the Dutch trial use cases.
- **Vehicle position (JSON object / ROS 2 message):** Represents the real-time position updates received from the vehicle side and ingested by the digital twin through the communication interfaces used in the active use case. Depending on the UseCase, these updates are received over MQTT or UDP and converted into ROS 2 messages for internal DT processing and visualization.
- **Virtual object position (JSON object / ROS 2 message):** Represents the position of objects generated or maintained by the digital twin and, where applicable, transmitted back to the vehicle-side controller to support mixed-reality use cases such as Dt-UC4. These objects are published through the ROS 2 middleware and exported via the relevant external interface.
- **Dynamic object positions (JSON object / ROS 2 message):** Represents the positions of dynamic actors detected by external sources such as the roadside perception set, RSU, or VRU-related devices. These data are received through MQTT and/or UDP interfaces, converted into ROS 2 messages, and used by the digital twin to maintain a synchronized representation of the surrounding environment.

At the Dutch site, the digital twin logger runs on the remote station PC and records the relevant interface traffic associated with the active use case. ROS 2 acts as the middleware layer for the DT, enabling the ingestion, internal exchange, and publication of data between the CARLA-based

simulation environment and the external systems. Incoming MQTT payloads are converted into ROS 2 messages, while UDP-based data such as vehicle position updates and DT outputs are logged at the corresponding communication interfaces. The resulting JSON logs are stored locally on the remote station PC and are used to evaluate the DT-related end-to-end latency and jitter KPIs.

Data collection & management: V-TRON Vehicle

This data logger is located at the vehicle used for the Dutch trial use cases. The vehicle will log the vehicle states based on the use case. The time synchronisation is achieved via the GPS system installed in the car and hence is synchronized across all use case equipment.

Table 14: KPI measurement at the test vehicle central gateway.

Item	Description
KPI ID	KPI_vehicleStateswithTimestamps
Data logger / PoP (location in architecture)	Test Vehicle
Log data type	CAN messages
Log format	JSON
Logging event	Periodic: Vehicle telemetry with time stamps Event-driven: TSR Speed signs.
Sampling policy	Event-driven: upon detection 100Hz for vehicle telemetry
Logging storage	Locally on vehicle
Log data fields	Timestamp, vehicle position, TSR detection values

The KPI_vehicleStateswithTimestamps related metrics are computed and logged as follows:

- **Timestamp (time/date):** Extracted directly from the onboard GPS synchronization system to enforce absolute time alignment across all use case equipment, bypassing local gateway reception latency.
- **Vehicle position (JSON object):** Sampled continuously at 100Hz. It comprises latitude, longitude, GPS Fix, GPS RTK fix, GPS speed, heading, horizontal accuracy, and the update frequency derived from the local script and GPS input.
- **TSR detection values (JSON object):** An event-driven payload appended to the continuous telemetry stream upon perception triggers. It consists of the specific recognized traffic sign value and the corresponding system confidence score.
- **CAN messages (JSON structure):** Raw vehicle states are processed via a direct mapping of the CAN ID and payload into the resulting JSON log format.

4 Greek Living Lab

4.1 Living Lab Description

4.1.1 Description of the infrastructure

The Greek Living Lab is a federated infrastructure geographically spanning across both the NCSR Demokritos (NCSRD) campus and the OTE premises in Athens, as illustrated in Figure 35. The figure provides a geographical overview of the trial area and highlights the main infrastructure elements involved in the ENVELOPE experimentation activities. Building upon the foundational capabilities established in previous European framework projects, such as 5GENESIS and EVOLVED-5G, the ENVELOPE network at the Greek trial site has been significantly extended. The deployment integrates the established OTE network (serving as the Home PLMN) with the NCSRD campus area and its surrounding road infrastructure (serving as the Visited PLMN), interconnected via a dedicated dark-fiber link.

The deployed infrastructure enabling the execution of the Greek use case (GR-UC6) and Open Calls 1 and 2 (OC1 & OC2) under controlled and realistic mobility conditions incorporates: i) the Radio Access Network (RAN) infrastructure, which includes advanced components, such as Ericsson Radio Dots deployed to ensure robust indoor and outdoor 5G coverage across the trial zones, ii) the Core Network (CN) infrastructure, which hosts a distributed 5G Core network setup that facilitates inter-PLMN handover scenarios. It features the HPE 5G Core deployed at the OTE premises, acting as the HPLMN, and the Open5GS core deployed at the NCSRD campus, acting as the VPLMN and iii) the edge and cloud infrastructure, where a federated edge computing continuum based on aerOS, is distributed across the two sites. aerOS hosts trial-specific applications requiring low-latency access and supports dynamic service migration during mobility events.

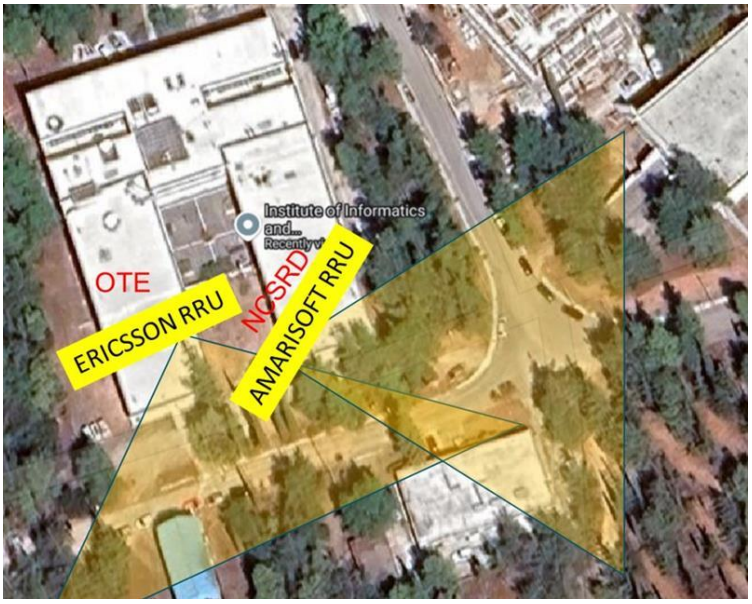


Figure 35: Antenna coverage of the Greek trial site.

Trial area and coverage

The trial area is highlighted in Figure 36 and includes the NCSRD campus, the OTE premises and their surrounding road infrastructure. This geographically distributed setup provides 5G radio coverage across the two administrative domains, for a total coverage of around 500 square meters. The environment features a mix of indoor and outdoor spaces, vehicular routes with a maximum speed of 30 km/s and overlapping network footprints with varying line-of-sight (LOS) and non-line-of-sight (NLOS) conditions. These realistic connectivity scenarios are highly representative of actual urban mobility, providing an ideal testing ground for evaluating the seamless inter-PLMN handover and service continuity required by the ENVELOPE (GR-UC6) use case.

More specifically, the Inter-PLMN handover will be validated using the outdoor mid-size cells of the ENVELOPE overlay network. These cells operate within the spectrum allocated to the COSMOTE TELECOM network, in full compliance with the Greek National Regulatory Authority (NRA) requirements (EETT) and 3GPP specifications for roaming and spectrum usage.

Due to the proximity of high-power commercial base stations operating in adjacent frequency bands, the ENVELOPE overlay cells are inherently vulnerable to interference. While more than 10 commercial COSMOTE TELECOM sites operate in the vicinity, as can be seen in Figure 36, due to the area clatter, the suspect base station to dominate and interfere with ENVELOPE sites is AOTHEOLO_2 located Periferiaki Imitou.

Figure 36: NCSRD Site - COSMOTE Telecom Coverage Overlap.

Figure 37 represents 5G SA signal-strength heatmap generated by outdoor mobility tests, illustrating the measured RSRP levels along the UE's trajectory from the OTE home network towards the NCSRD visited network over a few-minute (4) interval. The route is overlaid on a site map and color-coded from green (around −80 dBm, strong signal) through yellow and orange to

red (below −110 dBm, weak signal), while a dark marker at the road intersection denotes the point where the inter-PLMN handover from the serving OTE gNB to the NCSRD gNB was triggered as the OTE signal degraded. As consistently observed across three test runs, segments where the UE maintained direct line-of-sight to the serving gNB appear in green or light yellow, whereas areas with buildings or other obstacles correspond to lower RSRP values and red segments, yet in every case the handover occurred reliably at the intersection and connectivity remained stable on the visited network thereafter.

To mitigate this risk and ensure stable performance during the demonstration, it is critical that the RAN parameters configured at ENVELOPE sites are carefully planned to avoid overlapping or conflicting with commercial network frequencies. To ensure seamless mobility, particularly for ENVELOPE's inter-PLMN handover scenarios and to mitigate interference with surrounding commercial networks, the following RAN parameters are critical:

- **Physical Cell Identity (PCI):** Proper PCI planning ensures unique identification of ENVELOPE cells and prevents ambiguity during cell selection and reselection. PCI collisions or confusion can lead to handover failures and degraded mobility performance. PCI allocation must follow 3GPP TS 38.300 guidelines for NR cell identity planning.
- **Synchronization Signal Block (SSB) Frequency**: Selecting SSB frequencies away from the center of commercial allocations reduces the likelihood of interference during initial access and synchronization procedures. Frequency planning must adhere to NRA spectrum allocation rules and 3GPP TS 38.211 for SSB configuration.
- **TDD Special Slot Pattern**: Configuring an appropriate TDD special slot pattern ensures compatibility with surrounding networks and minimizes interference during simultaneous transmissions. Slot pattern selection must follow 3GPP TS 38.213 and consider coexistence requirements in shared spectrum environments according to national NRA recommendations.

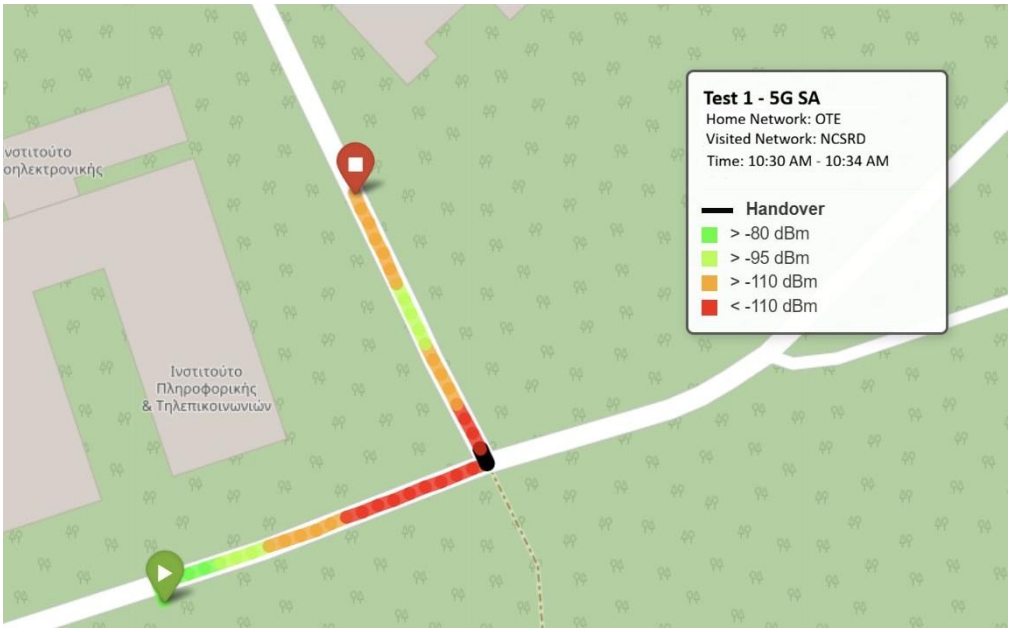


Figure 37: 5G Standalone (SA) signal-strength heatmap in the Greek trial site.

In co-operation with the COSMOTE Telecom planners, the following RAN parameters are dedicated to the two ENVELOPE base stations:

Table 15: Parameters of the two RANs of the Greek trial site.

Parameter	Home Network (ENVELOPE-OTE)	Visited Network (ENVELOPE-NCSRD)
Cell Name	NCSRD	NCSRDTESTE1
nrPCI	3	2
rachRootSequence	66	44
nrTAC	15100	1
arfcnDL/UL	636666	636666
bSChannelBwDL/UL (KHz)	100.000	50.000
configuredMaxTxPower	4x50mW	2x1mW
3 [6:4:4]1 (UL/DL pattern is DDDSUUDDDD)ssbFrequency	634080	636666
tddSpecialSlotPattern	3 [6:4:4]	3 [6:4:4]
tddUlDlPattern	1 (UL/DL pattern is DDDSUUDDDD)	1 (UL/DL pattern is DDDSUUDDDD)
NR Numerology	30KHz	30KHz
Frequency Band	3500	3500

It must be noted that based on the implemented roaming architecture, to initiate the handover (HO) process, an area of degraded signal quality must be intentionally created between the two ENVELOPE base stations. Because the UE natively prefers to remain camped on the Home Network, it will only trigger the network reselection process once the Home network's signal quality drops below acceptable thresholds. Despite this induced signal degradation, service continuity is successfully maintained throughout the transition.

