

D5.1 Use Cases - Scientific Report - a

Version 1.0

28 April 2023

Abstract

COGNIT is an AI-enabled Adaptive Serverless Framework for the Cognitive Cloud-Edge Continuum that enables the seamless, transparent, and trustworthy integration of data processing resources from providers and on-premises data centers in the cloud-edge continuum, and their automatic and intelligent adaptation to optimise where and how data is processed according to application requirements, changes in application demands and behaviour, and the operation of the infrastructure in terms of the main environmental sustainability metrics. This document describes the Use Cases and user requirements that will guide the innovative development of COGNIT, and explains the demonstration scenarios that are being employed for testing and validating the integration of the components of the COGNIT Framework.



Copyright © 2023 SovereignEdge.Cognit. All rights reserved.



This project is funded by the European Union's Horizon Europe research and innovation programme under Grant Agreement 101092711 – SovereignEdge.Cognit



This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

Deliverable Metadata

Project Title:	A Cognitive Serverless Framework for the Cloud-Edge Continuum
Project Acronym:	SovereignEdge.Cognit
Call:	HORIZON-CL4-2022-DATA-01-02
Grant Agreement:	101092711
WP number and Title:	WP5. Adaptive Serverless Framework Integration and Validation
Nature:	R: Report
Dissemination Level:	PU: Public
Version:	1.0
Contractual Date of Delivery:	31/03/2023
Actual Date of Delivery:	28/04/2023
Lead Author:	Thomas Ohlson Timoudas (RISE)
Authors:	Monowar Bhuyan (UMU), Marek Białowąs (Phoenix), Dominik Bocheński (Atende), Aritz Brosa (Ikerlan), Idoia de la Iglesia (Ikerlan), Sébastien Dupont (CETIC), Torsten Hallmann (SUSE), Joan Iglesias (ACISA), Rafał Jurkiewicz (Phoenix), Tomasz Korniluk (Phoenix), Johan Kristiansson (RISE), Antonio Lalaguna (ACISA), Martxel Lasa (Ikerlan), Marco Mancini (OpenNebula), Alberto P. Martí (OpenNebula), Philippe Massonet (CETIC), Nikolaos Matskanis (CETIC), Behnam Ojaghi (ACISA), Daniel Olsson (RISE), Holger Pfister (SUSE), Tomasz Piasecki (Atende), Francesco Renzi (Nature4.0), Bruno Rodríguez (OpenNebula), Kaja Swat (Phoenix), Gerard Świdorski (Phoenix), Paul Townsend (UMU), Iván Valdés (Ikerlan), Riccardo Valentini (Nature4.0), Constantino Vázquez (OpenNebula), Shuai Zhu (RISE).
Status:	Submitted

Document History

Version	Issue Date	Status ¹	Content and changes
0.1	14/04/2023	Draft	Initial Draft
0.2	21/04/2023	Peer-Reviewed	Reviewed Draft
1.0	28/04/2023	Submitted	Final Version

Peer Review History

Version	Peer Review Date	Reviewed By
0.1	21/04/2023	Prof. Rubén S. Montero (OpenNebula/UCM)
0.1	21/04/2023	Mr Philippe Massonet (CETIC)

Summary of Changes from Previous Versions


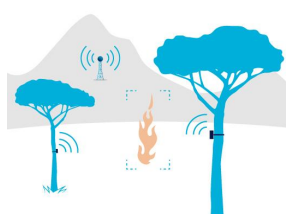

First Version of the "Use Cases - Scientific Report" Deliverable

¹ A deliverable can be in one of these stages: Draft, Peer-Reviewed, Submitted, and Approved.

Executive Summary

This is the first version of Deliverable D5.1, the Use Cases Scientific Report in WP5 (“Adaptive Serverless Framework Integration and Validation”). It includes the collection of requirements from Use Cases performed in Task T5.2 (“Use Cases Technical Coordination, Support, and Assessment”) and describes each of the Use Cases, including an initial architecture design and a plan for the demonstration and validation that will take place as part of the execution of Tasks T5.3, T5.4, T5.5, T5.6. A further analysis of the Use Case requirements, in the context of the initial definition of the COGNIT Framework Architecture, is presented in Deliverable D2.1.

These requirements were collected through a combination of information provided by each Use Case leader through an initial template plus a series of regular collaborative workshops with each Use Case (including participants from both WP2 and WP5). Table 1.1 below summarises the four Use Cases and the main research challenges that they pose for the COGNIT Framework.

	<p>Smart Cities: Connected vehicles and autonomous driving are expected to revolutionise transportation systems, improving road safety and traffic efficiency while reducing accidents. These systems need to be interoperable, intelligent, secure and support the development of multi-tier edge applications that can be deployed across the cloud-edge continuum. The main challenges will be managing dense networks of edge infrastructure resources, as well as a variety of services with different QoS requirements.</p>
	<p>Wildfire Detection: Europe has witnessed a significant increase in the number and ferocity of so-called ‘mega-fires’, a phenomenon linked with climate change. Edge/IoT devices, coupled with AI/ML, can play an important role in preventing and fighting wildfires. The main challenges will be rare events (suspected fire) with sudden peaks of extremely high offloading demands, and effective energy management to counter the lack of dedicated power supply.</p>
	<p>Energy: Supporting the energy transition in Europe requires new solutions providing wide and open accessibility of energy data, and individual energy independence of a household and/or small energy clusters. Smart edge applications using advanced AI/ML algorithms could be used for monitoring, predicting, and managing both energy production and consumption. The main challenges will be resource-constrained environments (i.e. energy meters) and the very high security requirements for distribution system operators (DSO), as the energy sector moves</p>

away from a hierarchical, centralised structure towards a more decentralised and distributed way of managing energy assets and networks.



Cybersecurity: Moving computation and data processing services to the edge, far from secured data centers, leaves systems exposed to new threats. Edge Computing requires a new generation of intelligent security mechanisms to be deployed along with edge applications, implementing advanced authentication and authorisation policies. This Use Case explores resilient anomaly detection and remediation in a smart mobility context. The main challenges will be the migration of workloads between edge nodes, and managing a DevSecOps pipeline in a multi-provider edge context with dynamic (geo-dependent) security policies.

These Use Cases and their requirements will guide the design and development of the COGNIT Framework, and provide scenarios for testing and validating the agile development and integration of its different components and new features during the execution of the project. Furthermore, a description of the testbed setup that will be used for demonstration purposes is provided at the end of this document.

This deliverable has been released at the end of the start phase (M3), and will be updated with incremental releases at the end of each research and innovation cycle (i.e. M9, M15, M21, M27, M33).

Table of Contents

Abbreviations and Acronyms	6
1. Introduction	8
PART I. Use Case Descriptions	9
2. Methodology	9
3. Use Case 1: Smart Cities	11
3.1. Reference Scenario	11
3.2. Objectives	14
3.3. System and Architecture Description	15
3.4. Use Case Infrastructure, Demonstration, and Validation	19
3.5. Risks and Mitigation Plan	21
4. Use Case 2: Wildfire Detection	23
4.1. Reference Scenario	23
4.2. Objectives	24
4.3. System and Architecture Description	25
4.4. Use Case Infrastructure, Demonstration, and Validation	26
4.5. Risks and mitigation plan	28
5. Use Case 3: Energy	30
5.1. Reference Scenario	31
5.2. Objectives	33
5.3. System and Architecture description	34
5.4. Use Case Infrastructure, Demonstration, and Validation	38
5.5. Risks and Mitigation Plan	40
6. Use Case 4: Cybersecurity	41
6.1. Reference Scenario	41
6.2. Objectives	44
6.3. System and Architecture Description	44
6.4. Use Case Infrastructure, Demonstration, and Validation	46
6.5. Risks and Mitigation Plan	48
7. Use Case Requirements	49
8. Summary of Use Case Cloud-Edge Infrastructure	52
PART II. Test Infrastructure	53
9. Testbed	53
9.1. Background	53
9.2. RISE ICE Cloud-Edge Infrastructure	54
9.3. COGNIT Testbed Architecture	57
10. Conclusions and Next Steps	60

Abbreviations and Acronyms

3G	Third Generation Mobile Network
4G	Fourth Generation Mobile Network
5G	Fifth Generation Mobile Network
AI	Artificial Intelligence
API	Application Programming Interface
AWS	Amazon Web Services
CAM	Cooperative Awareness Message
CC	Control Center
C-V2X	Cellular Vehicle to Everything communication
FaaS	Function as a Service
GNSS	Global Navigation Satellite System
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
IaaS	Infrastructure as a Service
IP	Internet Protocol
ITS	Intelligent Transport System
LAN	Local Area Network
LTE	Long-Term Evolution
MCU	MicroController Unit
MEC	Multi-Access Edge Computing
ML	Machine Learning
M-Hub	Mobility Hub (advanced TLC)
OBU	On Board Unit
OS	Operating System
PaaS	Platform as a Service
QoS	Quality of Service
RES	Renewable Energy Source
REST	Representational State Transfer
RSU	Road Side Unit
SaaS	Software as a Service
SLA	Service Level Agreement
SOAR	Security Orchestration, Automation, and Response
SREM	Signal Request Extended Message

SSEM	Signal request Status Extended Message
TCP	Transmission Control Protocol
TLC	Traffic Light Controller
V2X	Vehicle to Everything communication technology
VLAN	Virtual Local Area Network
VM	Virtual Machine
VPN	Virtual Private Network
WLAN	Wireless Local Area Network

1. Introduction

The general purpose of Deliverable D5.1 is to describe the evolving requirements of the Use Cases that are part of the COGNIT Project, and summarise the demonstration and validation work performed in WP5 during its execution. To this end, D5.1 will be incrementally updated and released at the end of each research and innovation cycle (i.e. M9, M15, M21, M27, M33), incorporating updates (if any) to the requirements of the Use Cases, a description of the open source integration and certification process, and a description of the research, development, and validation work done in each cycle.

This report is the first version of D5.1, describing the Use Cases and their requirements for the COGNIT Framework. It documents the activities performed in WP5 to characterise and specify the Use Cases and their requirements, which have served to guide the design and specification of the COGNIT Framework Architecture defined by WP2 in Deliverable D2.1. The Use Cases will furthermore serve as integration scenarios for the COGNIT Framework, with the purpose of testing and validating the integration of its different components and new features along the execution of the project. The methodology used to collect and analyse the Use Case requirements is described in more detail in Section 2.

D5.1 is a living document that is composed of an introductory section and eight additional sections organised in two main blocks of content:

- **Part I** focuses on the characterisation and analysis of the Use Cases, along with a specification of their requirements. Each Use Case has a section dedicated to it (Sections 3 to 6), containing a detailed description of the Use Case scenario and system under consideration, and a realisation of the scenario within the COGNIT Framework. The requirements for all the Use Cases are summarised in Section 7, which lists a number of common requirements, as well as requirements specific to each Use Case.
- **Part II** focuses on the setup and architecture of the project's edge cloud testbed. This testbed will be used to connect the edge nodes of the Use Cases to the core data center resources at the facilities of the RISE ICE Data Center.

The document ends with a conclusion section.

This report makes references to the components of the COGNIT Framework (e.g. Serverless Runtime, Provisioning Engine, etc.) defined in Deliverable D2.1.

PART I. Use Case Descriptions

2. Methodology

The general approach aimed to facilitate a collaborative process between understanding the Use Case scenarios and their requirements in WP5, and the design and specification of the COGNIT Framework in WP2. To this end, a Use Case template was jointly drafted in the very beginning, to harmonise the methodology and presentation across the four Use Cases, and a series of collaborative workshops were set up between WP2 and WP5.

The Use Case template addressed information including a detailed description of the Use Case scenario, objectives and Key Performance Indicators (KPIs), actors and system specifications, technical requirements, and how to realise the scenario in the COGNIT Framework. The Use Case leaders were responsible for gathering and filling out the information in the Use Case template, and this material was discussed and evolved during the regular workshops. Thus, each Use Case maintained its own version of the template, which was open to the whole consortium to read and comment on. These templates served as the basis for the presentations of the Use Cases in the following sections, as well as the collection and analysis of their requirements, presented in Section 7.

USE CASE TEMPLATE	5. Requirements
<p>1. Use case summary <i>A short description/summary of the use case</i></p> <p>1.1 Ambition <i>Ambition, what would you like to achieve/demonstrate within this project, as use case owners – useful as a guide, even if we don't reach there with Cognit</i></p> <p>2. Objectives <i>Which aspects of the Cognit solution/architecture/framework do we want to demonstrate and validate in this use case</i></p> <p>2.1 KPIs <i>What are the KPIs for the use case demonstration and validation</i></p> <p>3. What help will you need from others <i>This is for internal communication, so that the different partners can easily find out what help you will need to realize the use case demonstration and validation</i></p> <p>4. Reference scenario <i>Isolate and describe a use case scenario that should be used to demonstrate the Cognit solution</i></p> <p>4.1 Actors, stakeholders, roles <i>List actors and stakeholders in the scenario, and describe their roles</i></p> <p>4.2 Context and system overview <i>Put the reference scenario in a context</i></p> <p>4.3 Processing tasks and steps <i>List the processing tasks and their relationships, including conditions</i></p> <p>4.4 External dependencies, communication, constraints <i>Are there any external dependencies/communication? Any other constraints?</i></p> <p>4.5 Current solution <i>If possible, could you describe your current solution (if you have one), and what needs to be kept as it is</i></p>	<p>5.1 Technical requirements <i>Describe and list any technical requirements needed to set up and run the use case, e.g:</i></p> <ul style="list-style-type: none"> • Infrastructure involved: Private cloud / Public cloud / Public edge / Private edge / 5G edge • Need for shared data resources / storage abstraction? (e.g. S3/MinIO-based?) • Execution environment, e.g. Default location of the computation • Connectivity of the device / existing infrastructure • Specific applicable regulation / security / data sovereignty considerations? • Key expectations/requirements: access to GPUs, ultra-low latency, local data processing capacities, etc. • Application architecture: single device / distributed • Language support? Python/C/Java? • Robustness • Continuous/streaming data processing / Intermittent? <p>5.2 Security and privacy <i>Identify security and privacy risks/requirements</i></p> <p>5.3 Requirement KPIs <i>Any KPIs that need to be satisfied?</i></p> <p>6. Proposed use case architecture <i>Describe the architecture we will use for the use case. Explore the possibility to use FaaS serverless computing as a complement/replacement to existing cloud computing</i></p> <p>7. Demonstration and validation <i>Plan for what and how we should demonstrate and validate the Cognit solution for this use case</i></p> <p>7.1 Milestones <i>Key milestones for the demonstration and validation of the use case</i></p> <p>7.2 Testbed requirements <i>Requirements for the testbed</i></p> <p>8. Risks <i>What are the risks we foresee, in particular for Milestones 1 (M3) and 2 (M15)</i></p> <p>8.1 Risk mitigation plan <i>Risk mitigation measures</i></p>

Figure 2.1. Templates used for the collection of information from each Use Case.

The workshops served two purposes: to discuss and add to the information in the templates, as well as to guide the design and specifications of the COGNIT Framework. With the exception of the first workshop, in which all Use Cases were presented along with the overall architectural vision and the testbed at RISE, the workshops were held with each Use Case individually. These workshops were open to participation from the whole consortium, with the leaders of WP2 and WP5 as core participants, but also including developers of the core components of COGNIT. The focus of the first few workshops was mainly on understanding the details of each of the Use Cases, including their current system solutions and scenarios. This focus subsequently shifted towards architectural discussions, addressing both the design of the COGNIT Framework, and how each Use Case could be realised within it, while addressing their unique requirements and challenges.

Several common requirements emerged from these discussions, in addition to a few requirements specific to each Use Case. They are summarised in Section 7 of this report, and also in Deliverable D2.1 on the COGNIT Framework Architecture.

3. Use Case 1: Smart Cities



Connected vehicles and autonomous driving are expected to revolutionise transportation systems, improving road safety and traffic efficiency while reducing accidents. This Use Case will demonstrate the capabilities of edge computing for future smart city platforms and smart mobility services—the main challenge will be managing dense networks of edge infrastructure resources, as well as a variety of services with different QoS requirements. The transportation systems need to be interoperable, intelligent, secure and support the development of multi-tier edge applications that can be deployed across the cloud-edge continuum, leveraging data locality to reduce overhead, and be managed securely using cloud-native practices.

ACISA is developing a new generation of Intelligent Transport System (ITS) systems for urban and interurban environments, aspiring to leverage open technologies in the cloud and edge computing fields to achieve a fastest pace of innovation, more resilient systems and less siloed proprietary technologies. This will benefit both their customers, by avoiding vendor lock-in scenarios, and other stakeholders by attracting the vibrant open source community to the ITS domain. For the last 2 years, ACISA's R&D team has been working on a new concept of Traffic Light Controller (TLC) that aims to leverage the privileged location held by each TLC in the layout of a city, to provision an enhanced distributed computing capacity, which enables real low-latency, high-capacity infrastructure to deploy advanced services for traffic management, video analytics, and mobility.

