A Lightweight Software Stack and Synergetic Meta-Orchestration Framework for the Next Generation Compute Continuum

D2.1 Requirements, Use Cases Description and Conceptualization of the NEPHELE Reference Architecture

Document Identification			
Status	Final	**Due Date**	30/05/2023
Version	1.0	**Submission Date**	30/06/2023

Related WP	WP2	**Document Reference**	D2.1
Related Deliverable(s)	D2.2, D3.1, D4.1, D6.1	**Dissemination Level (*)**	PU
Lead Participant	INRIA	**Lead Author**	Adriana Arteaga Arce
Contributors	All partners	**Reviewers**	Guillermo Gómez (ATOS), Leonardo Militano (ZHAW), Rudolf Susnik (ININ), Panagiotis Papadimitriou (UOM)

Keywords:
Virtual Object, VOStack, Synergetic Orchestration, Computing Continuum, IoT Interoperability, IoT Virtualization, Application Graph

(*) Dissemination level: **PU**: Public, fully open, e.g., web; **CO:** Confidential, restricted under conditions set out in Model Grant Agreement; **CI:** Classified EU RESTRICTED, EU CONFIDENTIAL, **Int** = Internal Working Document, information as referred to in Commission Decision 2001/844/EC.

Document Information

List of Contributors	
Name	Partner
Symeon Papavassiliou	NTUA
Anastasios Zafeiropoulos	NTUA
Eleni Fotopoulou	NTUA
Nikolaos Filinis	NTUA
Dimitrios Spatharakis	NTUA
Ioannis Tzanettis	NTUA
Ioannis Dimolitsas	NTUA
Constantinos Vassilakis	NTUA
Giacomo Genovese	CNIT
Antonella Molinaro	CNIT
Antonio Iera	CNIT
Alessandro Carrega	CNIT
Darko Anicic	SIEMENS
Kirill Dorofeev	SIEMENS
Mahda Noura	SIEMENS
Guillermo Gomez	ATOS
Nathalie Mitton	INRIA
Adriana Arteaga Arce	INRIA
Hazem Chaabi	INRIA
Lefteris Mamatas	UOM
Georgios Papathanail	UOM
Angelos Pentelas	UOM
Panagiotis Papadimitriou	UOM
Ilias Sakellariou	UOM
Rafael Marin Perez	ODINS
Elena Maria Torroglosa	ODINS
Rudolf Sušnik	ININ
Janez Sterle	ININ
Marco Jahn	ECL
Konstantinos Almpanakis	WINGS
Konstantinos Lessis	WINGS
Sofiane Zemouri	IBM
Luca Russano	ESAOTE
Jurij Mirnik	LUKA KOPER
François Daoust	ERCIM
Giovanni Toffetti	ZHAW
Leonardo Militano	ZHAW

Document History			
Version	Date	Change editors	Changes
0.1	15/03/2023	Anastasios Zafeiropoulos, Adriana Arteaga Arce	ToC preparation and first draft
0.2	30/03/2023	Adriana Arteaga Arce, Anastasios Zafeiropoulos	Editing of Sections 1, 2
0.3	30/04/2023	Leonardo Militano, Adriana Arteaga Arce, Anastasios Zafeiropoulos	Editing of Sections 3, 4 and Appendix
0.4	30/05/2023	Leonardo Militano, Adriana Arteaga Arce, Anastasios Zafeiropoulos	Full version of the deliverable available for internal review
0.5	15/06/2023	Guillermo Gómez, Leonardo Militano, Rudolf Susnik, Panagiotis Papadimitriou	Comments by the internal review
0.6	20/06/2023	Leonardo Militano, Adriana Arteaga Arce, Anastasios Zafeiropoulos	Updated version with revisions
1.0	28/06/2023	Symeon Papavassiliou	Final version to be submitted

Quality Control		
Role	Who (Partner short name)	Approval Date
Deliverable leader	Nathalie Mitton (INRIA)	15/06/2023
Internal reviewers	Guillermo Gómez (ATOS), Leonardo Militano (ZHAW), Rudolf Susnik (ININ), Panagiotis Papadimitriou (UOM)	20/06/2023
Project Coordinator	Symeon Papavassiliou (NTUA)	28/06/2023

Table of Contents

List of Tables

List of Figures

List of Acronyms

Abbreviation /Acronym	Description
2D	Two dimensional
3D	Three dimensional
5G	Fifth Generation
AI	Artificial Intelligence
API	Application Programming Interface
ARE	Ambulance in a Rural Environment
ASP	Application Service Provider
ASV	Application Service Vendor
BAS	Building Automation System
CCNM	Computing Continuum Network Manager
CD	Continuous Delivery
CFS	Container Freight Stations
CI	Continuous Integration
CICD	Continuous Integration and Continuous Delivery
CPU	Central Processing Unit
cVO	Composite Virtual Object
DID	Distributed Identifier
DL	Deep Learning
DLT	Distributed Ledger
DPR	Data Processing Requirements
DT	Digital Twin
E2E	End-to-End
EHR	Electronic Health Record
ERP	Enterprise Resource Planning
FR	Functional Requirement
FRM	Federated Resource Manager
GB	Gigabyte
GNSS	Global Navigation Satellite System
GPU	Graphics Processing Unit
GUI	Graphical User Interface
GW	Gateway
HDA	Hyper Distributed Application

Abbreviation /Acronym	Description
HDAR	Hyper Distributed Application Repository
HVAC	Heating, Ventilation and Air Conditioning
HW	Hardware
IaaS	Infrastructure as a Service
IoT	Internet of Things
KPI	Key Performance Indicator
LP	Local Processing
MB	Megabyte
mHWDev	Minimal HW Device
ML	Machine Learning
MQTT	MQ Telemetry Transport
NB-IoT	Narrowband Internet of Things
NFR	Non-Functional Requirement
OBU	On Board Unit
OS	Operating System
PIS	Port Information Systems
PS	Primary Screen
QoS	Quality of Service
ROS	Robot Operating System
SLA	Service Level Agreement
SLAM	Simultaneous Location and Mapping
SMO	Synergetic Meta-Orchestrator
SR	System Requirement
SW	Software
TD	Touchscreen Display
TDD	Test Driven Development
TRL	Technology Readiness Level
TSN	Time Sensitive Networking
UC	Use Case
UHD	Ultra High Definition
VO	Virtual Object
XACML	eXtensible Access Control Markup Language
ZSM	Zero Touch Network and Service Management

Executive Summary

NEPHELE is a Research and Innovation Action with a duration of 36 months involving 17 partners from 9 countries and several sectors. The project aims to “enable the efficient, reliable and secure end-to-end orchestration of hyper-distributed applications over programmable infrastructure that is spanning across the compute continuum from Cloud-to-Edge-to-IoT, removing existing openness and interoperability barriers in the convergence of IoT technologies against cloud and edge computing orchestration platforms, and introducing automation and decentralised intelligence mechanisms powered by 5G and distributed AI technologies”.

This document presents the set of requirements that are identified for the development of the main artifacts of the NEPHELE project, as well as the initial design of the NEPHELE conceptual architecture. Initially, the NEPHELE ecosystem is detailed focusing on the identification of the main stakeholders and their role. Following, a set of definitions and specifications are provided for both the software stack that is under development in the project, called as VOStack, as well as the synergetic orchestration mechanisms. In both cases, functional and non-functional requirements that have to be fulfilled are detailed considering their importance and difficulty level.

The set of requirements is enriched based on the initial specification of the scenarios and challenges to be considered in the four use cases of the NEPHELE project that refer to emergency/disaster recovery, AI-assisted logistic operations in a port, energy management in smart buildings/cities and remote healthcare services. Per use case, initial specification of the application graphs to be considered for deployment over programmable resources in the computing continuum is presented. Each application graph includes the defined Virtual Objects (VOs) and Composite Virtual Objects (cVOs), highlighting the support of interoperable Internet of Things (IoT) solutions as well as the convergence of IoT technologies with edge/cloud computing technologies.

This document will be the basis for the specification of the revised version of the NEPHELE reference architecture in D2.2 that is due in M18, while the identified requirements will feed the development of the VOStack in WP3 and the synergetic orchestration mechanisms in WP4. The initial specifications of use cases will be used as a basis for the definition of the use cases framework and their development in WP6.

1. Introduction

NEPHELE is a Research and Innovation Action (RIA) project funded by the Horizon Europe programme under the topic "Future European platforms for the Edge: Meta Operating Systems". NEPHELE vision is to enable the efficient, reliable and secure end-to-end orchestration of hyper-distributed applications over a programmable infrastructure that is spanning across the compute continuum from IoT-to-edge-to-cloud.

This deliverable reports on the activities of Work Package 2 (WP2). WP2 is devoted to collecting the requirements for intelligent IoT device management and coordination and synergetic orchestration of cloud and edge computing applications to provide the breakthrough reference architecture of NEPHELE. To reach this objective, this WP is divided into four Tasks: Task 2.1 is related to identifying, documenting, and prioritizing a set of requirements associated with the management of intelligent IoT devices, the Virtual Objects (VOs) specification, and the VOStack (software stack of IoT virtualization) layers. Task 2.2 is focusing on identifying, documenting, and prioritizing requirements related to the synergetic orchestration of hyper-distributed applications over programmable infrastructure in the compute continuum. Task 2.3 specifies the set of use cases that will be supported within NEPHELE, including requirements for the efficient provision of hyper-distributed applications in the compute continuum per use case. Finally, Task 2.4 specifies and details the NEPHELE reference architecture based on the requirements identified in Tasks 2.1 – 2.3. These Tasks involve the participation of all the partners of the consortium with exception of FundingBox.

In this deliverable, we first describe the NEPHELE vision and the stakeholders involved (Section 2). We continue with the Virtual Object description, the multiple-layer Virtual Object Stack specification, and the associated requirements (Section 3). Subsequently, we present the synergetic orchestration and the requirements for managing deployments of hyper-distributed applications across the compute continuum from IoT-to-edge-to-cloud (Section 4). The proposed architectural approach in NEPHELE will be validated, evaluated, and demonstrated through four use cases (Section 5). The use cases domains are disaster/emergency management, AI-assisted logistic operations in a port environment, energy management in smart buildings, and remote health care services. We identify the applications' components and services to manage the use cases' various challenges and objectives. Finally, we present an overview of the NEPHELE reference architecture based on the requirements identified in Tasks 2.1 – 2.3. A detailed description of the NEPHELE reference architecture is presented in Deliverable 2.2. At the end of the document, there is an Appendix with the description of the four use cases, where the details of the HDAs are presented, as well as the analysis of the system requirements and data processing.

The work achieved in this deliverable will serve as a base reference document to the other WPs and deliverables of the project since it gathers main information useful for the project.

2. NEPHELE Vision

The next generation Internet of Things (IoT) and Edge Computing technologies are evolving at a rapid pace (in 2023, the global number of connected IoT devices is expected to grow by 16% to 16.7 billion of active endpoints according to IoT Analytics [1]) and are transforming businesses and peoples' lives, producing solutions targeted at many industrial sectors, and forming the foundation of a totally interconnected world . This evolution moves in parallel with the increase in the heterogeneity of the IoT technologies [3][2] in terms of the production of different types of intelligent IoT devices, the support of diverse communication protocols, and the conceptualization of various information models for semantically representing entities in the IoT world. These trends make inherent the need for novel architectural approaches, able to support by design a full convergence and integration among existing and evolving IoT and edge computing technologies. In parallel, data processing and analytics, even if today taking place in centralised computing facilities (i.e., cloud data centers) at 80%, are foreseen to rapidly shift towards edge computing facilities (by 75% in the upcoming five years)[4]. To efficiently manage data management and analysis over such distributed environment, novel hyper-distributed applications (HDAs) are even more embracing microservices-based and cloud-native computing technologies, while distributed computing principles are deeply evolving their lifecycle orchestration paradigms to efficiently exploit resources in the continuum from Cloud-to-Edge-to-IoT.

Two main challenges arise in this transition. The first challenge regards the need for convergence of IoT technologies based on novel architectural approaches, able to guarantee continuous and seamless openness and interoperability of the plethora of existing and emerging solutions, models and devices, while enabling analytics for their lifecycle's costs, measured in time and resources (from seconds or watts to CO_2). The second challenge regards the need for the provision of an integrated meta-orchestration environment for HDAs, where a synergy between cloud and edge computing orchestration platforms takes place to optimally manage applications' end-to-end deployment and data provision over the continuum. For tackling these challenges, the NEPHELE project aims to introduce two core innovations, namely:

- An IoT and edge computing software stack for leveraging virtualization of IoT devices at the edge part of the infrastructure and supporting openness and interoperability aspects in a device-independent way. Through this software stack, management of a wide range of IoT devices and platforms can be realised in a unified way, avoiding the usage of middleware platforms, while edge computing functionalities can be offered on demand to efficiently support IoT applications' operations.
- A synergistic meta-orchestration framework for managing the coordination between cloud and edge computing orchestration platforms, through high-level scheduling supervision and definition, based on the adoption of a "system of systems" approach.

With both innovations, NEPHELE aims to provide an integrated environment for the next-generation HDAs management, where IoT and edge computing platforms and orchestration mechanisms will interoperate securely and be trusted. Besides, NEPHELE aims to release the produced artifacts as open source, targeting their wide adoption from the research community and the industry. Both the software stack and the synergetic orchestration framework are going to be disseminated to existing open-source initiatives, while an open-source community is going to be built.

2.1. Stakeholders

The different stakeholders involved in the NEPHELE ecosystem identified here are based on the proposed reference architecture presented in this document. This section provides an overview of those actors that participate in the NEPHELE ecosystem from a technical and business perspective as a way to help depict the storyline of how the developments of the project will be used and their individual objectives and challenges. This information contributes to the collection of requirements and sets the grounds for the business analysis of the NEPHELE ecosystem which will be covered in successive WPs. It should be noted that, depending on the deployment scenario, one entity may undertake more than one of the following roles.

1. NEPHELE Operator: Is the owner of the NEPHELE Platform from the perspective of the management of the software components (orchestration) that are horizontal to all application service providers (ASPs) as well as the underlying infrastructure required to keep them running, either entirely or via a hyper-scaler in the cloud.
    a. Objectives:
        i. Ease of scalability of the underlying infrastructure to extend the coverage of the ecosystem while maintaining security.
        ii. High-availability and resilience of the synergetic orchestration platform.
        iii. Lifecycle management of the Virtual Objects (optional) and the HDA components.
        iv. Ease of usability and Use Case (UC)-oriented functionalities to promote the use of the platform.
        v. Efficient resource utilization of the compute and network resources for a cost-effective and efficient operation of the platform.
    b. Challenges:
        i. Compatibility and interoperability of resources across the continuum.
        ii. Achieve promised security and monitoring commitments in a complex environment built as a system of systems.
        iii. Convergence of IoT, edge and cloud orchestration mechanisms.
2. Infrastructure Provider: Is the owner of the infrastructure used to host the HDA of the ASPs in the compute continuum (IoT, edge, far-edge, cloud resources). There are two complementary stakeholders in this category, the compute and the network infrastructure providers. They are the ones that provide the virtualized infrastructure and custom orchestrators for their environments which are then connected to a Compute or Network Manager as a gateway to the Nephele Platform. The infrastructure may span from IoT to (far-)edge to cloud resources across the continuum.
    a. Objectives:
        i. Offer an infrastructure compatible with the HDA objectives and the Managers' requirements.
        ii. Host as many HDAs as possible, providing proper interfaces for management of resources.
    b. Challenges:
        i. Interoperability between compute and network infrastructure providers.
        ii. Interoperability between IoT HW, VOs and the HDA SW deployed within their infrastructure.
        iii. End-to-end (E2E) isolation and multi-tenancy to support multiple applications at the same time.
3. Application Service Providers: These are the HDA developers and providers who offer an E2E application to service consumers. They provide the IoT with the pre-configuration required as well as the HDA components. They should provide the Application Graph to the NEPHELE Operator as well as any additional information required to correctly orchestrate and automate the lifecycle of the different parts of the HDA. This stakeholder might be further composed of two distinct actors, the

AP Vendor (APV), which develops part or the totality of an HDA, and the ASP which uses assets from the APV and of their own to create a consumer-tailored service offering.

a. Objectives:
    i. Leverage the compute continuum to the full extent to improve the service consumer experience.
    ii. Assert that the Service Level Agreements (SLAs) are maintained to the service consumer.
    iii. Publish/Subscribe services in the NEPHELE ecosystem to extend their applications.
b. Challenges:
    i. Customisation of the HDA for each scenario.
    ii. Complexity to expand and rollout new versions of deployed HDA in the compute continuum.
    iii. Expose application-level data required for reporting and management by the NEPHELE platform.

4. Service Consumer: They are the final users (consumers) of the HDAs.
    a. Objectives:
        i. Connect to services running as HDAs from a variety of locations and devices.
        ii. Experience seamless and uninterrupted service continuity even in mobility and high-service-traffic scenarios.
    b. Challenges:
        i. Have information about E2E SLAs in an HDA.
        ii. Ensure that sensitive data is kept secured throughout the continuum.

2.2. Definition of Requirements

Requirements are studied from different perspective in the NEPHELE Ecosystem, mainly in the definition of the IoT and edge computing software stack (see Section 3.5, the Synergetic Orchestrator framework (see Section 4.5), and the use cases where the NEPHELE outcomes will be validated (see Section 5.5); Nevertheless, requirements will be categorized throughout the document into two main groups.

- Functional Requirements (FR): Defines features of the system that can be mapped to technical functionalities provided by one or more components which need to be evaluated against an expected threshold. Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that defines what a system is supposed to accomplish.
- Non-Functional Requirements (NFR): Defines features of the system that can be mapped to business functionalities which target those objectives and challenges from stakeholders that cannot be expressed as a functional requirement. Non-Functional requirements specify criteria that can be used to judge its overall operation capabilities and constrains in a pass/no pass type.

Since the NEPHELE outcomes are going to be demonstrated, validated and evaluated in a set of use cases across various vertical industries, each use case defines its own specific requirements depending on the applications. See the Appendix for details of the use case requirements.

Key Performance Indicators analysis

As software complexity grows, monitoring becomes more important and is nowadays an intrinsic part of the development process itself. Furthermore, projects such as NEPHELE face two complementary challenges in the context of monitoring, first it needs to monitor the Platform itself to gain valuable insights into how the system is performing and, to take action to improve its performance and deliver better outcomes. Then, it also needs to monitor the underlaying heterogeneous and distributed compute and network continuum as well as the VSP applications that are orchestrated, to meet the service level objectives and the overall availability and resilience of the continuum itself. Out of the complete set of monitoring metrics that could be extracted, KPIs are selected as the best representation of the goals of the system and the applications.

Evaluation criteria

The selection of the system goals and therefore the monitoring metrics should correctly represent the requirement that it is associated with, and in a way that is Specific, Measurable, Achievable, Relevant and Time-Bound (SMART)[1], as is shown in Figure 1. SMART goals help to clarify what success exactly means which in turn results in a clear and easy to understand monitoring system.

Figure 1. SMART Goals[2]

[1] Smart goals, https://www.atlassian.com/blog/productivity/how-to-write-smart-goals
[2] Smart metrics, https://www.profit.co/blog/kpis-library/what-are-smart-metrics-why-are-they-important/

3. Virtual Object and Virtual Object Stack (VOStack)

The evolution of the Internet of Thing (IoT) and Edge Computing technologies is leading to the heterogeneous development of IoT devices, the support of diverse communication protocols, and the conceptualization of various information models for semantically representing entities in the IoT world. The development of an IoT and edge computing software stack in the NEPHELE project is motivated by the need of supporting a full convergence and integration among existing and evolving IoT and edge computing technologies. One of the main challenges is to develop virtual counterparts of IoT devices, called Virtual Object (VO), to provide a set of abstractions for managing any type of IoT device through a virtualized instance, while augmenting the supported functionalities through the hosting of a multi-layer software stack. The Virtual Object Stack (VOStack) shall provide VOs with edge computing and IoT functions, like – among others – distributed data management and analysis based on machine learning (ML) and digital twinning techniques, authorization, security, and trust based on security protocols, autonomic networking and time-triggered IoT functions, taking advantage of ad-hoc groups management techniques, service discovery and load balancing mechanisms. At the same time, edge computing functionalities can be offered on demand to support IoT applications' operations efficiently.

This section presents an overview of the VO, including definition of the interfaces between the physical device, the VO, the application components, and the orchestration mechanisms. We also describe the multi-layer VOStack and the general requirements to be fulfilled along the VOStack.