The process included tuning of the following parameters:

- **Transmission Power:** The bigger the power the stronger the signal quality and the wider the area covered. However, this directly impacts interference footprint as overpowered neighbouring commercial cells can severely disrupt the ENVELOPE operations.
- **Antenna Tilt and Beamforming:** These control the coverage shape and are important to appropriately cover the ENVELOPE demo route.
-

Radio Access Network (RAN) infrastructure

In the Greek site, at the NCSRD campus for ENVELOPE two New Radio (NR) sites are deployed as described in Table 16.

Table 16: ENVELOPE Greek Site Base Stations.

	Home Network (ENVELOPE-OTE)	Visited Network (ENVELOPE-NCSRD)
PlmnId	20202	99970
Cell Name	NCSRD	NCSRDTESTE1
gNB Vendor	Ericsson – Baseband 6630 & IRU 8848	Amarisoft
Antenna	Outdoor - Radio 4408 Indoor – Dot 4479	Outdoor – Octoport Multi-Band Antenna Indoor – Amari Callbox Classic 2x2 MIMO
Latitude	37,99649	37.996356
Longitude	23,81868	23.818910
Bandwidth	100 MHz	50 MHz – (max hardware limit)
CellId	131 (outdoor), 132 (indoor)	1

The indoor Amari Callbox Classic 2x2 MIMO Antenna (Figure 38) is used to test and verify the functionality of Inter-PLMN and ENVELOPE APIs at the Greek trial site premises. It is part of the Amarisoft RAN and operates in TDD with a 50 MHz channel Bandwidth. This antenna is only used for testing and verification purposes; it is not part of the final scenario/demo.

Figure 38: Illustration of the indoor Amari Callbox Classic 2x2 MIMO Antenna.

The VPLMN outdoor antenna is an Octoport Multi-Band Antenna (Figure 39) that supports the inter-PLMN handover testing at the Greek trial site premises. It is connected to the Amarisoft RAN and operates in TDD with a channel Bandwidth of 50 MHz, which is the maximum permitted by the hardware. It is placed in around 2-5 meters height and 30-35 meters vertically away from the road in which the Inter-PLMN handover testing takes place, the road is uphill so as the relative height of the antenna starts at around 5 meters and by the time the UE is at the “handover zone” the height is 2 meters. The tilt is set at 0 degrees. This antenna covers half of the testing path, to create a small coverage overlap with the Ericsson antenna, where the handover takes place.

Figure 39: Outdoor Octoport Multi-Band Antenna.

The HPLMN outdoor antenna is an Ericsson Baseband 6630 & IRU 8848 unit (Figure 40) that covers the other half of the total coverage provided by the Greek trial site. It is connected to the HPE core at OTE premises via dark fiber. This part of the road is straight, and the height of the antenna is at 3.5-4 meters. It also operates in TDD and it has a channel Bandwidth of 100 MHz.

Figure 40: Outdoor Ericsson Baseband 6630 & IRU 8848 unit.

Core Network (CN) infrastructure

The Greek Living Lab incorporates two core networks, the HPLMN that is located at OTE Academy premises and the VPLMN in the NCSRD campus that is the demonstration area for the Inter-PLMN Handover functionality. The rack hosting the ENVELOPE VMs at NCSRD premises is shown in the following picture (Figure 41).

Figure 41: Rack server at NCSRD hosting VMs and network function.

The VPLMN is deployed on dedicated server infrastructure at NCSRD premises and is connected to the gNB via a dedicated fiber backhaul link. The backhaul provides sufficient capacity and low latency to support the ENVELOPE use cases and associated edge services.

The CN hosts the required 5G network functions and is configured in accordance with the ENVELOPE architecture defined in WP2 supporting the ENVELOPE innovations developed in WP3.

For the ENVELOPE open core implementation (VPLMN), and to meet the operational requirements of the GR site activities, we deployed and customized selected 5GC network functions from both Open5GS and Free5GC. Both platforms are open-source implementations of the 3GPP 5G Core, primarily targeting research, prototyping, and experimental deployments. They implement SBA network functions, including AMF, SMF, UPF, NRF, PCF, AUSF, and N3IWF, and are cloud-native solutions typically deployed in containerized environments to support modular, scalable, and flexible core network experimentation.

In the modified ENVELOPE open core network, the N3IWF component from Free5GC (v4.2.0) is employed to enable non-3GPP access integration, while the remaining core network functions are instantiated using the Docker-based distribution of Open5GS 3GPP Release 17 (v2.7.2). Furthermore, the Open5GS AMF has been extended to correctly process the Initial UE Message and the User Location Information (ULI) information elements, in accordance with 3GPP TS 38.413. These enhancements were required to support N3IWF-specific user location reporting, which is not natively supported in the standard Open5GS implementation.

In addition to the core network functions, a Network Exposure Function (NEF) has been implemented to provide controlled exposure of selected 5G Core capabilities to external applications in accordance with the 3GPP service-based architecture. The implemented NEF follows the specifications defined in 3GPP TS 29.122, which defines the northbound APIs exposed at the T8 reference point for enabling AFs to access network capabilities in a standardized and secure manner. The NEF operates as an intermediary between external applications and internal core network functions, enabling the controlled retrieval of subscriber-related information as well as application-driven policy control.

The current implementation supports two NEF services defined in 3GPP TS 29.122, namely the Monitoring Event service and the Application Session with QoS service. The Monitoring Event functionality allows external applications to subscribe to network monitoring events associated with specific subscribers. In the current implementation, this capability focuses on subscriber location monitoring, enabling applications to request notifications related to the current location of a subscriber identified by its MSISDN. Applications can create monitoring subscriptions that specify a callback endpoint to receive monitoring reports, as well as parameters defining the reporting behavior such as periodic reporting intervals or the maximum number of reports to be delivered. When location updates are detected by the network, the NEF generates MonitoringEventReport notifications and delivers them to the subscribing application. In addition to subscription-based monitoring, the NEF also supports the retrieval of the last known location of a subscriber, allowing an Application Function (AF) to obtain the most recently recorded location information associated with a given subscriber and SCS/AS identifier.

The second implemented NEF capability concerns application-driven Quality of Service configuration through the Application Session with QoS API, defined in 3GPP TS 29.122 (subclause 5.14.1). This interface enables an Application Function to request the establishment of an application session with specific QoS requirements by specifying parameters such as the desired 5G QoS Identifier (5QI). Upon receiving such a request, the NEF interacts with the Policy Control Function (PCF) through the Npcf_PolicyAuthorization service, as specified in 3GPP TS 29.514, in order to authorize and enforce the requested policy. Through this procedure the PCF installs the corresponding Policy and Charging Control (PCC) rules associated with the subscriber's PDU session, enabling the network to apply the requested QoS treatment to the application traffic. The data models and common information elements used by the implemented APIs follow the definitions provided in 3GPP TS 29.122 and 3GPP TS 29.571, ensuring compatibility with the standardized 5G service-based interfaces.

Figure 42 illustrates the open core network deployed for the visited PLMN, highlighting the functional components adopted from Free5GC and Open5GS, as well as the additional core network elements developed within the ENVELOPE framework. To establish the Inter-PLMN functionality a SEPP-like function was developed (cf. HPLMN description).

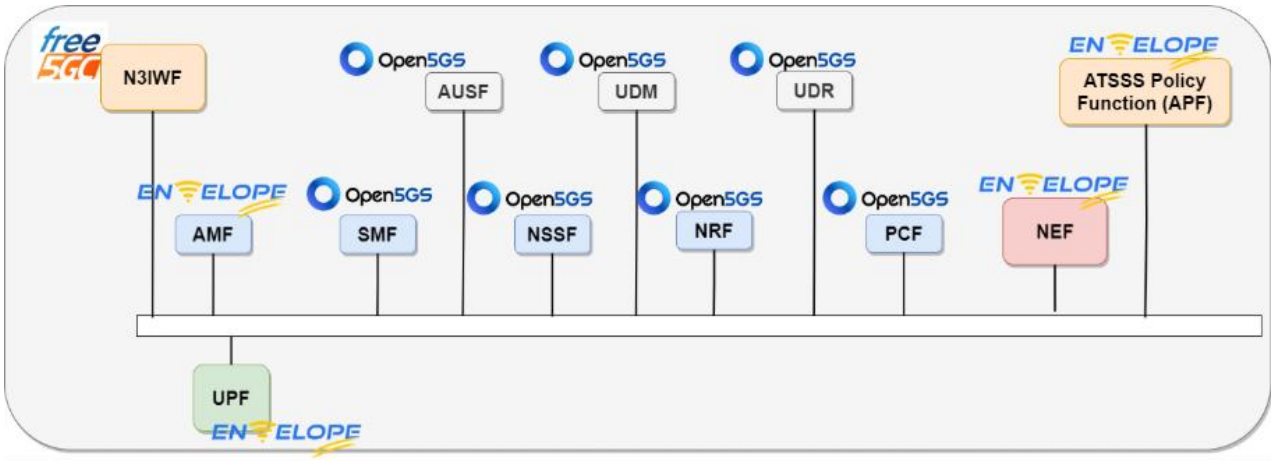


Figure 42: Modified ENVELOPE Open Core Network – VPLMN.

Figure 43: HPLMN Redhat Openstack Infrastructure.

HPE Private 5G Core Network is used as HPLMN in the roaming scenario, with the same capabilities already described in the Italian Living Lab section. Since the HPE core does not natively provide roaming functionalities, additional mechanisms were required to enable the exchange of information needed for roaming procedures. To address this, a SEPP-like function was introduced to ensure secure and controlled signaling between the two networks, acting as an adaptation layer that enables the exchange of messages and policies in the absence of standard roaming interfaces. The HPE Private 5G Core Network is hosted at OTE Academy premises in the rack shown at Figure 43.

To achieve full interoperability, two enhancements were implemented on the Open5GS side, which operates as the VPLMN. First, a selective routing mechanism was added so that all Network Functions - except SMF and AMF - automatically send their SBI requests through the SEPP-like function. This allows these NF to obtain the necessary information from the HPLMN when roaming is active, ensuring correct interaction with the home network. SMF and AMF, instead, always retain their original local endpoints so that mobility and session management remain fully anchored within the VPLMN. When no roaming is involved, the SEPP-like function simply redirects the requests back to Open5GS, maintaining normal local operation and ensuring consistent behavior across both roaming and non-roaming conditions. Second, the URI parsing logic within the SBI layer of Open5GS was improved to correctly handle requests that may arrive without a complete or properly formatted URI. When this condition occurs, the system now adjusts the URI before processing it,

ensuring robust behavior and preventing failures that would otherwise interrupt signaling flows during roaming and non-roaming operations.

The HPLMN ENVELOPE-OTE Core (HPE Aruba) is dimensioned as follows:

Table 17: ENVELOPE-OTE Core Compute Infrastructure.

VM Name	VM Role	vCPUs	vRAM GB	Stor-age GB
NGC_CP	Combo 4G/5G Control Plane	4	8	40
NGC_UP	Combo 4G/5G User Plane	8	8	40
OAM_Con-sole	HPE Manage-ment Console	2	4	120
Aragosta	VPN Client	1	1	10

The HPLMN and VPLMN networks setup is depicted in Figure 44 while the details of the two Core networks deployed within the Greek trial site are tabulated in Table 18.