Their Mobility-Hub (henceforth M-Hub) is designed as an advanced traffic light controller at the edge, aiming to reduce response times and improve the efficiency of public transportation and emergency services. Considering that this infrastructure is often owned by the city councils themselves, providing open and standards-based technology will allow cities to deploy robust smart-city IoT services on top of its already existing traffic infrastructure. Other stakeholders include traffic/mobility departments inside city councils, and other local authorities, such as police, ambulance operators, and fire-fighters. ACISA acts as the technology integrator that installs, sets up, and maintains the ITS systems throughout the term of the contract.

3.1. Reference Scenario

This Use Case will explore the scenario of traffic signal priority (green light) for emergency vehicles and public transportation in a smart city context. This is especially important in densely populated areas, where traffic congestion can significantly impact response times for emergency services, or cause delays to public transportation. The solutions of the future can help save lives and reduce property damage, as well as improve public transportation by reducing travel times and improving schedule adherence. This can encourage more people to use public transportation, which can reduce traffic congestion and improve air quality in urban areas.

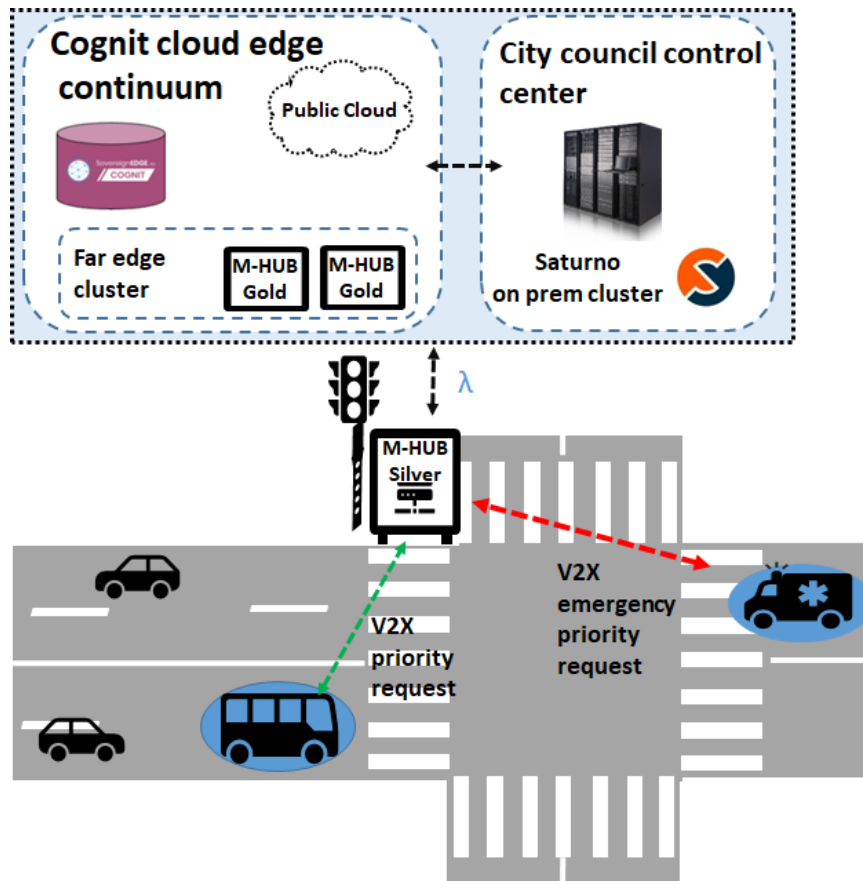


Figure 3.1. Reference Scenario for the Smart City Use Case

In this scenario, a vehicle initiates a priority request by sending a standardised V2X priority request message. Such messages are intercepted by a Road-Side Unit (RSU)—these will be installed along the smart roads of the future—that forwards V2X messages to a nearby M-Hub, as illustrated in Figure 3.1. This M-Hub can then decide whether to adjust the traffic signals to grant priority to the approaching vehicle. For example, when an active emergency vehicle approaches an intersection, the corresponding M-Hub will change the traffic signal to green almost immediately in their direction (always ensuring a safe transition), possibly breaking the synchronism of the traffic plan, while other traffic will be held at a red light, as illustrated in Figure 3.2. If the approaching vehicle is instead a bus from the public transport system of the city, the M-Hub will make a decision based on various parameters, e.g. prior delays, the time in the cycle at which the demand is received, and the expected time of arrival at the traffic light. The controller chooses the most efficient method, without altering the synchronism of the overall traffic plan: in case the signal is green, it will try to extend the time of green to allow passage within the same signal phase, as illustrated in Figure 3.3a; in case the signal is red, it will instead try to minimise the time to green, as illustrated in Figure 3.3b.

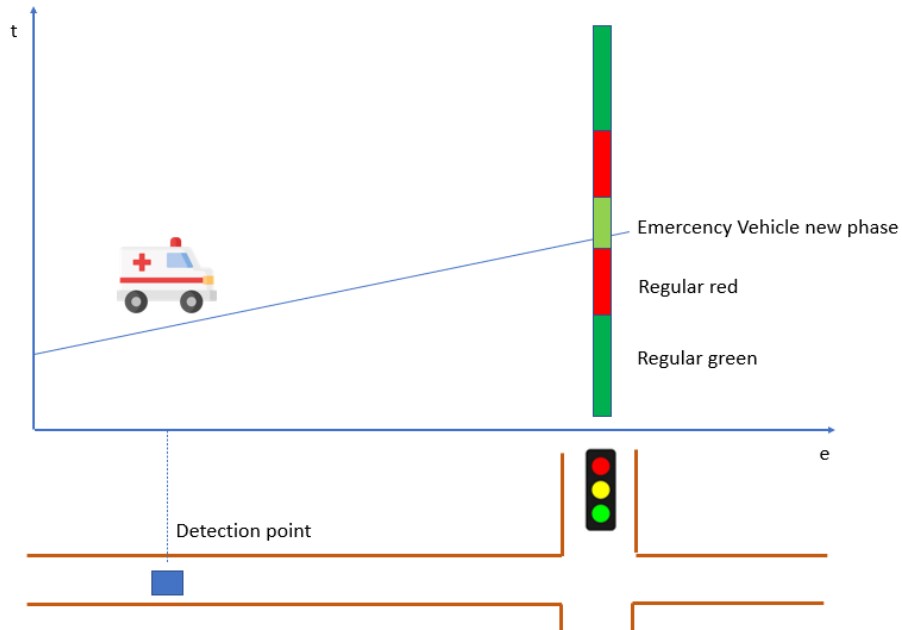
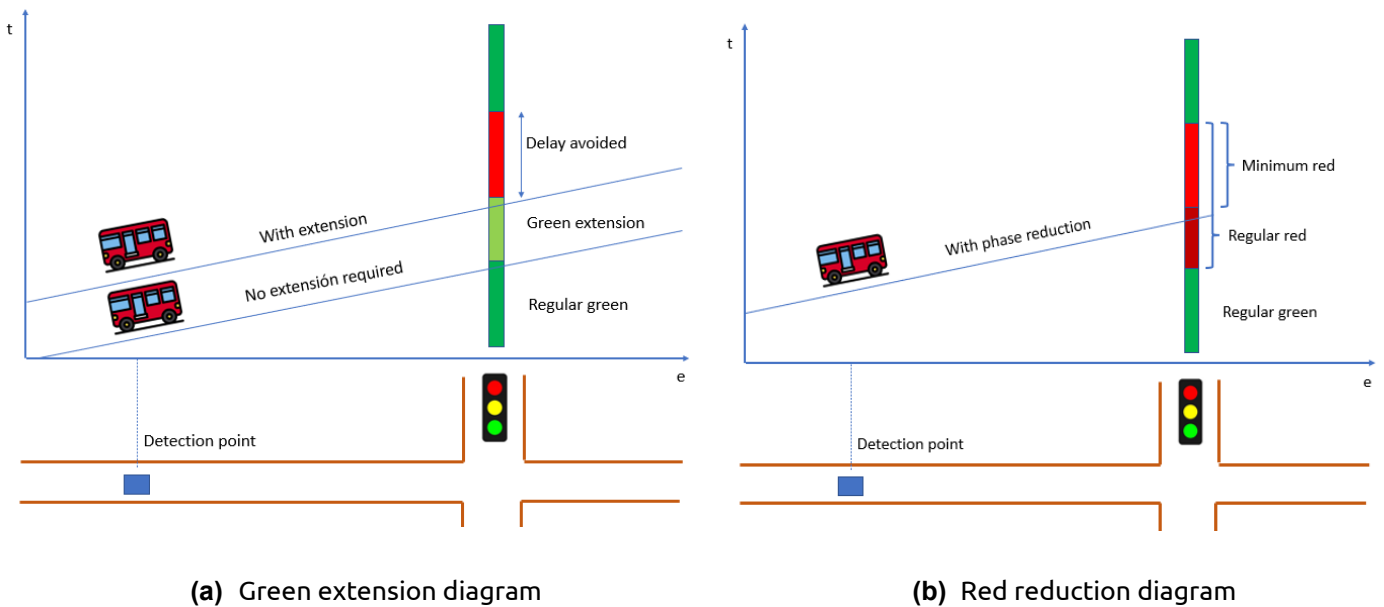


Figure 3.2. Traffic phase diagram for emergency vehicle priority request

Today, the necessary information to process all the traffic priority requests is held centrally in ACISA's traffic management platform Saturno (in the Control Center), but the aim is to distribute it among some of the more powerful M-Hub nodes, to bring this decision closer to the requestor. The less powerful of the M-Hub nodes will instead act as clients that offload computations to the cloud-edge continuum, which combines these powerful M-Hub nodes (the far edge) with the on-prem data center in the Control Center (near edge) and the Cloud, where more computation capacity is available, based on edge applications' requirements.



(a) Green extension diagram

(b) Red reduction diagram

Figure 3.3. Traffic phase diagrams for public transport vehicle priority request

For reference, current solutions for implementing bus priority, without the use of standardised V2X technology, typically use one of several technologies, such as direct radio communication between the bus and the traffic light, or on board legacy technologies (see Figure 3.4) that asks for priority when the bus or the emergency vehicle is located in a certain area or detection point near to the intersection. The basic architecture of the current solution used by the Use Case is based on an on-premises server located in the Control Center that is running a monolithic application that manages all the priority requests.



Figure 3.4. Legacy sensor for bus detection

Unique Challenges

Implementation of the scenario described above presents a number of unique challenges and novelties, including:

- Edge services in general and V2X in particular represents a new business case for both ACISA and city councils.
- V2X services are not yet generally adopted by city operators.
- Novelty use of a network of traffic light controllers in a city to deploy advanced mobility services.
- Deploy research development in a controlled real pilot.
- Use FaaS technology in the cloud continuum, integrated with V2X and TLC technologies to provide advanced mobility services.

3.2. Objectives

The objectives of this Use Case are to:

- Implement a solution for priority management in public transport and emergency vehicles based on V2X technology, through direct interaction between the on-board unit (OBU) and the road-side unit (RSU) and the integration with the M-Hub and with the Saturno cloud platform, both developed by ACISA.
- Develop and deploy our applications on the COGNIT platform.
- Automate resource provisioning for handling V2X messages using COGNIT, to dynamically allocate Serverless Runtimes between the hierarchy of M-Hubs and the

Control Center. As the traffic load increases, increasingly dynamic resource orchestration would be needed.

- Integrate infrastructure encompassing far edge nodes, on-premise data center (near edge), and public cloud, all managed as a single cloud-edge continuum under one single framework.

Also, this specific Use Case is expected to provide a unique contribution towards the validation and achievement of the following global KPIs of the COGNIT Project:

- **[KPI1.1]** Framework tested to scale up to tens of thousands of distributed nodes;
- **[KPI5.4]** Contributions to, and pilot implementation of, at least 1 standard in the field of Edge Computing.

3.3. System and Architecture Description

The Use Case involves the following system components, as illustrated in Figure 3.5:

- **Saturno:** The traffic management platform of ACISA. It exposes a REST API, and is hosted on an on-prem data center in the Control Center and public cloud.
- **Mobility-Hub (M-Hub):** M-Hubs have the capability to receive and process the priority request from the vehicle detection system, besides its conventional functionality as a traffic light controller. The M-Hubs will be divided into two categories, based on their computing capacity: an enhanced version of M-Hub (i.e. M-Hub Gold) with increased computation power will form a cluster in the far edge to federate computational resources within the COGNIT cloud continuum.
- **M-Hub Silver**—Acting as COGNIT Client: A basic M-Hub node deployed at every intersection, capable of directly controlling the traffic lights at intersections, and storing data from nearby sensors. These are the devices initiating FaaS requests.
- **M-Hub Gold**—Far edge infrastructure managed under the COGNIT Framework: Enhanced Mobility-Hub with increased computational power. These operate as far edge nodes in the COGNIT cloud-edge continuum, capable of executing Serverless Runtimes. The information required for processing a traffic priority request (today held centralised in Saturno in the control center) will be distributed among the M-Hub Gold edge nodes. These nodes should therefore in general have all the data needed for granting or denying a priority request. Should the M-Hub Gold need extra data from external systems, it will be able to request the missing info either to ACISA's traffic management platform Saturno, the AVL (Automatic Vehicle Location) system from the public transport operator control center (CC) or any other related information system.

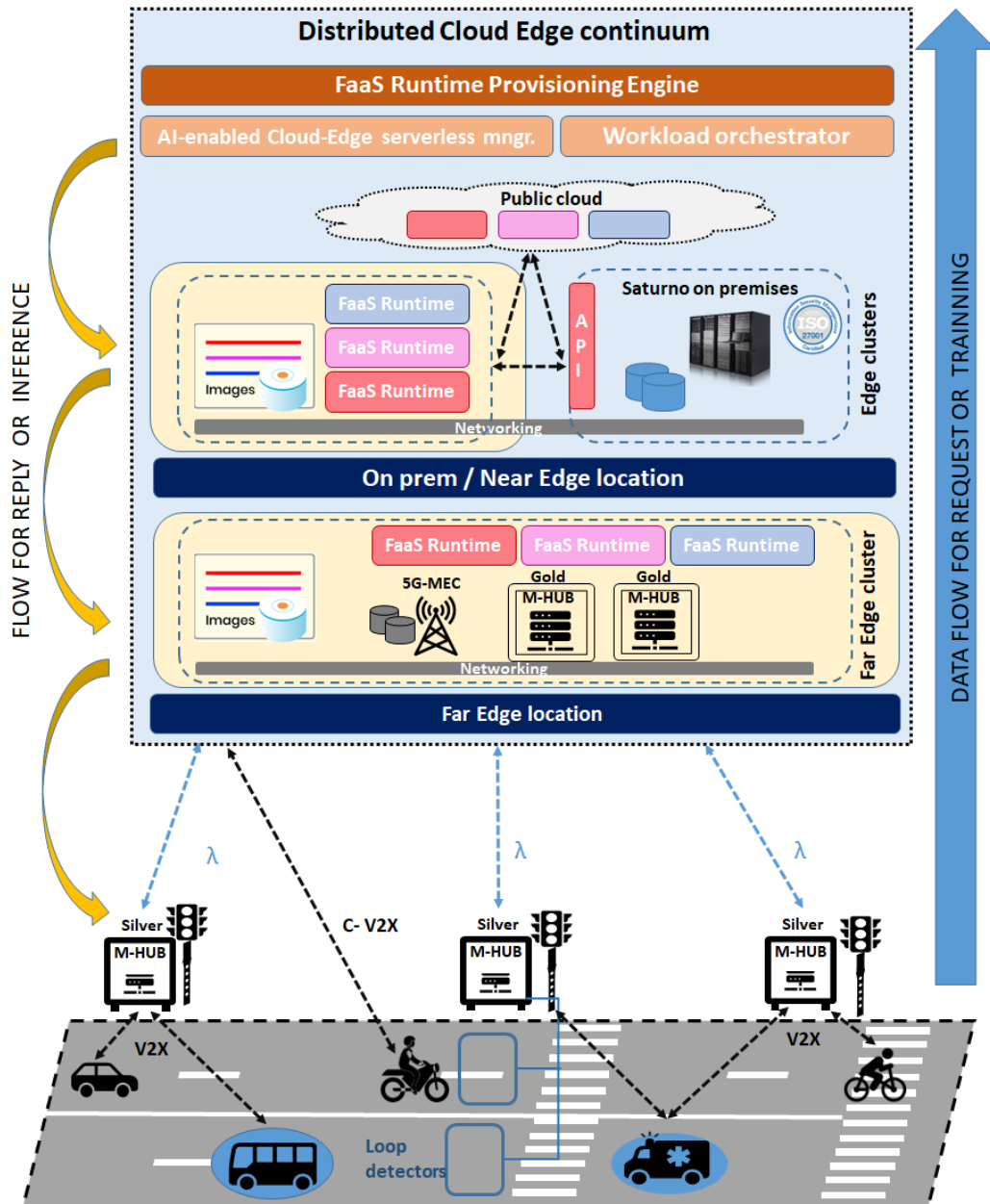


Figure 3.5. Architecture for the Smart City Use Case.