3.1. Virtual Objects Definitions and Motivation

The Virtual Object (VO) as a digital counterpart of a physical IoT device has experienced an evolution of its functionality over the years. Since its introduction, in most of its deployments, the VO concept has been commonly intended to promote the interoperability of heterogeneous devices, facilitate the deployment of new services, improve reachability, and achieve self-management of devices [6][5]. Several other features further enhanced the VO as part of a variety of fit-for-purpose introduced management frameworks: common semantic representation of the device's data and functionalities for enhanced interoperability, device augmentation with compute and storage capabilities, device augmentation with context awareness and cognitive management, device offloading and energy consumption optimization, are just a few examples [6].

As a further step, a more effective collaboration of several physical devices is enabled in the virtual world by the introduction of the Composite Virtual Object (cVO) as an aggregation of trusted VOs illustrating a new set of functions out of the interaction of several member devices through their virtual counterparts. Deployments presented so far do not always consider a single corresponding VO to a physical device but have also introduced solutions, adapted to their reference scenario, where a single VO may correspond to multiple physical devices, each of them performing different functions/services, or multiple VOs correspond to a single physical device [7]. Furthermore, the combination of several VOs and cVOs along with other services, results into a new higher level of IoT services and applications, while their orchestration and execution in the cloud and/or edge have triggered the introduction of several methods and frameworks often targeted to a specific application area **Error! Reference source not found.**[6].

The concept of Digital twin (DT) also bases its definition on the mapping of a physical object onto a virtual space and builds upon it to illustrate a synchronous bidirectional data exchange to monitor, simulate, predict, diagnose, and control the state and behaviour of the physical object within the virtual space [9]. One could think of a VO as a DT with a set of advanced features and tight synchronicity and state matching with the physical object. It is believed that an open VO design establishing it as a potential building block of a DT or even future cyber-physical systems would promote significant advances in the area [6].

Emergent applications developed in a cloud-native/microservices-based fashion, as service chains of scalable components-microservices, leverage a hyper-distributed execution of their interconnected components over a computing continuum of orchestrated resources in different network domains (IoT, edge, cloud) [10]. In light of these new requirements and potential, the VO design should be revised to set the VO as a facilitator for: (i) a unified devices management, overcoming interoperability issues; (ii) the development of computing continuum native IoT applications where convergence aspects with edge and cloud computing technologies are tackled; and (iii) the development of new cyber-physical paradigms and new IoT-driven business models [6][6].

To facilitate the development and efficient orchestration of new emergent applications, we argue that the VO/cVO should be part of the application graph having itself a cloud-native nature. We consider a single correspondent VO per physical device while we expand the definition of cVO not only to be able to illustrate the collaboration of several VOs but also to be able to augment and customize a single VO. This approach provides the required flexibility to efficiently illustrate new IoT-driven business models and address diverse application areas. We also introduce a multi-layered lightweight software stack (named VOStack) for leveraging the virtualization of IoT devices at the edge part of the infrastructure and supporting openness and interoperability aspects in a device-independent way [6].

3.2. Main Definitions

Prior to delving into details for the VO specification, we provide a set of definitions that we consider helpful for a better understanding of the considered terms. These terms regard the IoT Device, the IoT Gateway, the IoT Device Cluster, the Digital Twin, and the Client of a VO [6].

With the term **IoT Device** we refer to a device with one or more sensors, computational capabilities (optional) and a communication interface. The IoT Device can act as a sensor (produces data based on the observations by sensors) and/or an actuator (applies/enforces actions).

In various cases, to achieve network connectivity and enable management capabilities for IoT devices, the use of a smart IoT gateway is necessary. With the term **IoT Gateway**, we refer to a device that is used as an aggregation point in the IoT domain, connecting various on-the-ground devices with an edge Point of Presence (PoP). The IoT Gateway may support different communication protocols for interacting with the IoT devices. To represent cases where a set of identical IoT devices create clusters and operate in a given area (e.g., a set of humidity sensors that monitors humidity in a smart agriculture scenario), we use the term **Cluster Manager** which is a software component whose objective is to allow many similar devices to be managed as a single IoT Device, this meaning that the state and actions of the Manager refer to all the devices in the cluster at the same time.

A **Digital Twin (DT)** is a virtual representation of a real-world physical system or product (a physical twin) that serves as the indistinguishable digital counterpart of it for practical purposes, such as system simulation, integration, testing, monitoring, and maintenance. A DT can but has not necessarily to be used in real-time and regularly synchronized with the corresponding physical system.

From the perspective of the edge/cloud application or service providers that aim to deploy applications based on IoT-related data and actions, the VO must be considered as a part of their application's graph. To represent this interaction, we introduce the term **VO Client**. The VO Client is considered as an application component that requests data or dictates actions to the IoT Device represented by its VO.

Given the definitions regarding IoT-related concepts, we proceed by defining the proposed **VO** and the associated software stack (**VOStack**).

3.3. Virtual Object Specification

A **Virtual Object (VO)** is considered as a virtual counterpart of a physical device on the Internet of Things domain. It provides a set of abstractions for managing any type of IoT device through a virtualized instance while augmenting the supported functionalities through the development of a multi-layer software stack, called Virtual Object Stack (VOStack) [6].

The relationship between a VO and an IoT device is one-to-one. To meet the strict requirements of the IoT-specific scenarios, the VO is intended to be deployed in small-scale data centres at the network edge, in the proximity of the corresponding IoT device. The integration of the appropriate communication protocols within the VO is essential to ensure efficient interactions with the device layer. From the client's perspective, the VO is considered part of an application graph, providing application-related functionalities (e.g., accessing the IoT data or enforcing actions on IoT devices). The VO is *orchestratable* as part of the application graph and, thus, can be managed, monitored, scaled, and migrated.

As the proposed VO is intended to operate near its hardware counterpart, the resource constraints of small-scale infrastructures at the network edge must be accounted for in view to matching the VO's requirements. On the other hand, various application scenarios may demand computing-intensive tasks to be scheduled or data from different IoT devices to be aggregated for further processing. In such cases, the custom, application-specific configuration is necessary per client and the basic VO capabilities must be enhanced.

To keep the VO isolated and as lightweight as possible while, at the same time, allowing clients to deploy custom functionalities in the IoT device proximity, or combine data from different IoT devices, which will be part of the application graph and fully configurable by the clients, we introduce the concept of Composite VO.

A **Composite Virtual Object (cVO)** is a software entity that can manage the information coming from one or multiple VOs and provide advanced functionalities. Two modes of operation are designed for a cVO. In the first mode, a cVO is connected with multiple VOs that manage IoT devices of several types. The cVO interacts with the VOs, processes the collected information, and can contextually produce advanced knowledge, by enabling the communication and collaboration of several VOs, toward the production and exposure of a combined set of data outputs.

Thus, a cVO can be used as an aggregation entity of multiple VOs of various types to provide clients' applications with a single point of access to the information of multiple IoT devices. As a result, it is part of an application graph undertaking the role of a VO Client, while the relationship between a cVO and multiple VOs is one-to-many. The cVO can guide the behaviour of the managed VOs by enforcing kind of policies for their optimal collaboration (e.g., synchronization in data acquisition processes). In the second mode, the cVO enhances the capabilities of the VO through the provision of application-oriented functionalities, once again under the role of a VO client. For instance, in case of a video camera, the cVO can support image recognition features over the streams managed by the VO. Under the scope of this mode, the cVO can operate also as a DT. In this case, the enhanced capabilities of the VO through the cVO refer to the support of simulation, integration, testing, monitoring, and maintenance activities for an IoT device.

Figure 2 provides a high-level convergence view across the computing continuum, wherein the proposed VO can communicate with IoT devices, thus enabling their interconnection with application-oriented parts (application graphs). Representative cases of IoT devices managed by VOs are illustrated. For example, complex IoT devices with multiple sensors, such as a robotic arm, could be virtually represented by a VO, while a cVO might extend the device's functionalities and capabilities regarding the application's requirements or operate as a DT for the robotic arm. Furthermore, leveraging the capabilities of the Cluster Manager, the VO could manage a cluster of identical IoT devices. The data generated from different IoT devices (such as a camera and a temperature sensor), could be aggregated and processed by a cVO, in accordance with the application's objective. As it is also shown, a VO could provide the retrieved data from the corresponding IoT device to several application-oriented components of the same or distinct service graphs.

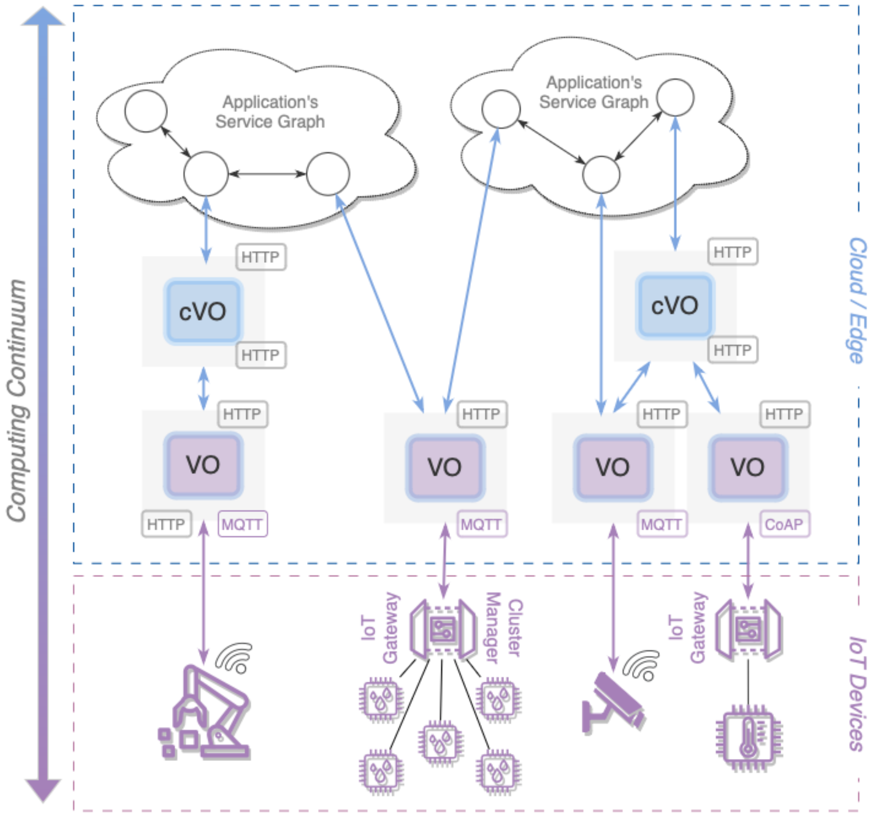


Figure 2: High Level View of VO Usage [6]

As illustrated in Figure 3, to support all the envisioned capabilities, a VO has the following main types of interactions with the computing continuum world:

VO-to-IoT-Device Interaction: the objective is to address interoperability and convergence challenges with the IoT ecosystem.

VO-to-Application Interaction: the objective is to enable the interaction between the VO and cVOs, as well as the interaction between the (c)VO and application components that provide part of the distributed application business logic.

VO-to-Orchestration Interaction: the objective is to enable the development of edge/cloud computing distributed applications, where the (c)VO is an integral part of a distributed application graph and, thus, manageable by cloud/edge computing orchestration mechanisms.

VO-to-Storage Entity Interaction: the objective is to keep track of device metadata, status and messages exchanged with other devices and applications. Moreover, the VO must be able to support basic operations regarding data management by integrating multiple data sources to produce contextual information about the devices that can be used by the clients.

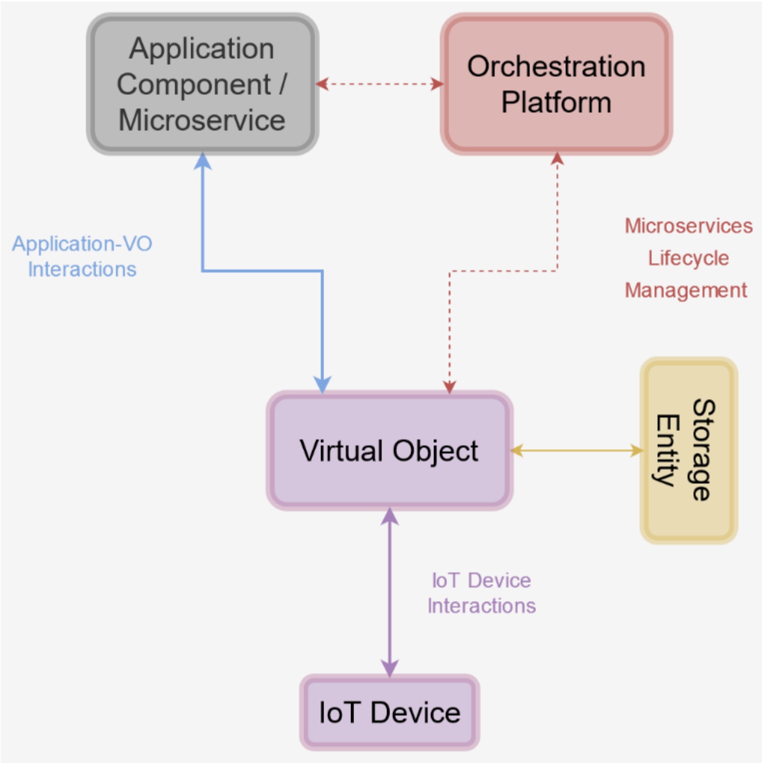


Figure 3: VO Interactions [6]

3.4. Virtual Object Stack Specification

A software stack (VOStack) is under development to flexibly support interaction with both physical IoT devices and edge/cloud computing orchestration platforms, considering both the VO and cVO concept. The VOStack is implemented in the context of stateless pluggable micro-services. The main incentive is that the VOs must be lightweight and modular while supporting basic functionalities that most devices and applications need. Hence, the VOStack has three main architectural layers namely: (i) the Physical Convergence Layer, (ii) the Edge/Cloud Convergence Layer, and (iii) the Backend Logic Layer. Following, we analyse them along with the functionality envisaged per layer. In Figure 4 we present an overview of the layered approach of the VOStack.

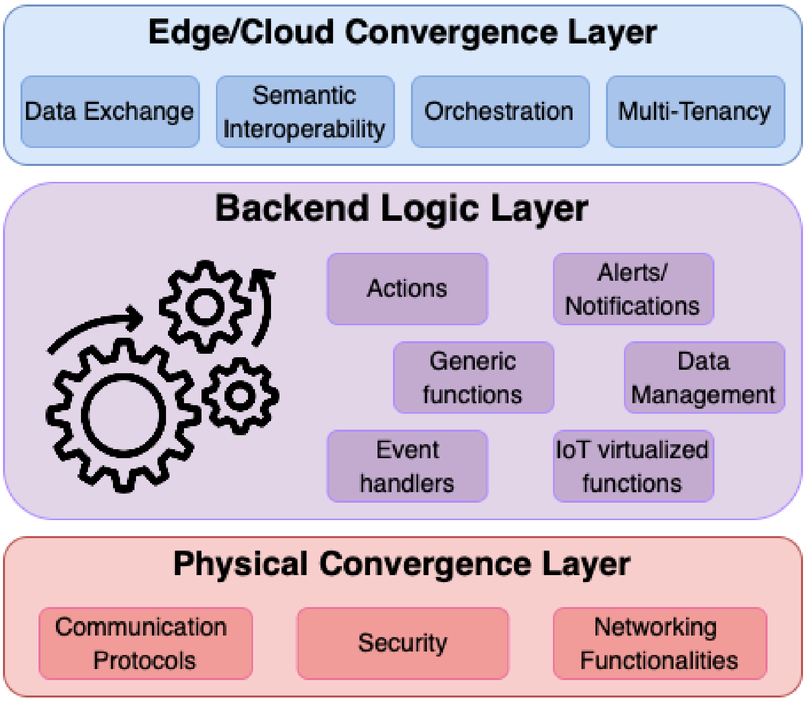


Figure 4. VOStack Layers [6]

Physical Convergence Layer

This layer is responsible to tackle the major challenges of connecting the IoT devices with the computing continuum infrastructure [6]. First and foremost, the VO can address device registration issues (e.g., registering a new device to a VO, bootstrapping of a connection, etc.). Regarding connectivity, a VO supports different types of communication protocols among those most widely used in the IoT domain at: (i) the application layer (e.g., MQTT, CoAP, HTTP, etc.), (ii) the network layer (e.g., IPv4, IPv6, etc.), and (iii) transport layer (e.g., TCP, UDP, etc.). In such a way, the majority of IoT devices can be connected and communicate with their virtual counterpart. However, as many devices have restricted security capabilities, authentication, and authorization functionalities (e.g., OAuth 2.0) are provided to solve secure communications between the devices and the applications. Moreover, this layer simplifies the coordination of multiple IoT devices or IoT clusters, by providing autonomic and self-* functionalities [11]. Furthermore, a set of network-oriented functionalities are available to facilitate intermittent connectivity of the devices, manage dynamic routing protocols, time-sensitive networking mechanisms or tackle mobility aspects. On the one hand, by keeping the VO synced with the IoT device, clients can access the device's information uninterruptedly even if the device suddenly loses connection with the VO.

Backend Logic Layer

This layer is responsible for augmenting the functionalities and capabilities of IoT devices [6]. Here, we include all the logic related to the IoT device's operational behaviours, enhanced functionalities, and services that the Object/Device can perform. Primarily, the VOs can declare alerts on the IoT devices' state (e.g., a device suddenly restarted), and/or data-driven notifications (e.g., the temperature of a sensor rapidly increased). This functionality is closely related to the interaction with the storage entity, since, for instance, it is typical in many scenarios to observe past data values. Naturally, in trying to implement the virtual counterpart of an IoT device it is mandatory to introduce a set of actions and behaviours that the VO can dictate to the IoT device. To this extent, a VO can reconfigure or try to remotely heal a device. Moreover, following an event-based logic, actions are also either triggered by the monitored data (e.g., alerts and notifications coming from a sensor) or activated by commands received from the application and/or orchestration side (e.g., an application provider may want to dictate a different behaviour of a sensor, such as changing the polling period of measurement when a given threshold is exceeded). Finally, for each defined action, a mechanism is designed to support the action-related policies implementing multi-tenancy characteristics. It is crucial that the set of actions, alerts, and notifications are reconfigurable, and their definitions are not limited or heavily depend on the respective use case.

Moreover, the modularity of IoT functions in the edge and the cloud infrastructure is considered a key challenge to enabling modern applications and cloud native IoT solutions. To this end, two main categories of functionalities are defined to support the basic operations that the interplay between IoT and Applications require, namely (i) IoT Device Virtualized Functions and (ii) Generic/Supportive Functions.

The IoT Device Virtualized Functions refer to functions that tackle part of the business logic of an application. They are responsible for the deployment and management of IoT-specific functionalities (e.g., video transcoding for a camera, image processing and analysis in case of a remote healthcare device or a face detection sensor). The objective is to alleviate IoT devices from executing computationally heavy tasks and transfer this responsibility to the VOs that are deployed in a nearby edge computing infrastructure. Virtualization of IoT functions enables their integration into edge computing applications and their dynamic management by edge and cloud computing orchestration platforms. The IoT Device Virtualized Functions are mostly envisaged to be provided in the form of cVOs, as an advanced capability of a VO.

The Generic/Supportive Functions consider a set of supportive functions that can be horizontally applied over all the instantiated VOs for an application. These functions can support IoT-oriented functionalities in a generic way (e.g., distributed data management, data aggregation, filtering, firewalling, authentication, failure handling), as well as functionalities at the edge part of the infrastructure (e.g., service discovery, telemetry). As a result, the Generic/Supportive VO functions should be part of the basic VO implementation and may be activated on demand according to the application's needs.

Edge/Cloud Convergence Layer

This layer is responsible for bringing the VO closer to the application and orchestration layer [6][6]. As the VO is part of the application graph, it communicates with entities, such as data consumers, applications, or users, through suitable interfaces. Various communication protocols (e.g., HTTP, MQTT, CoAP, etc.) are going to be supported. Hence, via this layer, IoT devices are exposed and consumed. More specifically, a set of functionalities may be provided through this layer for managing incoming requests and providing responses (e.g., requesting data, triggering actions, declaring new alerts), and handling multi-tenancy aspects (e.g., multiple requests for IoT device information). Besides, this layer supports a set of functions related to orchestration and addressing the monitoring of the status of the VO (e.g., container monitoring), the management of the deployment of the VO over the computing infrastructure (e.g., start, stop, restart, destroy), and the management of elasticity and migration actions.