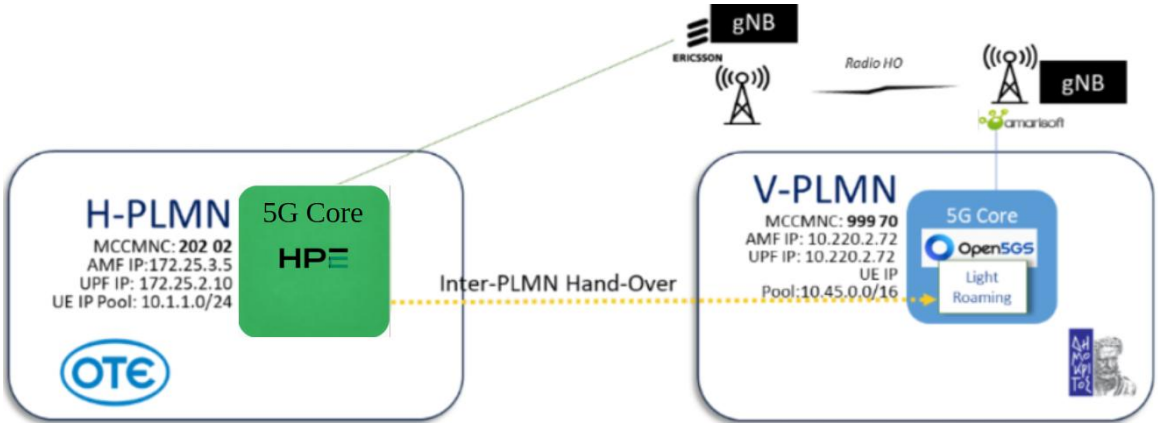


Figure 44: ENVELOPE Greek Site HPLMN & VPLMN Setup.

Table 18: Core Products per PLMN.

	Home Network (ENVE-LOPE-OTE)	Visited Network (ENVE-LOPE-NCSRD)
PlmnId	20202	99970
Vendor	HPE	Open5GS
Product	HPE Aruba Networking	open5gs v2.7.5-41-g52ca325ed

	Private 5G Core 1.24.2	
Host Platform	11-nodes RedHat Open-stack on premises Cloud	Open Nebula 1 node

Edge computing infrastructure

To support the deployment of applications close to the 5G infrastructure, an edge computing domain is deployed at each site participating in the Greek ENVELOPE deployment. These edge domains provide the compute and storage resources required to host applications that interact with the mobile network and benefit from low-latency execution near the radio access and core network functions. The two sites participating in the Greek deployment, namely the NCSRD and OTE infrastructures, each host their own 5G Core network and associated gNB, forming two independent operational environments. The corresponding edge environments are deployed alongside these infrastructures, enabling application instances to run near the network entry point used by connected devices. To efficiently coordinate these two edge domains and support application placement across them, the aeriOS Meta-OS is employed within the Greek ENVELOPE site. aeriOS is the open-source release of the Meta-Operating System developed in the Horizon Europe CL4 project aerOS for the CEI continuum. The platform provides a distributed software substrate that enables the coordinated operation of heterogeneous infrastructures spanning cloud, edge, and on-premises environments. Through its core services, aeriOS (also see D3.1 of ENVELOPE) supports orchestration of containerized workloads across federated domains, enabling deployment models in which application components are dynamically placed according to operator-defined requirements and resource constraints, allowing deployment on the most appropriate infrastructure resources based on multiple criteria.

In the ENVELOPE Greek deployment, aeriOS is used to integrate the edge resources of the NCSRD and OTE sites that are onboarded as aeriOS domains into a unified operational continuum that supports coordinated application orchestration across the two sites. Each site remains independently managed and operates its own 5G Core and radio access infrastructure, while only a subset of the local compute resources is integrated into aeriOS as an orchestration domain. These selected resources form the edge environments on which applications can be deployed and managed through the aeriOS orchestration services. In this way, the 5G network functions remain deployed and managed independently of the aeriOS platform, while the edge compute resources participating in the aeriOS domains provide the infrastructure layer for application hosting. Through this coordinated environment, aeriOS enables application lifecycle management across the two sites and allows applications to be deployed on the edge infrastructure associated with the specific PLMN to which a device is currently connected, ensuring that application components execute close to the corresponding core network and radio access network.

One of the main functionalities of the Greek ENVELOPE use case is the support of roaming and inter-PLMN handover between the two deployed networks at NCSRD and OTE, while maintaining the proximity of the application services to the user. The scenario supports the transition of a device from one PLMN to the other, where the UE performs registration to the core network of the new serving infrastructure. Following this transition, the application instance associated with the device can be reallocated to the edge environment that is closer to, or co-located with, the core network of the serving PLMN. Through the orchestration capabilities provided by aeriOS across the two federated edge domains, the platform enables the migration of application components between

the NCSRD and OTE domains according to the UE connectivity state. As the UE registers to the core network of the new PLMN, the application instance can be migrated to the edge domain associated with that infrastructure. This coordinated orchestration ensures that the application components remain close to the serving network infrastructure, maintaining low latency and efficient interaction with the network functions and exposed capabilities.

This capability is realized through the deployment of a federated edge environment across the two sites, where application workloads can be orchestrated and dynamically relocated between the available domains. In the Greek ENVELOPE deployment, the described architecture is implemented through two aeriOS domains hosted at NCSRD and OTE, respectively. Each domain operates on top of a Kubernetes-based edge environment where the core aeriOS services are deployed and expose the standard aerOS APIs required for federation and orchestration across the continuum. The NCSRD domain operates as the EntryPoint Domain, additionally integrating the singleton management services, namely the identity management services and the developer dashboard used to interact with the platform. The domain is deployed on three virtual machines provisioned through OpenNebula, each configured with 2 vCPUs and 12 GB of RAM. The domain provides public access to the aeriOS API exposure endpoint (http://api-continuum.front-research-group.eu), the identity management service (https://keycloak.front-research-group.eu), and the aeriOS dashboard (https://portal-continuum.front-research-group.eu). All services are proxied through the public IP address **143.233.227.105**.

The OTE aeriOS domain hosts the edge resources used for the deployment and execution of application components associated with the ENVELOPE use cases. The domain is deployed on virtual machines hosted on a Proxmox-based infrastructure, with two main virtual machines providing the Kubernetes cluster used for the aeriOS deployment. The first VM is configured with 4 vCPUs and 12 GB of RAM, while the second VM provides 4 vCPUs and 16 GB of RAM. The broader Proxmox infrastructure offers additional capacity available for application deployments, including approximately 80 physical CPUs (160 vCPUs), 110 GB of RAM, and 1.5 TB of storage. Unlike the NCSRD domain, the OTE domain exposes its APIs internally within the OTE local network. The OTE LAN is directly connected to the NCSRD site through a dedicated dark-fiber interconnection with programmed routing, enabling full bidirectional reachability and cross-domain API communication between the two domains. Figure 45 below shows a diagram of the topology implemented in Athens site.

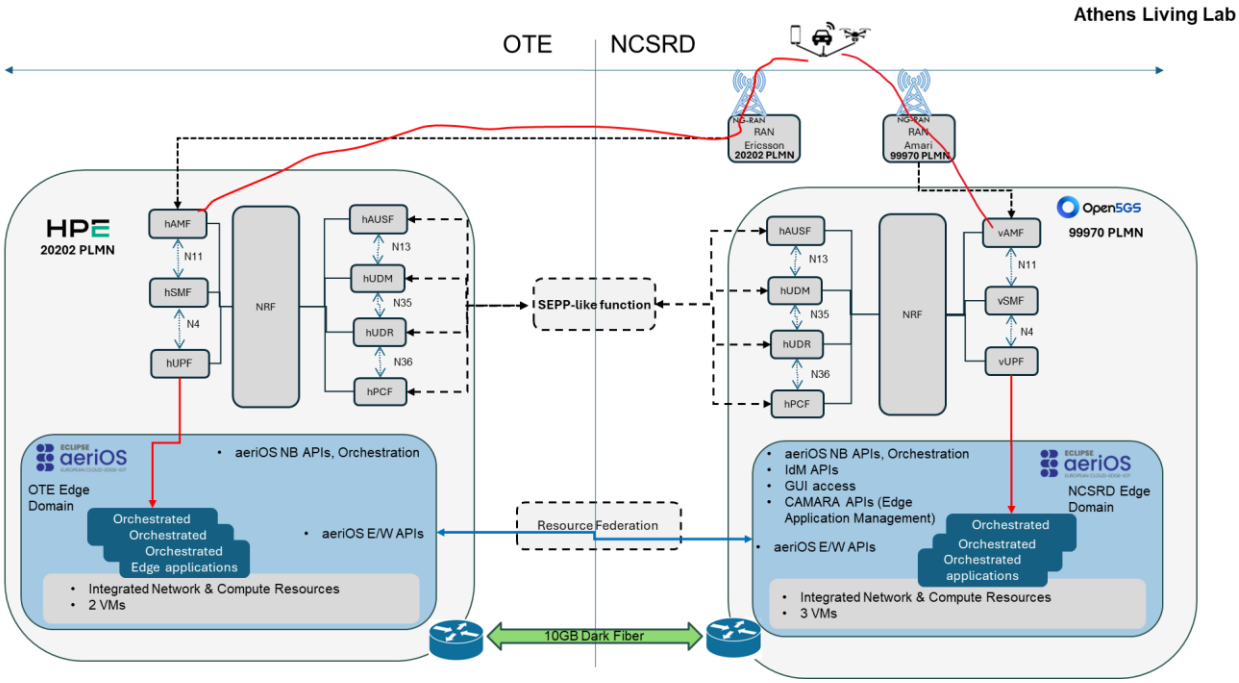


Figure 45: ENVELOPE Greek Site edge infrastructure topology.

Consistent with the aeriOS architectural design, the two domains operate as symmetrical peers within the continuum. Each domain exposes the same APIs and supports the same orchestration workflows, allowing services to be deployed on either site based on resource availability, policy constraints, or contextual information. Placement decisions are supported by the shared continuum-wide knowledge graph, which aggregates contextual information about resources, services, and capabilities across both domains.

In addition to the native aeriOS APIs, the platform also exposes higher-level interfaces designed to simplify interaction with the continuum for application developers and vertical service providers. CAMARA Edge Application Management APIs have been implemented on top of the aeriOS APIs, providing simplified mechanisms for onboarding, deploying, and managing applications at the edge. These APIs are exposed at https://camara-continuum.front-research-group.eu, allowing external services to request edge application deployment without directly interacting with the internal orchestration mechanisms. Applications deployed within the local site networks can also directly access all APIs (aeriOS and CAMARA) through their internal LAN addresses, enabling efficient integration with locally deployed services and network functions.

4.1.2 Description of vehicles and end devices

This subsection presents the vehicles and end devices used at the Greek Living Lab, including the onboard hardware, sensing equipment, and user devices involved in the ENVELOPE trials. The focus is on the physical and logical integration of equipment on the vehicles, as well as the connectivity between sensors, onboard computing platforms, and network interfaces.

4.1.2.1 Vehicles used in the Greek Living Lab

The Greek Living Lab includes one vehicle provided by ISFM (from Milla Group), which is used as the primary platform for the execution of GR-UC6. ISFM shuttles are built on proven and safety-

certified vehicle platforms that are already approved for operation on public roads. The integration of the autonomous driving system does not alter the vehicle's original safety functions, such as airbags, ABS, or steering assistance, which remain fully operational as provided by the base vehicle manufacturer.

The vehicles are designed to operate in two modes depending on operational requirements and route configuration:

- Manual mode, where the vehicle is driven by a human operator.
- Autonomous mode, where the vehicle operates as a driverless shuttle with remote supervision.

The ISFM automation system is designed to function without the need for dedicated infrastructure or specific road markings, allowing deployment in a wide range of real-world environments and facilitating flexible integration into existing road networks.

ISFM autonomous shuttle deployed on the Greek living lab is MILLA CAR, a passenger car M1 category.

Figure 46: MILLA CAR designed, constructed and equipped with autonomous driving system by ISFM (Milla Group).

The vehicle is equipped with an Onboard Unit (OBU) providing 5G connectivity and supporting the execution of ENVELOPE-related software components on the vehicle side. The OBU interfaces with the 5G Radio Access Network and enables communication between the vehicle and the ENVELOPE network and edge infrastructure.

Technical Specifications

- Vendor: VEDECOM
- Hardware Platform: Gateworks Venice 7401 SBC
- Modem: Quectel RM530Q-NL
- 3GPP Release: Release 16
- Supported Radio Technologies: 5G NR (Standalone compatible), Wi-Fi, Bluetooth (BT/BLE) and GNSS

External high-gain antennas are installed to optimize signal reception and transmission under vehicular mobility conditions.

Integration

The OBU interfaces with the vehicle's Autonomous Driving (AD) computer via Ethernet, ensuring high-bandwidth and low-latency data exchange are shown in Figure 47.

Figure 47: Onboard unit deployed in the autonomous vehicle.

Onboard Computing Platforms

In addition to the OBU, the vehicles integrate dedicated onboard computing systems supporting vehicle-side applications and ENVELOPE enablers.
Automation System Architecture consists of:

- A robotic ECU
- Perception sensors (AD) including LiDARs and cameras distributed:
  - On the front, roof, and sides of the vehicle
  - Inside the cabin above the driver position and laterally
- GNSS and cellular antennas mounted on the roof
- Independent electrical wiring harnesses, dedicated to automation and perception, installed in parallel to the original vehicle wiring

In addition to the OBU, the vehicles are equipped with onboard computing platforms used to host vehicle-side applications, data processing components, and, where applicable, ENVELOPE enablers requiring local execution.