- On-premise Data Center**—Near edge infrastructure managed under the COGNIT Framework: On-premise data center in the Control Center (CC) where Serverless Runtimes can run and communicate with Saturno via its API REST. This may be placed in the City council traffic control center or in a private cloud data center, and will have connections to other information systems that are related to the traffic management of the city. These powerful near edge nodes could potentially train large artificial intelligence models and perform inference to determine the criticality of the prioritisation requests, based on many more features.
- Public Cloud**—Cloud infrastructure managed under the COGNIT Framework: Additionally, Serverless Runtimes could be deployed on a public cloud offering bare-metal resources, managed under the COGNIT Framework.

- **On Board Unit (OBU):** Vehicle communication is supported by a V2X On Board Unit, equipped with a Global Navigation Satellite System (GNSS) receiver and its antennas, that can communicate with the RSU through different V2X radio standards, or connect straight away with an M-Hub Gold instance deployed at a Telco Multi-Access Edge Computing (MEC) through Long-Term Evolution (LTE) or Fifth generation mobile network (5G).
- **Road-Side Unit (RSU):** A device capable of intercepting, reading and transmitting V2X messages. RSUs will be installed in close proximity to smart road infrastructure.

The reference scenario involves the following steps:

- 1) A vehicle initiates a priority (green light) request by sending a standardised V2X Signal Request Extended Message (SREM) message (containing, e.g. bus ID, Lane ID, delay information, and other related data if available), first intercepted by an RSU that forwards the request to the relevant M-Hub, which filters incoming V2X messages for priority requests.
- 2) Upon receiving a priority request, a FaaS request will be initiated by the M-Hub, to offload the processing to decide whether priority should be given.
- 3) The function will make use of either traditional algorithms or AI models to decide whether to grant or deny priority, assessing the traffic situation at the intersection under consideration based on the following data from the offloading M-Hub, included as arguments to the Serverless Runtime:
 - Data stored in the particular M-Hub Silver node that initiated the FaaS request, e.g. traffic light controller status, loop detector data, and V2X Cooperative Awareness Messages (CAMs) from other vehicles;
 - Data from the vehicle contained in the V2X SREM message received by the M-Hub silver node, e.g. bus/vehicle ID, lane ID, other related data and delay information if available.
- 4) In case the function needs extra data from external systems, it will be able to request the missing info either from ACISA's traffic management platform Saturno, SAE system from the public transport operator control center (CC), or any other related information system.
- 5) Once the priority request has been processed and analysed, a message will be sent back to the requestor through another standardised ETSI message for V2X, Signal request Status Extended Message (SSEM).

The following sequence diagram (Figure 3.6) depicts the communication messages involved in the prioritisation decision-making process at the far edge node. Specifically, it illustrates how M-Hub Gold nodes can gather information from other nodes and Control Center nodes to make a decision about traffic prioritisation at intersections. This decision-making process considers various factors such as traffic status, bus line delay, emergency network information, weather conditions, and pollution levels in other parts of the city that may be affected by the prioritisation. By considering these factors, M-Hub

Gold devices can make informed decisions about traffic prioritisation that optimise traffic flow and reduce congestion in the city.

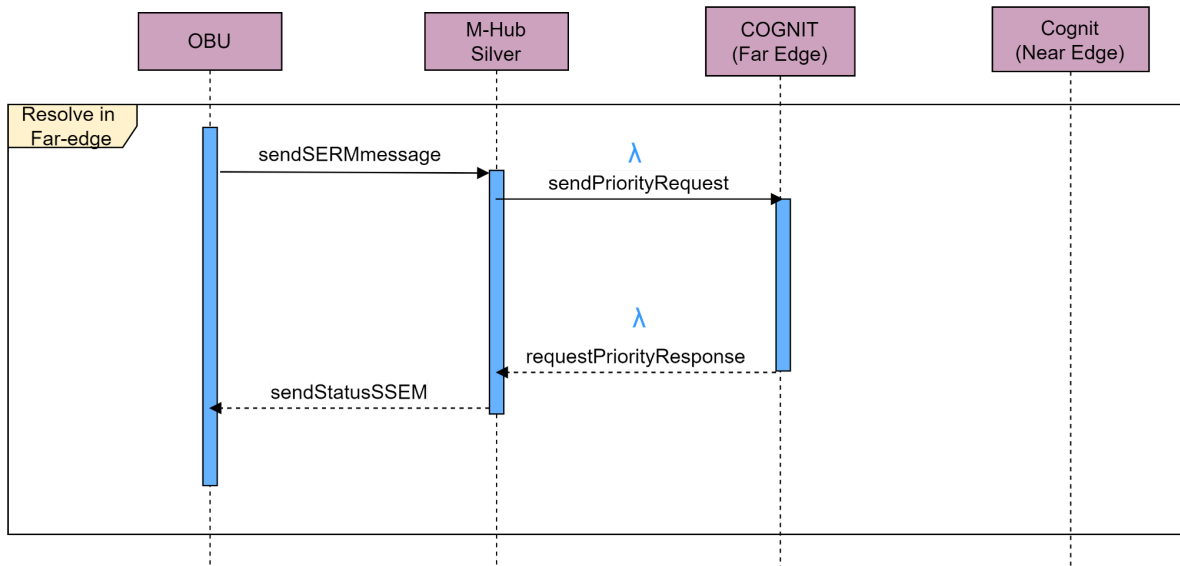


Figure 3.6. Sequence diagram of the prioritisation decision-making process at the far edge node

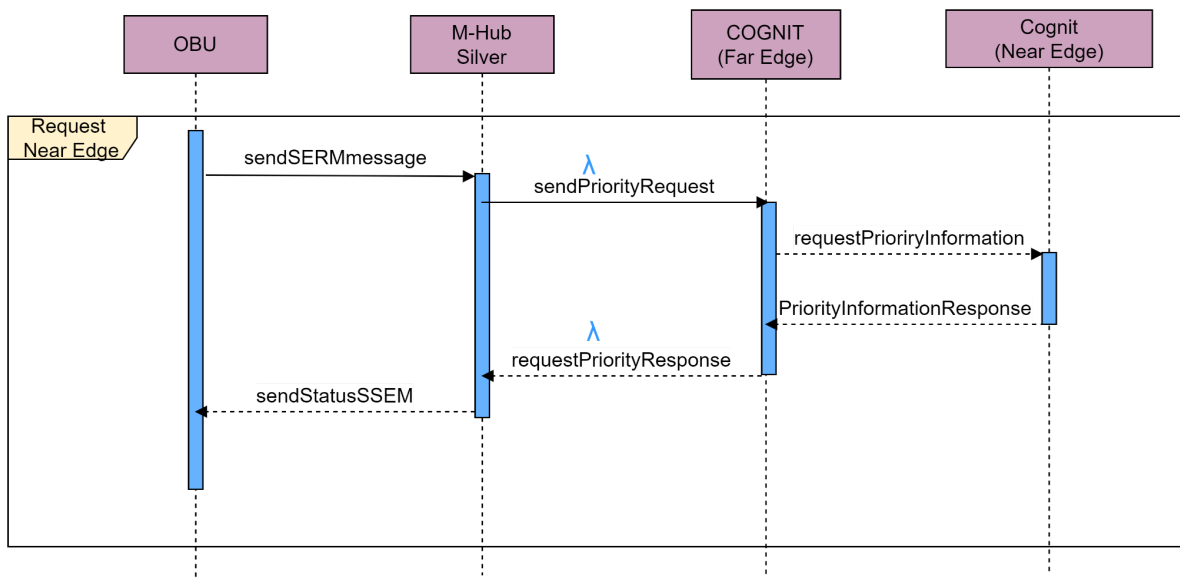


Figure 3.7. Messaging sequence with invalid SREM in the near-edge COGNIT node

In contrast, Figure 3.7 illustrates the prioritisation resolution that occurs whenever additional information is requested from COGNIT’s near edge nodes. The near edge nodes may be located in the City Council traffic control center or in a private cloud data center, and it is connected to other traffic management information systems within the city like Saturno. In some cases, near edge nodes with high processing power may be equipped

with artificial intelligence models that can be trained to infer the criticality of prioritisation based on a range of features. Through these connections and advanced capabilities, COGNIT and the near edge node are able to provide M-Hub Gold with detailed and highly-relevant information that enables accurate prioritisation decisions.

For several ITS services, communication in vehicular networks is eminently public, but confidentiality, integrity and non-repudiation must be guaranteed. As an example, V2X messages may not be encrypted for the sake of efficiency, latency, or limited computational power within the vehicles. However, to ensure the integrity of the messages, all V2X messages will carry a digital signature. This measure guarantees that the message was not tampered with during transmission and provides an added layer of security to the communication system.

3.4. Use Case Infrastructure, Demonstration, and Validation

Demonstrations and validations of the Use Case will either be conducted in a controlled environment dedicated to vehicle mobility tests or in a real crossing. The basic infrastructure of the testbed for the Use Case (as shown by Figure 3.5) will consist of:

- 2 heavy M-Hub nodes (M-Hub Gold) acting as the farthest COGNIT edge nodes;
- 6 light M-Hub nodes (M-Hub Silver) acting as the client and users of COGNIT;
- On-premise data center in Control Center (CC) running Saturno platform;
- Optionally, the Saturno platform may be deployed on a Public Cloud.

The number of M-hub (either Silver or Gold) nodes could be scaled up within a virtualization environment in the on-premises installation. This scenario could be scaled up to tens of thousands of distributed nodes to achieve **[KPI1.1]**.

The data injected by public buses and emergency vehicles will be sent to RSU which is integrated within our M-Hub. This communication is done through two different interfaces, namely, ITS-G5 (ETSI Standards) and C-V2X (3GPP Standards), wherein these interfaces are standardised by ETSI and 3GPP well-known standardisation associations in telecommunications and communications domain. Therefore, this Use Case will contribute towards achieving **[KPI5.4]**.

Hardware

The hardware elements necessary for the validation and demonstration of the use case:

- OBU with 802.11p (optionally also C-V2X) connectivity, and 3G/4G connection for maintenance and GNSS receiver.
- Human-Machine Interface (HMI) connected to the OBU in which the relevant information related to the Use Case is displayed.
- RSU installed at an intersection with 802.11p, C-V2X and GNSS connectivity, as well as an internet connection for integration with the cloud.
- M-Hub Silver integrated with the RSU that is connected via ethernet or 4G to Saturno deployed in the CC or private cloud.

Software

The software elements necessary for the validation and demonstration of the use case:

- Software embedded in OBU with functionality close to a real scenario and Application in the OBU capable of sending V2X messages corresponding to the priority request.
- Software application in RSU with functionality close to a real RSU scenario capable of receiving priority request messages, processing them and sending them to the priority management service deployed in the cloud or in the M-Hub. This application is also in charge of sending the response of requests to the OBUs once the decision of the cloud or the traffic light regulator has been received.
- Service in charge of collecting data from the RSU, M-Hub and Saturno, determining priorities and sending a response to the RSU regarding the requests received, and their level of priority.
- Communications systems for connection of M-Hub Silver & Gold to control center.

Milestones and Validation Criteria

Successful demonstration of the Use Case will be evaluated against several achievements and milestones.

For the Project's second milestone at M15, at which point the first demonstration of the Use Case will take place, the Use Case is expected to have reached the following achievements:

- Identify and develop a lambda function that should be offloaded to COGNIT cloud-edge.
- Integrate the COGNIT Client with existing applications, and install it on M-Hub Silver nodes.
- Integrate M-Hub Gold nodes with the COGNIT Framework to enable running Serverless Runtimes on them, and have them managed under the COGNIT cloud-edge.
- Pilot deployments of COGNIT-enabled M-Hubs in a real scenario.
- Deploy the Saturno Platform to control the M-Hubs and priority.
- Gather data from a real scenario and test basic functionality.

These advanced features are expected to be incorporated later on by Milestone M27:

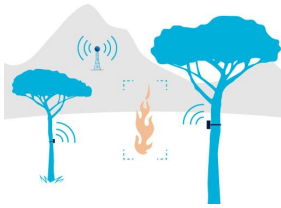
- With the recorded real data make a simulator to test the scalability of the solution.
- Simulate COGNIT edge-cloud with tens of thousands of edge nodes using the developed simulator.

3.5. Risks and Mitigation Plan

N° Risk	Potential risks		Contingency plan	
	Level (1:low; 5:high)	Impact	Description	Responsible
1	3	Cybersecurity requirements put the security of the M-Hub at risk.	Use of best practices in cybersecurity, and ensure the inclusion of monitoring and control tools in edge application deployment pipelines. Use of ISO 27001 control that may apply.	Technical lead
2	2	HTTP or MQTT protocol is not efficient enough for certain devices or its implementation is not feasible.	Develop a protocol conversion server with the possibility of maintaining direct TCP/IP links to specific protocols.	Technical lead
3	3	Problems related to the integration of V2X communication systems, including cyber security aspects.	V2X equipment documentation shall be requested and analysed.	Project lead
4	3	Delays related to equipment procurement.	ACISA already has V2X equipment and a private cloud with the capacity to allocate the rest of the equipment.	Project lead
5	3	Implemented functionality does not give good results when applied in a real environment.	ACISA sees this framework useful in other types of real Use Cases like federated learning, urban digital twin and transfer learning related to traffic and mobility or multi-tenant smart-city infrastructure.	Project lead
6	3	No collaboration agreement with any city council or permission to implement the pilot test in real city conditions is obtained.	ACISA already has agreements in real scenarios for pilot testing, but it is also possible to Install a M-hub and RSUs in a laboratory or controlled environment.	Project lead
7	3	The results of the pilot test indicate that there are stability and scalability problems.	The equipment in the private cloud will be oversized to accommodate with enough slack the computational resources required. ACISA has an AWS	Technical lead

			account that might be used if scalability is needed. M-hub can be virtualized, and thousands of edge nodes could be used in order to test the COGNIT Framework at scale.	
8	2	Delays in developing the specific parts inside mHUB to provide access to COGNIT Framework or developing the specific parts inside mHUB to provide access to COGNIT Framework. Delays implementing FaaS or the Runtimes.	Developers will be trained as soon as the COGNIT Framework is available in order to implement and test it.	Project & Technical leads

4. Use Case 2: Wildfire Detection



Europe has witnessed a significant increase in the number and ferocity of so-called ‘mega-fires’, a phenomenon linked with climate change. Real-time information gathered from edge/IoT devices deployed in forests, coupled with AI/ML, can play an important role in preventing and fighting wildfires. However, the lack of dedicated power supply makes low energy consumption and effective energy management top priorities. This Use Case will demonstrate the

capabilities of edge computing for monitoring scenarios in remote areas—the main challenge will be rare events with sudden peaks of extremely high offloading demands, and effective energy management to balance sensor utilisation, on-board processing, communication, and edge offloading.

Nature 4.0 develops IoT devices for remote monitoring applications, using sensor data to create accurate digital twin models of forests and other natural environments. They have developed a device called TT-Fire specifically to detect early signs of wildfires in large forests, to support and warn the civil protection in case of wildfire emergency. The real-time sensor data collected from these devices are combined with simulation models and satellite images to monitor wildfire risks and forecast wildfire spread. Since the devices are powered by photovoltaics, without any dedicated power supply, proper energy management is critical, utilising edge offloading capabilities to optimise battery lifetime. The system should warn the civil protection in case of wildfire emergency, providing (when possible) forecasting through the use of wildfire spreading models with real-time data.

Their ambition is to design and install wildfire detection systems able to detect the presence of flames, validating and locating them from the information received by different devices, and running wildfire behaviour prediction models, combining edge, cloud, and HPC resources, using real-time measurements as input.

4.1. Reference Scenario

This Use Case will explore a wildfire detection scenario in remote forests, using the IoT device TT-Fire developed by Nature 4.0. TT-Fire is equipped with various gas and flame detection sensors, a camera, and a wireless internet module (NB-IoT or 5G). To preserve energy, the fire monitoring device TT-Fire operates in one of two modes: the default mode, in which only the basic sensors are activated, and data transmissions are less frequent; and the high-alert mode, which is activated if there’s a suspected fire nearby.

In the default operational state, each device wakes up at given time intervals checking for possible signs of fire, and if nothing is detected it returns to sleep for energy saving purposes. Additionally, the device transmits its measurements to an external database at a scheduled time once every hour.

The high-alert mode of a device is activated if it detects signs of suspected fire, upon which it sends a distress (wake-up) signal to other nearby devices. In high-alert mode, the device will start gathering measurements from all sensors, including images from the camera, and a FaaS request will be sent to the COGNIT Framework every minute until a

certain condition is met (no flame/fire detected in the N past offloading requests). The function will use a sophisticated ML model to assess the presence of fire, using the measurements and images supplied as arguments to the function.

A device that receives a distress signal wakes up and starts scanning the environment with all sensors available (camera included) to check if it is able to detect the fire; if so it enters high-alert mode, otherwise after some detection attempts it returns to sleep. This potential sudden cascade of offloading requests to the edge is a challenge the COGNIT Framework will have to overcome.