VO Data Store

The adoption of a microservices-based approach for HDAs creates a need for following a coherent approach for data storage. In detail, each microservice or VO will host data in its data store, while distributed data storage functionalities may be also applied when and if requested. Data management functions can be applied in the IoT device or the VO according to the requirements of each IoT application (e.g., constraints due to data privacy). Flexible and distributed data sharing among VO data stores will be supported by exploiting block-chain techniques for assuring the trustworthiness of each data store.

Intelligence on IoT Devices and Interplay with VOs

The concept of the VO is going to substitute/augment part of the functionalities offered by physical IoT devices, based on the provision of such functionalities in virtualized environments in the edge part of the continuum. By taking advantage of virtualization, network softwarization and 5G technologies, guarantees regarding QoS characteristics (e.g., very low latency, high bandwidth) can be provided, making the transition from the physical IoT device to its virtual counterpart straightforward. However, the VO has not to be considered as a full replacement of the IoT device. In many cases, it is required or suggested to perform part of the data processing and analysis at IoT device level, especially when there are severe concerns regarding data security and privacy or very strict requirements in terms of delay (e.g., mission-critical, delay-intolerant applications). Existence of enough computational resources in the IoT devices cannot be considered as granted. Intelligence at both IoT device and VO level must be introduced, taking advantage of decentralized AI and TinyML techniques. Decentralized AI can be applied for moving intelligence and learning at both VOs and IoT devices, while TinyML can support models that run on small, low-powered devices like microcontrollers and enable low-latency, low power, and low bandwidth model inference at edge devices. The interplay between IoT and edge computing resources reservation must be continuously considered by the VOs and edge computing clusters. To support this interplay, an analogy between the functionalities provided per layer of the proposed stack by the VO and the IoT device must be specified. It should be noted that, by taking advantage of the VOs implementation and the supported VOStack, we avoid the need for any type of middleware between the IoT device and the VO.

3.5. VO and VOStack Requirements

Table 1 presents the Functional Requirements (FR) of the VOStack. The requirements are mapped to the VOStack related feature that can support them, as well as the interfaces needed to interact across the computing continuum, the difficulty and the priority. The implementation of these requirements will depend on the characteristics and needs of each use case. We describe the use cases and the consolidated requirements in 5 and the detailed analysis of the requirements for each case is in the Appendix section.

Table 1. VOStack Functional Requirements

ID	Description	VOStack related feature	Associated Interface(s)	Difficulty	Priority
FR_VOS_001	The VOStack shall provide interfaces to connect heterogeneous IoT devices directly or through an IoT gateway	Interoperability, Security and IoT Device Management	VO-to-Device	High	High

ID	Description	VOStack related feature	Associated Interface(s)	Difficulty	Priority
FR_VOS_002	The VOStack shall provide linking and collaboration mechanism between VOs across the compute continuum	Autonomicity and Ad-Hoc Networking	VO-to-VO	Medium	High
FR_VOS_003	The VOStack shall provide multi-tenant access to IoT devices	Interoperability, Security and IoT Device Management; Autonomicity and Ad-Hoc Networking; Generic Functions	VO-to-VO-to-Device; VO-to-VO	High	High
FR_VOS_004	The VOStack shall provide offloading functions between VO across the compute continuum	IoT Device Virtualization Functions	VO-to-Application; VO-to-VO-to-Device	Medium	Medium
FR_VOS_005	The VOStack shall allow the integration of resources across the compute continuum to enhances the capabilities of the IoT device	IoT Device Virtualization Functions	VO-to-Application	Medium	Medium
FR_VOS_006	The VOStack shall provide mechanisms for the VO migration across the compute continuum	Orchestration Management	VO-to-Orchestration	High	High
FR_VOS_007	The VOStack shall allow multiple instances of a VO across the compute continuum, and support linking and collaboration between them	Orchestration Management	VO-to-VO; VO-to-Orchestration	High	High
FR_VOS_008	The VOStack shall provide a proxy	Generic Functions	VO-to-Application	Medium	Medium

ID	Description	VOStack related feature	Associated Interface(s)	Difficulty	Priority
	service for IoT devices				
FR_VOS_009	The VOStack shall provide security for the connection of the IoT device and data management	Interoperability, Security and IoT Device Management; Generic Functions	VO-to-Device; VO-to-Application	High	High
FR_VOS_010	The VOStack shall define device management premises for configuration and control functions, monitoring and diagnostics, software maintenance and updates	Interoperability, Security and IoT Device Management; Autonomicity and Ad-Hoc Networking; Orchestration Management	VO-to-Device; VO-to-Orchestration	High	High

Table 2 presents the non-functional requirements supported by the multiple-layer VOStack. The implementation of these requirements will depend on the characteristics and needs of each use case. We describe the use cases and the consolidated requirements in Section 5 and the detailed analysis of the requirements for each case is in the Appendix section.

Table 2. VOStack non-Functional Requirements

ID	Description	NEPHELE related feature	Difficulty	Priority
NFR_VOS_01	The system should enable low latency and high bandwidth communications, and high computational power for rapid response on data processing	Cloud and Edge Synergetic Orchestration; Computing Continuum Network Management; Federated Resource Management	High	High
NFR_VOS_02	The system should guarantee data security and privacy in transmission and storage	Security and IoT Device Management; VO Storage Space	High	High
NFR_VOS_03	The system should support and store various IoT data sources with varying workloads	Generic/Supportive Functions; Interoperability; VO Storage Space;	High	High

ID	Description	NEPHELE related feature	Difficulty	Priority
		Computing Continuum Network Management		
NFR_VOS_04	The system must be able to represent IoT devices as extended Digital Twins offering additional features and functionalities	IoT Device Virtualized Functions; IoT Device Management	Medium	Medium
NFR_VOS_05	The system must be able to receive and process data from IoT devices and the environment	Generic/Supportive Functions; Interoperability	Medium	High
NFR_VOS_06	The system must be able to store IoT data	VO Storage Space	Medium	High
NFR_VOS_07	The system must be able to monitor devices and networks to trigger alerts when an error on a task occurs or a specific event is detected	Generic/Supportive Functions IoT Device Management	High	High
NFR_VOS_09	The system must be able to detect objects and humans and predict future values of associated risks/motion/condition	Generic/Supportive Functions AI models	Medium	Medium
NFR_VOS_10	The system should be able to monitor devices and networks to deploy additional elements when needed	Generic/Supportive Functions IoT Device Management	Medium	High

4. Synergetic Orchestration

The development of the Synergetic Orchestration Platform in the NEPHELE project is motivated by the need to address the requirements and challenges of the use cases (UCs) in a complex, heterogeneous and distributed environment such as the computing continuum. One of the main challenges is the complexity of managing and coordinating, in a distributed manner across different domains and infrastructures, the large number of resources, services, and applications that conform the proposed UCs, while ensuring efficient and effective operation. The platform raises needs for scalable and efficient orchestration mechanisms that can adapt to the user demand in real-time yet maintaining the end-to-end security, interoperability and seamless integration of the different components and services in the continuum.

This section describes the Synergetic Orchestration Framework, the development patterns used for the platform development and the requirements.

4.1. Synergetic Orchestration Challenges

In the context of computing continuum (CC), the challenges faced push toward the distribution of orchestration across distributed cloud environments, aiming to address management from a holistic perspective [12]. Each stakeholder in the computing continuum, (i.e., application/network/infrastructure providers or end users), sets their objectives for the operation of applications. However, a key question arises regarding how these objectives can be effectively measured or enforced [13]. Additionally, the computing continuum comprises a vast number of resources distributed across various computing tiers and layers of abstraction [14]. Maintaining the fulfillment of objectives while keeping the system in balance poses another open question.

To tackle these challenges, a novel orchestration paradigm must strike a balance between local autonomy and centralized control. While local autonomy brings several benefits, it alone is insufficient to ensure that objectives related to distributed applications deployments and multi-cloud infrastructures will be met. This is primarily due to the challenge of achieving system goals when agents are completely autonomous or even when they cooperate locally with each other [10].

The emergence phenomenon arises from the inherent nature of complex systems where the behavior of the entire system cannot be reliably predicted based solely on the actions of individual agents. This unpredictability stems from the nonlinear interactions between agents within the computing continuum. As a result, autonomous agents operating within the continuum may face challenges in acquiring the required resources. Additionally, their actions might inadvertently disrupt or interfere with the operations of other agents, thereby impeding the successful achievement of shared goals.

Hence, the integration of centralized control becomes necessary to effectively achieve system goals with sufficient performance. In other words, a balance of loose (weak) coupling is required [10], where the agents operate autonomously to a significant extent, but fair resource allocation and agent cooperation are ensured through minimal centralized control. This minimal centralized control enables the achievement of system goals that often require compromises from the agents. Moreover, centralized control allows for global optimization, facilitating the balancing of resource usage to reach goals while fulfilling performance requirements.

Based on the above, the importance of orchestrating the computing continuum in a manner that leverages the benefits of local autonomy while ensuring that system-level objectives are met, is highlighted. Under this setting, a hierarchical orchestration architecture could be employed, aiming at organizing the individual agents (planners) which operate throughout the continuum into orchestration layers, where each of them would be responsible for specific management and coordination functionalities. On the top of the hierarchy, centralized control operates to guarantee the overall system's objectives, by modeling complex multi-cloud environments and massive IoT application deployments and providing solutions for multi-objective optimization problems [10].

The main challenges identified in the orchestration domain include the following [16]:

- ***Application Lifecycle Management.*** Application Lifecycle Management in the compute continuum involves managing applications throughout their entire lifecycle, from provisioning and deployment to scaling, monitoring, and decommissioning across multiple cloud domains [17]. Challenges in lifecycle management include ensuring seamless application deployment across different cloud environments, managing dependencies and configurations, and providing mechanisms for automated deployment and decision-making. Application lifecycle management also involves facilitating continuous integration and deployment and ensuring proper resource scaling to meet dynamically changing demands.
- ***Efficient Resource Allocation.*** In the context of multi-cloud orchestration, the challenge of resource allocation encompasses various operations, related to typical edge/cloud orchestration problems, such as service placement, task scheduling, and virtual network embedding [16]. Placement refers to the process of determining which physical or virtual resources (such as edge cloud infrastructure nodes or containers) should host an application (or a part of it) or workload. Task scheduling involves determining when and where specific tasks or workloads should be executed to optimize resource utilization and applications' quality of service. For IoT applications, an emerging problem related to task scheduling is the computational offloading, when resource and energy constrained -end-devices, can offload intensive computational tasks to edge servers in their proximity [18]. Virtual network embedding pertains to mapping virtualized application components or services onto physical network resources. Challenges in resource allocation include efficient load balancing, considering resource constraints and performance requirements, optimizing resource utilization, and providing mechanisms for resource autoscaling, and container migration across multiple domains, to meet SLAs and performance targets [16].
- ***Interoperability.*** Efficient orchestration of distributed infrastructures and heterogeneous resources in the computing continuum requires ensuring interoperability between all the involved components, which operate on various layers (Cloud, Edge, IoT) [19]. Interoperability is the ability of different systems to understand and utilize each other's functions and is one of the most important requirements in the design of emerging computing continuum orchestration mechanisms. The challenge of interoperability arises due to the massive number of heterogeneous devices running different protocols, and, in parallel, the diverse hardware and software components, which are available to deploy applications at the edge and the cloud. To enable seamless operation, CC architectures must be able to provide support for interoperability, by exploiting inter-operable interfaces and open-source frameworks.
- ***SLA management.*** SLA guarantees refer to the assurance of meeting service-level agreements between orchestration agents and applications in the compute continuum. Challenges in this area involve negotiating and defining SLAs that align with the requirements of applications and stakeholders [20]. Orchestration agents must monitor and enforce SLAs, ensuring that applications receive the agreed-upon levels of performance, availability, and resource allocation. This includes handling SLA violations, managing performance degradation, and providing mechanisms for remediation or compensation in case of non-compliance.

- ***Observability.*** Monitoring is crucial for ensuring the health, performance, and security of applications and resources in the compute continuum [15]. It involves collecting and analyzing metrics such as resource utilization, application performance and network traffic related to specific elements, i.e. applications or resources. Distributed tracing, on the other hand, focuses on the workload applied to these elements and provides information regarding the performance of specific workflows, aiming to identify possible issues at the time they occur. Logging adds to this analysis by providing system and application logs that may indicate the root of the issue. Metrics, traces, and logs constitute a set of observability signals that provides a fully observable view of the deployed applications and their environment and offers a framework enabling precise orchestration in the continuum. When considering such a plethora of available information, new challenges arise such as the selection of appropriate observability tools and techniques [21], the establishment of efficient data collection and analysis processes, while also the exploitation of such observations and the correlation between them for detecting anomalies or performance issues, generating actionable insights for proactive management and troubleshooting.
- ***Network Orchestration.*** Network orchestration involves managing and coordinating the networking aspects within the compute continuum, including network connectivity, configuration, and security [10]. Challenges in network orchestration include establishing and maintaining reliable network connectivity across multiple cloud environments, ensuring consistent network policies and security mechanisms, and optimizing network performance [22]. Network orchestration also encompasses tasks such as load balancing, traffic optimization, and network function virtualization to achieve efficient and secure communication between applications and resources.

To face the aforementioned challenges in the context of the compute continuum in a decentralized manner, future research should think in terms of different entities with different objectives, i.e., Multi-Agent Systems (MASs). Multi-agent systems distribute the functionality of an application and its deployment among several agents, either collaborating or competing ones. A MAS can have an open or closed organizational structure, which governs relationships, rules, objectives, policies and authority and its behavior emerges through the actions and interactions of autonomous or partially autonomous individual agents, with the guidance of an orchestrator or through a choreography of the autonomous participants.

Orchestration in the computing continuum can be modeled as a hierarchical network of intelligent, autonomous agents that manage resources in a decentralized manner. According to their place in the hierarchy, agents are given specific responsibilities and objectives for managing specific resources i.e., infrastructure, network, or application elements. Agents can also play the role of the multiple stakeholders taking part in the continuum's operation. Each stakeholder, be that infrastructure or network service provider, or an application, is represented by a stakeholder agent on the highest level of the hierarchy.

These agents have service level objectives (SLOs) to fulfill, related to cost, quality, and resource usage, set by the application developer or domain manager. The stakeholder agents break down and pass on their objectives to the agents below them in their administrative domain, organized as clusters and further sub-clusters. Using this classification, the following synergies arise:

- **Infrastructure-Application**: Application providers may make requests for allocating resources to the infrastructure provider.
- **Infrastructure-Network**: Infrastructure providers may make requests for QoS assurance.

- **Application-Application**: Application providers may coordinate their allocation requests to maximize social welfare.
- **Infrastructure-Infrastructure**: Infrastructure providers may form synergies through negotiation to collectively support application deployments.

Handling such synergies and developing mechanisms that can facilitate their coordination is the big challenge that synergetic orchestration architectures will have to face very soon.

In complex systems, such as the spectrum of the computing continuum, a MAS needs to autonomically reorganize itself to adapt and evolve, in response to changes in the participating agents or in the external environment. Artificial Intelligence and Machine Learning have shown significant results in building autonomy based on collected real-time and historical information from their environment guiding decision making. AI agents can learn to be reactive, proactive, and collaborative based on what they observe in previous experiences and thus, have the potential to demonstrate significant performance in heterogeneous and continuously changing environments. Thus, new challenges arise when considering the application of autonomic intelligence in distributed environments in the compute continuum [9]:

- *Flexible representation*. To orchestrate the resources in the computing continuum, the management entity needs a flexible and adaptive representation of those resources able to reflect any changes in the system architecture.
- *Infrastructure*. Applications running in the computing continuum are highly dependent on the underlying resources, which may be heterogeneous and dynamic, comprising for example a wide variety of different IoT devices as well as edge, fog, and cloud configurations.
- *Temporal evolution*. The computing continuum is constantly changing and evolving. Any management model must consider this change and allow for concept and data drift. Moreover, the rate and direction of this change (say, in terms of the orchestration objectives related to costs, quality and resource usage) may also be considered to allow for appropriate reporting and consequent action.
- *Causality relations*. The computing continuum comprises an ecosystem of multiple interacting resources and stakeholders. A global perspective thus must consider the whole of this ecosystem, not restricting to individual resources or their activities. To understand how actions propagate across this ecosystem, the computing continuum orchestrator needs to keep track of causal relationships between the resources.
- *Proactive adaptation*. The complex causal relations between the resources may lead to a cascade of failures as issues can propagate across the computing continuum. Maintaining stakeholder objectives thus requires prompt and proactive action to prevent such failure propagation.
- *Learning framework*. The ecosystem complexity and scale make it impossible to draw a complete management plan in the design phase. Therefore, setting management methodologies inside a learning framework is required to provide incrementally better solutions and adaptations.
- *Degree of centralization*. The identification of the optimal balance between independent local decision-making and centralized control in a large system of systems is crucial for modern application deployment in the computing continuum. The balance can vary over time, for example, when abrupt changes are experienced in the operation environment, making its identification even more challenging.
- *Emergence*. In a distributed, loosely coupled system, patterns of activity between the agents may emerge. How can harmful emergence be avoided in such systems? How can emergence be used in achieving the system-level goals?

- *Convergence*. The identification of the conditions under which agents converge to a behavior and/or to a shared language when learning from data can guarantee their efficient and valuable operation in the computing continuum.
- *Scarcity*. Dealing with the fact that knowledge will be limited per agent in terms of incomplete information, limited compute power, memory for training, inference, reasoning and planning can be troublesome and the design of intelligent solutions to handle this can provide many benefits.
- *Security:* The complexity and scale of the computing continuum require guaranteeing physical security, network security, and application security, as well as authorization, availability, confidentiality, integrity, and trust.

4.2. Synergetic Orchestration Mechanisms

NEPHELE aims to provide an integrated ecosystem for the next-generation HDAs management, where IoT and edge computing platforms and orchestration mechanisms will interoperate in a secure and trusted way. A Synergetic Meta-Orchestrator (SMO) is envisaged to undertake the role of efficiently coordinating the management and orchestration of distributed compute and network resources via the Federated Resources Manager (FRM), the Computing Continuum Network Manager (CCNM), and the enforcement of AI-assisted orchestration mechanisms in the various parts of the compute continuum. 5G and beyond technologies will be exploited for serving end-to-end strict Quality of Service (QoS) needs of Hyper Distributed Applications (HDA), while IoT resources will be considered as a part of the available end-to-end infrastructure. Intelligence is going to be continuously injected within the orchestration actions, exploiting advances provided by AI technologies in features detection and inference and leading to the optimal management of the interplay among edge and cloud resources. The overall synergetic orchestration solution, NEPHELE Platform, is going to be open and extensible since it is going to be based on evolving open-source orchestrators with wide support communities.

A “system of systems” management approach will be adopted by the SMO for coordinating and assigning responsibilities to cloud and edge computing and networking orchestration Managers. A “system of systems” is a collection of independent systems, integrated into a larger system that delivers unique capabilities. The independent constituent systems collaborate to produce global behaviour that they cannot produce alone. In the case of NEPHELE, with the term system we refer to open-source modular orchestration platforms that will be adopted and appropriately extended to be interoperable with the SMO to form the global NEPHELE Platform.

This layer of abstraction allows for the creation of a harmonised ecosystem based on the standardisation of the information and characteristics about the lower-level systems and what is running throughout them. By using a common information model between the SMO, the Managers and the VOs, the Platform becomes open and extensible from the perspective of the NEPHELE ecosystem, third parties and even the HDA providers themselves since they can leverage the information and tools provided to create innovative solutions in the continuum.

4.3. Orchestration Automation and Coordination Patterns

Business opportunities enabled by the continuum come with unprecedent operation agility and automation requirements [23] as the continuum introduces new challenges, such as heterogeneity, mobility, and dynamicity of the underlaying infrastructure.

The proposed automation solution is inspired on the Zero Touch Network and Service Management (ZSM) framework defined by the ETSI [24] which is based on a robust Service-Based Architecture (SBA) built on top of a Cloud Native microservice architecture. Systems built in this way can implement Closed-Loops (CLs) by offering and consuming management services (monitoring, analytics, planning, execution and knowledge) that can be easily extended and adapted [25], as is shown in Figure 5. By combining several CLs stages, it is possible to create automated processed, also known as CL Automations (CLAs) which need to be complemented with a governance and coordination layer to ensure proper operation of the CLA in complex scenarios. The resulting system represents the proposed structured in [23] for a distributed CLA that constantly monitors and assesses the managed element and takes corrective actions when the goals are not fulfilled. Figure 5 also shows an additional key component in the ZSM framework, the Integration Fabric, provided both at the intra and inter (cross) levels to further allow coordination of CLAs in complex scenarios with one or more distributed infrastructure combining edge and cloud resources. Intent-based interfaces are proposed for the coordination and governance services within the NEPHELE Platform. The intent-based interfaces exchange high-level information, typically translated into multiple, low-level operations by an AI-driven engine. Due to the use of such interfaces, the information exchange between the management system components is minimized [26].