Sensors and data sources

The vehicles are equipped with a set of sensors and data sources supporting the ENVELOPE use cases, as illustrated in Figure 48. These may include, but are not limited to, cameras, GNSS receivers, inertial sensors, and vehicle internal data sources.

The Milla Car relies on numerous external sensors, including LiDARs, radars, cameras, and GNSS and cellular antennas, which provide real-time data enabling the system to anticipate and respond appropriately to the behavior of various road users, particularly pedestrians and cyclists, while ensuring a high level of redundancy:

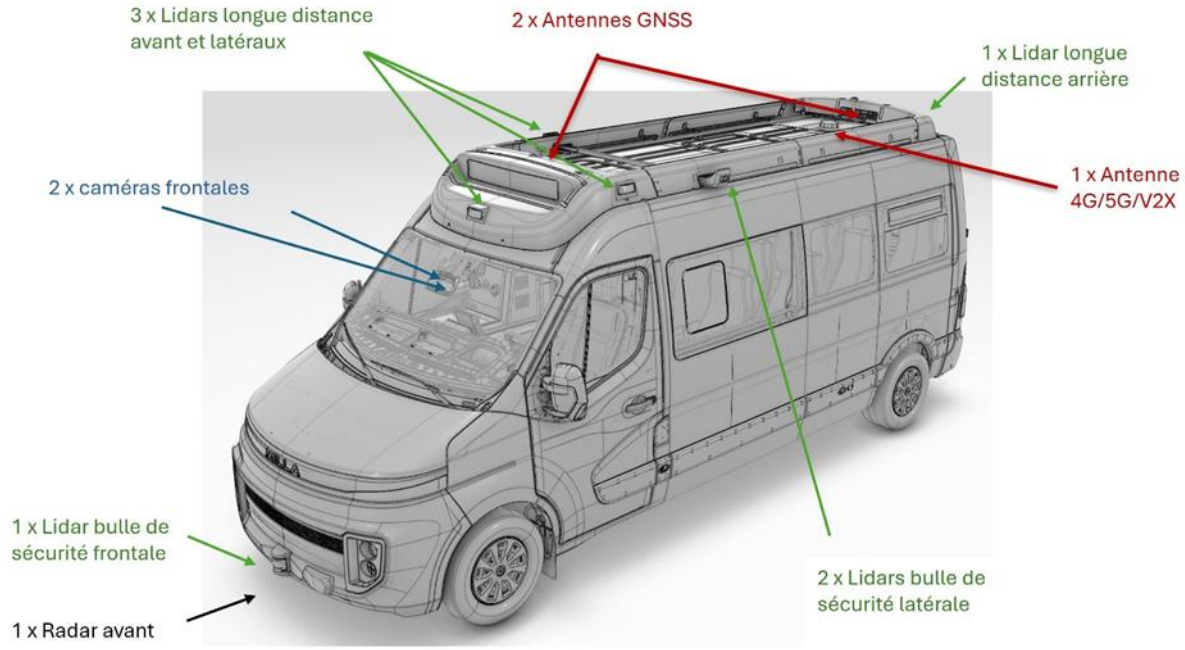


Figure 48: Overview of the vehicle's sensors.

In-vehicle integration and interconnection

All onboard components are interconnected through the vehicle's internal communication infrastructure. Sensor data and vehicle status information are made available onboard computing platforms and OBUs via native vehicle interfaces.

The vehicle relies on a combination of standard in-vehicle communication buses to ensure reliable data exchange between sensors, onboard computing units, and native vehicle systems:

- CAN bus (Controller Area Network): Used to access vehicle internal data (e.g., speed, steering angle, braking status) and to interface with native vehicle ECUs in a secure and controlled manner.
- Automotive Ethernet: Used for high-bandwidth data transmission, particularly for perception sensors such as LiDARs and cameras, as well as for communication between onboard computing units.
- USB interfaces: Used for specific peripherals, diagnostics, GNSS modules, or auxiliary communication devices where appropriate.

This multi-bus architecture ensures separation between safety-critical vehicle systems and autonomous driving subsystems.

4.1.2.2 Other UEs and end devices

Currently for the ENVELOPE Greek LL experiments at NCSRD premises the following UEs are available:

Huawei ELS-NX9, 5G smartphone: This is a Huawei smartphone used for testing the functionality of the components of the Inter-PLMN Handover scenario (HPE modified Open5gs core, External Antenna, HPE/OTE core, etc.) at NCSRD premises

A Xiaomi 15 ULTRA, used for the Situation Awareness CAM Application, with OS version 2.0.3.0 and Android 15.

Rpi + 5G Dongle modem (Waveshare) (2): Two Raspberry pi + Quectel RM530N-GL modems. The 5G Quectel modem has four attached antennas and a USB 3.1 port, through which it connects to the Rpi. Right now, these UE are plugged into a wall socket and are used for static testing of several components at NCSRD premises. A high-voltage battery can be attached to this setup so that it can become mobile and utilized for testing the handover scenario with the external antenna at NCSRD premises.

OBU of MILLA as a physical UE:

It is a Raspberry pi with extended capabilities, equipped with an RM530N-GL Quectel modem which is used for internal and external testing. The external testing is supported by an EcoFlow Delta 2 portable power station. It is used to connect to the setup at NCSRD premises.

4.2 Data collection & management

Within the Greek site ENVELOPE architecture, several data loggers are deployed to support the validation and evaluation of both individual components and the overall system. These logging points correspond to the APIs, the ENVELOPE technology enablers, and the site's primary use case components (GR-UC6). The placement of the data collection tools is illustrated in the figure below (Figure 49).

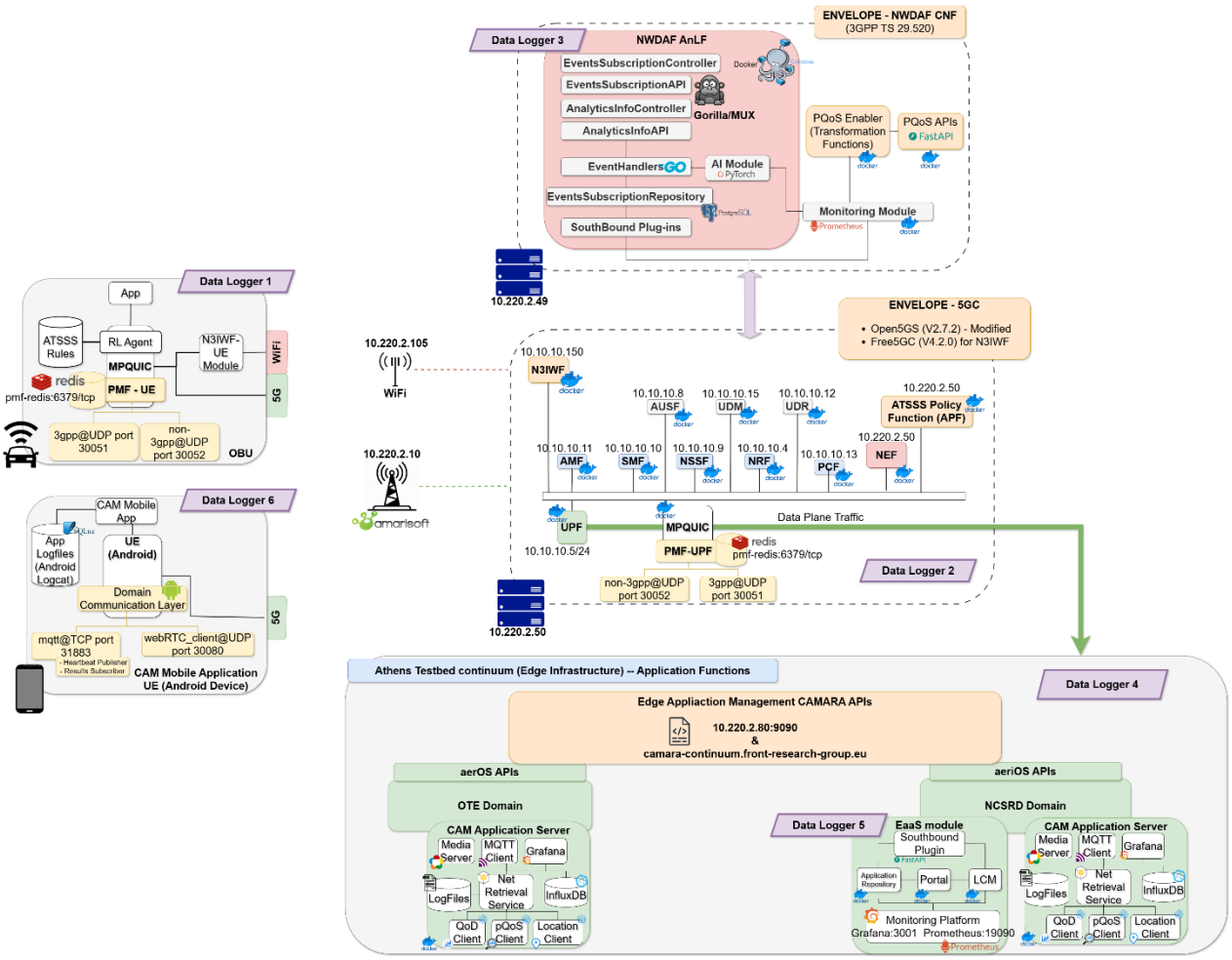


Figure 49: Data logging architecture at the Greek Trial Site.

In total, the architecture includes six distributed data loggers positioned across different system layers to capture operational, network, and application-level metrics. Each logger is associated with the software components deployed at its respective location. These logging points correspond to the key points of presence within the GR site infrastructure and support the monitoring of the KPIs defined in Deliverable 2.3. The collected data will subsequently be used for evaluation within the activities of WP6.

Data Logger 1 (On-Board Unit of the MILLA vehicle): The OBU hosts the software stack responsible for implementing the ATSSS functionality on the UE side. A local Redis-based database is deployed on the OBU to store measurements corresponding to network performance metrics, including latency (derived from ping measurements) and packet loss, when the

communication links are active during GR-UC6 operation. These measurements are collected through the Performance Measurement Function (PMF) developed in WP3 (see Deliverable D3.3 of ENVELOPE for further details).

The PMF Redis database can be accessed in real time, enabling the retrieval of live link performance metrics. The use of Redis is particularly suitable in this context due to its in-memory architecture, which enables low-latency data access and high write throughput, making it well suited for storing rapidly generated performance measurements from multiple network interfaces. Additionally, Redis provides lightweight data structures and efficient publish-subscribe mechanisms, facilitating real-time monitoring and integration with downstream monitoring tools.

The data logger deployed at this location collects ATSSS-related operational metrics used for monitoring the KPIs defined for the ATSSS enabler, including component availability, session setup time, packet statistics per link (sent, received, and lost), achieved latency and throughput, and the reinforcement learning (RL) decision time. Logging is event-driven and is triggered when an ATSSS session is initiated by an application function.

In addition, the same Redis-based database instance is used to collect pQoS-related operational metrics associated with the interaction between the OBU-side components and the predictive QoS service. These metrics are used to monitor the KPIs defined for the pQoS enabler, including service availability, request end-to-end response time, and the end-to-end latency between CAM-safe mode initialization and the reception of a QoS degradation event notification from the edge server. In this context, the logger records pQoS service requests and corresponding responses, enabling the measurement of whether requests are successfully served, the elapsed time between request issuance and response reception, the elapsed time between CAM-safe mode initialization and QoS degradation event notification reception, and the prediction accuracy, which is calculated by logging the predictions generated by the pQoS service together with the corresponding actual OBU network metrics and comparing the predicted degradation events against the observed network behaviour.

The collected data is structured in JSON and/or CSV format and forwarded to the Prometheus server for storage and further analysis.

Data Logger 2 (Open5GS Core Network - VPLMN): Like Data Logger 1, part of the ATSSS monitoring framework is deployed within the CN infrastructure. At this location, a Redis-based logging instance is used following the same approach described for the OBU, enabling the collection and temporary storage of ATSSS-related operational measurements generated by the network-side components of the framework, logs regarding Inter-PLMN, QoD, Device location APIs and other CN metrics. This logger complements the OBU-side measurements by capturing performance and session-related events observed within the 5G core network.