The type and the volume of data also differs between the above mentioned cases, in particular in the high-alert scenario, in which the images collected by the cameras of the devices are also sent, and the sending period is shortened to better follow up the event bringing to the overall increase in transmitted data to manage.

Unique Challenges

Implementation of the scenario described above presents a number of unique challenges and novelties, including:

- The devices and communication gateways are located in highly remote areas, which means that minimising maintenance and energy use are highly prioritised.
- Due to energy conservation measures, the devices will rarely make offloading requests during normal/default operation—however, in the unpredictable event of a suspected fire, many nearby devices could potentially wake up within a very short amount of time, causing a sharp peak of heavier-than-usual FaaS offloading requests to the edge.

4.2. Objectives

The key objective here is to test how COGNIT deals with an unforeseen/sudden and overwhelming peak of requests. Specifically, in default mode the main focus is to analyse and save the collected data in the most cost-efficient and energy-efficient way. On the other hand, during a wildfire event, low-latency features and advanced data analysis capabilities become important—this scenario causes a rapid change in service level requirements.

To support the development of smart energy management strategies for the devices, the goals of this Use Case are moreover to:

- Evaluate the feasibility of performing image analysis directly on the device.
- Potentially compare the energy consumption of performing image analysis directly on the device, with performing it on the edge.
- Evaluate the impact of the offloading on the battery of the devices, in particular assess the increase in operating time if the tasks are offloaded.
- Develop a smart energy management strategy when deciding whether to offload or not, taking the battery levels of the device into account.

Also, this specific Use Case is expected to provide a unique contribution towards the validation and achievement of the following global KPIs of the COGNIT Project:

- **[KPI2.1]** Meet experience level agreements with dynamic needs without intervention of the user.
- **[KPI3.3]** Automatic scale up/down of the microVM running the Serverless Runtime.
- **[KPI4.3]** Reduction of at least 10% energy consumption compared with manual edge deployment.

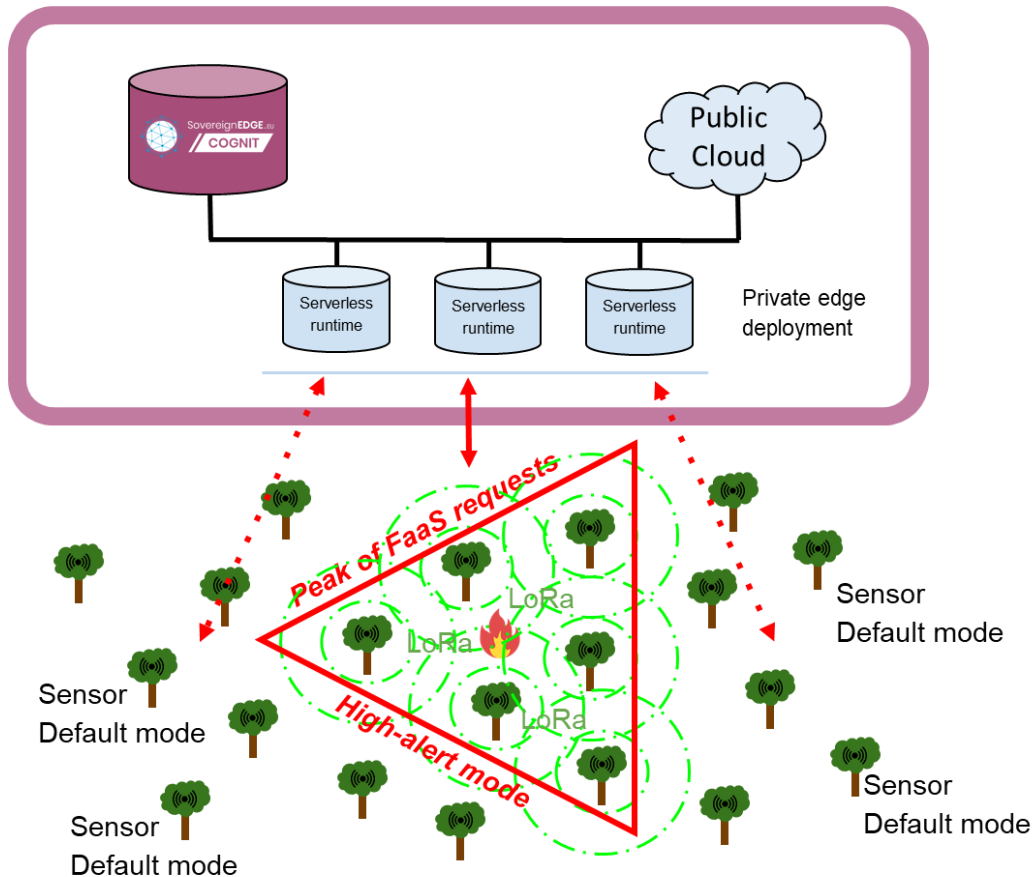


Figure 4.1. Architecture for the Wildfire Use Case.

4.3. System and Architecture Description

This Use Case involves the following system components:

- Monitoring device (TT-Fire): The device will be equipped with O_3 , CO_2 and PM (2.5, 5 and 10) sensors, a TRH (air Temperature and Relative Humidity) sensor, a flame detection sensor (with a range of approximately 150 m) and a camera. An internet module (NB-IoT or 5G) will be used to connect each device with the cloud-edge continuum.
- Private edge deployment close to device (managed by COGNIT): E.g., using 5G or NB-IoT connectivity.

- Private edge node deployment (managed by COGNIT) in close proximity to the forest.
- Public cloud: Possibility to manage public cloud resources under COGNIT.
- External cloud/database: The results can be sent to an external database hosted on a public cloud service, for long-term data storage and integration with visualisation tools to simplify wildfire management (Fire maps).

The COGNIT Client will run on the monitoring devices, and the private edge nodes will be managed by the COGNIT Framework. In the default operation mode, the devices intermittently monitor their surrounding environment for signs of fire, using only the basic sensors (not the camera). If a device detects measurement values indicating the presence of a flame, a suspected wildfire event is triggered and the device is put in high-alert mode, during which it will activate its camera and send a FaaS request to COGNIT every minute, until a certain condition is met (i.e. no fire detected in the past N offloading requests).

The Serverless Runtime will perform machine learning inference and image segmentation using the following data from the device as arguments to the Serverless Runtime:

- O_3 , CO_2 and PM (2.5, 5 and 10) concentrations,
- Air humidity and temperature,
- Fire intensity from fire detection sensors,
- Images from cameras.

For now, the function will not require any input data from previous FaaS requests. The data and results will be transmitted to an external service hosted on a public cloud. A temporary data storage could be required if data from more than one sensor should be taken into account with the aim to increase the accuracy of the information provided.

When high-alert mode is activated in a device, the device will furthermore send a distress call to wake up nearby devices. The devices receiving the distress call will activate and use all sensors to scan for possible presence of fire, using FaaS requests to offload computations. If the presence of fire is confirmed, the high-alert mode is triggered and the chain goes on, otherwise the sensor returns to default mode after a given time.

4.4. Use Case Infrastructure, Demonstration, and Validation

The demonstration scenario will be based on a deployment scenario of 120 devices covering the monitoring needs of a forest of 350 hectares. As previously mentioned, this Use Case will be used to test the COGNIT Framework under sudden and overwhelming peaks of requests. COGNIT features could be tested in three iterations:

1. Realistic simulation of a wildfire event on a virtual TT-Fire network in order to test the reaction of the COGNIT Framework to the sudden increase in resource demand.
2. Use of TT-Fire in an open field to check the reaction of the sensors and the COGNIT Framework in a real scenario.
3. Installation of a network in a real environment by the end of the project.

For the first iteration, a suitable algorithm will be developed in order to virtually simulate the devices and their behaviour. The simulation will take into account the distribution of devices in the forest, and a realistic wildfire scenario will be simulated, including the waking up process with the subsequent upload of realistic images.

Hardware

The hardware elements necessary for the validation and demonstration of the use case:

- TT-Fire platform with 5G or NB-IoT module for connectivity purposes

Software

The software elements necessary for the validation and demonstration of the use case:

- TT-Fire firmware for managing the device modes and integration with the COGNIT Client.
- Image recognition and fire detection software.
- Software for TT-Fire network simulation in order to test COGNIT Framework under different scenarios.

Milestones and validation criteria

Successful demonstration of the Use Case will be evaluated against several achievements and milestones.

The following achievements are expected for the Project's second milestone at M15 (based on a demonstration of a wildfire event scenario using virtual TT-Fire network):

- Identify and develop a lambda function that should be offloaded to COGNIT cloud-edge.
- Integrate the COGNIT Client with existing applications, and install it on a TT-Fire device.
- Demonstrate a TT-Fire device offloading data analysis and image segmentation to the COGNIT Framework.
- Create a manual edge deployment procedure using best practices, and consult an impartial expert to validate the procedure.
- Simulate a wildfire event on a virtual TT-Fire network, based on a realistic deployment scenario, to test the ability of the COGNIT Framework to adapt to sudden peaks of FaaS requests.

These advanced features are expected to be incorporated later on by Milestone M27:

- Scientific comparison of the energy consumption when using the manual deployment procedure, compared to COGNIT AI-enabled workload orchestration.
- Pilot deployment and demonstration of a TT-Fire network in a controlled, real environment.

4.5. Risks and mitigation plan

N° Risk	Potential risks		Contingency plan	
	Level (1:low; 5:high)	Impact	Description	Responsible
1	3	Insufficient connectivity	The location of the test could not provide a good NB-IoT or 5G coverage. Possible mitigation strategies could be testing COGNIT features using a virtual TT-Fire network, deploying a private 5G or NB-IoT access or changing the location of the test, can mitigate the risk.	R&D team
2	1	Issues in integrating COGNIT Client with the TT-Fire existing application	COGNIT Client is a new application and should be integrated with the existing device firmware. The risk can be mitigated through continuous collaboration and confrontation with other project partners and the required changes can be directly implemented and tested iteratively being the software directly managed by Nature 4.0.	R&D team
3	2	Functionalities tested in laboratory does not perform as well in the field	Every detail will be thoroughly tested at different levels more and more similar to the real case scenario with an increase in complexity. However, both hardware and software will be modified in a short time if an issue arises, being the development of the devices totally managed internally.	R&D team
4	1	Issue in running the image recognition algorithm on the TT-Fire device	Compared to the first version some upgrades will be made on the platform such as the addition of cameras. Performing image recognition tasks directly on the microcontroller unit (MCU) will help	R&D team

			<p>assess and compare power consumption offloading the task to COGNIT architecture or performing it on the platform. The MCU will be upgraded in order to grant the needed resources.</p>	
5	3	<p>Low memory on the device to run the COGNIT Client, the image recognition algorithm and the firmware</p>	<p>As in the previous cases, the MCU will be chosen taking into account the memory and computational power requirements. Also, the microphyton and the C++ programming languages will be both supported.</p>	R&D team
6	3	<p>Delays related to equipment procurement</p>	<p>During recent years, electronic supply shortages create difficulties in finding particular electronic components on the markets. The risk can be mitigated by securing the amount of components needed by the project as soon as the product is defined.</p>	UC Leader
7	4	<p>Unable to obtain permission for installing a private deployment supporting 5G or NB-IoT</p>	<p>The private deployment supporting 5G or NB-IoT is a nice feature for reliability purposes and for the possibility to guarantee coverage in a challenging environment. However, it has to obtain the country's permission to be installed. It is an additional feature being the main connection assured by public networks, moreover the country test location can be changed in order to obtain permissions. The risk can be mitigated starting to ask permission in advance assuring long times for fulfil possible requirements.</p>	Admin

5. Use Case 3: Energy



Supporting the energy transition in Europe requires allowing the wide and open accessibility of energy data. Due to diminishing fossil fuel resources (i.e. oil, gas, carbon) and the current geopolitical context, it is critical to develop solutions for individual energy independence of a household and/or small energy clusters. In order to do that it is necessary to develop methods for monitoring, predicting, and managing both energy production and consumption in a given environment. For this, the implementation of smart edge applications that make use of advanced AI/ML algorithms is a clear need.

Current deployments in the energy sector are limited by edge applications being deployed in resource-constrained environments (i.e. energy meters) and the very high security requirements for distribution system operators (DSO), which is usually a result of centralised architectures based on private DC infrastructures. Advanced AI/ML applications would benefit from being able to leverage, in a secure way, additional resources across the cloud-edge continuum that are much closer to the data sources and edge/IoT devices on the ground.

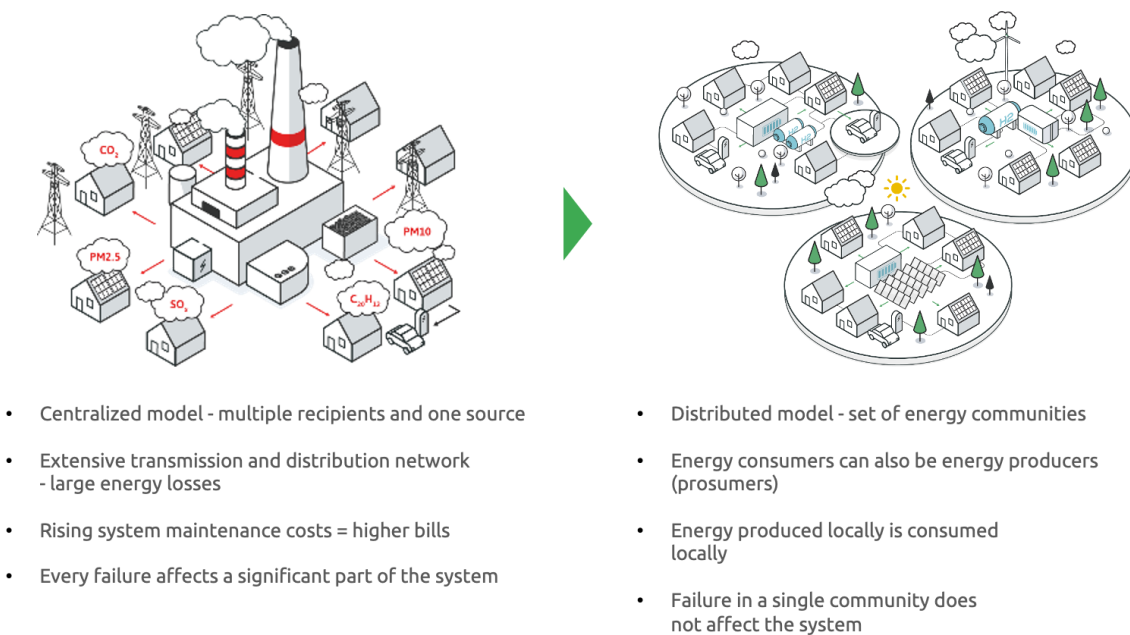


Figure 5.1. Transformation of the European energy sector.

This Use Case will demonstrate the capabilities of edge computing to support the ongoing transformation of the energy sector. It is moving from a hierarchical, centralised structure towards a more decentralised and distributed way of managing energy assets and networks.

Traditionally, the main functionality of an electricity meter has been to provide reliable measurement data and an interface for billing purposes—these functions were

implemented even in old electromechanical energy meters. Modern electricity meters, on the other hand, are powerful devices with functions such as providing measurement data using power line communication through a data concentrator unit and up to a complete data acquisition system.

Phoenix Systems develops the open source, real-time operating system Phoenix-RTOS, designed specifically for resource-constrained platforms, such as IoT devices. They are developing next-generation electricity meters, leveraging the Phoenix-RTOS platform, with the ambition to turn it into an Energy Assistant capable of running user applications directly on the electricity meters and manage appliances relevant to household energy balancing. Phoenix-RTOS stands out for its reliability, flexibility and scalability of software projects for low resources platforms. The transformation of the European energy sector brings new challenges and requirements that electricity meters equipped with Phoenix-RTOS can meet.

Atende Industries develops the *besmart.energy* cloud platform, which allows energy communities, such as energy clusters or energy cooperatives, to balance energy and increase self-consumption rates, thereby reducing electricity costs for their members. Built-in high accuracy weather forecasts allow to predict production from PV or wind farms. Artificial intelligence algorithms can predict the energy demand of energy community members, resulting in active control of the demand side.

5.1. Reference Scenario

This Use Case will explore the scenario of using smart electricity meters to optimise green local energy usage in a household context, in which energy consumers are also energy producers (*prosumers*). The Use Case will be developed and tested by meeting its goals in Poland. In most Polish locations, the current energy system is carbon-intensive and centralised, which means that there are only a few locations in the country where energy is generated—electricity must be transmitted over long distances through the transmission and distribution network resulting in high losses. Bottlenecks and disruptions in the network have the potential to affect huge areas and populations.

The energy industry of the future will be based on distributed systems, relying on renewable energy sources (RESs) and energy storage solutions. This highly distributed model of the energy network features many small producers of energy, aiming to reduce costs, risks, and intensity of greenhouse gas emissions, as well as to eliminate transmission energy losses because energy is produced and consumed locally. To make this a reality, there is a strong need to manage energy consumption as well as its production to optimise usage of local energy. Electricity meters, already at the interface between the building and the power grid, are ideally positioned to manage such distributed smart energy systems.