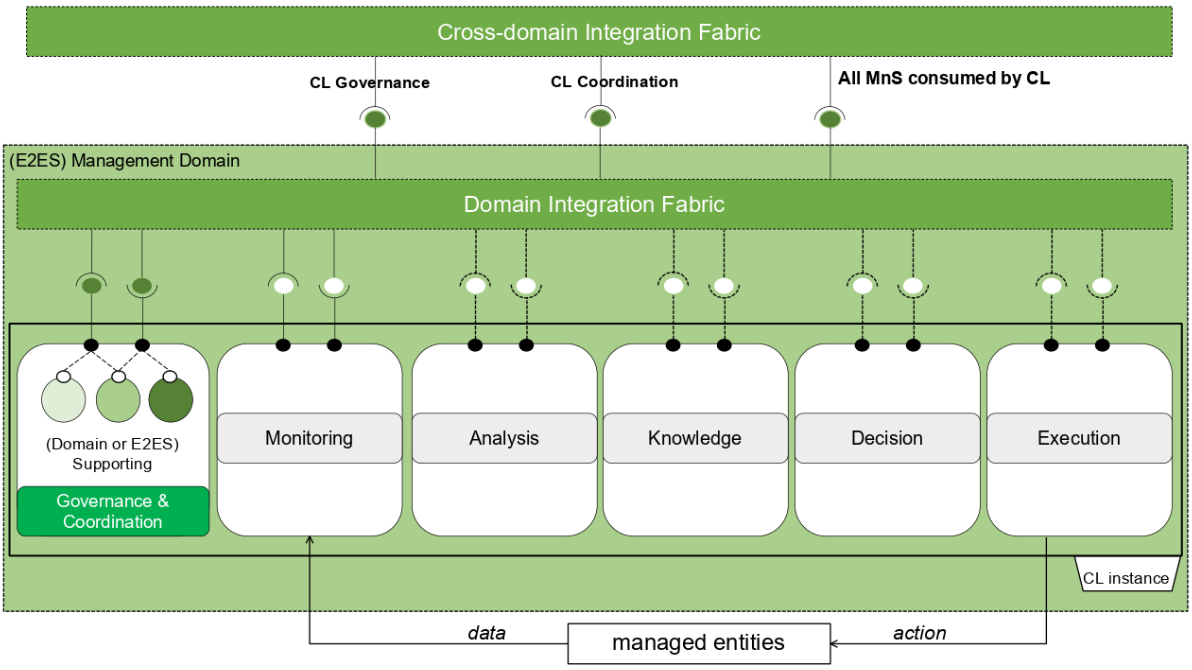


Figure 5. Closed-Loop Automation architecture in ZSM

4.4. Development Patterns

The delivery of the NEPHELE Platform will be based on a modern software methodology supported by a common Continuous Integration and Continuous Delivery (CICD) and DevOps reference implementation. The NEPHELE project is also committed to the Open-Source principles; transparency, collaboration, early and frequent release, inclusive meritocracy, and community[3]. This will ensure that the developed solution has a continuity in the industry and receives valuable feedback from all involved stakeholders at all stages of the project life cycle.

The contributions from partners towards the NEPHELE Platform will follow an automated validation framework in the form of CICD pipelines built in GitLab under two main environments. Firstly, in an isolated CICD development and testing environment fully managed by the GitLab runners, and secondly in a common integration environment where the latest version of each component will be available to run pre-production E2E tests. New features and improvements originated from the interaction between WPs in the project will be identified and requested to the corresponding partner to be included in their component in the form of a feature request or bug report GitLab issue[4]. Thus, enabling an early iteration of NEPHELE Platform components release to facilitate the proper operation of UCs.

Continuous Integration (CI) is the practice of correctly combining the code as it is being developed by a team into a shared repository by building/testing each change automatically. This must happen as early as possible and usually several times a day. Its main purpose is to detect errors as quickly as possible and to reduce integration debugging time by addressing smaller problems found at early stages. This description can be translated into three very simple bullet points that should make it easy to understand the benefits from the CI methodology[5]:

- Fail fast. Code conflicts and integration problems are discovered soon and if not, the CI pipeline should be modified. It is better to fix small problems often than to fix large problems seldom.
- Always releasable. Even after an unfruitful development period there should be at least something that is releasable.
- Simple. All team members will be using this scheme every day, so the rules and routines must be clear and simple.

Continuous Delivery (CD) goes one step further to enable a way to generate a verified version of the code and update the production environment with the latest features. According to [27], a CD pipeline takes the artefacts built by the CI pipeline and makes them available with a known tag and version to be later deployed. This pipeline prepares the SW release and significantly reduces the manual tasks to just a few clicks. The literature often refers to Continuous Deployment as an interchangeable term but, in reality, this last technique shall only be used when the human intervention is fully removed from the deployment life cycle. Due to the characteristics of the NEPHELE project, CD will be used to refer to the Continuous Delivery process.

Test Driven Development (TDD) is a software development pattern that nicely complements the CICD principles. Its endorsement makes the programmer focus on the requirements before writing the code[6]. This way, the developer codes a certain feature against a test which, when finished coding the feature, will prove that the functionality is achieved. TDD helps to build code that is correct and yields a reliable progress of the code as well as maintenance tasks such as refactoring without fear of regression. Writing good tests that are not too slow, give decent coverage and feedback and have very little dependency is not trivial and does not have a one-fits-all solution. It is also important to find the trade-off between the degree of encapsulation and abstraction that makes a block of code more testable

[3] Open source, http://opensource.com/open-source-way
[4] Use issues to collaborate on ideas, solve problems, and plan work. Share and discuss proposals with your team and with outside collaborators, https://docs.gitlab.com/ee/user/project/issues/
[5] GitLab Documentation. https://about.GitLab.com
[6] Test Driven global Lifecycle, https://commons.wikimedia.org/wiki/File:TDD_Global_Lifecycle.png

and the test-induced design hurdles [28]. TDD is sometimes referred to as the “Red, Green, Refactor” development cycle for the steps involved (see Figure 6).

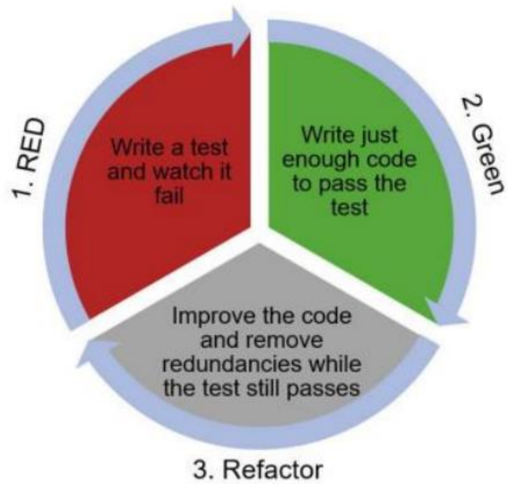


Figure 6. Test Driven Development pattern cycle

Based on the above considerations, it is expected that the development of the software components of the NEPHELE synergetic orchestration platform comply with the types of tests specified in Table 3.

Table 3. Test Suite description

Test Type	Description
Unit and Functional Tests	Unit Tests are those that check small portions of code in an isolated way. Their scope must not be very broad, and no end-to-end business logic is expected at this point. A good design of the SW should enable unit tests with almost no mocks because it does not have external dependencies. Their scope is to assert that, given some static input, the expected output is received. "Every line of code that we put in a test is like a blob of glue, holding the system in a particular shape. The more low-level tests we have, the harder it will be to change things."[7] Functional tests are those that check bigger pieces of code, potentially an entire functionality. Each functionality should be tested independently and mocking any external dependency to ensure that everything else other than the functionality under test, is using static, well-known data. Just like unit tests, it should assert that the expected output is achieved. They make use of limited operational versions of the code's dependencies (dry-run versions) which ensure that the interaction between components is correct but will not carry-out the underlaying operation.
Conformance Tests	These tests aim to validate that the release of the SW is compatible with the integration strategy of the corresponding technology. As the NEPHELE Platform leverages the Cloud-Native ecosystem, a set of validations will be performed using existing tools to check the conformance of the Helm Charts.

[7] Version control concept, https://www.atlassian.com/git/tutorials/what-is-version-control

Other tests	There also exist others that may be required, namely Linter Smoke Tests, System tests, Regression tests and Performance & Load tests.
Integration tests	These are the ones where the Platform as a whole will be tested to check the usage of the APIs in a manual way leveraging a close-to-production environment.

4.5. Synergetic Orchestration Requirements

The functional requirements in Table 4 account for the entire HDA synergetic orchestration Platform of NEPHELE. One requirement might be offered by one or more of its components (SMO, FRM, CCNM) or even by the (c)VO. Figure 27 presents the non-Functional requirements of the synergetic orchestrator specifying the goal that should be achieved.

Table 4. Synergetic Orchestration Functional Requirements

ID	Description	NEPHELE's architecture component(s)	Difficulty	Priority
FR_SO_001	Support AI-empowered decision making at local, regional, and global level based on the allocation of management areas across the continuum.	VO, FRM, SMO respectively	Medium	High
FR_SO_002	Manage network resources across the continuum to provide a network infrastructure that can cover the application needs.	CCNM	Medium	High
FR_SO_003	Offer a smart resource discovery functionality for placement of HDA components based on the actual status of the continuum.	All	Low	Medium
FR_SO_004	Initial AI-empowered placement of the HDA components according to an intent-based description of the application.	All	Medium	High
FR_SO_005	Support for the AI-empowered re-allocation of HDA components after instantiation to allow mobility scenarios and follow-me placement strategies.	All	High	Medium
FR_SO_006	The platform shall support local redundancy/high availability by endorsing a microservice and stateless	All	Low	Medium

ID	Description	NEPHELE's architecture component(s)	Difficulty	Priority
	approach for the architecture components.			
FR_SO_007	The platform shall support the deployment of cloud native compute workloads using Helm and K8s.	FRM	Low	High
FR_SO_008	The platform shall support the deployment of cloud native and VM-based network workloads using a ETSI complaint MANO stack.	CCNM	Low	High
FR_SO_009	The platform shall offer a way to describe the different workloads already deployed for a specific tenant to allow the interplay of functions from one or more HDA graphs. It shall include the required information to reach them and connect to them from additional VOs.	SMO	Medium	Medium
FR_SO_010	The platform shall offer a way for all workloads deployed to publish and exchange the required metrics, logs and KPIs in a secure manner that could be leveraged by the Service provider of the platform itself.	ALL	High	Medium
FR_SO_011	The platform shall be able to verify that the workloads offer the interfaces associated to the orchestration features that the workload will use.	SMO	High	Medium
FR_SO_012	The overall behaviour of the platform shall be described in an intent-driven approach by using a declarative human-understandable language.	SMO	High	High
FR_SO_013	The platform shall be able to react to events described in the HDA graph and automatically apply the call-back action specified which may include additional deployments, rollouts and decommission of parts of the HDA graph.	SMO	High	High
FR_SO_014	The platform shall be able to collect some basic performance and resource	ALL	Medium	High

ID	Description	NEPHELE's architecture component(s)	Difficulty	Priority
	usage metrics from all of the workloads in addition to those explicitly exposed by them.			
FR_SO_015	The platform shall offer multi-tenancy to isolate several verticals leveraging the same infrastructure.	All	Medium	High
FR_SO_016	The platform shall be able to report the complete inventory of underlaying systems and resources available to HDA developers.	All	Medium	Medium
FR_SO_017	The platform shall ensure that the interfaces that it offers to other parties (e.g. service providers) are provided using common definitions publicly available using the OpenAPI specification.	SMO	Low	Medium
FR_SO_018	The platform shall ensure that workloads deployed throughout the continuum are always tracked and accessible to the service provider and consumer.	All	Medium	Medium
FR_SO_019	The platform shall offer a history of operations performed to each of the HDA components for its auditing.	All	Low	Low
FR_SO_020	The AI-empowered SMO shall be able to decompose the high-level E2E performance constraints metrics of an application to resource allocation decisions at individual layers.	SMO, FRM	High	Medium
FR_SO_021	Support for appropriate action coordination modules in order to assess and manipulate the joint decision vectors of individual resource orchestrators.	SMO, FRM	Medium	Medium
FR_SO_022	AI-empowered modules shall support centralized training and decentralized execution schemes in order to minimize its dependence on a centralized action coordinator	SMO, FRM	High	Medium

ID	Description	NEPHELE's architecture component(s)	Difficulty	Priority
FR_SO_023	Support efficient, secure, and cost-effective AI-based resource allocation training schemes with real time monitoring to trigger re-training when discrepancy happens.	SMO, FRM	High	Medium

Table 5. Synergetic Orchestration Non-Functional Requirements

ID	Description	NEPHELE related feature	Difficulty	Priority
NFR_SO_001	Time to provision and deploy infrastructure from the VSP request to the readiness of the application shall be controlled and minimized.	Initial E2E elapsed time for deployment	Medium	Medium
NFR_SO_002	Time to react over an existing AG component once an alarm has been triggered shall be controlled and minimized.	SMO LCM reaction time	Low	Medium
NFR_SO_003	Internal components of the NEPHELE Platform shall be able to automatically scale resources to keep a controlled average CPU utilization measured at least every minute. Reaching the max threshold shall trigger a scale-out action while achieving the min threshold on scaled pods shall trigger a scale-in action.	NEPHELE Platform scalability	Medium	Medium
NFR_SO_004	Internal components of the NEPHELE Platform shall ensure a minimum service availability measured in a daily basis.	NEPHELE Platform high availability	Medium	Medium
NFR_SO_005	Internal components of the NEPHELE Platform shall be able to report their metrics in a controlled time range.	NEPHELE Platform Performance Monitoring	Low	Medium
NFR_SO_006	VOs shall be able to make requests to internal components of the NEPHELE Platform in real time.	NEPHELE Platform latency	Medium	High

ID	Description	NEPHELE related feature	Difficulty	Priority
NFR_SO_007	Internal components of the NEPHELE Platform shall be able maintain a low error rate of requests from the VOs measured in a daily basis.	NEPHELE Platform error rate	Medium	High
NFR_SO_008	The hosts where the internal components of the NEPHELE Platform are running must keep a controlled average CPU utilization measured at least every minute. Actions should be taken to scale-up and scale-down when reaching the max and min thresholds respectively.	NEPHELE Platform saturation	Medium	Medium
NFR_SO_009	Internal components of the NEPHELE Platform shall be configured to ensure that a high number of requests are supported from VOs and other services measured in every minute.	NEPHELE Platform traffic	Medium	Medium

5. Use Cases Description and Requirements

In NEPHELE, a set of four use cases tackling challenges in various industrial domains are included. Table 6 provides a summary of the main functions, IoT enablers and VOs that are going to be used. A detailed description of each use case is provided in the following subsections.

Table 6. NEPHELE use cases cartography

Use case	Use Case #1	Use Case #2	Use Case #3	Use Case #4
Industrial Domain	Emergency/Disaster Recovery	AI-assisted Logistics Operations	Energy management	Remote healthcare services
Edge/Cloud Functions	Risk assessment, Victim/Object Detection, Mission Management, Mapping	Route optimisation, Traffic management	Decision making, Secure access, Radio offloading	Dashboard, AI Algorithm, Image PRF (Process, Rendering and Filtering)
Generic/ Supportive Functions	Data Aggregation, Authentication, Telemetry	Load balancing, Live migration	Distributed AI, Authentication, Distributed Authorisation	Load balancing, Data management
Virtualized IoT Functions	Object detection, monitoring, Image processing	Video transcoding, Object detection	Image analysis, Video transcoding, Consumption analysis	Image processing, Data streaming
IoT Management Functions	Bootstrapping, self-configuration, self-healing	Self-healing, cloud-based management, zero-touch configuration	Blockchain, Encryption	Authentication, Network isolation
Intelligent IoT Devices	Ground Robots, Drones Cameras, Sensors	5G IoT gateway, UHD cameras, Sensors	System-on-chip devices, cameras, power meters	Medical imaging, Ultrasound HW
Participant Testbed	INRIA, ZHAW	LK, ININ, UOM	ODINS, SIEMENS	CNIT

5.1. Use Case #1: Emergency/Disaster Recovery

When a natural or human disaster occurs, time is critical and often of vital importance. Data from the incident area, containing the information to guide first response operations and improve the intervention effectiveness, should be collected as fast as possible and with the highest possible accuracy. The main objective is to rescue as many victims as possible in the shortest possible time whereas ensuring secure operations through risk assessment. To this aim, the rescue team needs to 1) deploy network infrastructure and devices for the mission, 2) map the area and locate and identify victims, and 3) assess the damages and comprehend the remaining or upcoming risks to prioritize rescue operations.

The high-level goal for this use case is to enhance situational awareness for first responders. To this aim, data collected in the area is of utmost importance. On the data coming from the IoT devices, image recognition, AI-powered decision-making, path planning, and other technological solutions can be implemented to support rescue teams. Sensor data fusion can help to provide precise 2D/3D representations of emergency scenarios in real-time, integrating the inputs from multiple sensors, equipment, and actors. Furthermore, all the information that is being extracted from the heterogeneous data should improve the effectiveness of decision-making and emergency response, increasing safety and coordination.

Robotic platforms have features that are highly appreciated by first responders, such as the possibility to generate 3D maps of a disaster scene in a short time. Open-source technologies (i.e., ROS – Robot Operating System) offer the tools to aggregate sensor data from different coordinate frameworks. To achieve this, precise localization and mapping solutions are needed, together with advanced sensor data fusion algorithms. The envisaged real-time situation awareness is only possible through substantial research advancement with respect to the state of the art in cooperative localization, mapping, and perception in emergency environments. The ability to provide information from a single specialized device (e.g., drone streaming) has been demonstrated, whereas correctly integrating multiple heterogeneous moving data sources with imprecise localization in real-time is still an open challenge.

In the following we present an overview of Use Case 1, whereas the extended description is presented in the Appendix.

Case study: Post-disaster in a container port

In this use case, the technologies and solutions will be tailored for a post-disaster scenario in a container terminal environment. In very complex container terminal operations, the risk of work accidents is inevitable and can happen at any time. Traffic accidents, work accidents, fires, environmental causes are among the other highly rated risks in container terminals. However, there are several other causes that can lead to severe accidents and disasters. On the one side natural conditions such as heavy rain, storm, earthquakes, floods, and wind can cause containers stacks collapsing or vessels accidents in approaching the terminal. On the other side, workers accidents due to human factors especially negligence in operating vehicles and equipment can lead to traffic accidents being one of the biggest potential risks. Finally, damage of equipment occupies an important place in the common causes of accidents in the container terminals. All the mentioned causes and accidents are potential factors that make this study case of high interest for containers ports.

In Figure 7 we summarize the main stakeholders, the location (physical or virtual) and the constraints, challenges, and risks for this case study.