Data Logger 3 (NWDAF Layer): At the NWDAF layer, a Redis-based database is deployed to log analytics- and prediction-related events generated by pQoS framework. In particular, this logger records every detected QoS degradation event, as well as every prediction notification indicating that a future QoS degradation is expected to occur. By correlating predicted and observed degradation events, the logger enables the calculation of the confusion matrix and, consequently, improve overall prediction accuracy.

In addition, this logging instance supports the computation of the pQoS availability rate. In this context, a successful pQoS request is defined as any inference service response returned with an

HTTP status code in the 2xx range. Based on these records, the logger can determine the proportion of successful pQoS responses over the total number of prediction requests issued.

Furthermore, the logger collects and stores a set of model-level and service-level performance indicators, including average inference confidence, inference response time, training convergence time, and training loss. These measurements provide visibility into both the runtime behavior and the training efficiency of the pQoS models deployed at the NWDAF layer. The collected data can be structured in JSON and/or CSV format and forwarded to the Prometheus server for persistent storage, monitoring, and further analysis.

Data Logger 4 (aeriOS Framework) and Application Functions Layer: The Edge KPI logger will be implemented as a dedicated test application deployed within the local environment, with network access to the CAMARA Edge Application Management APIs exposed by the enabler. Its role will be to execute controlled and repeatable API-based test sequences and to record the data required for the evaluation of KPI1 to KPI4, namely enabler availability, application deployment time, handover time, and handover success rate. Depending on the KPI under evaluation, the logger will issue periodic health or functional probes, deployment requests, instance status queries, and handover-related operation sequences, while capturing the corresponding timestamps, response codes, identifiers, and observed lifecycle states.

To support scalability validation, the KPI logger can operate in parallel with a separate load-generation script that produces configurable background API load, for example through repeated onboarding, retrieval, and deletion of application manifests, as well as repeated queries to supporting endpoints such as edge zone discovery. In this way, the same KPI measurements can be collected under both no-load and high-load conditions, allowing the behaviour of the enabler to be compared under different operational stress levels. The overall process is parameterized, so that aspects such as the number of iterations, polling interval, timeout values, workload configuration, handover mode, request rate, and concurrency level can be adjusted without changing the implementation.

The produced measurements will be stored in structured output files, such as JSONL or CSV, containing one record per probe or test attempt, together with summary records per run. These files can then be processed offline by dedicated parsing scripts in order to derive the final KPI values and supporting statistics, such as average and percentile deployment or handover times, success rates, and availability estimates. The full campaign can also be automated through a shell script orchestrator, which sequentially executes the KPI measurement runs, optionally activates the background load generator, and invokes the post-processing scripts to produce consolidated result files. This enables repeatable execution of the full validation workflow and facilitates systematic comparison of the measured KPI values across configurations and test campaigns.

As part of the aeriOS, a set of backend scripts are implemented to support the communication of the CAM application and the OBU with relevant network metrics via the aforementioned CAMARA APIs. This script regards the NetworkRetrieval script. The script received heartbeats from subscribers, which are registered internally. For all active subscribers the script keeps a log regarding their subscription datetime and status. Periodically it communicates with the CAMARA APIs and request location and pQos information as well as QoD when prompt. All the requests and replies are also stored. All storing is performed locally, in JSON related format. Moreover, an

InfluxDB is installed as part of the backend services, that stores data produced via the Android device and CAM application (see Data Logger 6). Those data are accessible either directly or from the Grafana framework.

Data Logger 5 (EaaS module): The EaaS module has been deployed at the NCSRD domain and includes the following components: EaaS Portal, the Application Repository, the Lifecycle Manager (LCM), the Southbound Plugin, and the Monitoring Platform (please see D3.1 for more details). The relevant components of EaaS module are designed to compute and expose KPI-related metrics with dedicated Prometheus endpoints. A Prometheus server instance scrapes these endpoints, collects and stores time-series metrics for the KPI evaluation. The collected data supports the evaluation of EaaS availability, application onboarding time and experiment startup and stop times.

Data Logger 6 (Android device on vehicle): The CAM application on board of the back car generates continuous data regarding its status and relevant Network metrics. Specifically, the application sends periodic heartbeat messages via the MQTT, in order for the backend service (Net Retrieval Service) at Data Logger 5 to register the specific device. Additionally, the application locally stores all network related values produced, including RSRP, RSRQ, RSSNR, RTT and ping metrics. It also stores data regarding the location and speed. Regarding the KPI measurement, the RTT (Round Trip Time) metric, is utilized to verify flowless and low latency connection between the device and the services provided. All data are stored locally in the device in SQLite database. Data are accessible via the Grafana framework or receive the raw data from the InfluxDB stored in the backend services, as described in Data Logger 5.

The following tables (Table 19 - Table 25) summarize the logging configuration used for the monitoring and evaluation of the defined KPIs. For each technology enabler, the tables indicate the data loggers involved in KPI monitoring, their location in the system architecture (point of presence-PoP), the type and format of the collected telemetry, and the corresponding logging policies. The listed log data fields represent the set of telemetry attributes used for KPI computation across the involved data loggers. Depending on the deployment configuration and the specific use case, individual data loggers may collect only the subset of fields relevant to their role in the system architecture. The collected telemetry is stored in a Prometheus server and can be exported in JSON or CSV format for further analysis.

Table 19: ATSSS data logging framework.

Item	Description
KPI ID	KPI_Enab_Availability, KPI_Enab_ATSSS_SuccessRate, KPI_Enab_ATSSS_SetupTime, KPI_Enab_ATSSS_PLRreduction, KPI_Enab_ATSSS_LatencyReduction, KPI_Enab_ATSSS_ThroughputGain, KPI_Enab_ATSSS_RLDecisionTime.
Data logger / PoP involved in KPI monitoring	Data Logger 1 ISFM (OBU), Data Logger 2 (CN), Data Logger 4 (Edge).

Log data type	ATSSS components availability, ATSSS session initiation timestamp, ATSSS packets sent/received/lost per link, ATSSS achieved latency per link, RL decision time.
Log format	JSON *and/or* CSV.
Logging event	Activated when ATSSS session is triggered and used by an application function.
Sampling policy	Event driven.
Logging storage	Redis, Prometheus server.
Log data fields (consolidated across involved data loggers)	session_id, session_status, timestamp, link_id, packets_sent, packets_received, packets_lost, latency, throughput, session_setup_time, RL_decision_time.

The ATSSS-related metrics are computed as follows:

- **session_id (string)**: unique identifier of the ATSSS session for a UE.
- **session_status (string)**: current state of the ATSSS session, for example initiated, active, failed, or terminated.
- **timestamp (datetime)**: time when the event or measurement was recorded.
- **link_id (string)**: identifier of the access link/path used by ATSSS, for example cellular, Wi-Fi, or another access leg.
- **packets_sent (numeric):** number of packets transmitted per ATSSS link
- **packets_received (numeric):** number of packets successfully received per ATSSS link.
- **packets_lost (numeric):** number of packets lost per ATSSS link
- **latency (ms):** measured packet delay on a specific ATSSS link**.**
- **throughput (Mbps):** achieved data rate on a specific ATSSS link, or aggregated over multipath.
- **session_setup_time (s):** elapsed time from ATSSS session trigger to successful session establishment.
- **RL_decision_time (ms)**: time needed by the RL engine to calculate and output the steering/splitting decision.

Table 20: pQoS data logging framework.

Item	Description
KPI ID	KPI_Enab_pQoS_EffectLatency KPI_Enab_pQoS_SuccessRate KPI_Enab_pQoS_InferenceAccuracy KPI_Enab_pQoS_InferenceIQNMessages KPI_Enab_pQoS_InferenceDuration KPI_Enab_pQoS_InferenceConfidence KPI_Enab_pQoS_TrainingDuration KPI_Enab_pQoS_TrainingConvergence KPI_Enab_pQoS_CAMSafeModeTA

Data logger / PoP involved in KPI monitoring	Data Logger 1 ISFM (OBU), Data Logger 3 (Edge). Logger 4 (NWDAF)
Log data type	Timestamped probe results (HTTP status code, success/failure outcome), timestamped prediction evaluation records, timestamped service availability metrics, timestamped model evaluation metrics, timestamped training execution metrics
Log format	JSON *and/or* CSV.
Logging event	Activated when: (a) a PQoS service subscription or request is enabled (or existing is modified), (b) a PQoS notification or response is triggered, (c) a training request is enabled, (d) a training completion event occurs. (e) a backend event/failure occurs.
Sampling policy	Event driven.
Logging storage	Redis, Prometheus server.
Log data fields (consolidated across involved data loggers)	Timestamp, http_probe_target, http_status_code, EffectLatencyMs, pqos_predicted_values, pqos_actual_values, pqos_notification_sink, InferenceDurationSec, InferenceConfidenceUnit, TrainingDurationSec, GpuFlag, RetrainFlag, TrainLoss, SafeModeFlag, ActualQoSValues, QosValueType, QosValueUnit.

The pQoS-related metrics are computed as follows:

- **Timestamp (datetime):** Recorded whenever a logging event occurs (e.g., service request, prediction generation, training execution, or notification reception). This field enables chronological correlation of events and supports the computation of time-based KPIs.
- **http_probe_target (string), http_status_code (numeric):** Captured during interactions with the pQoS service endpoints. The http_probe_target identifies the specific service instance or endpoint being accessed, while the http_status_code indicates the response status of the request. Responses with status codes in the 2xx range are considered successful and are used to compute the pQoS service availability and success rate KPIs.
- **pqos_notification_sink (string):** Identifies the destination endpoint that receives prediction or QoS degradation notifications generated by the pQoS service.
- **pqos_predicted_values (list of numeric), pqos_actual_values (list of numeric), ActualQoSValues (list of numeric):** These fields store the predicted QoS values produced by the pQoS inference model and the corresponding actual network measurements collected by the OBU. By comparing predicted and observed values, the system evaluates the prediction accuracy and constructs the confusion matrix used for inference accuracy KPIs.
- **QosValueType (enum: Throughput DL, Throughput UL, Latency), QosValueUnit (enum: Mbps, ms, etc.):** Specify the type of QoS metric being evaluated and the unit in which the metric is expressed.
- **SafeModeFlag (boolean):** Indicates whether the CAM application entered safe mode in response to predicted or detected QoS degradation events.
- **EffectLatencyMs (numeric):** Computed as the elapsed time between CAM safe mode initialization and the reception of the QoS degradation notification from the pQoS service.

- **InferenceDurationSec (numeric), InferenceConfidenceUnit (numeric):** These fields capture the runtime performance and reliability of the pQoS inference process. InferenceDurationSec represents the elapsed time between receiving an inference request and generating the prediction response, while InferenceConfidenceUnit records the confidence score produced by the model for the predicted QoS degradation event.
- **TrainingDurationSec (numeric), TrainLoss (numeric), GpuFlag (boolean), RetrainFlag (boolean):** These fields capture the execution characteristics and outcome of the model training process. TrainingDurationSec measures the elapsed time between the start and completion of a training task, while TrainLoss represents the final loss value indicating model convergence quality. GpuFlag specifies whether GPU acceleration was used during training, and RetrainFlag indicates whether the training corresponds to a retraining cycle triggered by updated datasets or model performance degradation.

Table 21: Inter-PLMN data logging framework.

Item	Description
KPI ID	KPI_InterPLMN_AppRTT KPI_InterPLMN_ServiceDisruption
Data logger / PoP (location in architecture)	Data Logger 2: Open5GS Core Network - VPLMN Data Logger 4: aeriOS Framework Data Logger 6: Android device on vehicle
Log data type	MQTT RTT Messages, NGAP Messages
Log format	JSON *and/or* CSV, PCAP files
Logging event	Activated when the UE mobile App starts, Activated when the UE's PDU session in the HPLMN drops and a new PDU session is established in the VPLMN
Sampling policy	periodic (1 s, 200ms etc.)
Logging storage	InfluxDB, Local storage (Sqlite3),
Log data fields (consolidated across involved data loggers)	application_id, lpn, send_time_epoch, received_time_epoch, trip_time, UEContextReleaseRequest, UTC Arrival Time, PDUSessionResourceSetupResponse.

The InterPLMN-related metrics are computed as follows:

- **application_id (string):** Unique identifier set to distinguish the application .
- **lpn (string):** Unique identifier or plate numbers related to the UE running the application.
- **send_time_epoch (string):** Timestamp (in epoch format) recorded at the moment an application-layer message is transmitted from the UE. Used as the reference start time for RTT calculations.
- **received_time_epoch (string):** Timestamp (in epoch format) recorded when the corresponding response or message is received. Used together with the send timestamp to compute latency-related KPIs.
- **trip_time (ms):** Computed as the difference between *received_time_epoch* and *send_time_epoch*. Represents the application-level Round Trip Time (RTT) in milliseconds.