In this scenario, the smart electricity meters will run a number of user applications to manage important appliances and energy assets installed (from grid topology perspective) behind the meter, adjusting and optimising operation in real time, according to user preferences. These appliances and assets include energy storages, photovoltaic installations, heat pumps, electrical vehicle chargers, and electric floor heating. By empowering electricity meters with apps, connected to services equipped with advanced decision-making algorithms (and eventually pretrained AI models), they turn into highly

personalised Energy Assistants. Running user apps is possible thanks to the Phoenix-RTOS operating system, which offers all needed mechanisms for effective partitioning and as a result to reach full safety in separation of the partition for user apps and Distribution System Operator (DSO) partition, including the legally relevant Measuring Instruments Directive part. The separation between user space and DSO space is shown in Figure 5.2.

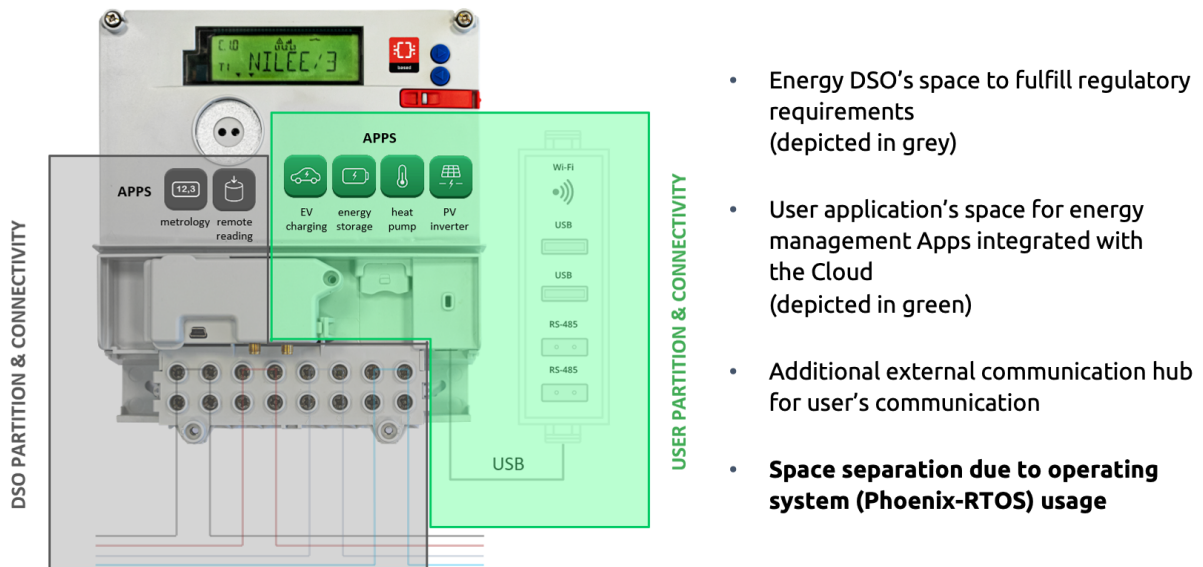


Figure 5.2. Smart energy meter

To optimise green local energy usage, decision algorithms should consider weather forecasting data and other relevant predictions, e.g. energy cost, energy consumption or production. Such services are provided by the external smart energy management cloud platform *besmart.energy*, that provides weather forecasts data and predictions about energy consumption and production, as well as the energy cost predictions. Beyond that, the *besmart.energy* platform offers a graphical interface, which allows for unified system management, analysing historical data about e.g. energy consumption and production statistics.

Even though electricity meters are nowadays much more powerful than ever, resource limitations remain one of the main challenges for software developers targeting electricity meters. Thanks to the use of the lightweight operating system Phoenix-RTOS as the basis for developing applications for resource constrained platforms, such as electricity meters, we can manage these resources optimally and additionally benefit from software modularity. Offloading decision-making and optimisation tasks to the edge allows these resource-constrained devices to take advantage of low latency benefits of highly distributed edge nodes and be powered up with additional external high performance computational resources.

Ultimately, this approach leads to cost savings because of more effective usage of energy and lowering overall demand for coal energy, as an example.

Unique Challenges

Implementation of the scenario described above presents a number of unique challenges and novelties, including:

- The devices (electricity meters) have very limited computing resources, e.g. only 500 kB of memory available for the entire user partition.
- Highly dynamic and varying processing and offloading needs that require relatively frequent offloading (every few minutes, depending on user preferences).
- Maximisation of local energy usage.
- Scaling the number of Serverless Runtime instances, since there will be many smart meters making concurrent requests.
- Sufficient computation resources needed for building and training AI models processing data for Use Case functionality.

5.2. Objectives

To sum it up, the goals of this Use Case scenario are to:

- Transform the regular energy meter into the Energy Assistant.
- Develop AI/ML algorithms for decision making in terms of managing energetically important appliances.
- Maximise local energy consumption by using the energy meter as an actuator based on data from the *besmart.energy* platform and algorithm run thanks to the COGNIT Framework.

Main aspects of demonstration and validation are to test performance of the COGNIT Framework:

- in case when there are large amounts of concurrent requests in the area of the same local edge node;
- in case of dynamic changes in Serverless Runtime performance requirements due to changes in user preferences.

Also, this specific Use Case is expected to provide a unique contribution towards the validation and achievement of the following global KPIs of the COGNIT Project:

- **[KPI2.1]** Meet experience level agreements dynamic needs without intervention of the user.
- **[KPI3.3]** Automatic scale up/down of the microVM running the Serverless Runtime.

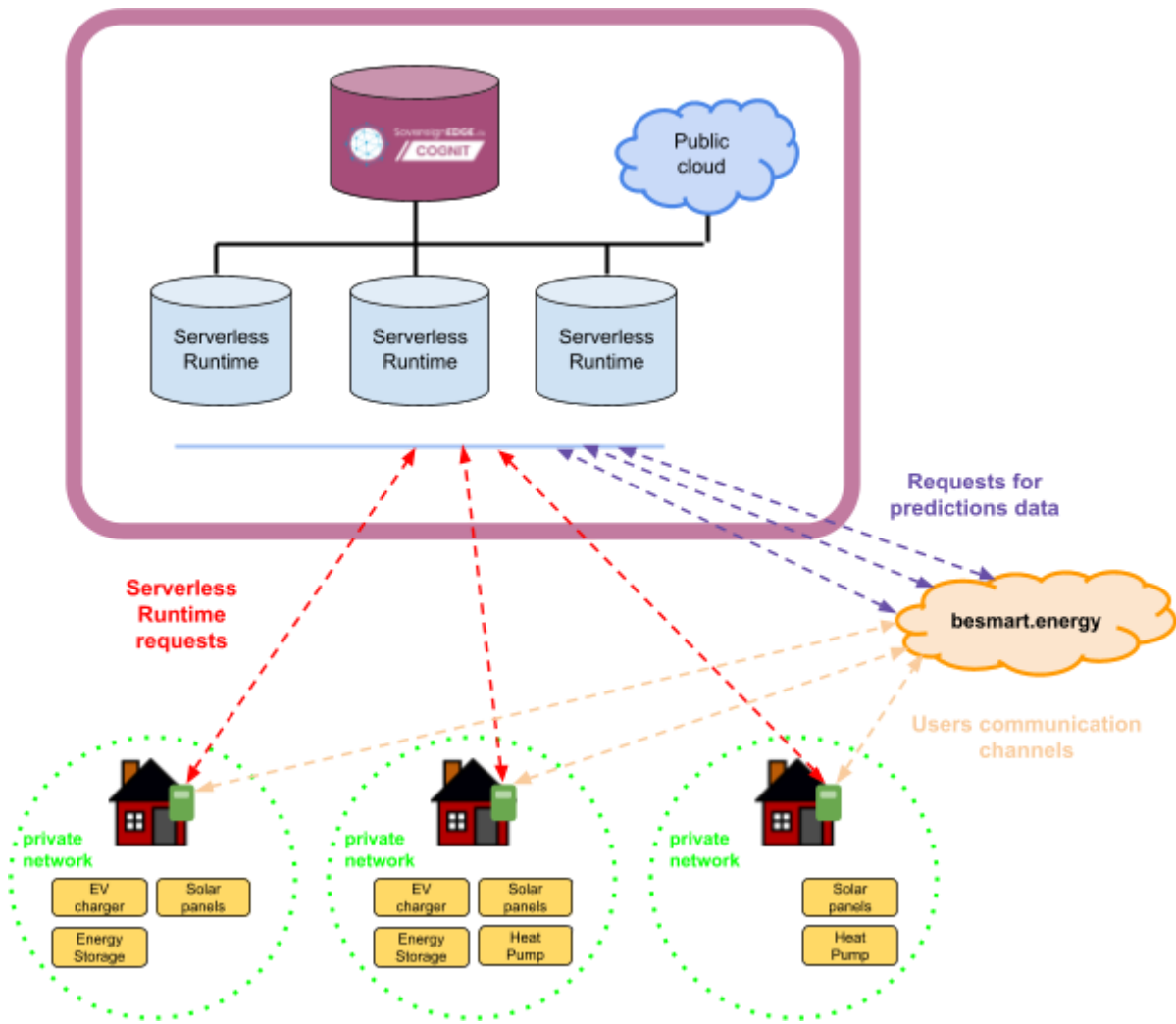


Figure 5.3. Architecture for the Energy Use Case

5.3. System and Architecture description

The Use Case involves the following system components:

- **besmart.energy** - a cloud platform for smart energy systems
- **Devices Controllers** - actuating devices
- **Electricity Meter-COGNIT Client** - current data provisioners, manager of actuators
- **Provisioning Engine** - Serverless Runtime manager
- **Serverless Runtime** - consists of a FaaS Runtime which is computation environment and a Data Service providing data storage for FaaS Runtime usage
- **User** - stakeholder

The Use Case scenario will require communications between the following components:

- **User** <-> **besmart.energy** - user preferences, control supervising process etc.
- **Electricity Meter** <-> **besmart.energy** - on/off supervising process

- **Electricity Meter <-> Provisioning Engine** – requests for Serverless Runtime, responses for requests
- **besmart.energy <-> Data Service (Serverless Runtime)** – weather forecasts, energy predictions
- **Electricity Meter <-> Devices Controllers** – current parameters of process
- **Electricity Meter <-> FaaS Runtime (Serverless Runtime)** – current energy data, current supervised process parameters, offloaded function, initiation of computing

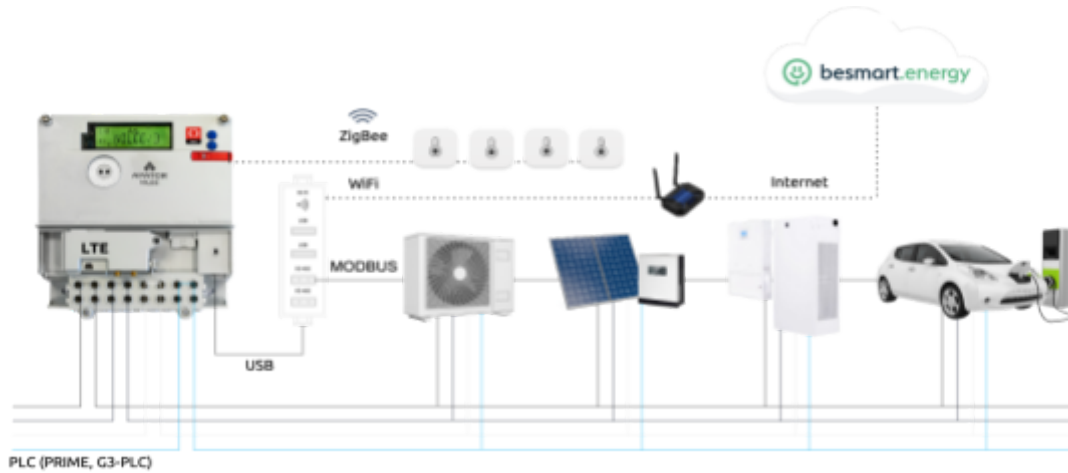


Figure 5.4. Examples of appliances connected to the smart energy meter

Appliances and energy assets important for the energy system will be connected to the electricity meter (see Figure 5.4) which, from the point of view of the appliances, is treated as their manager controlling their operational parameters. All the above features of the Electricity Meter equipped with the Phoenix-RTOS operating system combined with the possibilities that the COGNIT Framework offers, will result in achieving the goals defined in this Use Case.

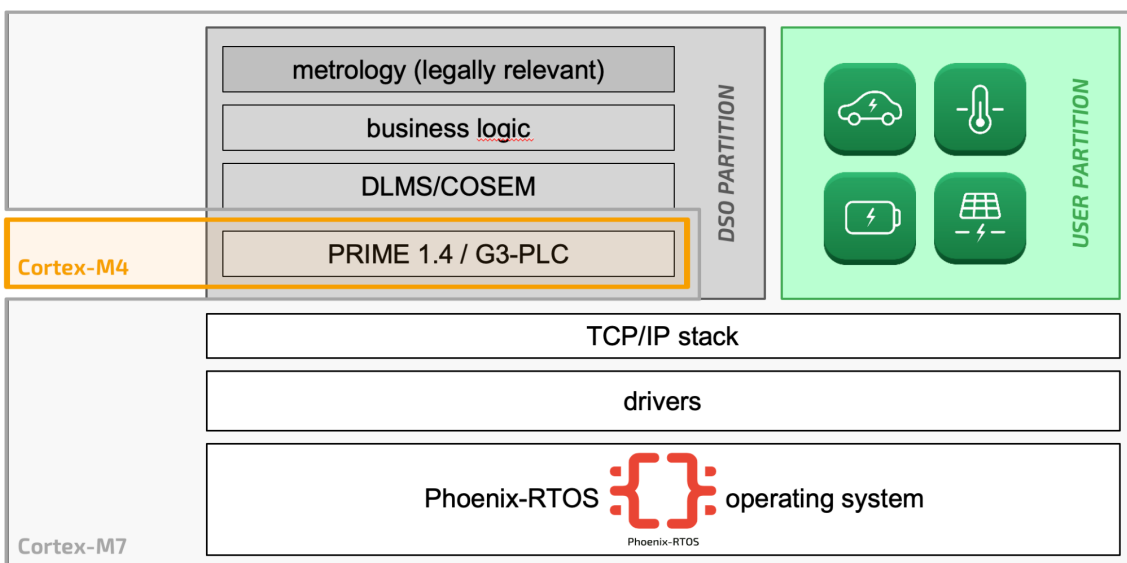


Figure 5.5. Logical partitioning of the functionality of the smart energy meter.

The Electricity Meter has to take care of implementing energy sector specific data models (e.g. COSEM), usage of standardised communication protocols (e.g. DLMS), security keys management, data encryption etc. From the software perspective, this means that the electricity meter should host applications from the Distribution System Operator (DSO), which provide all functionalities for meeting business needs. The necessary isolation between system-critical functionality (DSO applications), and user applications, is illustrated in Figure 5.5.

Supervising process will be running in a loop in which a single iteration will be triggered once per certain period of time according to user preferences (e.g. every 1/3/5 minutes). In this loop, the Electricity Meter will send current data to Serverless Runtime and initiate processing of the offloaded function.

The function to be offloaded is a management function deciding how to utilise household appliances at the given moment (e.g. based on current state of the devices and energy usage/production the function will return an hour when the car should start charging). Data which will be used in processing tasks are:

- Weather forecasts
- Energy production predictions
- Energy consumption predictions
- Energy cost predictions
- Current data about usage of energy
- Current data about supervising process

Offloaded processing task will be a decision algorithm which in conduct with above data should be able to produce a decision as result of computation. This decision effectively will be the current parameters of the supervised process which should be set.

A process diagram describing the Use Case is presented in Figure 5.6.