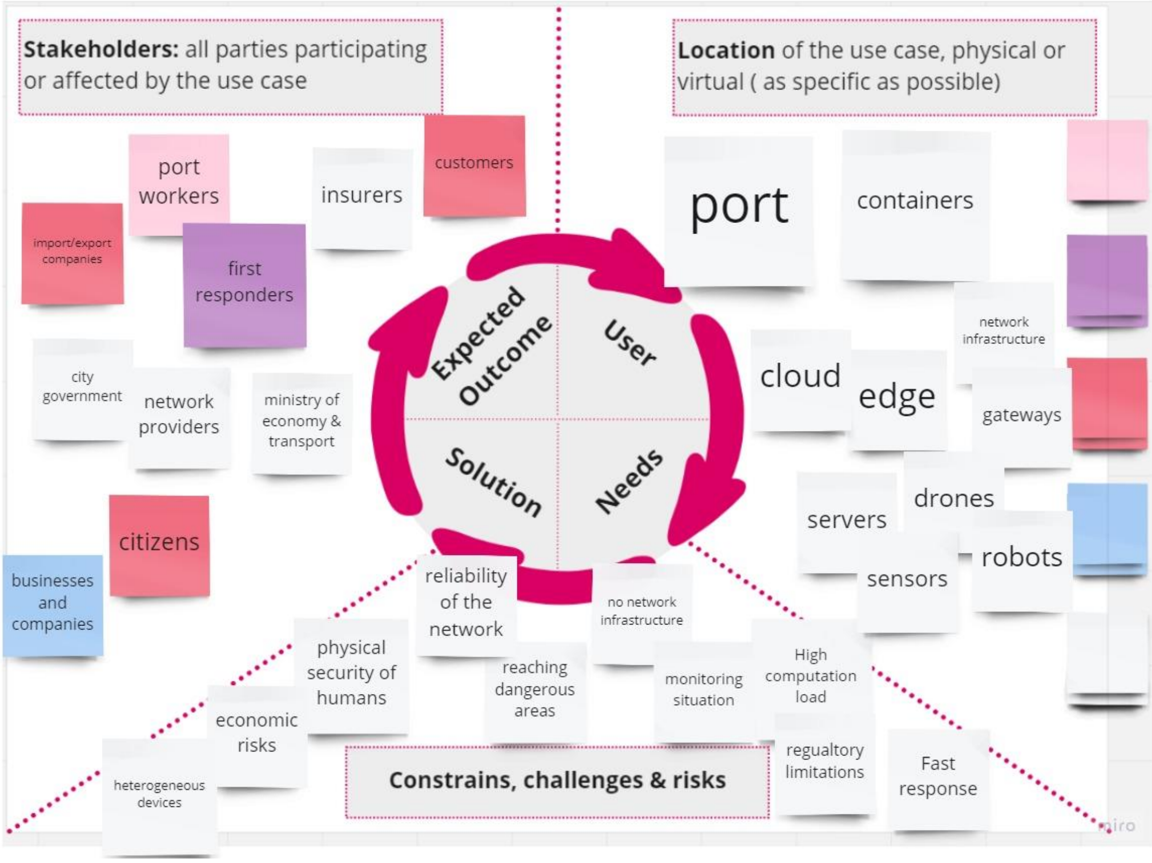


Figure 7. UC1 - Stakeholders, location, constraints, challenges and risks for the use case

Several stakeholders are involved in the scenario in focus in this study case. These range from the port workers and the business/companies active in the area to the citizens and customers present in the port. All of them may be categorized as potential victims to be rescued or in general persons at risk. A firefighter brigade is an example of first responders as the main stakeholder in this use case. They are typically based in the container port and own a set of physical devices (robots, drones, and sensors). Besides the hardware, the firefighter brigade also defines the logic of a first response application to be deployed and executed over the NEPHELE platform. The application logic is represented as a HDA graph which will be available on the NEPHELE repository. The application logic will define the high-level goal and the Key Performance Indicator (KPI) requirements for the application. The application graph will require the deployment of one or more VOs to represent IoT devices like robots or sensors and one or more application components supporting the operations with movement, sensing, and mapping capabilities. The VO description required by the HDA graph will be available on the NEPHELE Hyper-distributed Applications repository (HDAR).

The main physical location for the study case is a container terminal. After an accident in a terminal, there might be victims, due to explosions or collapsed containers/equipment/buildings, that need to be rescued or helped, there might be high-risk areas as a consequence of collapsed/damaged containers carrying dangerous materials or due to gas/liquid leakage, there might be lack of networking infrastructure and any available map of the port area may be not usable (or not completely) as the landscape was modified. The implementation of first response operations in this scenario will require the use of hardware and virtualized locations. The following hardware is used for the scope:

- **Mobile ground robots** that will be used to map the post-disaster area, to monitor the area and approach victims, to deploy sensor networks, to take samples of non-identified liquids leakage, to react to network disconnections by deploying additional nodes.
- **Drones** that will be used to fly over the area and send video streams and pictures from the integrated camera to identify objects/victims/leakages and make an early map of the area of interest.
- **Sensor networks** that are deployed by the ground robots in the post-disaster area are used to monitor the area for potentially dangerous situations through, e.g., gas detection, leakage detection, temperature detection, collapse detection and others.
- **Depth/Thermal cameras** mounted on the robots and the drones for risk assessment, risk prediction in the area, and victim identification and monitoring.

Moreover, networking and computation devices such as a 5G gateway, IoT gateways, edge servers and Wi-Fi routers will be used for this use case (see Figure 8). Part of the computation will also occur in virtualized environments at the edge and cloud using containerized application components.

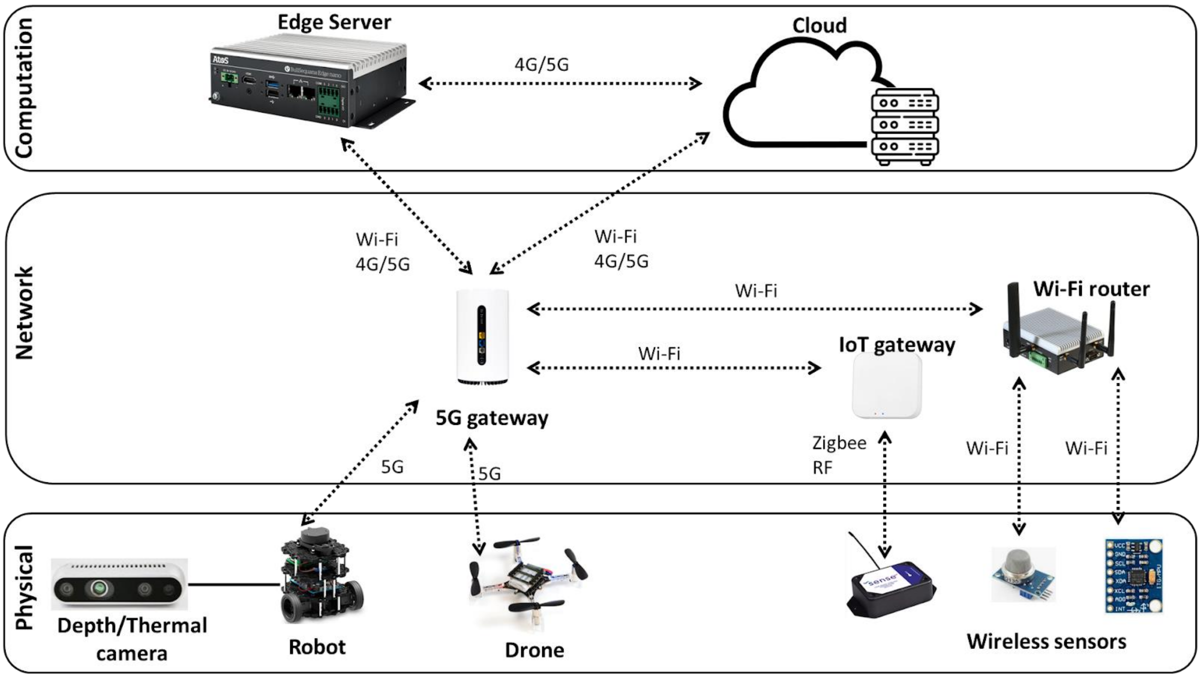


Figure 8. UC1 - Devices at the physical, networking and computation levels

Technical requirements and challenges

There are several technical requirements and challenges for this use case [29]. One of them is the heterogeneity of devices and time strong requirements. Data should be transmitted, filtered, and processed at different levels of the compute continuum to guarantee short delays while maintaining full knowledge of the situation. Therefore, communication technologies and protocols should guarantee low latency. Devices are heterogeneous in terms of CPU, memory, sensors, and energy capacities, some of the hardware (HW) and software (SW) components are use-case specific, while others are common to multiple scenarios (see Figure 9). Different complementary application components can be run on top of the same devices but exploit different sets of data, services, and application components. The network is dynamic because of link fluctuations, energy depletion of devices and device mobility (which can also be exploited when controllable) and this should be dealt with. How to use VOs, where to deploy edge computing for what application in such a context is a tremendous challenge that NEPHELE can address. The orchestration of VOs and their performance are highly related to the hardware that manages them.

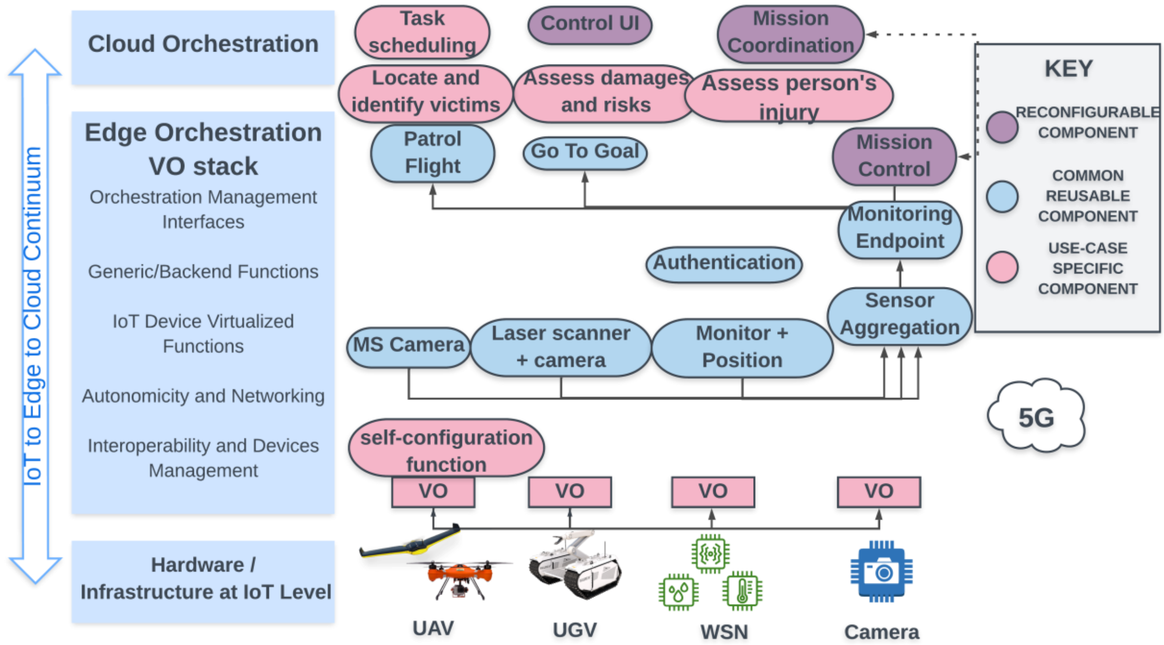


Figure 9. UC1 - VOStack mapping to emergency/disaster recovery application scenarios [29]

To best face the needs of the operations in the use case above and offer solutions to reach the overall goal for the solution we can summarize the following main technical requirements and challenges.

- **Orchestration of software components:** given the application graph, a dynamic placement of software components should be enabled based on service requirements and resource availability. This will require performance and resource monitoring at the various levels of the continuum and dynamic components redeployment.
- **Device Management:** some application functionalities can be pre-deployed on the devices or at the edge. The device management should also enable bootstrapping and self-configuration, adding and removing devices on the fly, supporting hardware heterogeneity, and guaranteeing self-healing of software components.
- **Low latency communication:** communication networks to/from disaster areas towards the edge and cloud should guarantee low delays for fast operation in first under mobility conditions and possible disconnections.
- **Dynamic multi-robot mapping and fleet management:** coordination, monitoring, and optimization of the tasks allocation for mobile robots that work together in building a map of unknown environments or executing tasks in a collaborative manner.
- **Computer vision for information extraction:** AI and computer vision enable people/object detection, position detection and localization from image and video data.
- **Smart data filtering/aggregation/compression:** a large amount of data is collected from sensors, robots, and cameras in the intervention area for several services (e.g., map building, scene, and action replay). Some of them can be filtered, others can be downsampled or aggregated before sending it to the edge/cloud. Smart policies should be defined to also tackle the high degree of data heterogeneity.

Application Graph Specification

The graphical representation of the application graphs for the use case are detailed in the Appendix. Three different levels of detail are used to show: i) the high-level view of a specific scenario for the application with the involved devices and overall objectives; ii) the application graph reporting the logical application components, the VOs/cVOs and the conditions and requirements for them to communicate among each other and with the identified VOs/cVOs, and iii) the service graph where the single services and the links among them are shown.

All in all, five different application scenarios are required to manage the various challenges and objectives for the study case in UC1, as detailed next.

Mapping

This scenario refers to the application components and services needed to map a given area using ground robots and/or drones through cameras and lidars. The resulting map and its graphical representation will be used to give the first responder commander a graphical overview of the area and by this enhance his situational awareness. Drones will be used to make an aerial 3D map of the area and detect the condition of buildings and containers, the location of people in safety areas, fires, liquid leakages and their progress, and other potential risks. Ground robots will be used to map the area from the ground in 2D with greater detail of analysis as done with the drone. A VO should be deployed for each ground robot and drone at the edge of the network. A network connection fulfilling data rate and latency requirements for video streaming is required between drone and NEPHELE through the corresponding VO to send the videos and process them. The communication between physical devices, the virtual counterparts at the VO and the other application components is enabled through the Zenoh protocol. Some services will be running on the physical devices, whereas others on the edge and cloud continuum and will have to be configured through the VO. The storage and analysis of the collected data may be demanding, reason for which having the edge/cloud data storage support is important. Nonetheless, robots and drones should have local storage to save the video in case they lose connection and should be sent when the connection is recovered. A fleet management service, a GUI with alerting, monitoring, task control, replaying and dashboards services through a web interface complete the application graph.

Victim detection and injury assessment

This scenario refers to the application components and services to detect victims and assess their injuries in the post-disaster area using ground robots and/or drones. Like the previous application scenario, a VO should be deployed for each ground robot and drone at the edge of the network, a network connection fulfilling data rate and latency requirements for video streaming is required between drone and NEPHELE through the corresponding VO, storage is required and computation at the edge are required. The Zenoh protocol will be used also in this case for communication between the application components and services running either on the physical devices or on the edge and cloud continuum. A trajectory planner for the ground robots and drones used is needed. Additionally, services are required for object/person detection and assessment of their injury. Using AI-supported algorithms, the map of the area can be enhanced with a graphical add-on about the detected information. By this, the first responder using a GUI will experience an enhanced situational awareness. Managing multiple robots for this task also requires a fleet management service for the application. Alerting, monitoring, task control, replaying and dashboards services through a web interface complete the application graph.

Risk prediction

This scenario refers to the application components and services to predict possible risks in a post-disaster area using ground robots, drones, and sensor nodes. Once the wireless sensor network is deployed, the sensed data is sent to the edge of the network using the appropriate gateway or the access point depending on the network configuration and technology. Multi-hopping is available for sending data from the physical sensors deployed to ensure the connectivity and monitoring of all sensitive areas to the VOs when the application allows it. Robots and drones are used to further monitor the area and identify risks using their cameras. Liquid leakages are monitored over time to verify their movement and detect potential risks. A trajectory planner for the ground robots and drones used is needed, whereas AI-supported algorithms are used to detect risks to be shown on the GUI. Similar requirements as for the previous applications exist in terms of VOs, networking, communication protocols, fleet management, and storage. Alerting, monitoring, task control, replaying and dashboards services through a web interface complete the application graph.

Device deployer and liquid sampler

This scenario refers to the application components and services to physically deploy sensor nodes in a certain area or take liquid samples using ground robots equipped with a manipulator. A single VO should be deployed for the robot with the manipulator. Whenever a wireless sensor node is to be deployed in selected areas of the port, the ground robot with manipulator will be used. The selected areas for deployment are the most pertinent places according to the physical conditions of the port and the areas that have been identified as needing monitoring (e.g., based on the built maps). Sensors could be gas detectors, temperature, air quality, microphones, cameras, motion detectors, seismic detectors, and infrared sensors and thus may request different network capacities in terms of bandwidth, latency, etc. A gateway should be placed near the sensor network to send data from physical sensors to the Internet and a VO should be deployed for the wireless sensor network gateway at the edge of the network. The sensor network is pre-configured before the deployment, and it must ensure connectivity with the corresponding VO guaranteeing the needed bandwidth and latency. A network connection is required between robots and NEPHELE through the corresponding VO to receive instructions for the deployment of the wireless sensor network and to report feedback on the executed tasks. After the first deployment, sensors are continuously monitored to prevent disconnections. Data sent to NEPHELE through the VO should, therefore, also include information on the sensor/robot status itself (sensor temperature, battery level). Once a disconnection, or in general a need for additional devices is detected, the deployment of additional sensors can be triggered.

The same technology and hardware will be used in case some non-identified liquid leakage was detected with the camera. A robot with a manipulator can be sent to take samples of the liquid for further analysis. This will avoid this risky operation to be performed by first responders directly. Fleet management, trajectory planner, storage, alerting, monitoring, task control, replaying and dashboards services through a web interface complete the application graph.

Network and device monitoring

This scenario refers to the application components and services to monitor the network connectivity for the IoT devices deployed in the post-disaster area. Network devices, sensors and robots are continuously monitored to prevent risks of device and network disconnection. To this aim, networking and device information are sent through the VO or directly (depending on the device) to the application components monitoring the network status. Similar requirements as for the previous applications exist in terms of VOs, networking, communication protocols, and storage. Alerting, monitoring, task control, replaying and dashboards services through a web interface complete the application graph.

5.2. Use Case #2: AI-assisted Logistics Operations in the Port of Koper

Port of Koper is a multi-purpose deep-sea freight port located at the Northern end of the Adriatic Sea. Logistic and port services are provided by the company Luka Koper d. d. Total maritime throughput in 2020 topped over 19,5 million tons and the company has strengthened its position becoming the most important container port in the Northern Adriatic. Port terminals are equipped with state-of-the-art transhipment and warehousing equipment, such as ship-to-shore cranes, reach-stackers, forklifts, utility tractor rigs, etc. Port of Koper has excellent connections to road and railway network. Continuous monitoring and optimization of the traffic within the port poses a daily challenge (traffic congestions, unplanned road closures, etc.), which will be addressed in the use case.

The main objective of this use case is to optimize the routing of containers from the Container terminal yard or Depo area to different Container Freight Stations (CFS) within the port, where the cargo is stuffed/stripped, and vice-versa. This is one of the most important operations in the port. This objective will bring business value in terms of reduced routing times, lower CO_2 emissions, higher truck/forklift utilization, and service level agreements (e.g., times of delivery, compliance with goods sensitivity, etc.).

The exploitation of the VOStack layers will allow to exchange and aggregate data among the physical components involved in the use case (e.g., forklifts, trucks, cameras, sensors). The application of decentralized machine learning techniques at a VO level will satisfy requirements regarding security and low latency regarding a set of port operations (containers routing optimization, traffic detection and classification). The integrated meta-orchestration framework will allow the orchestration of the deployed microservices between the cloud and edge computing orchestration platforms ensuring the self-healing, portability and elasticity of the complete solution (Figure 10).

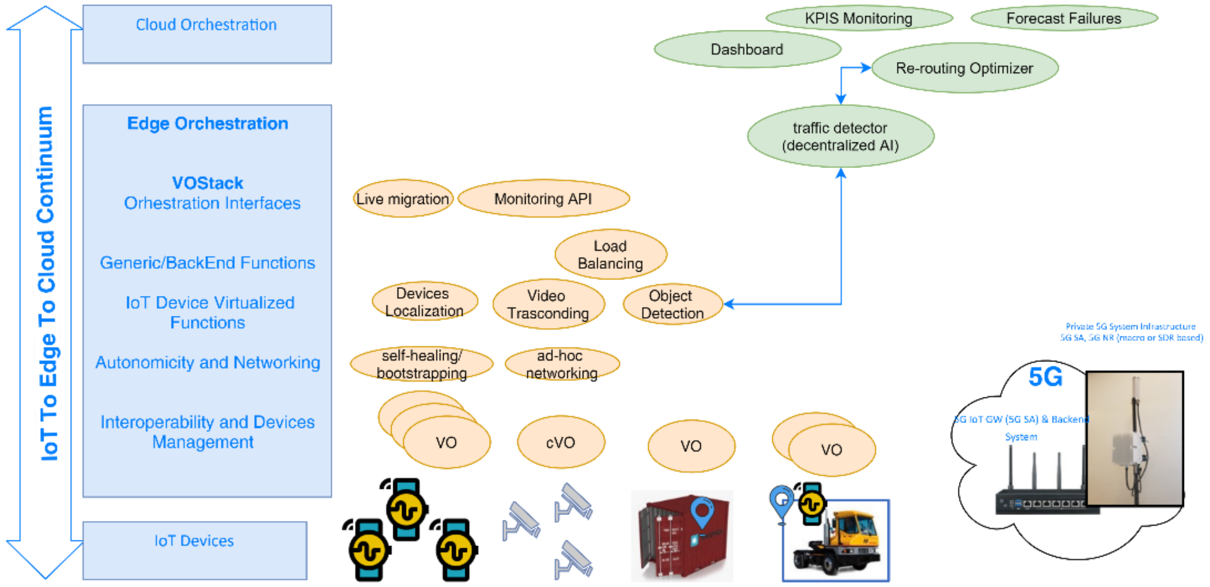


Figure 10. UC2 - AI-assisted Logistics Operations in port

In the following we present an overview of Use Case 2, whereas the extended description is presented in the Appendix.