- **ping (ms):** A network-level latency measurement obtained via ICMP echo requests between the UE and a target endpoint. It provides a baseline connectivity indicator independent of application-layer protocols.
- **UEContextReleaseRequest (string):** Extracted from NGAP signaling messages. In an inter-PLMN mobility context, this message is used to detect when the UE context is released in the HPLMN. The release of the UE context marks the start of a potential mobility event, such as a handover or re-registration, which may lead the UE to connect to a visited PLMN (VPLMN). Monitoring this message allows tracking the beginning of service disruption or transition, serving as a reference point to correlate with subsequent signaling events during the mobility procedure.
- **UTC Arrival Time (string - ISO 8601 timestamp in UTC, with fractional seconds):** The exact UTC timestamp when a signaling or application event is recorded by a data collection system. In an inter-PLMN scenario, this field is essential to track the timing of all messages, ensuring proper sequencing and correlation of events across HPLMN and VPLMN. It enables precise reconstruction of mobility procedures and the duration of service interruptions, supporting analysis of roaming transitions and handovers.
- **PDUSessionResourceSetupResponse (string):** Extracted from NGAP signaling messages. In an inter-PLMN mobility context, this message is used to detect when a PDU session has been successfully established in the target VPLMN. Completion of PDU session setup indicates that the UE has reconnected in the new network and marks the end of the service disruption or mobility transition. By correlating this message with preceding UEContextReleaseRequest events, it is possible to measure the duration of the service interruption and confirm that the UE is fully operational in the visited PLMN.

Table 22: EaaS data logging framework.

Item	Description
KPI ID	KPI_EaaS_Availability KPI_EaaS_AppOnboardingTime KPI_EaaS_ExperimentStartupTime KPI_EaaS_ExperimentStopTime
Data logger / PoP (location in architecture)	Data logger 5: EaaS module Server (in NCSRD domain where EaaS components are deployed)
Log data type	EaaS components availability, application package onboarding time, experiment start and stop duration times
Log format	Time-series Prometheus metrics
Logging event	Periodic logging, application package onboarding, experiment start and stop
Sampling policy	Periodic (1 s) and event driven
Logging storage	Prometheus
Log data fields (consolidated across involved data loggers)	PortalAvailable, AppRepositoryAvailable , ExperimentLcmAvailable, MonitoringAvailable, AppUploadingTimeDuration, AppProcessingTimeDuration, AppSouthboundTimeDuration, ExperimentStartupTimeDuration, ExperimentStopTimeDuration.

The EaaS-related metrics are computed as follows:

- **PortalAvailable, AppRepositoryAvailable, ExperimentLcmAvailable, MonitoringAvailable (binary):** Are binary indicators computed via periodic health-checks functions to reflect the operational status of each EaaS component.
- **AppUploadingTimeDuration (ms):** Capture the duration of the application package upload and validation process, from the moment the HTTP request is received until the artifact is stored in the database.
- **AppProcessingTimeDuration (ms):** Computed by measuring the duration from when the application repository starts downloading the app package from the database, validating its content, and uploading each app package artifact to the database.
- **AppSouthboundTimeDuration (ms):** Captures the time from when the application repository sends the onboarding request to the Southbound Plugin until it is complete.
- **ExperimentStartupTimeDuration (ms):** Measure the elapsed time between the reception of a start command and the confirmation that all underlying experiment resources have reached a running state.
- **ExperimentStopTimeDuration (ms):** Measure the elapsed time between the reception of a stop command and the confirmation that all underlying experiment resources have reached a terminated state.

Table 23: Edge data logging framework.

Item	Description
KPI ID	KPI_Enab_availability KPI_Enab_Edge_ServiceDeploymentTime KPI_Enab_Edge_HandoverTime (Option: Make-before-break) KPI_Enab_Edge_HandoverTime (option: Break-before-Make) KPI_Enab_Edge_HandoverSR
Data logger / PoP (location in architecture)	Data logger 5 (aeriOS framework) / ENVELOPE CAMARA APIs
Log data type	Timestamped records of deployment request, status polling, handover procedure events, CAMARA API responses, event records derived from CAMARA API interactions an associated metadata Timestamped event records for each handover attempt plus CAMARA polling observations (optional)
Log format	JSON.
Logging event	Probes run at fixed intervals, counting success based on timely 2xx HTTP responses. Deployments start with a CAMARA API request, followed by polling until the instance is ready. Start / outcome of the handover procedure.
Sampling policy	Periodic sampling during the defined service time window for the test run (e.g., every 5 seconds). Event-based logging for every deployment request / handover attempt executed during the test scenario.

Logging storage	Local structured log file (JSON or CSV) stored on the test controller host, organised per test run and workload configuration (no-load / high-load). Local structured log file (JSONL) stored on the test controller host, organised per test run and workload configuration (no-load / high-load).
Log data fields (consolidated across involved data loggers)	request_timestamp, response_timepstamp , response_status, availability_window, successful_response, deploy_request_timestamp, app_registration_timestamp, instance_ready_timestamp, deployment_outcome, target_edge_id, handover_request_timestamp, source_edge_id, new_instance_ready_timestamp, handover_outcome, handover_attempt_id, failure_reason.

The Edge/MEC-related metrics are computed as follows:

- **request_timestamp (datetime):** Time at which the API request is issued by the KPI logger.
- **response_timestamp (datetime):** Time at which the corresponding API response is received.
- **response_status (string):** Outcome of the API call (e.g. HTTP status or mapped status such as *success*, *failed*).
- **availability_window (ms):** Time interval between request and response, used to assess responsiveness within the availability measurement window.
- **successful_response (boolean):** Indicates whether the request was successfully processed (e.g. HTTP 2xx).
- **deploy_request_timestamp (datetime):** Time at which an application deployment request is initiated.
- **app_registration_timestamp (datetime):** Time at which the application is successfully registered/onboarded in the system (if applicable).
- **instance_ready_timestamp (datetime):** Time at which the application instance reaches the *ready/running* state.
- **deployment_outcome (string):** Final result of the deployment attempt (e.g. *success*, *failed*).
- **target_edge_id (string):** Identifier of the edge node or domain where the application is deployed.
- **handover_request_timestamp (datetime):** Time at which the handover (migration) process is initiated.
- **source_edge_id (string):** Identifier of the source edge node/domain where the application was initially running.
- **new_instance_ready_timestamp (datetime):** Time at which the new application instance in the target edge reaches the *ready* state.
- **handover_outcome (string):** Final result of the handover attempt (e.g. *success*, *failed*).
- **handover_attempt_id (string):** Unique identifier assigned to each handover attempt for traceability.
- **failure_reason (string):** Optional description of the reason for failure in deployment or handover cases.

Table 24: Device Location data logging framework.

Item	Description
KPI ID	KPI_Enab_availability KPI_Enab_DL_EffectLatency KPI_Enab_DL_CellidLocationRetrievalSR
Data logger / PoP (location in architecture)	Data Logger 4: aeriOS FrameworkData Logger 2: Open5GS Core Network - VPLMN
Log data type	Device Location API logs Device Location API logs, NEF MonitoringEvent API Logs Device Location API
Log format	Log file
Logging event	Deployed server, able to serve requests Location Information Requested Successful Location Retrieval requests
Sampling policy	Per http request
Logging storage	Docker logs
Log data fields (consolidated across involved data loggers)	request_timestamp, response_timestamp, response_success, effect_latency, ue_identifier, retrieval_success, cell_id, failure_reason

The Device Location Enabler metrics are computed as follows:

- **request_timestamp (datetime):** Computed via a fetched timestamp that corresponds to the request datetime via container logs
- **response_timestamp (datetime):** Computed via a fetched timestamp that corresponds to the response datetime via container logs
- **response_success (boolean):** Computed via http operations that responds with Positive status codes to the request that has been performed.
- **effect_latency (ms):** Computed by tracking the time it takes for the enabler to query the underlying 5G core network for location data and receive the answer, excluding internal API overhead.
- **ue_identifier (string):** The identifier that comes from the request body that a location is requested
- **retrieval_success (boolean):** Computed via the condition if a location retrieval was successful or not
- **cell_id (string):** A string value that comes from the Open5Gs Core and corresponds to the cell_id that the requested UE is connected.
- **failure_reason (string):** A string value that correspond to the reason a location request has failed and computed via the error messages that the application triggers.

Table 25: QoD data logging framework.

Item	Description
KPI ID	KPI_Enab_QoD_QoSSessionRetainability

	KPI_Enab_QoD_QoSProfileEstablishmentTime KPI_Enab_QoD_SuccessRate KPI_Enab_availability
Data logger / PoP (location in architecture)	Data Logger 2: Open5GS Core Network - VPLMN
Log data type	QoD API logs, open5gs PCF logs, open5gs AMF logs
Log format	JSON/plain text
Logging event	On each QoD request On session create and session end/expiry (or delete) On each QoD API request completion On each QoD API request
Sampling policy	Event-driven
Logging storage	In-memory (runtime)
Log data fields (consolidated across involved data loggers)	session_id, qos_profile, session_start_time, session_end_time, total_session_duration, request_timestamp, api_http_status, qos_established, pcf_http_status, response_timestamp, successful_response

The Quality-on-Demand Enabler metrics are computed as follows:

- **session_id (string)**: A unique identifier assigned to each QoS session, generated as a UUID when a QoD request is initiated and used to track the session throughout its lifecycle.
- **qos_profile (string)**: A string value that specifies the requested Quality of Service profile (e.g., QOS_E, QOS_L, QOS_M, QOS_S) extracted from the QoD API request body.
- **session_start_time (datetime)**: Computed via a fetched timestamp that corresponds to the datetime when the QoS session was successfully created, captured when the session status transitions to AVAILABLE.
- **session_end_time (datetime)**: Computed via a fetched timestamp that corresponds to the datetime when the QoS session is scheduled to expire or has been terminated, calculated based on session start time plus the granted duration.
- **total_session_duration (ms)**: Computed as the total session duration in milliseconds.
- **request_timestamp (datetime)**: Computed via a fetched timestamp that corresponds to the datetime when the QoD API request was received and session creation was initiated.
- **api_http_status (numeric)**: A numeric value extracted from the HTTP response indicating the status code returned by the QoD API (e.g., 201 for created, 200 for success, 404 for not found, 403 for forbidden).
- **qos_established (boolean)**: Computed via a boolean condition that evaluates whether the QoS session was successfully established in the 5G core network, based on the NEF/PCF response status code being 200 or 201.
- **pcf_http_status (numeric)**: A numeric value extracted from the NEF/AsSessionWithQoS API response indicating the HTTP status code returned by the Policy Control Function (e.g., 201 for successful QoS policy creation, 404 for not found, 500 for internal error).

- **response_timestamp (datetime)**: Computed via a fetched timestamp that corresponds to the datetime when the QoD API response was sent back to the client after processing the session request.
- **response_status (string)**: A string value that corresponds to the current QoS session status (REQUESTED, AVAILABLE, or UNAVAILABLE) as tracked in the SessionInfo model and updated based on network events.
- **successful_response (boolean)**: Computed via a boolean condition based on HTTP operations that respond with positive status codes (2xx) to the QoD request that has been performed.

5 Technical guidelines for data management

Data is fundamental to every phase of a project lifecycle. From design and requirements elicitation through development, validation, and evaluation, data plays a critical role in ensuring each phase is executed effectively and informs subsequent activities. Notably, the data required for later stages of a project, such as evaluation that will be conducted in WP6 for ENVELOPE, must be defined, or at least anticipated, during the initial phases of the project, particularly during design and requirements specification.

The following section proposes guidelines for data management, with a particular focus on evaluation and validation data. These guidelines have been derived from the expertise and best practices gathered from ENVELOPE partners across past and ongoing projects. Consequently, they are tailored to research and innovation contexts and are aligned with the application of the FAIR principles [1], in accordance with the ENVELOPE project Data Management Plan (DMP) defined in Deliverables D1.4: Data management plan [2], and D1.5: Updated data management plan [3].

As a basis, the following data-related activities are proposed, to structure the guidelines and group them into specific categories

- Data acquisition
- Data transmission
- Data pre-processing
- Data protection and storage

Accordingly, the following section should be understood as a set of guidelines to be considered and tailored, as necessary, to the specific requirements, constraints, and operational limitations of each Living Lab.