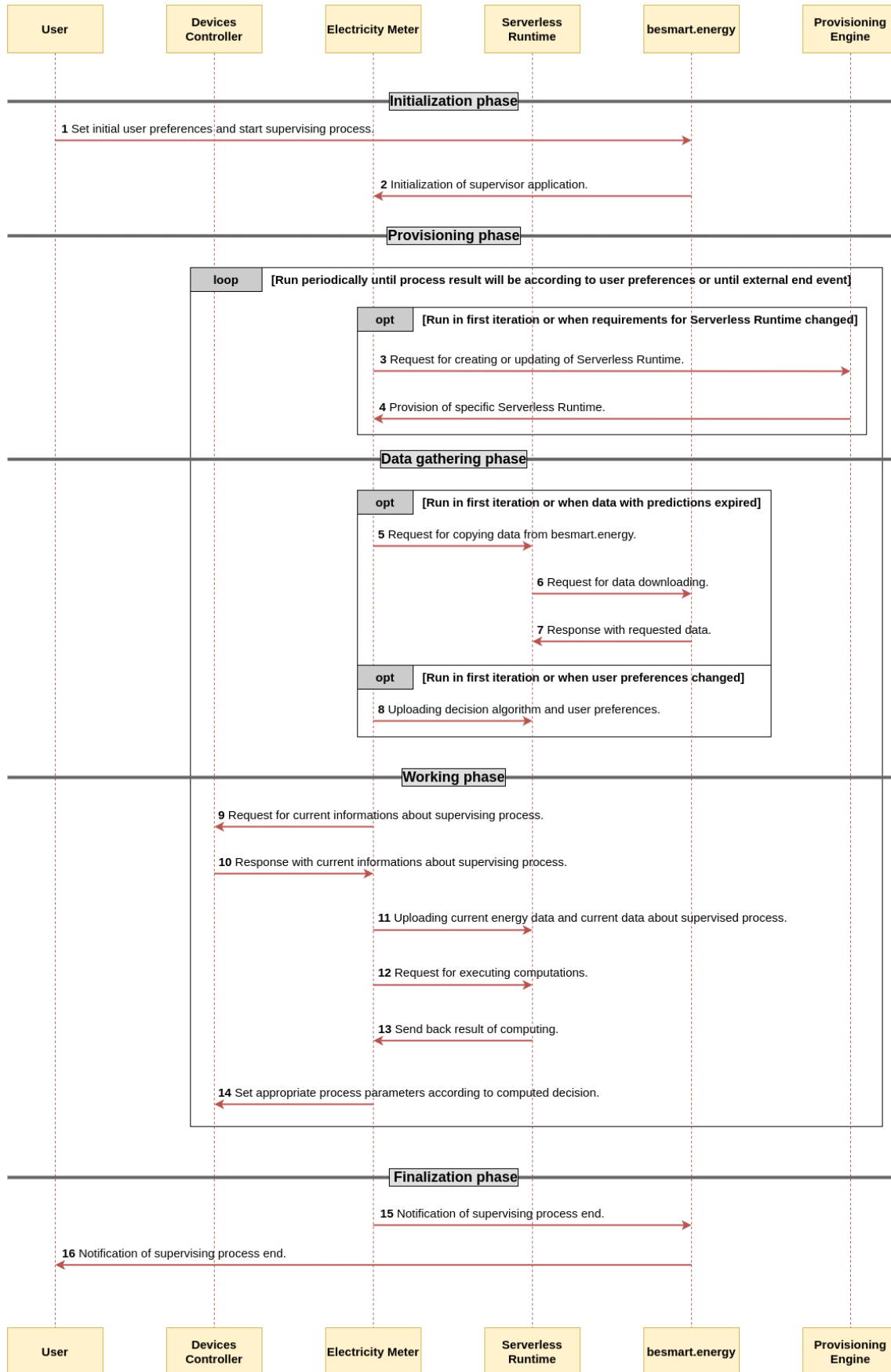


Figure 5.6. Process diagram for the Energy Use Case.

- 1) A user of the *besmart.energy* platform sets user preferences and starts the supervising process (e.g. EV car charging, Energy Storage charging etc.).
- 2) Besmart.energy platform communicates with user electricity meter, starts appropriate application for specific process and feeds it with initial user preferences.
- 3) Electricity Meter communicates with Provisioning Engine with request for Serverless Runtime provision.
- 4) Provisioning Engine response with provisioned Serverless Runtime for Electricity Meter.
- 5) Electricity Meter request for copying of data from external source to the Serverless Runtime. These data are predictions data from *besmart.energy* platform: weather forecasts, energy consumption/production predictions or energy costs predictions.
- 6) Serverless Runtime request for data from the *besmart.energy* platform.
- 7) Besmart.energy sends data about predictions and/or AI pretrained models to the Serverless Runtime.
- 8) Electricity meter uploads the decision algorithm and user preferences to the Serverless Runtime.
- 9) Electricity Meter requests for current information about the process which is supervised. Electricity Meter communicates with Devices Controllers.
- 10) Devices Controllers respond with current process parameters to the Electricity Meter.
- 11) Electricity Meter uploads current energy data and current data about supervised processes to the Serverless Runtime.
- 12) Electricity Meter requests for execution of the offloaded decision algorithm to the Serverless Runtime.
- 13) Serverless Runtime sends back to the Electricity Meter result of computing.
- 14) Electricity Meter with accordance to the computed result sets process parameters.
- 15) Electricity Meter sends notification about end of supervision to *besmart.energy*.
- 16) Besmart.energy sends notification to the user about end of supervision.

5.4. Use Case Infrastructure, Demonstration, and Validation

The demonstration environment will have a single WLAN network and include the following equipment:

- Two Photovoltaic (PV) inverters are connected to the WLAN.
- An Energy Storage unit connected to the PV inverter using the RS485 interface and Modbus protocol.
- An EV charger connected to the WLAN network.

- Smart home HUB connected to the WLAN network, for controlling electric floor heating.
- Resistive heating mats are connected to the smart home HUB using relays by wireless radio protocol Z-Wave (868MHz).
- Temperature sensors are connected to the smart home HUB by wireless radio protocol Z-Wave (868MHz).
- The Electricity Meter is connected to the WLAN/LAN.

This test environment is expected to be in a real home equipped with the smart energy meter, or, alternatively, in a controlled lab environment. The electricity meter will be the point of coordination, managing and communicating with all these devices.

Milestones and validation criteria

Successful demonstration of the Use Case will be evaluated against several achievements and milestones.

For the Project's second milestone at M15, at which point the first demonstration of the Use Case will take place, the Use Case is expected to have reached the following achievements:

- Identify and develop a lambda function that should be offloaded to COGNIT cloud-edge.
- Integrate the COGNIT Client with existing applications, and install it on a smart electricity meter.
- Pilot deployments of COGNIT-enabled smart electricity in a lab environment.
- Gather data from a real scenario and test basic functionality.

These advanced features are expected to be incorporated later on by Milestone M27:

- Simulate concurrent FaaS requests from many smart electricity meters, with dynamic changes to user preferences and experience level agreements, to test the scalability of the solution.
- Demonstrate the Use Case in a real environment (running COGNIT Client and decision algorithm on the electricity meter and full integration with the COGNIT Framework).

Additionally, the Use Case will be evaluated against the following technical KPIs:

- Stable internet connection with ability to send data every TBD minutes.
- COGNIT IoT client using up to TBD RAM memory.
- Return of an offloaded function in maximum TBD seconds.
- Provisioning of Serverless Runtime in maximum TBD minutes/seconds.

5.5. Risks and Mitigation Plan

N° Risk	Potential risks		Contingency plan	
	Level (1:low; 5:high)	Impact	Description	Responsible
1	4	Insufficient connectivity	The location of the device may impact the connectivity as the coverage of GSM differs for different locations. Given that the goal is to feed the upper level system with energy consumption and production data within a one minute interval may the risk manifest it will be mitigated by developing means of device connectivity via end user LAN network instead of GSM.	Technical Lead
2	2	Insufficient resources on the device (electricity meter)	For the first phase of the project a simulator will be developed to check the concept within the simulation environment. Once the concept is verified the COGNIT Client (or its essential for the Use Case functionality) will be ported and used on the device.	Project Lead
3	2	Not achieving interoperability and orchestration	This risk is mitigated through stating the requirements for the Use Case and close collaboration with other parties involved in the project. Additionally, to mitigate this risk, an iterative approach for project development was assumed, allowing for the parties to identify any interoperability and orchestration problems in time.	Project Lead
4	3	Too high latency preventing correct performance of decision algorithm	In case the latency is too high it is foreseen to add an extra edge node in the test environment premises.	Technical Lead

6. Use Case 4: Cybersecurity



The new Edge Computing paradigm, while offering great advantages both from a technological perspective and in terms of bringing more balance to the cloud market, also comes with a number of intrinsic challenges and risks that have to be properly addressed through research and innovation. Moving computation and data processing services to the edge, far from secured data centers, leaves systems exposed to new threats. Edge Computing requires a new generation of intelligent security mechanisms to be deployed along with edge applications, providing low latency and resilient anomaly detection and remediation, including privacy preserving mechanisms that use federated learning and implement advanced authentication and authorization policies.

This Use Case will demonstrate an integrated cybersecurity solution for edge computing, focusing on smart mobility—the main challenge will be migration between edge nodes, in particular managing a DevSecOps pipeline in a multi-provider edge context with dynamic (geo-dependent) security policies.

CETIC, in collaboration with SUSE, will design and test a Use Case focused on securing the development and operation of an FaaS workflow on a smart mobility solution at the edge. They will implement a DevSecOps pipeline that integrates development, operations, and security activities to update and operate software deployed on simulated and physical autonomous rovers based on the [Robot Operating System](#) and the [KubeEdge](#) framework. This innovative DevSecOps approach will focus on providing anomaly detection at the edge by extending SUSE's [OPNI](#) AIOps engine, applying confidential computing to protect data in the rovers not only at rest or in transit but also in use (during runtime), and will orchestrate the security response using the [Vaccine](#) SOAR ("Security Orchestration, Automation, and Response) platform based on security policies.

6.1. Reference Scenario

This Use Case will explore the scenario of distributed edge security, focusing on anomaly detection and remediation in a smart mobility context. Platooning is a concept for coordinating a group of connected autonomous vehicles, usually for linking multiple trucks to act as a single convoy, in which one leader vehicle is followed by a number of other vehicles (followers). In a platoon, the vehicles can exchange information using a V2V (vehicle-to-vehicle) communication interface, and/or the follower can use sensors to keep distance and direction. In such a system, any intrusion in the communication can have serious consequences, and it is therefore important to detect anomalous communication behaviour. This Use Case will implement a DevSecOps pipeline that focuses on providing anomaly detection at the edge by extending SUSE's [OPNI](#) AIOps engine. The vehicles continuously collect data about all communication activities, and store them in a local logging system. When certain conditions are triggered, these logs are sent to this anomaly detection service.

As shown in Figure 6.1, cyber physical systems such as connected vehicles, are getting more and more connected, which means their attack surface increases, making them more

vulnerable. The increased computing capabilities in cars allow better protection strategies to counter those new threats. Examples of such strategies include filtering inbound and outbound network traffic or monitoring the car systems for unauthorised access or suspicious behaviour.

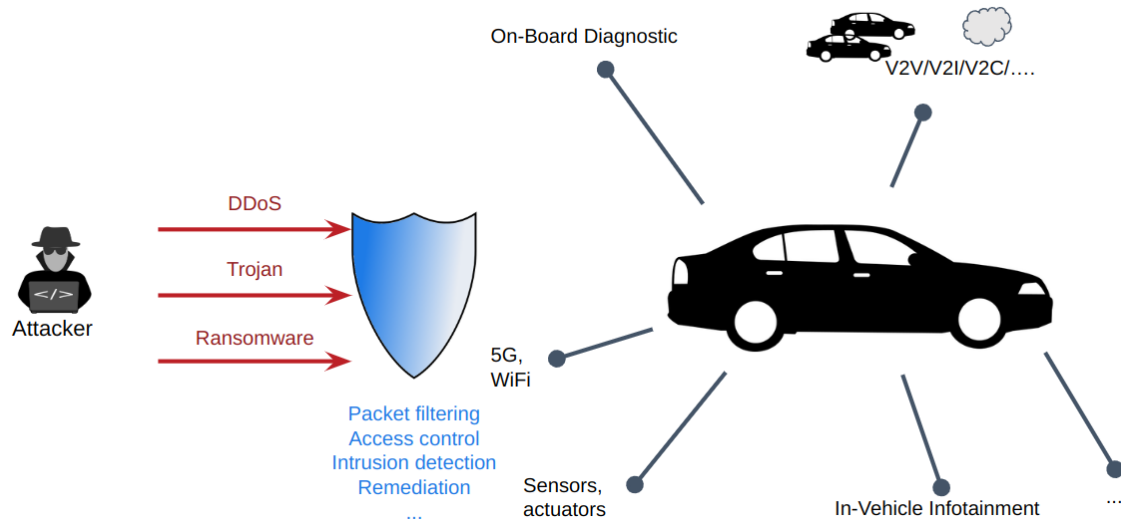


Figure 6.1. Cyber Physical Systems security

In this scenario, there are edge nodes with more compute power placed along highways and roads, as part of the smart city infrastructure. The platoon communicates with this road infrastructure using a V2I (vehicle-to-infrastructure) interface to get traffic information. As platoons move between zones (here, a geographic area with specific security requirements),² new requirements may apply, and it may be necessary to migrate Serverless Runtimes to edge nodes closer to the platoon.

The case study will be developed in the following iterations:

- **anomaly detection inference and remediation at the edge** with migration: rover data is copied to the edge where it is analysed by OPNI. When the platoon moves to a new zone and there is a closer edge node, the functions are migrated to the closer edge node. Rover data can be copied into a COGNIT Data Service.
- **distributed anomaly detection learning:** if OPNI has a learning phase that can be tailored to Rover data, then the distributed learning phase can be deployed across rovers, edge nodes and clouds.
- **federated learning:** if OPNI can be used in a federated learning, the case study can explore a federated learning approach for both the inference and learning phases. In a federated learning approach data is processed locally and only local models are shared in a cloud, not the data. Local models are then aggregated in a cloud into an aggregated model that can then be deployed back locally.

² See, for example, "How to Define Zones and Conduits": <https://gca.isa.org/blog/how-to-define-zones-and-conduits>

Unique Challenges

The scenario described above presents a number of unique challenges and novelties, including:

- **Edge devices mobility:** Rovers are driving from one zone corresponding to an edge node to another one, and require the Serverless Runtime (anomaly detection and remediation) to be deployed on the edge node that is closer to them and have access to the necessary data, such as models, logs, policies, remediation playbooks, etc.
- **Edge latency:** As rovers move from one edge node to another, anomaly detection and remediation need to be performed with low latency constraints. The process needs to be done as close as possible to the devices, which is a soft constraint that justifies the offloading of computation from the devices to the edge nodes.

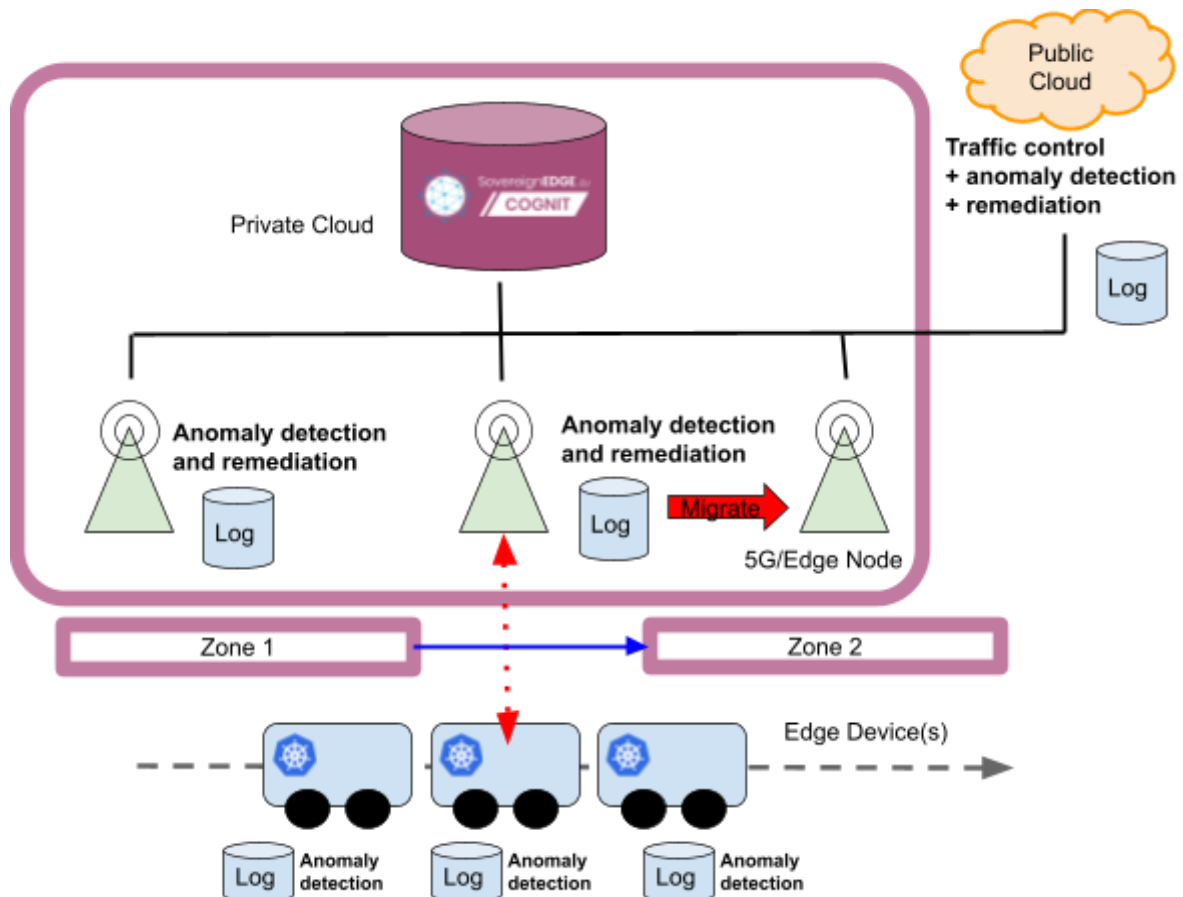


Figure 6.2. Architecture for the Cybersecurity Use Case to perform distributed anomaly detection (inference) and remediation.