Case Study: Containers routing optimization in the port

Freight forwarders place order to the Container Terminal to organize that the set of containers, both full (in import) or empty (for export), are timely delivered to the CFS in the port, where containers are loaded/unloaded of the cargo. Additionally, and in parallel, freight forwarders place orders to the General Cargo Terminal. On this basis, a common delivery plan is prepared, including a list of containers to be delivered to CFS (and vice versa), delivery equipment requirements and staff accounted for the task. When the plan is set, the algorithms for the “container route optimization” should define the work order list sequence and the optimum number of trucks/forklifts, taking into consideration safety rules, priorities regarding vessel schedule, priorities regarding rail operations, cargo sensitivity, client ranking, terminal equipment availability, daily traffic in the port (road and rail), work on other terminals, etc.

In the Figure 11 we summarize the main stakeholders, the location (physical or virtual) and the constraints, challenges, and risks for this case study.

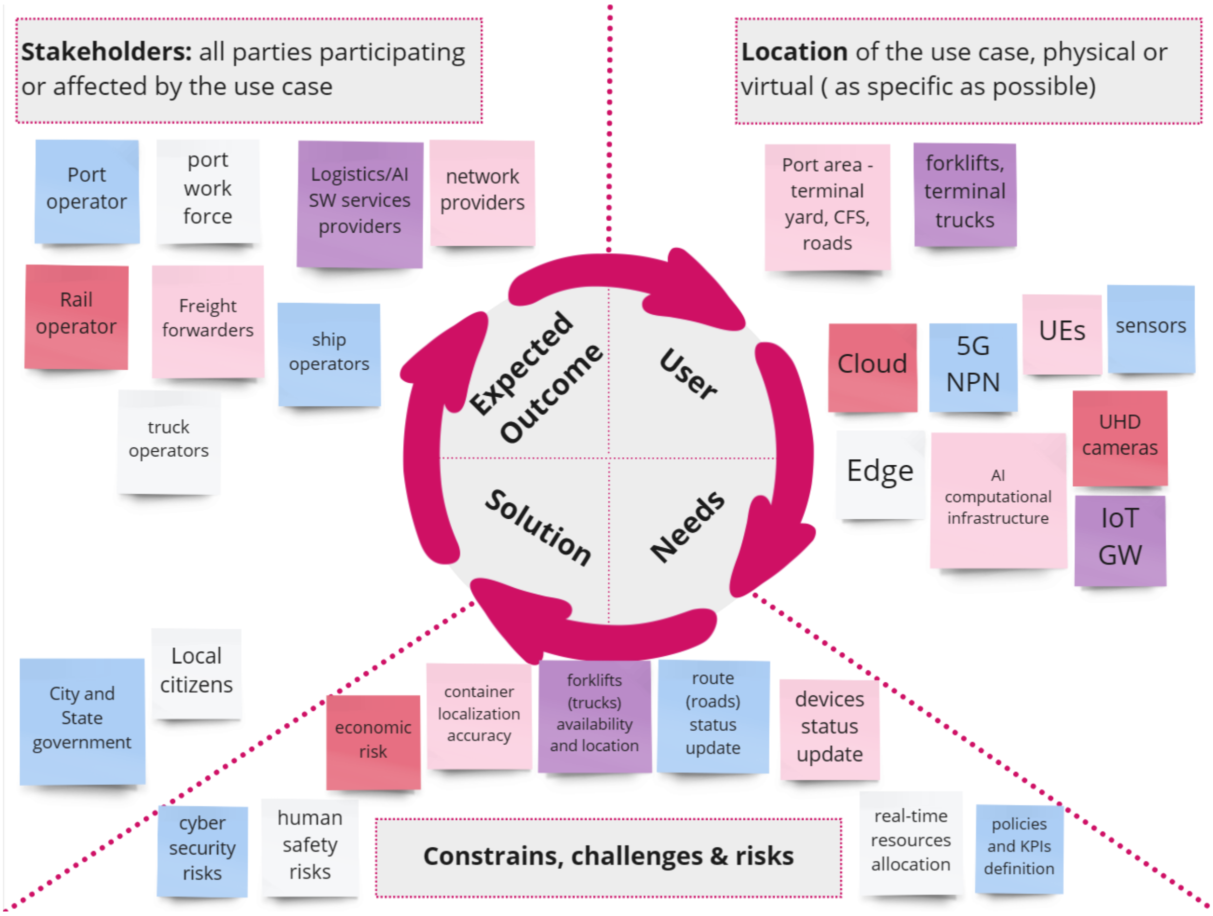


Figure 11. UC2 - Stakeholders, location, constraints, challenges and risks for the use case

Several stakeholders are related to the use case scenario, its actors and its impact to the port and its neighbourhood. Port operator is considered as a main stakeholder since its role as a main user of the solution explored within the use case. The port operator is therefore expected to gain certain economic benefits. Considering the latter, benefits are also expected for other stakeholders directly involved in the logistics process, i.e., freight forwarders, ship operators, rail operator, truck operators. All those will benefit out of the optimized business process(es) taking place in the port. The latter will also impact port work force in general, i.e., their productivity.

Since the use case requires specific software solutions, network, sensors and other equipment, at least two additional stakeholders are required, i.e., software/service provider(s) and network provider/operator. On the other side, their economic benefits come from providing required services.

Specific interest in the use case is also related to the local community, i.e., local citizenship and city/local government, as well as state government which all benefits due to the economic strength of the port. As well, optimized business processes in the port, as described, tends to reduce the greenhouse gas emissions thus contributing to greener environment and sustainable development.

The physical location of the use case study is a container terminal where we are looking for optimizing the routing of containers from the container terminal yard (or depo area) to different container freight stations (CFS) thus reducing routing times, lowering greenhouse gas emissions, enhancing truck/forklift utilization, and enhancing service level agreements (e.g., times of delivery, compliance with goods sensitivity, etc.). The implementation of the use case will require the use of hardware and virtualized locations, as is illustrated in Figure 12. The following infrastructure is used for the scope:

- **Infrastructure as a Service (IaaS):** cloud infrastructure provided by UoM, required for the cloud components of the cloud-continuum based solution to run properly.
- **Edge Infrastructure as a Service:** edge infrastructure provided by LKOP, required for the edge components of the cloud-continuum based solution to run properly.
- **Portable Infrastructure as a Service:** mobile network solution (5G) provided by ININ, required for providing data connectivity between IoT devices (sensors) in the field/port, its corresponding VOs and application level of the solution.
- **IoT devices:** provided by LKOP and ININ, required for acquiring real-time data and status (e.g., location of terminal trucks) from the field. Now, following IoT devices are expected to be involved in the use case:
- **Industry-grade 5G IoT gateway with additional computing capabilities (Far-Edge IaaS):** serves as a gateway providing 5G connectivity to non-5G devices and enables far-edge components of the cloud-continuum based solution to run properly, industry-grade UHD cameras, GNSS sensors providing location, truck speed and other GNSS related information in real-time, On-Board Units mounted on trucks/ forklifts.

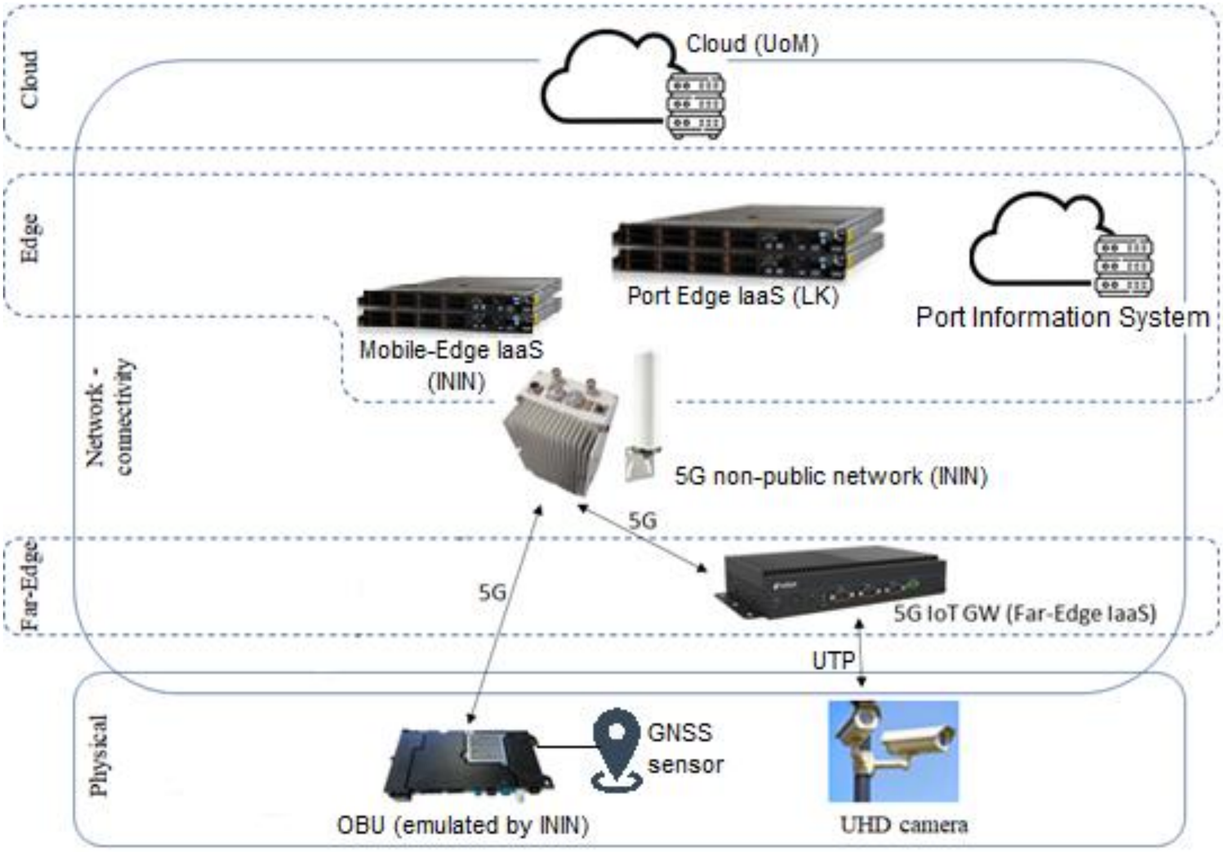


Figure 12. UC2 - Devices at the physical, networking and computation (cloud, edge, far-edge) levels

Technical requirements and challenges

Use case 2 main technical requirements and challenges are detailed as follows.

- **Radio network coverage (5G):** whole area where the containers routing optimization process takes place requires quality and stable radio network coverage providing data exchange between sensors in the field (e.g., stable UHD camera, on-board units mounted on trucks/forklifts), applications running in far-edge, edge and cloud environment, as well as applications running on end users' devices.
- **Orchestration of software components:** containers routing optimization application components (as well as VO stack components) require certain conditions and availability of resources which need to be considered during the automated deployment. This will further require performance and resource monitoring at the various levels of the cloud-continuum and possibility of dynamic redeployment of certain components.
- **Device Management:** some application functionalities can be pre-deployed on the devices or at the edge. The device management should also enable bootstrapping and self-configuration, adding and removing devices on the fly, supporting hardware heterogeneity and guaranteeing self-healing of software components.
- **Interface to data relevant for the business process:** containers routing optimization process requires data on freight forwarders demands and service level agreements stored separate databases.
- **Sensor data collection and aggregation:** data collected by sensor need to be properly stored in a secure place and available for further data processing.
- **Computer vision for information extraction:** since conditions relevant for containers routing optimization will be, among others, collected by cameras, algorithms for detecting relevant situations in video-stream or in still-pictures are required.
- **AI/ML supported data processing – containers routing optimization algorithm:** a key component of the system which considers all relevant data (sensors data, cameras data, freight forwarders demand, service level agreements) and produces optimal schedules for freight/containers transportation within the port.
- **Providing feedback to port personnel and freight forwarders:** an application component providing outputs of the optimization algorithm – schedule and routes for truck/forklift drivers, schedules for freight forwarders, etc.
- **Analytics:** based on data collected and output data from the optimization algorithm, post-analytics should be available to evaluate the successfulness of the solution.

Application Graph Specification

The graphical representation of the application graph for the use case is detailed in the Appendix. This includes the containers routing optimization process, the data collecting and forwarding to containers routing optimization application with the goal of processing them and reporting results (containers routing schedules) to the customer.

Data will be collected by video cameras, IoT sensors and queried from Port Information System. VO would be deployed for every camera and every OBU/GW equipped with various sensors. Also, VO will be deployed and adjusted to extract data related to port trucks from the port information system (i.e., OBUs not directly accessible). Considering certain requirements for data types provided through VOs (e.g., video stream), additional network related configurations, such as one for QoS, will be applied. Based on the data types provided by each sensor, corresponding data processing component will be applied (e.g., image/video extractor and detector of certain scenes, sensors' data collector) either at the far-edge, edge and/or cloud. Pre-processed data and data from port information system will be feed into the route optimization engine which outcome will represent a ground base for deciding on containers routing schedule ("dispatch decision making" component). The latter will be then distributed to the customers. Since the final decision on considering and applying proposed schedules to physical containers routing process depends on the customer/dispatcher, the "containers routing optimization" process keeps collecting and processing data in real-time which also enables adapting containers routing schedules to be updated in real-time.

5.3. Use Case #3: Energy management in smart buildings/cities

The physical increase of cities and their population and the continuous advancement of technology motivate the need and the increasing popularity of the concept of smart buildings/cities. If we put the focus on sustainability and the need to reduce carbon emissions, it has become increasingly important that buildings are designed and operated in a way that minimizes energy consumption. Thus, energy efficiency is a crucial aspect of smart buildings and cities. The integration of cutting-edge technologies such as IoT, machine learning and edge computing can make them even smarter, more efficient, and more sustainable.

One of the keyways that smart buildings achieve energy efficiency is using sensors and automation systems. Sensors can be used to monitor environmental conditions like temperature, humidity, and light levels, and this data can be fed into automated systems that adjust heating, cooling, and lighting to maintain optimal conditions while minimizing energy consumption.

In this sense, smart buildings are prepared to monitor and control energy use in real time to guarantee the desired energy efficiency. By collecting energy use data and analysing it in real time, building operators can identify areas where energy is wasted and take actions. This can be accomplished in a reduced amount of time, thereby achieving an efficient energy consumption.

With smart buildings, we can automatically adjust the use of heating, ventilation, and air conditioning systems, turn off lighting or other electrical devices when they are not in use, or even implement smart systems that allow us to dynamically adjust energy use based on the demand.

The integration of IoT to edge to cloud computing in smart buildings is thus important for energy efficiency reasons. By processing data closer to the source, edge computing can minimize the amount of data that needs to be transmitted to a centralized server, reducing energy consumption associated with data transfer and processing, too.

With the integration of the IoT, edge computing, and cloud computing, the possibilities for intelligent monitoring and remote energy management in these environments are expanding. This is where UC3, focus on energy management in smart buildings/cities, comes into play. Led by ODINS and supported by SIEMENS and IBM, this use case aims to design, develop, produce, and market products that leverage the entire IoT to edge to Cloud Continuum, to better address strict Service Level Agreement (SLA) requirements for the development of smart energy solutions. By implementing an automation scheme that gathers real-time information from a variety of IoT devices, such as appliances, sensors, and HVAC systems, along with edge nodes that instantiate Virtual Objects (VOs), this approach avoids bottlenecks caused by placing all the intelligence in a centralized Smart Building/City monitoring and control system, delving into the technical challenges, applications, and benefits of this innovative energy management solution.

The objective of UC3 is to develop different advanced applications and services leveraging on the VO Stack, to manage control actions of building equipment, providing user with customized services for energy-efficient, well-being and comfort, covering security aspects, too. The aim of this use case is also to show some of the security features that NEPHELE will offer, such as secure and authenticated access, secure and distributed access sharing of data, as well as higher level applications such as detection of people or objects.

In the following we present an overview of UC3, whereas the extended description is presented in the Appendix.

Case Study: Energy management in smart buildings/cities

In this use case, the technologies and solutions will be adapted for an energy management scenario focus on smart building, but that can even be exported to smart city scenarios. In Figure 13 we summarize the main stakeholders, the location (physical or virtual) and the constraints, challenges, and risks for this case study.

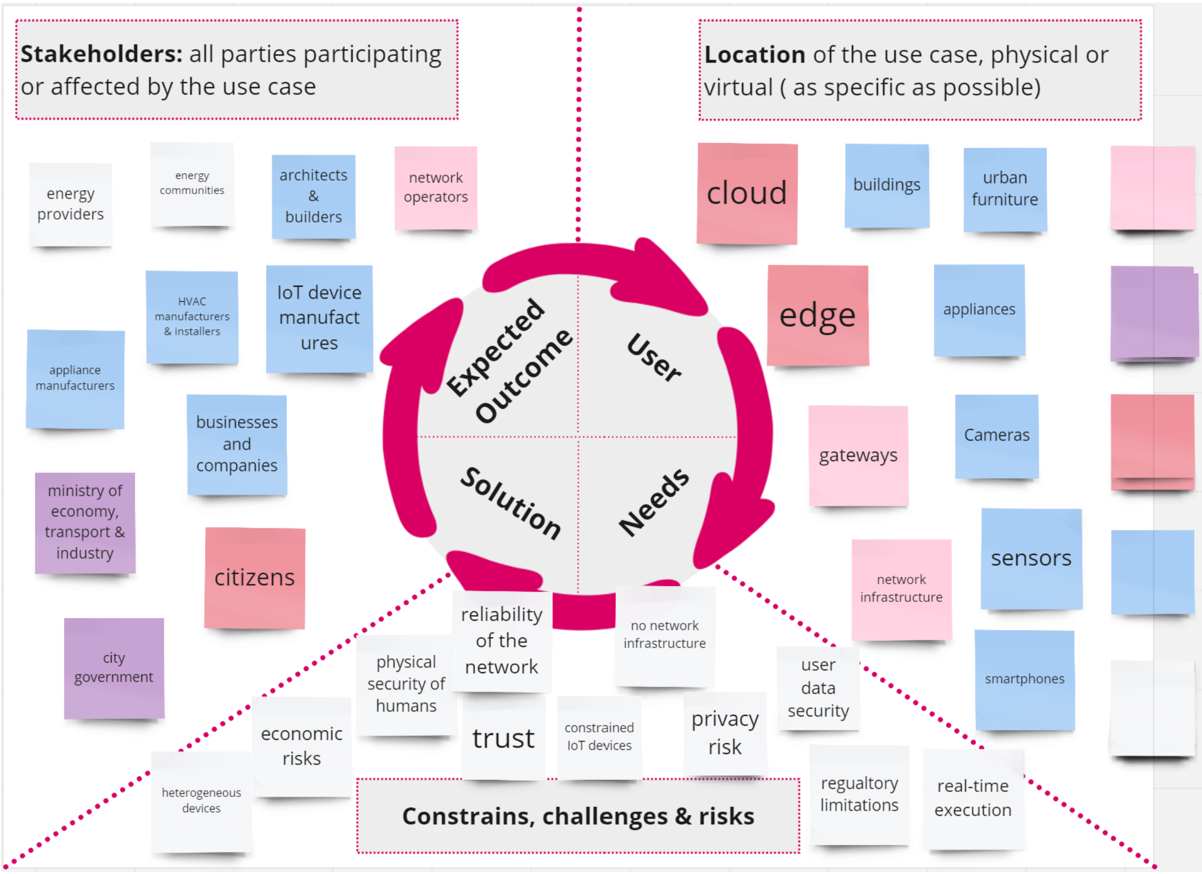


Figure 13. UC3 - Stakeholders, location, constraints, challenges and risks for the use case

Several stakeholders are involved in energy management solution in the context of smart buildings and cities. From the architects who design intelligent buildings to the citizens who finally live in them, there are different profiles that can benefit from the advantages that NEPHELE offers. Other examples are IoT providers for monitoring and actuation, household appliance manufacturers, network operators that connect the deployed systems and the energy operators, as well as governors, legislators and those responsible for building management as energy communities.