5.1 Guidelines for data acquisition

Data collection requires a clear definition of the why, what, when, and how. This section outlines the minimum requirements that should be addressed prior to the actual acquisition of data.

5.1.1 Define clear data collection objectives and requirements

This initial phase is critical for all subsequent stages through which the data will pass (e.g., transmission, quality control, storage, and evaluation). Clearly defining the purpose of data collection is essential to formulate the specific questions and hypotheses to be addressed and to determine how the data will support decision-making processes, the evaluation of KPIs/KVIs (based on D.2.1) and the overall project objectives. This, in turn, enables the selection of appropriate data formats, sampling rates, and other relevant data characteristics.

During this introductory stage, general logging data requirements must be defined, including specifications for formats, units, value ranges, resolution, and data sources or locations. These requirements should cover both technical data (e.g., UE-network-edge and CAM-related messages/data) and non-technical data (e.g., user questionnaires), where applicable.

Data quality requirements such as accuracy, precision, consistency, reliability, and acceptable levels of missing or incomplete data should also be clearly defined. Although these aspects are sometimes overlooked, they play a critical role in selecting appropriate sensors and ensuring that the collected data meets the expectations of subsequent analysis and evaluation activities. For example, vehicle positioning data obtained through the ENVELOPE/CAMARA Location API may require detailed User Equipment (UE) accuracy information when supporting use cases that depend on close positioning. In contrast, use cases focused on area-based constraints, such as geofencing, may be sufficiently supported by higher-level CAMARA Geofencing APIs, where strict positional accuracy requirements are not necessary.

Legal and ethical requirements shall be fully considered. Data handling must comply with applicable local, national, international, and/or global data protection regulations, depending on the scope and impact of the solution (see Section 5.4.1 for further details). Requirements should explicitly address aspects such as consent, bias and discrimination, transparency, data ownership, and consent revocation.

Within ENVELOPE, data logging serves two primary objectives: system validation and the evaluation of trial scenarios. Purpose-specific logging requirements have been defined for each objective, LL and use case. Where possible, the most restrictive or demanding requirements have been adopted to satisfy both objectives, thereby optimizing the overall data acquisition process.

5.1.2 Establish sampling rates, data formats and relevant information

Once the data logging objectives are clearly defined within the ENVELOPE framework, the next step is to identify the specific information to be captured. Depending on the KPIs/KVIs to be measured or the validation activities and trials to be performed, logging mechanisms and tools must be deployed at appropriate points in the system architecture (see also Sections 2.2, 3.2 and 4.2) to capture the required data elements. For instance, considering the objective is to measure an ENVELOPE enabler KPI, such as KPI_Enab_ATSSS_SetupTime, KPI_Enab_QoD_QoSProfileEstablishmentTime or KPI_Enab_DL_EffectLatency (see also D2.3 [4]), where these KPIs correspond to the setup time of an ATSSS or a QoD session, or the time required to retrieve the location of a UE, respectively. As these CAMARA-based (or MEC-based) APIs operate across multiple layers of the ENVELOPE network–compute continuum, data logging probes must be appropriately deployed to capture multi-level information related to both successful session establishment and failure conditions. As a further example, if the objective is to measure a network latency KPI, it is required to log a minimum of a timestamp and an identifier in the source and the destination points.

In line with this guideline, it is advisable to define and log a minimum common set of data across all logging activities. As a general rule, each log entry should include at least the measured value or metric, a timestamp indicating when the data was collected, and a unique identifier to enable correlation and traceability. These will be further calibrated according to the KPI/KVI requirements of the Trial sites, their respective infrastructure and use cases of focus.

A common logging mistake is to focus exclusively on the technical data required for evaluation, while overlooking complementary metadata that is essential for correctly selecting relevant records from large volumes of log entries. Typical examples include the data source and the timestamp: although these attributes may not be strictly necessary for the computation of a specific KPI, they

are crucial for filtering, processing, correlating, and comparing the appropriate subsets of technical data during validation and evaluation activities.

According to the Research Data Management (RDM) [5], mandatory for any Horizon Europe project, the metadata shall:

- At least include the following fields: author(s), dataset description or abstract, date of dataset deposit or publication date, dataset deposit venue, dataset license and dataset embargo period (if any).
- Also include information about Horizon Europe funding: grant project name, acronym and number. Ideally, the repository will have dedicated fields for this information. If not, you can include them in other appropriate fields, such as the abstract

In ENVELOPE, such mandatory metadata is expected to be added to the datasets during the upload process to the data warehouses (not at the collection phase) together with additional contextual information (metadata) such as the use case, the trial scenario, and the device or component responsible for logging.

Selecting appropriate data characteristics (e.g., numerical, categorical, textual) and clearly defining data units (particularly for numerical values) is essential to facilitate subsequent data management activities, including quality control, data filtering, and storage. For example, positioning data may be logged using different formats and reference systems, such as decimal degrees (WGS84), degrees–minutes–seconds (WGS84), or UTM coordinates. In practice, achieving a uniform logging format across systems can be challenging, as data representations often depend on the specific sensors or measurement devices employed (see Section 5.1.3). A similar issue arises with timestamps, which can be expressed in multiple formats. To ensure consistency across datasets, a common convention should be adopted for all logging activities. Unix time is a particularly effective choice, as it supports high temporal resolution, including microsecond or nanosecond granularity.

Depending on the use case and trial site, the volume of generated logging data can be substantial. Accordingly, requirements related to logging frequency and data volume should be defined when constraints in transmission, processing, and/or storage impose strict limitations and cannot accommodate excessive data loads. Typical examples of high-volume logging data include vehicle positioning information and records of transmitted and received messages, including video data, LIDAR data, rosbag data and so on.

5.1.3 Select appropriate sensors and measurement devices

The selection of appropriate sensor types and their placement for data acquisition plays a critical role in meeting both data requirements and the overall scope of logging activities including KPI/KVIs data collection. Sensor types vary according to the logging environment (e.g., outdoor, indoor, software-based, or hardware-based), measurement objectives (physical or digital), and data requirements such as range, accuracy, and resolution. These may include, for example, LIDAR sensors, GNSS devices, or software application loggers for the 5G CN or edge computing infrastructure and orchestration framework (e.g., Kubernetes logs).

Sensor and logger placement must be determined in alignment with the data management objectives, as system validation and evaluation often require the correlation of data captured at

multiple points. For instance, latency evaluation typically relies on the direct comparison of metrics collected at both source and destination locations.

For ENVELOPE, we consider software-based loggers for the validation of the project's network-compute continuum system, the APIs, technology enablers and the evaluation of the scenarios. In few cases raw data from sensors is recorded. Since within the CCAM world most of this data is merged and encapsulated in C-ITS messages, these messages are the final logged metrics for analysis, as they meet the data requirements for minimum sampling rate and accuracy.

5.1.4 Implement calibration and time synchronization

It is mandatory to verify the calibration of the selected sensors and measurement devices to ensure the accuracy of collected metrics. Calibration is a periodic activity and shall be incorporated into the regular maintenance procedures for all devices and equipment, were necessary.

As data logged from different components and devices is often compared or combined during validation and evaluation activities, all logged data shall be time-synchronized, where necessary. Synchronization should preferably be performed at the logger or component level using a common time reference, such as internal clocks, public or private NTP servers, GNSS-based timing, or equivalent mechanisms.

When a single time reference cannot be used, for example, because some components lack it is essential to ensure that the different time sources are sufficiently synchronized (e.g., within microsecond- or nanosecond-level accuracy) to allow meaningful comparison of timestamps across data sources.

If such synchronization cannot be guaranteed, time correction shall be applied during post-processing, provided that the time offsets between the different references can be accurately determined.

5.2 Guidelines for data transmission

Once logging data has been collected, it may either be stored locally (e.g., at the OBU or edge server) within the logging component or transmitted to an external data warehouse.

As discussed in previous sections, the volume of logged data can be substantial in certain scenarios or use cases. This necessitates the selection of a transmission protocol that is both efficient and sufficiently robust to meet performance and reliability requirements. Several widely adopted solutions are available, including HTTP (Hypertext Transfer Protocol), FTP (File Transfer Protocol), TCP (Transmission Control Protocol) / UDP (User Datagram Protocol), WebSocket, Syslog, and MQTT (Message Queuing Telemetry Transport). Indicative examples:

Protocol	Reliability	Speed	Overhead	Scalability	Use case
HTTP	High	Medium	Medium	High	RESTful APIs, web-based log-ging
TCP	Very High	Medium	Medium	Good	Reliable, ordered log transmission

UDP	Low	Very High	Very Low	Very High	High-volume, real-time logging where some loss is acceptable
MQTT	Configurable	High	Low	Very High	IoT devices, distributed systems
WebSocket	High	High	Low	Good	Real-time web applications, live log streaming
Syslog	Configurable	High	Low	High	Network device logging, server logging
FTP	High	Medium	Medium	Medium	Periodic batch transfer of large log files
SSH	High	Medium	Medium	Good	Secure remote log collection, encrypted log file transfer

When logging data is confidential, the use of secure transmission protocols is essential. Protocols such as SSL/TLS, applied on top of HTTP or MQTT as well as SSH or SFTP, provide encryption mechanisms that protect data during transmission and render it unreadable to potential interceptors. This protection is critical to preserving the confidentiality and integrity of sensitive log data and to ensuring compliance with applicable data protection regulations (see Section 5.4.1).

In ENVELOPE, most platform data logging services are deployed in the 5G domain (RAN and CN), cloud and MEC environments as well as the UE/OBU side, as containerized side-car applications. These services generate logging data that is transmitted to data warehouses e.g., via HTTPS (HTTP over SSL/TLS), in line with the access requirements of the target storage systems (e.g., Redmine and Zenodo), which expose RESTful APIs. Such APIs also support manual uploads for logging data that cannot be transmitted automatically, such as data generated by vehicle loggers or legacy systems. Specific details will be found at the particular LL, in Sections 2.2, 3.2 and 4.2.

5.3 Guidelines for Data Pre-Processing

It is advisable to perform processing operations on logging data prior to its storage in data warehouses and/or its use for analysis. Pre-processing of logging data is a critical step that can significantly enhance the efficiency, usability, and overall value of the collected information. Typical pre-processing activities include:

- Data validation: Relying solely on loggers and data acquisition processes to fully comply with all data requirements is not advisable. Verifying that the collected logs conform to the expected formats, value ranges, and other requirements defined during the design phase (see Section 5.1.2) can significantly reduce effort and rework in later stages. In addition, this validation process enables the detection of data duplication (arising from acquisition or

transmission issues) and provides mechanisms to handle errors by reporting, discarding, or correcting affected records.

- Data cleaning: Irrelevant or noisy data should be filtered during this processing stage. When data analysis is automated, any data generated during pre-testing or integration phases must be removed to avoid biasing or distorting the results. Additionally, the removal of duplicate records and the correction of data formats are recommended activities within this pre-processing step.
- Data normalization: As described in Data Acquisition (Section 5.1), logged data should contain normalized information to facilitate analysis. For example, timestamps and location data should follow a consistent format across all logging sources. When such consistency cannot be ensured during data acquisition, a pre-processing step is required to harmonize formats and guarantee consistency across the collected logs.
- Data enrichment: In some cases, captured data requires additional contextual information or metadata to be complete, such as the associated use case or executed scenario, ENVELOPE specific trial site, network configuration, etc. Furthermore, it may be necessary to correlate data originating from multiple sources to enable comprehensive analysis and evaluation.
- Data Categorization: It is important also to categorize the generated data based on sensitivity, and to apply appropriate storage and security measures. Co-funded research projects such as ENVELOPE are subject to specific data-related obligations, including compliance with applicable data protection regulations (see Section 5.4.1) and adherence to RDM requirements [5] under the Horizon Europe programme. Openness, transparency, and reproducibility are key principles that project-generated data shall uphold, in alignment with the FAIR principles [1] (see also D1.4 and D1.5).

  Therefore, the data generated should be classified as public, internal, confidential or highly confidential, where:
  - Public: Information that can be shared.
  - Internal: Data for use within the project members.
  - Confidential: Sensitive information with restricted access.
  - Highly Confidential: Critical data requiring the strictest protection.

  It is acknowledged that not all data can be openly shared, as confidentiality and privacy requirements must be respected. However, techniques such as anonymization, e.g., video blurring to conceal license plates and faces, or identifier modification to prevent traceability, can enable the transformation of sensitive data into shareable information of significant value for other research communities and stakeholders. This will be evaluated on a case-by-case basis at Living Lab level.
  - Time synchronization: Time synchronization across different logging sources is essential when time accuracy is critical for analysis and/or validation activities. As described in Data Acquisition (Section 5.1), if synchronization cannot be ensured at the logger level, it may be performed at a later stage, such as during pre-processing. However, any post-time correction requires reliable information about the time offsets or desynchronization between the different time references used by the logging sources and ENVELOPE tools.