6.2. Objectives

The objective of this Use Case is to demonstrate how the COGNIT Framework enables the secure mobility of edge devices. This is achieved by implementing distributed anomaly detection and security remediation on rovers (edge devices) that circulate on a smart road infrastructure (edge nodes) where compute-intensive security activities can be offloaded. To guarantee the confidentiality of those devices, a federated learning approach will be applied.

Also, this specific Use Case is expected to provide a unique contribution towards the validation and achievement of the following global KPIs of the COGNIT Project:

- **[KPI2.4]** Security processes and controls automated and integrated into the runtime lifecycle following a DevSecOps approach.
- **[KPI3.2]** Migrate workloads across the Continuum with minimal downtime.

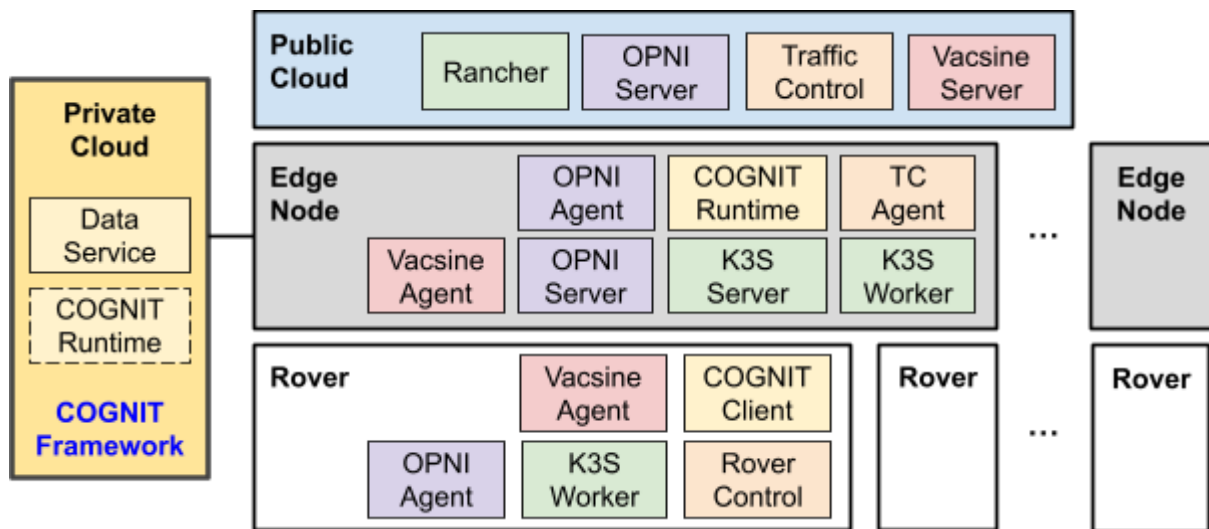


Figure 6.3. Components of the system.

6.3. System and Architecture Description

The overall architecture for the Use Case scenario is presented in Figure 6.2. The system consists of cloud nodes, edge nodes and rovers, as shown in Figure 6.3.

In the first iteration, a FaaS request to offload anomaly detection and remediation will be triggered by any of the following events:

- When the rover moves to a new zone and there is a closer edge node, depending on the security policy.
- Scheduled events.
- Uploading a new log.

The Serverless Runtime will require the following information as input:

- Logs and metrics transmitted by the Rovers as part of the FaaS request, this information is stored in OPNI (OpenDistro ElasticSearch/S3) at the edge node level for inference and at the cloud level for training;
- Anomaly detection models: The anomaly detection models created and managed by OPNI—the models can be updated on demand;
- Indicators of compromise, incidents and vulnerabilities as well as security policies and remediation strategies.

The Serverless Runtime will output/return the results of the anomaly detection to the COGNIT Client on the rover, and additionally send the results to the OPNI server present in the Public Cloud.

COGNIT Client (Rover)

The rovers generate data logs and metrics that contain traffic control and vehicle control messages (e.g. route information), as well as measurements from sensors and actuators (e.g. acceleration). Each edge device (rover) contains the following services:

- The COGNIT Client: an application that communicates with the COGNIT Provisioning Engine for offloading code and data to the Serverless Runtime.
- Vaccine Agent: apply security response for the edge devices.
- OPNI Agent: log collection and shipping.
- Rover Control: semi-autonomous driving control.
- Kubernetes worker node role that runs the Vaccine, OPNI, and Rover Control services—based on [SUSE's K3s](#) platform.

COGNIT Cloud-Edge Continuum

The edge nodes managed by the COGNIT Framework will perform anomaly detection and transmit rover logs and metrics to an external Public Cloud service. Each edge node contains the following services:

- Kubernetes (K3s) server node role that manages a K3s cluster local to the node.
- Kubernetes (K3s) worker node role that runs the Vaccine, OPNI and traffic control services.
- OPNI Agent: log collection and shipping.
- OPNI Server: retrieves ML models from the Data service for anomaly detection at the edge.
- Traffic Control agent: communication between the TC server and the rovers.
- Vaccine Agent: apply security response for the edge nodes.
- COGNIT Serverless Runtime.

External services on a Public Cloud

This external Public Cloud service is not managed by COGNIT as part of its cloud-edge continuum infrastructure. It receives rover logs and metrics to train anomaly detection

models, and deploy these anomaly detection models as a function managed by the COGNIT Framework. This external cloud resources will contain the following services:

- **SUSE Rancher**: manages the federation of K3s clusters.
- **OPNI server**: provides training, log storage, log aggregation (gateway), admin ui.
- **Traffic control server**: provides a traffic control web interface, and traffic control orchestration.
- **Vaccine Server**: security response orchestration.

6.4. Use Case Infrastructure, Demonstration, and Validation

In this use case, the edge devices are small robot cars (rovers) running along a track in a lab environment. The rovers are running on Raspberry Pi (ARM processor) with Linux/ROS, with the possibility of adding **Jetson Nano** when the GPU can be leveraged. The edge devices have the Kubernetes worker nodes role and connect to edge nodes that contain the Kubernetes server role. Concretely, the Use Case involves four edge nodes as Virtual Machines with 4 vCPU and 8GB RAM³ and 1 Public Cloud node as a Virtual Machine with 32 vCPU and 64GB RAM (all of them preferably based on **openSUSE**). The Public Cloud node will need a GPU for training purposes in the later iterations of the use case (OPNI recommends at least a NVIDIA k80).⁴ Table 6.1 shows the services contained by the edge nodes and by the private cloud node managed under COGNIT.

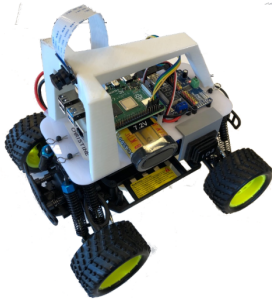
Edge Node	Private Cloud Node
<ul style="list-style-type: none"> ● Kubernetes (K3s) server node role ● Kubernetes (K3s) worker node role ● OPNI Agent & server ● Traffic control agent ● Vaccine Agent ● Serverless Runtime 	<ul style="list-style-type: none"> ● Kubernetes federated server role (SUSE Rancher) ● OPNI server ● Traffic control server ● Vaccine Server

Table 6.1. Services that are contained by edge nodes and by the private cloud node.

The rover platform, based on a **Donkey Car** design (see Figure 6.4), relies on the ROS framework for driving and interaction with its sensors and actuators. In order to provide a better localization of the vehicles, so the road infrastructure and the vehicles themselves know where they are, we may need to add additional sensors such as GPS. The first version of the rover platform was built during the H2020 project **SPARTA** as a demonstration scenario for continuous security assessment and certification.

³ <https://docs.k3s.io/installation/requirements>

⁴ <https://opni.io/architecture/highlevel/>



- Donkey Car chassis
- 2D Lidar
- Ultrasonic sensor
- Wide Lens camera
- RaspBerry Pi / Jetson Nano



Figure 6.4. The Rover platform

In later iterations of the use case, edge devices and edge nodes could communicate between each other and with the smart road infrastructure (V2V, V2I) using 5G technology. This would allow us to leverage 5G benefits such as lower latency, higher bandwidth or more intelligent network functions. This would require dedicated hardware, such as 5g antennas and modems that can be fitted on the rovers, as well as ideally a simulated 5G environment for testing.

Validation

To validate the anomaly detection across edge nodes, nefarious activity will be simulated while the rovers drive across road infrastructure edge nodes. The anomaly detection will need to detect this nefarious activity and distinguish it from the usual activity.

Security remediation will be validated by checking that the remediation strategies are working properly:

- The rover platoon keeps functioning after countermeasures have been applied,
- The platoon is stopped for safety reasons,
- A platoon member is quarantined.

Milestones and validation criteria

Successful demonstration of the Use Case will be evaluated against several achievements and milestones.

For the Project's second milestone at M15, at which point the first demonstration of the Use Case will take place, the Use Case is expected to have reached the following achievements:

- Create an offloading function that integrates OPNI anomaly detection engine.
- Edge-IoT-cloud system deployment model and security remediation.
- Simulations/first prototypes implemented for anomaly detection and remediation.
- Develop a distributed architecture for anomaly detection and remediation in the cloud-edge.

These advanced features are expected to be incorporated later on by Milestone M27:

- Federated architecture for detection and remediation.
- Cloud-Edge-IoT application prototypes (with Use Case functionality) implemented.

6.5. Risks and Mitigation Plan

N° Risk	Potential risks		Contingency plan	
	Level (1:low; 5:high)	Impact	Description	Responsible
1	4	Attack simulations risk adding vulnerabilities to the system that are exploitable by an outside attacker	Attack simulation will be performed in a secure environment (sandboxed), protected by good practices according to relevant ISO 27001 controls.	Technical lead
2	3	Delays related to equipment procurement	CETIC already has V2X equipment and a private cloud with the capacity to allocate the rest of the equipment.	Project lead
3	2	Implemented functionality does not give good results when applied in a real environment.	The framework and technology choices can be applied in other contexts, this Use Case is for example used and extended by CETIC in various research projects, notably industry 4.0 and space.	Project lead
4	3	The results of the pilot test indicate that there are stability, latency and scalability problems.	CETIC will procure spare rovers and sensors to ensure the resiliency of the pilot test. To reduce latency, better equipment can be fitted on the rovers (5G modem, more powerful wifi antenna, GPU on the rovers, etc.). The public cloud and CETIC infrastructure provides a scalability mechanism in case more compute resources are needed, and GPU services can be added to edge nodes to improve training performance.	Technical lead
5	3	Extending OPNI with distributed learning and federated learning proves too difficult due to its architecture.	The OPNI server instances can be moved to the edge nodes, and federated by a central instance in the cloud.	Technical lead

7. Use Case Requirements

This section summarises the results of the analysis of Use Case requirements described in previous sections. The requirements are given in five separate tables: the first one listing those requirements that all Use Cases have in common, plus four additional tables listing those requirements specific to each Use Case:

Id	Description	Source
UR0.1	Device applications should be able to offload any function written in C or Python languages.	All
UR0.2	Device applications should be able to upload data from the device ensuring data locality with respect to where the offloaded function is executed.	All
UR0.3	Device applications should be able to upload data from external backend storages ensuring data locality with respect to where the offloaded function is executed.	All
UR0.4	Execution of functions such as ML inference engines should be able to load machine learning models stored ensuring data locality with respect to where the function is executed.	All
UR0.5	Function execution can be executed in different tiers of the Cloud-Edge continuum according to network latency requirements.	All
UR0.6	Device application shall have the ability to define maximum execution time of the offloaded function upon offloading.	All
UR0.7	Device application shall have the ability to specify and enforce runtime maximum provisioning time and runtime shall be provisioned within the previously specified time.	All
UR0.8	Device applications must be able to request and obtain an authorization prior to establishing any further interaction with COGNIT.	All

Table 7.1. Common user requirements.

Id	Description	Source
UR1.1	Function execution shall be supported in shared, multi-provider environments (with different access and authorization procedures), and the execution must be isolated from other processes on the host system.	UC1
UR1.2	Device application shall have the ability to dynamically scale resources for offloading function execution to maximise exploitation of resources in shared environments, while avoid saturation or resources kidnapping.	UC1
UR1.3	Function execution should exploit data locality and prioritise edge nodes where the required data is already stored.	UC1
UR1.4	The whole life cycle of either function execution or code offloading should be auditable and non repudiable.	UC1
UR1.5	Device applications should be able to request execution over GPUs.	UC1

Table 7.2. User requirements for UC1 (Smart Cities).

Id	Description	Source
UR2.1	It shall be possible to obtain both a-priori estimates of expected, and actual measurements of, energy consumption of the execution of function.	UC2
UR2.2	COGNIT Framework should be able to adapt to rare events with sudden peaks of FaaS requests, in which the offloaded function requires much heavier computations and more frequent execution than usual.	UC2
UR2.3	Possibility for devices to request access to GPUs, when available, during high-alert mode.	UC2

Table 7.3. User requirements for UC2 (Wildfire Detection).

Id	Description	Source
UR3.1	Device Client and user applications shall share a maximum of 500 kB of available RAM in total.	UC3
UR3.2	It shall be possible for the user application to dynamically scale up/down resources for function execution due to changes in the user preferences.	UC3
UR3.3	The SDK for the Device Client shall have support for the C programming language.	UC3

Table 7.4. User requirements for UC3 (Energy).

Id	Description	Source
UR4.1	The Device Client should have the ability to dynamically set the permissible edge nodes for executing the function based on policy (e.g. geographic security zones, distance to edge node).	UC4
UR4.2	The COGNIT Framework should have the ability to live migrate of data/runtime to different edge locations based on policy and location of function execution (e.g. geographic security zones, distance to edge node).	UC4
UR4.3	The Device should be able to request the execution of a function as close as possible (in terms of latency) to the Device's location.	UC4

Table 7.5. User requirements for UC4 (Cybersecurity).

8. Summary of Use Case Cloud-Edge Infrastructure

Each Use Case features a unique combination of infrastructure resources along the cloud-edge continuum. Table 8.1 below summarises the configuration for each Use Case, described in previous “*System and Architecture Description*” subsections. In the table we distinguish between public cloud/edge resources (which refers to infrastructure operated by a provider, open to third-party use) and private infrastructure dedicated to the Use Cases. The core data center (Private Cloud) resources will be hosted by RISE at their [ICE Data Center](#) testbed facilities in Luleå, Sweden (see Section 9 for more details).

UC	Cloud-Edge Infrastructure			
	<i>Private Edge</i>	<i>Public Edge</i>	<i>Private Cloud / Data Center</i>	<i>Public Cloud</i>
Smart Cities	M-Hub Gold (far edge) Mini-DC at local traffic control center (near edge)	Potentially MEC (5G)	RISE ICE	Bare Metal resources
Wildfire Detection	Private deployment supporting 5G or NB-IoT (far edge) Edge server in facilities close to forest (near edge)	Potentially MEC (5G/NB-IoT)	RISE ICE	Bare Metal resources
Energy	Private on-prem supporting WiFi/Ethernet (far edge) Mini-DC for metering data storage (near edge)		RISE ICE	Bare Metal resources
Cybersecurity	Private on-prem (far edge) Private mini-DC for simulation and tests (near edge)	Potentially MEC (5G)	RISE ICE	Bare Metal resources

Table 8.1. Use Cases’ combination of infrastructure resources across the cloud-edge continuum.

PART II. Test Infrastructure

9. Testbed

The testbed involves Private Cloud resources, hosted at the RISE ICE Data Center in Luleå, Sweden, that act as the management cluster of the cloud-edge testbed, and supports integration of geo-distributed resources, such as private on-premise edge nodes. The edge infrastructure of each Use Case will be connected to and managed by the central Management Cluster located at the ICE Data Center.

9.1. Background



Figure 9.1. City of Luleå (Sweden).

The RISE ICE (“Infrastructure and Cloud research & test Environment”) data center testbed was inaugurated in 2016 in Luleå, Sweden—where Facebook has a data center—and is located within walking distance of Luleå University of Technology. The testbed is open for testing and demonstration experiments by clients and partners from all over the world, but is primarily intended to support European R&D projects, universities, and companies.



Figure 9.2. Location (N 65° 36' 43.558", E 22° 7' 56.554") and main access to the ICE Data Center.

The center coordinates national research projects as well as the testing and development of all features in the technology stack; the infrastructure and building of data centers, edge and cloud applications, IT architecture, and machine learning. ICE mission is to put Sweden at the forefront in the area of effective data center solutions, edge computing, cloud applications and data analysis.

The testbed has three different modules, open for testing and experiments, along with a demo space that can fit 50+ people. These modules offer isolated testing environments for different purposes, including testing of IT/cloud-related applications and big data experiments, and testing of facility/utility management and cooling innovations. Moreover, the testbed is equipped with its own microgrid infrastructure, which includes solar panels, fuel cells, batteries, thermal storage tanks, and heat pumps for heat re-use.