Citizens are the main users of buildings and cities, and as such, they are the ones who end up choosing which solutions best suit their needs. Among the main features they are looking for are real-time monitoring and decision-making, AI-assisted information analysis systems, as well as security and control of access to information to reduce costs, improve efficiency as well as simpler tools.

Local governments, in charge of promoting the deployment of solutions that optimize the use of resources to improve the services offered to citizens, are looking for tools that allow them to monitor and analyse energy consumption in real time, as well as intelligent mechanisms that allow them to take assisted and automated decisions, while guaranteeing users the highest standards of privacy and information control. Among the advantages they seek in new technologies is edge computing, which allows optimizing the use of the network, distributed processing, and latency reduction, allowing real-time communications. AI-assisted decision-making allows governments to offer better and more efficient services, while authentication and authorization mechanisms allow sharing the minimum necessary information and controlling who and for what.

Building managers and energy communities seek to improve the use of resources and monitoring systems for production and consumption as a starting point for offering more advanced services. Among the fundamental aspects that must be offered are the security and privacy of user information, real-time monitoring and data analysis, and AI-assisted decision-making systems, to offer an intelligent management solution on an interface intuitive and accessible.

Another of the stakeholders that arise from this use case are security companies that can use security devices and cameras to control access to buildings or help with locating tasks when a person disappears. For these scenarios, in the first place, it is necessary to be able to process the images in a distributed way, analysing the data and monitoring other sensors to improve computational efficiency. In addition, exquisite care of sensitive information and the privacy of users is essential. For this, the proposed solution must offer security, distributed analysis tools, real-time access to data and assisted decision tools. In addition, interfaces for access to information and tools should be offered, as well as advanced object detection tools.

This characteristic of the use case can be raised at different levels, from a high-level scope that would include buildings, urban furniture, electrical appliances, IoT devices and mobile devices for access to the network infrastructure that would provide connectivity from the cloud to the distributed processing nodes at the edge.

For the scope and regarding the physical layer (hardware) wireless sensors and actuators are used in the smart buildings, in this sense, these elements are the responsible for obtaining the real-time measurements of the environmental conditions as temperature, humidity and light levels.

Regarding the network infrastructure, the existing IoT and Edge/Cloud computing infrastructure is mainly composed by wireless microcontroller IoT devices communicating with Edge nodes and cloud platforms, as is illustrated in Figure 14. The IoT constrained devices include sensors (temperature, humidity, CO2), and actuators which monitor and control different environment parameters.

Technical requirements and challenges

To best face the needs of the UC 3 operations and offer solutions to reach the overall goal for the solution we can identify the following main technical requirements and challenges.

- **Software component orchestration:** Several components will be required to provide applications and services that have been used along the use case. To work together, it is necessary to provide resource monitoring at several levels together with mechanisms to coordinate and orchestrate this cooperation in the continuum.
- **Device customisation and management:** Devices or edge nodes may be needed to be re-configured or updated. It is desired that an IoT device may be extended with a set of functions. An execution of a virtual function (e.g., provided in a form of Complex Event Processing rule or a Neural Network) on a device would turn it into an intelligent IoT device.
- **Device Interoperability:** The access to VOs and their data should be provided via a standardized interface, e.g., W3C Web of Things. The functionality of IoT devices should be exposed

via a standardized and semantically enriched interface at the VO level. This will enable interoperability at the protocol and data level between diverse IoT devices and NEPHELE applications.

- **Control Access Management:** The access to resources, services or applications must be protected and controlled using access control policies that support distributed scenarios.
- **Identity Management:** the interaction between devices and services, the access to device data and more complex scenarios must be secured through advanced authentication mechanism that focus on privacy preserving mechanisms that allow controlled and limited disclosure and access of user or device attributes.
- **Data storage:** A distributed data storage system is necessary to store and share common information such as service public information, certifies, service policies or Distributed Identifiers (DIDs) in the specific case of identity data.
- **Low latency communication:** real-time video requires high bandwidths and low latencies for quality streaming. Likewise, the management and monitoring of electrical consumption must be carried out over a stable connection with low latency to adjust the reaction of the management tools to the maximum.
- **Computer vision for information extraction:** Persons and objects detection, their position and location from picture and video data are all made possible by AI and computer vision.
- **Intelligent data filtering/aggregation/compression:** A large amount of data will be collected from sensors and cameras in the UC3 environment. Some of them can be filtered out, others can be reduced or aggregated before sending them to the edge/cloud. Smart policies need to be defined to also address the high degree of data heterogeneity.

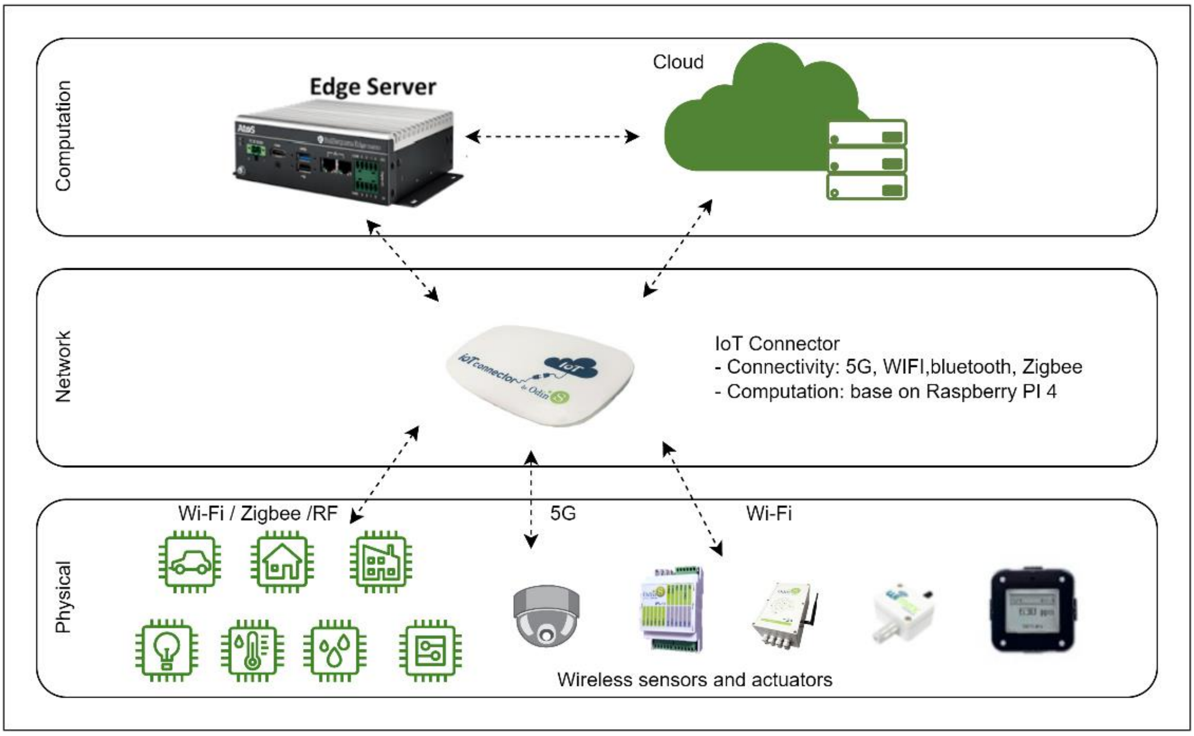


Figure 14. UC3 - Devices at the physical, networking and computation levels

Application Graph Specification

As for the previous use cases, the graphical representation of the application graphs is detailed in the Appendix with three different levels of detail. Six different application scenarios are identified to manage the various challenges and objectives for the study case as detailed next.

Secure group communication

This first application is used to provide secure attribute communication between two entities in a secured way, with the aim of offering minimum disclosure technics base on the use of W3C verifiable credentials. In this case the communication is done from a device to another device, but the target objective could be also an application or a service. This kind of protocols and other secure group communication protocols (e.g., CP-ABE) are computationally expensive and, in most cases, cannot be adapted to constrained IoT devices due to their computational limitations. Thanks to the VO in Edge nodes, the possibilities of instantiating security functions are much wider. The translation of credential emission and use scenario, are very simple making use of Virtual Objects (VOs). For each device, a VO will be deployed in the edge, which provides more computational and storage capacity.

Distributed complex decision making

This application scenario shows how VO edge nodes will perform intelligent energy saving actions not only based on the sensor measures collected by the IoT devices managed, but also the information and data coming from other nodes. This will drive the efficient use of renewable energy sources and the reduction of peaks in the energy consumption.

The objective is to control through the temperature and consumption sensors when the heating devices should be activated. To do this, decisions must be made based on multiple devices that must be able to coordinate. The system must also offer a management interface that allows the parameters to be met by the system to be indicated, such as the target temperature for each zone and the consumption limit established for the balancer. Thanks to the virtual objects and the possibility of composing them into more complex ones, it is possible to design temperature sensor-HVAC pairs, which makes it possible to define more complex and intelligent virtual devices at a first level (cVO1 and cVO2), resulting in a system with HVAC with a thermostat. But on top of this level, it is possible to create a more advanced one that groups several intelligent air conditioning systems (cVO3) that interacts with the Smart Energy Balancer, being able not only to work on temperatures but also to organize the switching on and off to limit the number of machines that are on at any given time, limit the power or prioritize one over the other depending on the complexity or intelligence that we want to implement on the composite system.

Distributed authorization scenarios

This application scenario shows where an access request to a resource is not decided exclusively by a centralized cloud platform but made by a back-end service leveraging in a Distributed Ledger Technology (DLT), that stores distributed access control policies, (e.g., distributed-XACML), employed at the edge nodes closest to the target resource to enforce access.

This application scenario shows, on the one hand, how policies can be configured by some entities in a distributed way using DLT, as well as on the other hand, how the policies are used in a distributed authorization process when a device tries to access a service using the application using the Service Access Control. The translation of this scenario to the field of Virtual Objects allows the handling of security policies and more complex and advanced credentials in the authorization processes. Application 1 offers an advanced access control system that retrieves access control policies from a DLT with several distributed peers. These policies can be updated on one of the DLT peers which automatically updates and distributes the information to the others.

Object/Person detection

This scenario shows how to provide object and person detection through AI-assisted image processing tasks running in distributed Edge nodes. This is achieved from the data collected by video cameras deployed in the scenario for finding dangerous or suspicious objects, and getting the location of vulnerable persons, such as missing children or lost elders.

The use of NEPHELE in this application case allows the definition of a complex virtual object (cVC1), which allows adding and processing several video streams at the same time, optimizing the processing and the required bandwidth, while allowing the application of quality control mechanisms. of service throughout all communication channels. cVO1 will offer an advanced interface to adapt to the people and object detection service based on ML/AI techniques deployed, Application 1 will provide access to the management, monitoring and alert configuration interface that will act on the VOs of the cameras as well as it will allow to define the detection parameters in 2.

Communication radio offloading

This section shows how communication radio offloading for battery-powered devices or subscription base technologies can benefit from the use of virtual objects; they can be used to optimize the use of communication channels. Decentralized AI-assisted orchestration of VOs may avoid certain radio channels to save either IoT device battery or subscriber data, by offloading the communication flows to auxiliary technologies (e.g., switching from NB-IoT to Wi-Fi access point).

In this application scenario, several devices are monitored to find out different parameters of energy consumption and available battery, as well as the available interfaces and the coverage they have. All this information is reported to the network monitoring nodes, which notify the distributed orchestration services of the changes. These should send instructions to the devices, indicating which interfaces they should use to properly manage network offloading.

The use of virtual objects as a digital representation of the device allows communication to be established on the physical device in situations of unstable connectivity, even when communication is intermittent, acting as a cache for the instructions sent from the infrastructure, or also storing the record of the monitored data. In turn, the VO can become part of the distributed orchestration system and make local decisions based on the specific context of the device or devices it represents.

On the other hand, the scenario presents Application 1 with a management interface to configure the orchestration system and check the status of the system and the devices connected to it.

Customizable IoT devices to support energy-efficiency and well-being in buildings

Building Automation Systems (BAS) has changed over time and thus need to be adapted. During their long lifetime their requirements change. For example, a high-energy efficient BAS turned not to be so efficient during the pandemic time or the post-pandemic time. Heating, ventilation, air quality and other building services designed prior to the pandemic may not be as energy-efficient nowadays. The reason for this is the occupancy of rooms, which in many cases has been changed. This application will demonstrate how the true presence in rooms can be easily determined and how that information can be used to make a BAS more energy efficient.

To achieve this, the application scenario will demonstrate two contributions of NEPHELE projects. First, new sensors need to be added and integrated into an existing BAS. This will be accomplished via the concept of the VO. In a plug and play fashion a VO needs to expose the functionality and data of a newly added sensor. Data access and semantics, which describes the device, should be standardized, e.g., with W3C standard Web of Things. Second, an existing device may need to be customized so that it can use the data from the new sensor. For instance, an existing thermostat will adapt its control based on the data from the new presence sensors. The thermostat can be customized to load and run a virtual function, provided in the form of Complex Event Processing rule or a Neural Network. The new function should be with no effort be exposed over an existing VO of that device. The concept of the VO will in this regard maintain the reality in terms of functionalities available in virtualized environments and play the role of the digital twin in the edge part of the continuum.

Intelligence at both IoT device and VO level will be demonstrated, taking the advantage of decentralized AI and TinyML techniques. Decentralized AI can be applied for moving intelligence and learning at both VOs and IoT devices, while TinyML can support models that run on small, low-powered devices like microcontrollers and enable low-latency, low power, and low bandwidth model inference at edge devices. This approach avoids bottlenecks caused by placing all the intelligence in a centralized Smart Building monitoring and control system. It also enables creation of complex virtual objects, which allows adding and processing several sensor data streams, optimizing the processing of data, and providing the added-value services such as for example the maintenance of energy-efficiency and well-being in buildings.

5.4. Use Case #4: Remote healthcare services

The current ultrasound medical imaging processes are constrained by both the technical features of the local device and the knowledge of the (local) healthcare operator performing the examination. In fact, Electronic Health Record (EHR) processes are currently bound to on-premises dedicated hardware/firmware components to fulfil the need of a real-time or an almost real-time execution of the process. As a result, acquisition costs/capital expenses are very high and limit the degrees of flexibility in upgrading the hardware and, consequently, the types and number of functions that can be (locally) provided. Functions refer to those EHR processes that elaborate ultrasound data to provide the operator with additional qualitative information (often visualized over coloured overlay images over the black and white video) or quantitative data (spatial measures, pattern identifications, etc.).

The goal of the UC4 is to connect, and somehow to decompose and virtualize ultrasound medical imaging systems into the cloud-edge continuum to lose any barriers due to the hardware capabilities and localization of current physical systems.

As depicted in Figure 15, by exploiting and leveraging on 5G and IoT technologies, the idea is to transform the ultrasound acquisition hardware and the medical imaging viewers into smart wireless-connected "things", that can be "plugged and played" through the cloud-edge medical imaging application: the essential functions of the ultrasound system, with the sole exception of the probe and the input/output devices (such as monitors, keyboards, etc.), must be dematerialized and migrated to the cloud/edge.

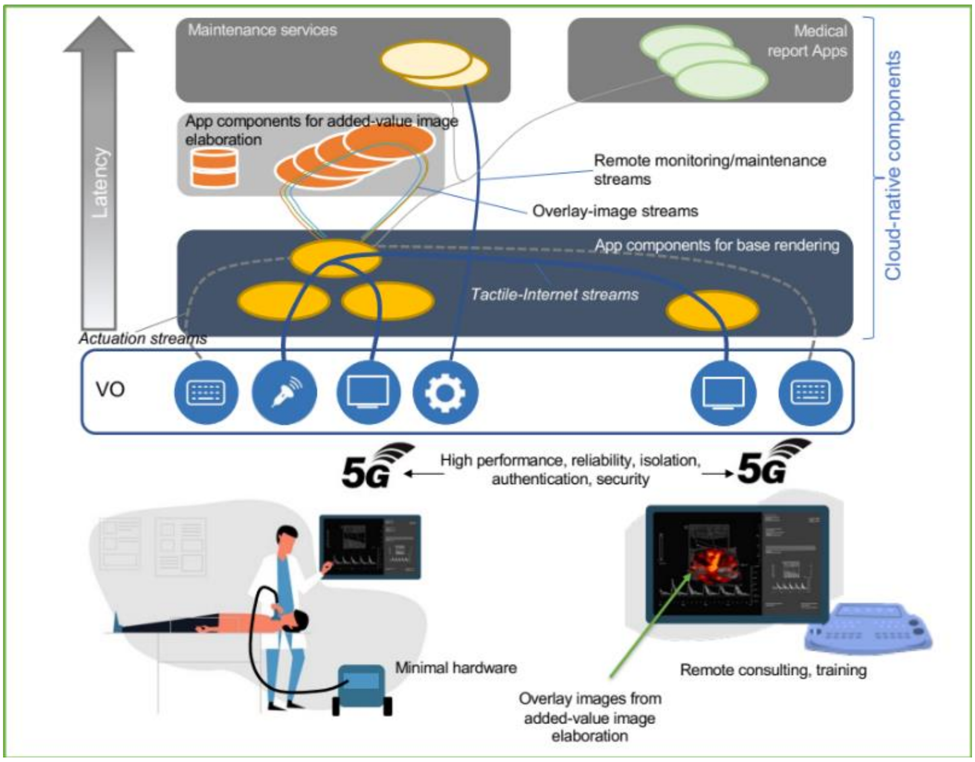


Figure 15. Overall architecture of the e-Health use-case

The ultrasound image processing currently involves a probe, image acquisition hardware, several local software functions devoted to the actual image processing and a monitor for displaying the images. The probe is a passive element cabled to the acquisition hardware in the device. The same device locally performs base and advanced image processing (typically using embodied GPUs) and renders the results on the local monitor. Since the image should have a high medical-grade resolution and that the whole imaging process is very complex, as it requires high volumes of data to be processed with strict latency (to react to human/operator-driven actions) and security constraints, present-day systems heavily rely on HW/FW tools. The decomposition of such systems into the cloud-edge continuum encompasses: a) the management through the Virtual Object stack of the connected physical acquisition and rendering devices, b) the possibility to "plug and play" physical systems (by means of their VOs) into different instances of the ultrasound medical app to execute visits even by involving remote operators (with their monitors and keyboards), c) the possibility of smartly manage (as-a-Service and at runtime) the processes for added value qualitative/quantitative analysis, medical reporting, and hardware maintenance.

As shown in Figure 15, the virtualized components, along with additional processes, will be deployed across the edge-cloud continuum, depending on the strictness of their time requirements. In more details, the base ultrasound image acquisition and visualization will become likely a tactile Internet application, and as such its proximity to the physical acquisition/rendering system will be crucial to provide the needed reactivity to the operator actions. On the other hand, the processes related to the medical reports' generation are less time-critical and can be deployed in the cloud. Between these two categories of applications, the edge-cloud continuum is accomplished with several services which still have strict latency requirements, but not as in the tactile Internet realm. The most relevant of these applications regard the overlay processes in charge of elaborating the raw images to identify known patterns or perform measurements, which are heavily based on Machine and Deep Learning (ML and DL) techniques.

Finally, the current system capabilities will be improved by pairing the physical components and, potentially, some of the virtualized functions, with a VO. Such digital counterparts will support and extend the capabilities of the IoT devices as well as helping with the interplay of physical and virtualized processes, for example, by adapting the image coding to the monitor resolution, providing data pre-processing, by managing caching, etc.

There are several possible advantages considering the goal described in the previous section. In particular, the benefits can be grouped in different categories based on the possible beneficiaries.

From the point of view of the clinical device manufacture (e.g., ESAOTE) there is a dramatic cost reduction: no plastics, mechanical boards, spare parts. The focus is on the software transducers and high parallel computing network. There is also a dramatic reduction of transport and installation cost and service management and maintenance. An additional benefit could be a less environmental impact.

Instead, from the point of view of the clinical staff and the hospital there is the possibility to use always up-to-date equipment with the support of remote control and diagnosis. This solution also allows a space reorganization.

Finally, there are benefits regarding the reduced time for reporting and training for the clinical staff and the end-user (in this specific case: the patients).

In the following we present an overview of UC4, whereas the extended description is presented in the Appendix.

Case Study: Ambulance in a Rural Environment (ARE)

In this UC, the technologies and solutions will be tailored for a 5G-enabled Ambulance in a Rural Environment (ARE) with the support of the mobility. Nonetheless, these can be adopted to a series of other similar scenarios where the mobility is involved, or where the environment has some connectivity limitations.