- Data compression: At this stage, data compression may also be applied to reduce transmission and storage requirements (e.g., compressing files into ZIP or RAR formats). Compression can be lossless, preserving all original information while reducing data size, or lossy, reducing data volume by discarding less critical or less perceptible information, depending on the data type and analysis requirements.
  In general, lossless compression preserves full data integrity and is therefore essential for data logging scenarios in which every piece of information is critical, such as Engine Control Unit (ECU) data, OBU data, high-precision sensor measurements, safety-critical system logs, and diagnostic information. However, lossless techniques typically achieve lower compression ratios.
  By contrast, lossy compression can achieve significantly higher compression ratios, making it suitable for automotive data logging involving high-volume data sources such as camera streams or non-critical telemetry. While this approach reduces storage requirements and transmission bandwidth, it does so at the expense of some data fidelity.
  Depending on the use case requirements and limitations, the compression will be evaluated on a case-by-case basis, where necessary.

5.4 Guidelines for Data Protection and Storage

Depending on the available infrastructure (ENVELOPE network-compute continuum system and the ENVELOPE vehicles) and the defined objectives, logging data may pass through multiple intermediate storage locations before reaching the final data warehouse. This situation commonly arises when logs cannot be automatically uploaded from the source logger, either because the target warehouse is not directly or technically reachable, or because the data format or content does not yet meet the requirements for validation or evaluation. In such cases, temporary storage is required to support pre-processing activities before the data is transferred to the designated repository. ENVELOPE does not directly collect personal data; however, for completeness, we provide a generic reference to the relevant GDPR provisions where applicable.

5.4.1 Comply with data protection policies

When processing personal data, compliance with applicable data protection regulations is not only a legal obligation but also a fundamental best practice for safeguarding information. Achieving compliance requires a clear understanding of the specific requirements imposed by the relevant regulatory frameworks, as well as the implementation of appropriate technical and organizational measures. In Europe, the General Data Protection Regulation (GDPR) governs the lawful and respectful handling of personal data, ensuring the protection of individuals' privacy rights. Further details are provided in the following Section 5.4.1.1.

5.4.1.1 General Data Protection Regulation (GDPR)

Regulation (EU) 2016/679 [6], commonly known as the General Data Protection Regulation (GDPR), is a comprehensive data protection framework that entered into force in 2018 and established a global benchmark for privacy and personal data protection. It applies to all organizations that process the personal data of individuals residing in the European Union, irrespective of the organization's geographic location.

The GDPR is founded on key principles such as data minimization, purpose limitation, and storage limitation, which require organizations to collect only data that is strictly necessary, process it for clearly defined purposes, and retain it only for as long as required. It grants individuals extensive rights over their personal data where applicable, including the rights to access, rectification, erasure, portability, and to object to certain forms of processing.

In addition, organizations are required to implement appropriate technical and organizational security measures, conduct Data Protection Impact Assessments (DPIAs) for high-risk processing activities, and notify supervisory authorities of personal data breaches within 72 hours of becoming aware of them. The regulation also mandates the appointment of a Data Protection Officer (DPO) in specific circumstances and generally requires explicit and informed consent for many processing activities. Non-compliance with the GDPR may result in significant administrative fines and other penalties.

A summary of the main principles is as follows:

- Scope: Applies to all organizations processing personal data of EU residents, regardless of the company's location.
- Consent: Requires clear, affirmative consent for data collection and processing.
- Data Subject Rights: Grants individuals rights including access, rectification, erasure, and data portability.
- Privacy by Design: Mandates data protection measures be built into products and services from the earliest stages.
- Data Protection Officers: Requires appointment of DPOs for certain organizations.
- Breach Notification: Obligates reporting data breaches within 72 hours.
- Penalties: Imposes significant fines for non-compliance, up to €20 million or 4% of global annual turnover.
- Accountability: Organizations must demonstrate compliance through documentation and impact assessments.
- Data Transfers: Restricts transfer of personal data outside the EU to countries with adequate protection levels.
- Special Categories: Provides extra protection for sensitive data like health information, political opinions, etc.
- Children's Data: Requires parental consent for processing data of children under 16 (or lower, as set by member states).
- Right to Explanation: Gives individuals the right to explanation about automated decision-making processes.

5.4.2 Use appropriate storage technologies and platforms

According to the RDM requirements [5], which are mandatory for all Horizon Europe projects, research data must be stored in trusted repositories and made openly accessible in accordance with the principle of “as open as possible, as closed as necessary.”

Such repositories include:

- Certified repositories, for example those with CoreTrustSeal [7], Nestor Seal DIN31644 [8] or ISO16363 [9] certifications.

- Domain-specific repositories (e.g. ZENODO [10]) that are internationally recognized, commonly used and endorsed by the research communities relevant to your project.
- General-purpose repositories or institutional repositories that, without official certification, present the essential characteristics of trusted repositories (e.g. security provisions, services for the creation of machine actionable metadata, long-term preservation of data, etc.).

The generated logging data should be stored in data warehouses that comply with the requirements dictated by the intended objectives of the data, such as validation and evaluation purposes. With respect to data transmission and storage, a variety of technologies may be employed, including:

Table 26: List of suitable storage technologies to store logging data.

Storage Option	Use Case scenario	Pros	Cons
Relational Databases (e.g. PostgreSQL, MySQL)	Structured log data, complex queries	- ACID compliance - Good for relationships - SQL querying	- Less scalable for high volumes - Schema changes can be challenging
NoSQL Databases (e.g., MongoDB, Cassandra)	Large volumes of unstructured data	- Highly scalable - Flexible schema - High write throughput	- Eventual consistency (in some cases) - Complex querying can be challenging
Elasticsearch	Full-text search, real-time analytics	- Powerful search capabilities - Real-time analytics - Scalable	- Resource intensive - Learning curve for optimization
Time-Series Databases (e.g. InfluxDB, TimescaleDB)	Time-stamped data, metrics	- Optimized for time-based queries - Efficient data compression	- Limited to time-series data - Less flexible for non-time data
Hadoop Ecosystem (HDFS, Hive)	Extremely large volumes, batch processing	- Highly scalable - Cost-effective for large data - Powerful analytics	- Complex setup - High latency for real-time queries
Cloud-based Solutions (e.g. AWS CloudWatch, Google Cloud Logging)	Cloud-native applications	- Managed service - Scalable - Integrated with cloud services	- Potential vendor lock-in - Can be costly at high volumes
Web-based Project Resources Repositories (e.g. Redmine, OPARU, ZENODO)	Structured and unstructured (raw) files storage	- Managed service - Easy to use	- File size and capacity limitations - Limited scalability
Specialized Log Management Tools (e.g. Splunk, Graylog)	Comprehensive log management	- Purpose-built for logs - Advanced analytics - User-friendly interfaces	- Can be expensive - Potential overkill for simple needs

Flat Files	Small-scale applications, temporary storage	- Simple and lightweight - No additional infrastructure needed	- Limited scalability - Difficult for complex queries

Depending on the requirements of the validation and evaluation activities, storage systems may also serve as support tools for advanced operations such as querying, filtering, comparison, and related analytical tasks. The ENVELOPE Living Labs will adopt a heterogeneous approach to data storage, relying on a combination of platforms and technologies. Most platform services transmit logging data to domain-specific repositories based on web technologies, such as Zenodo (see Section 5.4.2.1).

However, most vehicles, OBU/UE devices, and 5G and MEC infrastructure logs are initially stored locally on the respective machines and ENVELOPE devices. These files are retained temporarily until they are uploaded to the designated trusted repositories.

According to the ENVELOPE Living Labs plan, the validation and evaluation of the collected data will be performed through dedicated scripts and programs that analyze files stored across the different repositories (see also Sections 2.2, 3.2, 4.2). As a result, advanced storage-level functionalities, such as smart queries or built-in filtering mechanisms, are not required by the storage systems themselves.

5.4.2.1 ZENODO repository

Zenodo [10] is a general-purpose open-access repository developed under the OpenAIRE program and operated by CERN. It enables researchers to deposit a wide range of research outputs, including scientific publications, datasets, research software, reports, and other research-related digital artefacts. Each submission is assigned a persistent Digital Object Identifier (DOI), ensuring long-term accessibility and citability in line with the FAIR data principles.

For the EU research community, a dedicated repository within Zenodo, entitled Open Repository for EU-Funded Research, has been established as a pilot initiative. The ENVELOPE project is officially included in this pilot repository [link], where project logs and results will be published. These include public deliverables, scientific publications, and other publicly accessible project outcomes.

5.4.3 Establish access controls

Implementing robust user authentication and authorization mechanisms is essential to ensure that only authorized personnel can access stored data. This is a key requirement for compliance with data protection regulations and for safeguarding data integrity, preventing unauthorized modification or deletion.

Authorization mechanisms operate in conjunction with authentication to ensure that authenticated users can access only the data necessary for their assigned roles. Role-based access control (RBAC) is a widely adopted approach, whereby access permissions are defined according to predefined roles within the project partnership. This approach should be reinforced by the principle of least privilege, ensuring that users are granted the minimum level of access required to perform their responsibilities.

Monitoring and logging are critical components of an effective access control strategy and must not be neglected. All access attempts should be systematically logged and subject to regular review. In addition, real-time monitoring solutions can be employed to promptly detect and alert security teams to suspicious or anomalous activities.

The combination of strong authentication, fine-grained authorization, and continuous monitoring establishes a robust access control framework that significantly enhances the security and governance of stored data.

5.4.4 Implement a backup strategy

Logging data is essential for the evaluation and validation of the ENVELOPE system and platforms across the LLs. Any loss of such data may necessitate the repetition of tests, trials, or deployments, which typically entails significant consumption of resources, including equipment, personnel, time, and financial investment.

Implementing robust data backup strategies not only mitigates the risks associated with data loss caused by system failures, human error, or cyberattacks, but also supports compliance with data protection and governance policies by ensuring data availability and providing evidence required for audits or legal proceedings.

As a best practice, backups should never be stored within the same system, organization, or operational environment as the primary logged data. In line with this principle, backups are commonly maintained using off-site physical storage (e.g. HDDs or SSDs) as well as cloud-based storage solutions, such as Amazon S3, Google Cloud Storage, or Microsoft Azure Blob Storage, which serve as reliable and scalable backup repositories.

6 Conclusions

This deliverable presented the deployment of the ENVELOPE architecture across the three project Living Labs located in Italy, the Netherlands, and Greece. It described the infrastructure established at each site, including the radio access network, core and edge/cloud computing resources, as well as the vehicles, sensors, and end devices used to support the project use cases, experimentation activities, and the open-call projects.

Furthermore, the deliverable introduced the data collection and management mechanisms implemented across the trial sites. The defined logging points, monitoring tools, and data handling procedures enable the systematic collection of experimental data required for the evaluation of the ENVELOPE KPIs and the assessment of the developed enablers and services.

The results presented in this deliverable provide the basis for Deliverable D4.2, which will report on the integration of the ENVELOPE trial sites regarding the pre-evaluation testing activities. The outcomes of D4.2 will confirm the readiness of the experimental infrastructure and provide the necessary validation to proceed with the large-scale experimentation and evaluation activities carried out in Work Package 6.

7 References

[1] M. D. M. A. I. e. a. Wilkinson, The FAIR Guiding Principles for scientific data management and stewardship. Sci Data 3, 160018, 2016.

[2] D1.4 “Data management plan”. ENVELOPE project. Project No. 101139048..

[3] D1.5 “Updated data management plan”. ENVELOPE project. Project No. 101139048.

[4] D2.3 “ENVELOPE availability and cooperation enablers definition and evaluation data specifications”. ENVELOPE project. Project No. 101139048.

[5] OpenAIRE. "How to Comply with Horizon Europe Mandate for RDM”..

[6] E. Union., Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC, Official Journal of the European Union, 2016.

[7] CoreTrustSeal. https://www.coretrustseal.org/.

[8] nestor Certification Working Group. nestor Seal for Trustworthy Digital Archives. https://www.langzeitarchivierung.de/Webs/nestor/EN/Zertifizierung/nestor_Siegel/siegel.html .

[9] ISO 14721:2012 Space data and information transfer systems., https://www.iso.org/standard/56510.html: International Organization for Standardization, 2012.

[10] European Organization for Nuclear Research (CERN) and OpenAIRE. "ZENODO"., https://zenodo.org/..