9.2. RISE ICE Cloud-Edge Infrastructure

The testbed is using only open source technologies—including [OpenNebula](#)—to build a platform that offer both Infrastructure-as-a-Service (IaaS) and Container-as-a-Service (CaaS) across the cloud-edge continuum:

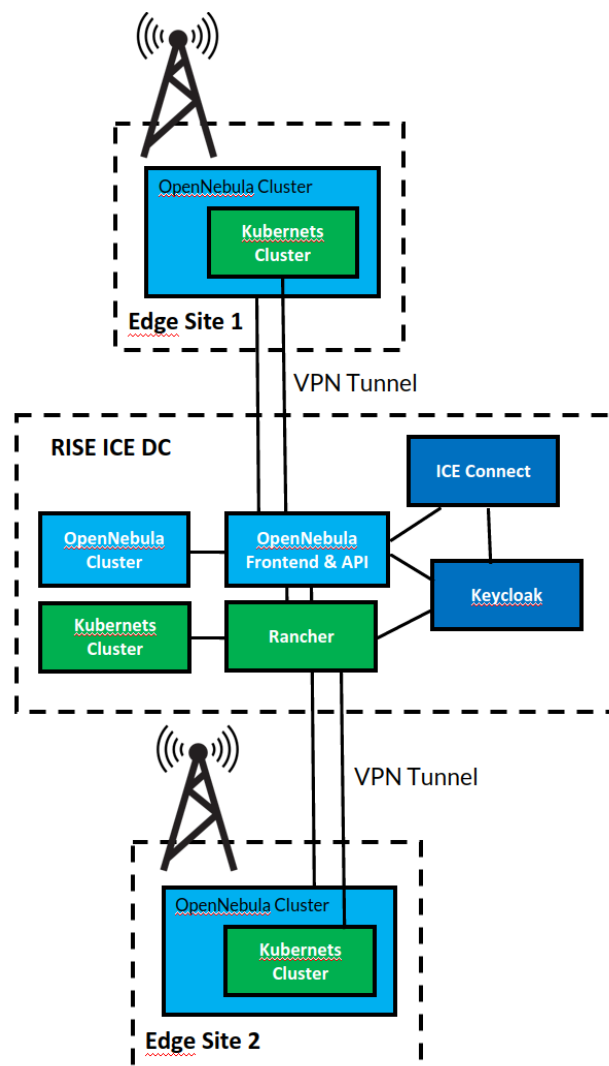


Figure 9.3. General cloud-edge architecture of the ICE Data Center.

RISE ICE Data Center (Private Cloud Nodes)

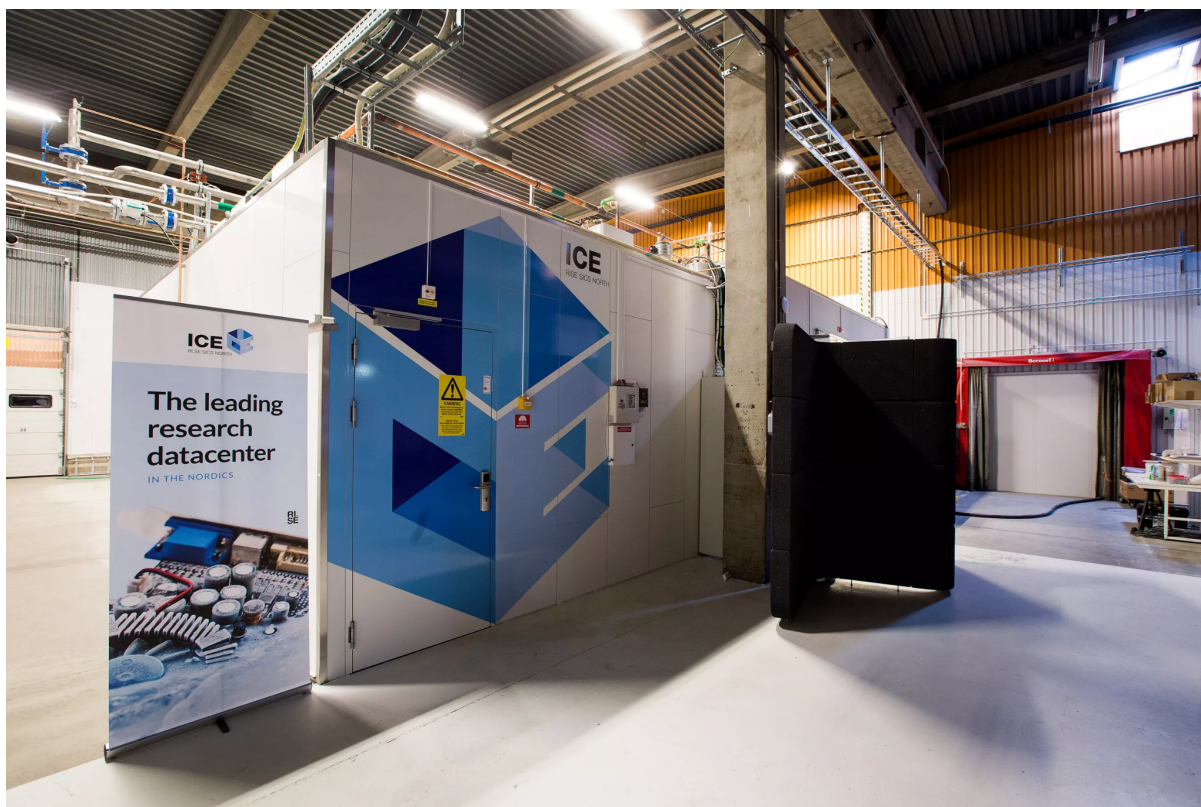


Figure 9.4. Inside view of the ICE Data Center.

This Private Cloud / DC offers access to a number of different resources and services:

- Kubernetes cluster with 200+ GPUs managed with SUSE Rancher.
- Virtual Machines of various sizes with and without GPU.
- Bare Metal servers of various sizes and models with and without GPU.
- S3 storage powered by a Ceph cluster with ~1.7PB raw storage capacity.
- Docker registry via Harbor.
- GitLab.
- Discourse.

Access to all services are provided via an in-house developed self-service portal called ICE Connect. The portal allows new users to register and create projects. Projects can be seen as isolated tenants that own resources like CPU, Memory, GPU, storage, etc. Users can have access to several projects.

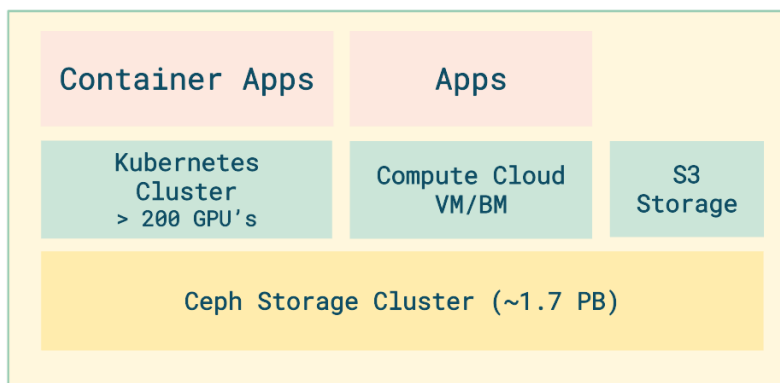


Figure 9.5. Cloud platform stack.

Edge Clusters (Edge Nodes)

Edge Clusters can be seen as smaller cloud instances that usually sit directly connected to some access technology like a 5G network. All Edge Clusters are managed from ICE Data Center via VPN tunnels. In addition to cloud resources, the users of ICE Connect will also have access to edge resources.

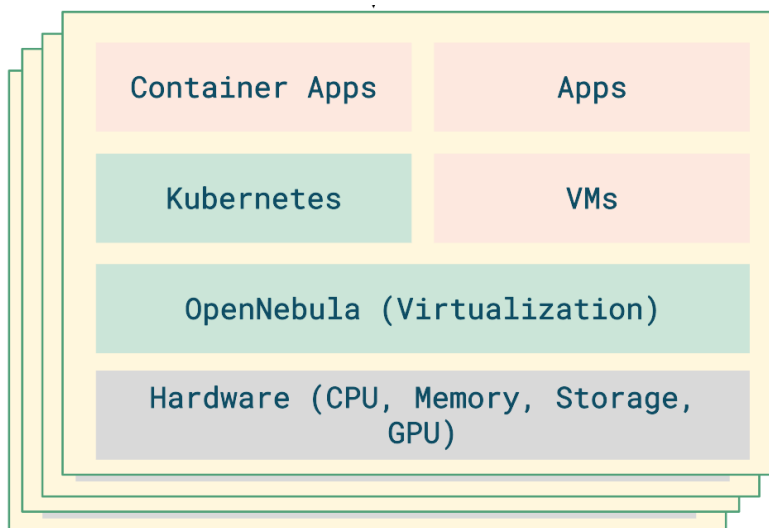


Figure 9.6. Edge platform stack.

Infrastructure-as-a-Service (IaaS)

The IaaS offering is based on OpenNebula, the open source virtualization platform. The cloud-edge spanning IaaS is centrally managed by an OpenNebula front-end running at the ICE Data Center. Because the front-end needs direct access to compute hosts, VPN tunnels are set up between Edge Clusters and the OpenNebula front-end. ICE Connect users can use the centralised management service to create and manage cloud/edge infrastructure resources (VMs, network, storage, etc.) using both GUI and API.

Container-as-a-Service (CaaS)

The features of the CaaS offering are very similar to the IaaS. The CaaS solution connects all multi-tenant Edge Kubernetes clusters to one common SUSE Rancher service. Rancher

implements centralised multi-cluster management and provisioning of Kubernetes. And similar to IaaS, it offers common management access to all sites through Kubernetes API and Rancher UI/API.

9.3. COGNIT Testbed Architecture

The current version of the COGNIT Testbed (M3) has been built on top of the RISE ICE Cloud-Edge infrastructure. Even though it was technically possible to use the original RISE OpenNebula deployment, the decision was not to do so because the OpenNebula deployment itself will be altered during the execution of the COGNIT Project, and the required administrator privileges for that instance cannot be granted to COGNIT Partners.

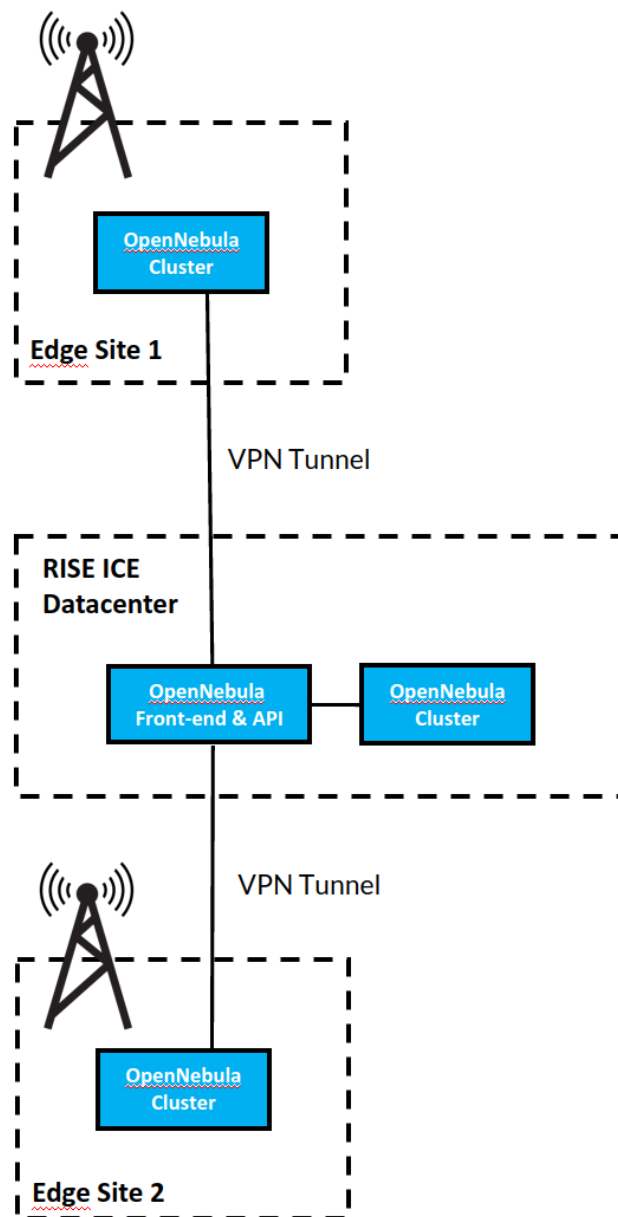


Figure 9.7. Current COGNIT Testbed architecture (M3).

OpenNebula Front-end & API

The OpenNebula central management components have been installed on Virtual Machines running on the ICE Data Center. The deployment itself has been configured in a classic OpenNebula high-availability configuration, consisting of three VMs.⁵

The screenshot shows the ICE Connect web interface. The top navigation bar includes the RI SE logo, ICE Connect, and menu items: SERVICES, PROJECT, PRICING, ADMIN, SUPPORT, cognit, and daniel.x.olsson@ri.se. The left sidebar has a 'Virtual' section selected, with sub-items: GPU, Bare-Metal, Security, Pricing, Docs, and Back. The main content area is titled 'Virtual Machines' and contains a table with the following data:

Name	Status	IP Addresses	Information	Actions
opennebula-1	ACTIVE ✓	Addresses ▾	Info ▾	Actions ▾
opennebula-2	ACTIVE ✓	Addresses ▾	Info ▾	Actions ▾
opennebula-3	ACTIVE ✓	Addresses ▾	Info ▾	Actions ▾
vpn-router-lb	ACTIVE ✓	Addresses ▾	Info ▾	Actions ▾

Below the table, it says 'Showing 1 to 4 of 4 entries'. There are two buttons: 'Launch new VM' and 'Help'. An 'Info' tooltip for the 'vpn-router-lb' VM shows the following details:

- OS: Ubuntu 22.04 LTS
- vCPUs: 2
- Memory: 2048 MB
- SSH IP: 213.21.96.180
- SSH Port: 28202
- SSH User: ubuntu

Below the VM table, there is a 'Volumes' section with the text 'Total size of volumes: 0.195 TiB'.

Figure 9.8. ICE Connect showing deployed front-end VMs.

The screenshot shows the ICE Connect web interface with the 'Bare-Metal' section selected in the sidebar. The main content area is titled 'Bare-Metal Servers' and contains a table with the following data:

Name	Status	IP Address	Information	Actions
p02r11srv01	DEPLOYED ✓	Addresses ▾	Info ▾	Actions ▾

Below the table, it says 'Showing 1 to 1 of 1 entries'. There are two buttons: 'Deploy Bare-Metal Server' and 'Help'. An 'Info' tooltip for the 'p02r11srv01' server shows the following details:

- OS: Ubuntu 22.04 LTS
- CPUs: 24
- Memory: 256.0 GIB
- Disk: 32605.8 GB
- SSH IP: 213.21.96.180
- SSH Port: 32001

Figure 9.9. ICE Connect showing bare-metal resources used as OpenNebula virtualization hosts.

⁵ https://docs.opennebula.io/6.6/installation_and_configuration/ha/frontend_ha.html

A DNS entry has been allocated which is pointing at load balancer (i.e. nginx) running in VM vpn-router-lb. A signed SSL certificate via *Let's Encrypt* for the domain has been installed. Traffic to following endpoints are load-balanced over the three front-end VMs:

- OpenNebula GUI (Sunstone): <https://cognit-lab.sovereignedge.eu>
- OpenNebula RPC API: <https://cognit-lab.sovereignedge.eu/RPC2>



Figure 9.10. Login screen of the OpenNebula GUI (COGNIT Testbed).

User accounts in OpenNebula have been created for COGNIT Partners that need access to this initial deployment (M3) of the COGNIT Testbed geo-distributed cloud-edge infrastructure, which is now ready to easily incorporate additional edge nodes as bare-metal resources in other locations (e.g. by OpenNebula in Madrid, Spain). For Use Cases that eventually need local compute resources, bare-metal servers can now be deployed on-premises and added under the COGNIT Testbed environment. After deployment, the whole process only requires a VPN tunnel (wireguard) to be created in order to connect the new edge node to the existing OpenNebula instance. Then, the bare-metal servers can be added as hosts to a new cluster instance within OpenNebula.

10. Conclusions and Next Steps

This document describes the Use Cases and their requirements, together with an initial plan for the demonstration and validation activities in the coming research and innovation cycles (until M15). It furthermore describes the initial setup of the common COGNIT Testbed and the geo-distributed cloud-edge architecture that will be available to the Use Cases for research, development, demonstration activities. The requirements of the Use Cases are further analysed in Deliverable D2.1, which specifies the initial COGNIT Framework Architecture in detail.

This is the first version of the Use Cases Scientific Report. This document will be incrementally updated and released at the end of each research and innovation cycle (i.e. M9, M15, M21, M27, M33), incorporating updates (if any) to the requirements of the Use Cases, a description of the open source integration and certification process, and a description of the research, development, and validation work done in each cycle.