In Figure 16, we summarize the main stakeholders, the location (physical or virtual) and the constraints, challenges, and risks for this case study.

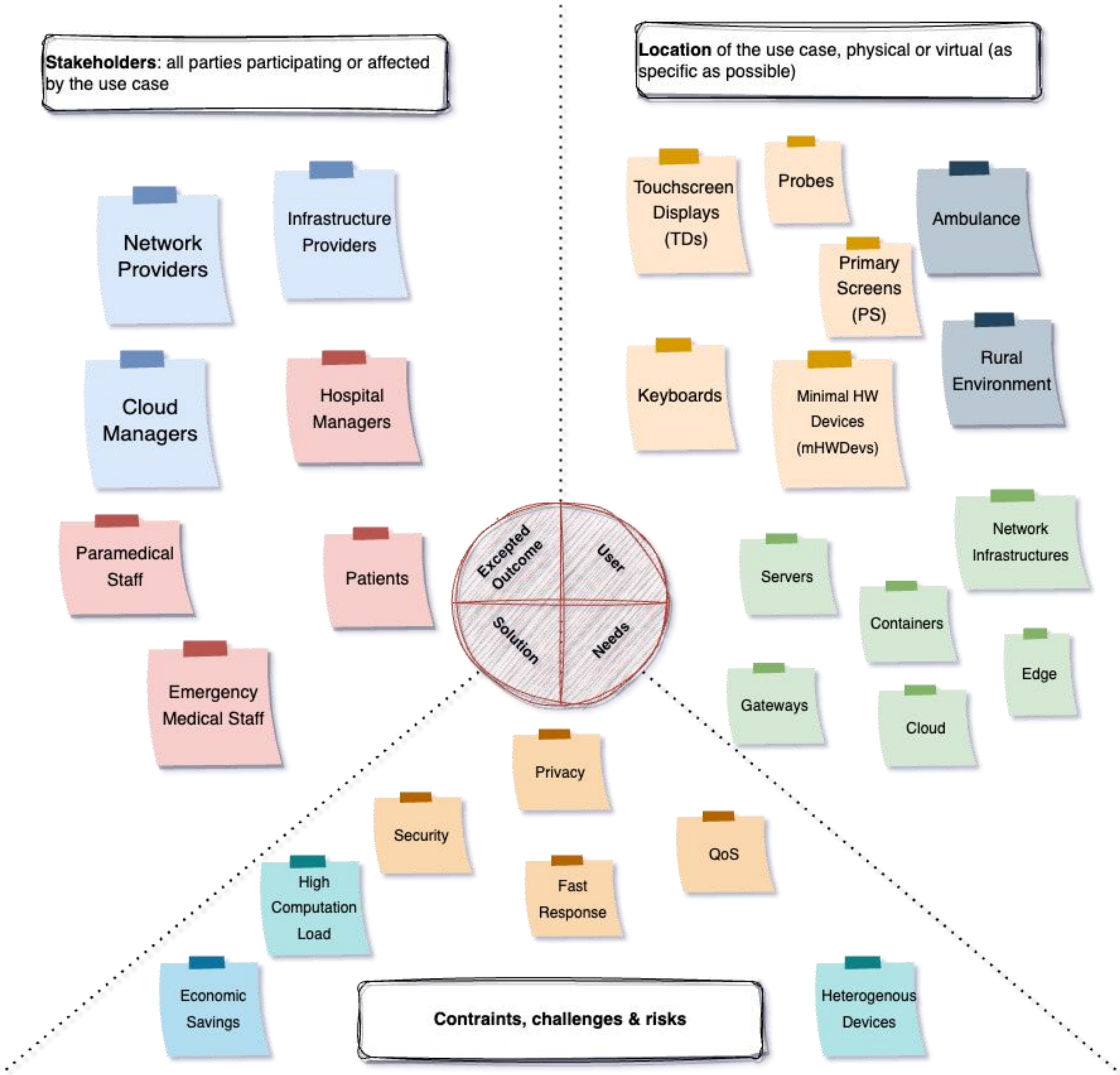


Figure 16. UC4 - Stakeholders, location, constraints, challenges and risks for the UC4

Several stakeholders are involved in an ARE scenario in focus in this UC. These range from the paramedical and the emergency medical staff to the network and infrastructure providers. All of them may be categorized as belonging to the medical staff or the ones involved to the infrastructure management. Besides them, the main actor in focus for this UC is the paramedical and emergency medical staff. For instance, the paramedical staff can use the dematerialized ultrasound system inside the ambulance consisting of the probe and the data processing part for local elaboration. Due to the 5G connectivity, the data obtained using the probe can be elaborated with further and advanced analysis in the cloud. Besides the hardware, the hospital defines the logic of a EHR application to be deployed and executed over the NEPHELE platform.

The application logic is represented as a Hyper Distributed Application (HDA) graph which will be available on the NEPHELE repository. The application logic will define the high-level goal and the Key Performance Indicator (KPI) requirements for the application. To run and deploy the HDA represented by the graph, some input parameters will be given. The application graph will require the deployment of one or more VOs/cVOs to represent IoT devices like probes, a Minimal HW Device (mHWDev) for Local Processing (LP), a touchscreen display, and one or more generic functions to support the application. These will support the EHR operations with scanning, processing, and displaying capabilities. The VO description required by the EHR HDA graph will be available on the NEPHELE HDAR.

The data processing part for local elaboration will be ready to be used with some basic software components running. For instance, this component already has Operating System (OS) installed and correctly set up, with some basic applications already running. Once the network connectivity is established the VO/cVO configuration will also enable some device management features to start and configure components on the devices and orchestration of software components according to the specific task to be executed over time. The paramedical staff will then use the physical devices and the HDA to guide them in their mission and benefit from the enhanced situational awareness offered thanks to the NEPHELE platform for the specific UC.

The main physical location for the study case is the ambulance in a rural environment. The ambulance is connected using the 5G to the central hospital where the emergency medical staff can make remote support with advanced analysis useful as feedback for the paramedical staff. The rural environment increases the complexity of this scenario, adding some possible limitations for the connectivity. In this regards the data processing elaboration can integrate the remote advanced analysis ensuring in any case an answer even when communication is not enough.

Figure 17 shows hardware is used for the scope. The probe is a passive element cabled to the acquisition hardware in the device. It locally performs base and advanced image processing (typically using embodied GPUs) and renders the results on the local primary screen.

- A **Touchscreen Display (TD)** to control and configure the probe.
- A **Minimal HW device (mHWDev)** for applying preliminary and Local Processing (LP) of the collected data stream.
- A **keyboard** to control and configure the probe.
- A **Primary Screen (PS)** to visualize the analysed and processed image data.

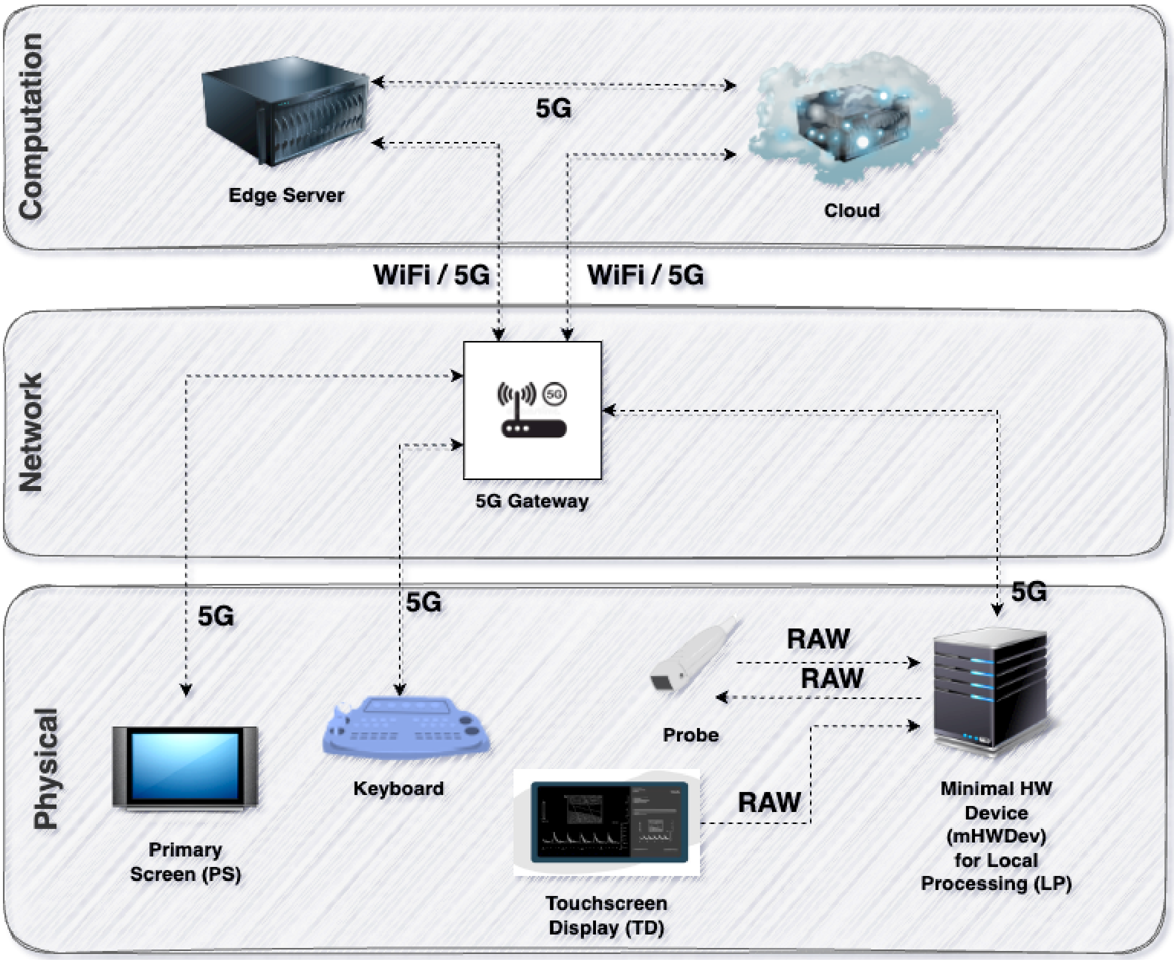


Figure 17. UC4 - Devices at the physical, networking and computation levels

Technical requirements and challenges

The distribution of ultrasound medical system into different application components in the cloud-edge continuum poses several challenges related heterogeneous performance levels required by the different functions (falling from "Tactile Internet" requirements to ones generally provided by current cloud systems), and to the way data is treated. Data security is of paramount importance for medical processes and so is the need of a real-time or an almost real-time execution of the process.

To best face the needs of the EHR operations in the use case above and offer solutions to reach the overall goal for the solution we can identify the following main technical requirements and challenges.

- **Orchestration of software components.** Given the EHR application graph a dynamic placement of software components should be enabled based on service requirements and resource availability. This will require performance and resource monitoring at the various levels of the continuum and dynamic components redeployment.
- **Device Management.** Some application functionalities can be pre-deployed on the devices or at the edge. The device management should also enable bootstrapping and self-configuration, adding and removing devices on the fly, supporting hardware heterogeneity, and guaranteeing self-healing of software components.
- **Low latency communication:** Communication networks to/from a rural environment towards the edge and cloud should guarantee low delays for fast response under mobility conditions and possible disconnections.

- **High bandwidth for edge/cloud:** The data collected from the probe and after some pre-processing with the mHWDev should be sent to the edge/cloud for advanced analysis and to obtain additional diagnosis from remote and skilled operators.
- **Smart data filtering/aggregation/compression**: Large amount of data is collected from the probe. A part of this data can be filtered, other ones can be down sampled or aggregated before sending it to the edge/cloud using the mHWDev for LP. Smart policies should be defined to also tackle the high degree of data heterogeneity.

Application Graph Specification

For this use case as well, the graphical representation of the application graphs is detailed in the Appendix with three different levels of detail. Three different application scenarios are identified to manage the various challenges and objectives for the study case as detailed next.

Real Time Cloud Elaboration

This scenario refers to the application components and services to provide real-time elaboration of the data collected the different devices (probe and keyboard). The collected data using the probe and some command sent with the keyboard are sent to the cloud for additional elaboration. The mHWDev is responsible to make preliminary elaboration.

Four different VOs should be deployed for the following IoT devices: PS, mHWDev, TD, and the Gateway (GW) that is used to send the command from the Keyboard to the cloud. The Keyboard and the Probe devices are not directly connected to the network, and, for this reason, a specific VO is not required.

A network connection fulfilling data rate and latency requirements for data streaming is required between the mHWDev and NEPHELE through the corresponding VO to send the data and process them. The communication between physical devices, the virtual counterparts at the VO and the other application components is enabled through the Zenoh protocol (some data communication can be integrated using HTTP and REST implementation). Some service will be running on the physical devices, whereas other on the edge and cloud continuum and will have to be configured through the VO.

The mHWDev is responsible to decrease the amount of data sent to the cloud for the processing. In addition, it includes local storage to save the data in case the connection is lost and should be sent when the connection is recovered.

Remote Support

This scenario refers to the application components and services to provide remote support for maintenance, tutorial, and training activities. The elaborated date is accessible in Real-Time using a dashboard on a Web Interface.

Like the previous application scenario, a VO should be deployed the following IoT devices: PS, mHWDev, TD, and the GW.

A network connection assuring data rate and latency requirements for data streaming is required between the Minimal HW Data Processing and NEPHELE through the corresponding VO to send the data and to process them. The communication between physical devices, the virtual counterparts at the VO and the other application components is enabled through the Zenoh protocol. Some service will be running on the physical devices, whereas other on the edge and cloud continuum and will have to be configured through the VO.

The Web interface includes a Dashboard that allows the following remote operations: monitoring, alerting, and replaying. In addition, with the Dashboard is it possible to make further remote elaboration for advanced analysis.

Off-Line Remote Consultation

This scenario refers to the application components and services to provide a data storage needed to perform off-line consultation of the elaborated data using a Web Interface. The web interface makes possible to program further elaboration that can be useful for maintenance, tutorial and training activities in a similar way of the previous application scenario.

A network connection fulfilling data rate and latency requirements for data streaming is required between the mHWDev and NEPHELE through the corresponding VO to send the data and process them. The communication between physical devices, the virtual counterparts at the VO and the other application components is enabled through the Zenoh protocol. Some service will be running on the physical devices, whereas other on the edge and cloud continuum and will have to be configured through the VO.

The Data storage includes a Time Series DB that allows to view the history of the elaborated data that can be used to simulate a stored cases useful to make additional analysis and elaboration.

5.5. Consolidated requirements from NEPHELE use cases

The extended description of the four use cases is presented in the Appendix. There a detailed analysis of data processing requirements, functional requirements, non-functional requirements, and system requirements is reported for each use case (UC).

Data-processing requirements (DPR) typically fall into two classes: system-oriented and user-oriented. System-oriented requirements measure the amount of information that your systems process. By contrast, user-oriented requirements measure the impact of data-processing services on the user. Service-level agreements reflect these expectations of performance. A functional requirement (FR) defines a function of a system or component which describes a particular behaviour, that is what the system should be able to do. Unlike a functional requirement, which defines what the system should do, a non-functional requirement (NFR) specifies how the system should work. Particularly, it defines criteria that judge the operation of a system, such as its performance, availability, etc. Finally, a system requirement (SR) defines the configuration that a system must have to run smoothly and efficiently.

In this section, we categorize the requirements coming from the single use cases and present them as a set of macro-categories that are of core interest for NEPHELE. In Tables 7-10 we will associate to these categories the data processing, the functional, the non-functional and the system requirements respectively from the specific use cases described in the Appendix. Further, a short description is provided and how NEPHELE will address these requirements.

Table 7. Data Processing Requirements by category

Category	Description	NEPHELE related feature	Use Case
Low latency in processing	The system should enable low latency and high bandwidth communications, and high computational power for rapid response on data processing	Cloud and Edge Synergetic Orchestration Compute Continuum Network Management Federated Resource Management Cloud continuum	DPR_UC1_01 DPR_UC1_06 DPR_UC2_03 DPR_UC2_04 DPR_UC2_07 DPR_UC2_08 DPR_UC2_09 DPR_UC3_01 DPR_UC3_06 DPR_UC4_01 DPR_UC4_05

Category	Description	NEPHELE related feature	Use Case
Data privacy and security in storage and communication	The system should guarantee data security and privacy in transmission and storage	Security and IoT Device Management VO Storage Space	DPR_UC1_07 DPR_UC2_10 DPR_UC3_07 DPR_UC4_06
Heterogeneity and dynamicity in data sources and workloads	The system should support and store various IoT data sources with varying workloads	Generic/Supportive Functions Interoperability VO Storage Space Compute Continuum Network Management	DPR_UC1_02 DPR_UC1_03 DPR_UC1_04 DPR_UC1_05 DPR_UC2_01 DPR_UC2_02 DPR_UC2_05 DPR_UC2_06 DPR_UC3_02 DPR_UC3_03 DPR_UC3_04 DPR_UC3_05 DPR_UC4_02 DPR_UC4_03 DPR_UC4_04

Table 8. Functional Requirements by category

Category	Description	NEPHELE related feature	Use Case
Virtualization of IoT Devices	The system must be able to represent IoT devices as extended Digital Twins offering additional features and functionalities	IoT Device Virtualized Functions IoT Device Management	FR_UC1_24 FR_UC2_11 FR_UC3_01-03
Input Data & Processing	The system must be able to receive and process data from IoT devices and the environment	Generic/Supportive Functions Interoperability	FR_UC1_11 FR_UC1_23 FR_UC2_01-02 FR_UC2_06-09 FR_UC3_14 FR_UC4_09-10
Data Storage	The system must be able to store IoT data	VO Storage Space	FR_UC1_12 FR_UC1_17 FR_UC1_21 FR_UC3_12 FR_UC4_02 FR_UC4_06-07

Category	Description	NEPHELE related feature	Use Case
Alerting and Monitoring	The system must be able to monitor devices and networks to trigger alerts when an error on a task occurs or a specific event is detected	Generic/Supportive Functions IoT Device Management	FR_UC1_25 FR_UC2_10 FR_UC3_15 FR_UC4_01 FR_UC4_11
Detection, Tracking & Prediction Result	The system must be able to detect objects and humans and predict future values of associated risks/motion/condition	Generic/Supportive Functions AI models	FR_UC1_02-09 FR_UC1_19 FR_UC1_26 FR_UC1_28 FR_UC2_04-05 FR_UC3_04-05 FR_UC3_09-11 FR_UC3_16 FR_UC4_05 FR_UC4_12
Deployment of on-demand devices/network s/protocols	The system should be able to monitor devices and networks to deploy additional elements when needed	Generic/Supportive Functions IoT Device Management	FR_UC1_10 FR_UC1_20 FR_UC1_22 FR_UC1_27 FR_UC1_04 FR_UC3_13 FR_UC4_08 FR_UC4_13

Table 9. Non-Functional Requirements by category

Category	Description	NEPHELE related feature	Use Case
Robustness	The system should perform correctly in dynamic conditions at all levels of the Cloud continuum	Cloud and Edge Synergetic Orchestration	NFR_UC1_01-02 NFR_UC1_06 NFR_UC1_09 NFR_UC1_15-17 NFR_UC2_04 NFR_UC2_07 NFR_UC2_09 NFR_UC2_11 NFR_UC3_01 NFR_UC3_06 NFR_UC3_09 NFR_UC4_01 NFR_UC4_05 NFR_UC4_08 NFR_UC4_13-15

Category	Description	NEPHELE related feature	Use Case
Privacy and security	The processed information shall be kept private and transmitted security	Security and IoT Device Management Generic/Supportive Functions	NFR_UC1_03-04 NFR_UC2_01-02 NFR_UC3_02-04 NFR_UC4_02-03
Time Response	The system shall be able to respond in a reasonable time. This time will be set by the different Use Cases	Cloud and Edge Synergetic Orchestration	NFR_UC1_10 NFR_UC2_06 NFR_UC2_10 NFR_UC4_09
Accuracy	The system shall be able to have reliable object detection accuracy with pretrained models and high computational capacity	Generic/Supportive Functions	NFR_UC1_11 NFR_UC3_10 NFR_UC4_10
High computational capacity	The system needs to delegate functionalities and computational tasks either vertically or horizontally	Cloud and Edge Synergetic Orchestration Compute Continuum Network Management Federated Resource Management Cloud continuum	NFR_UC1_05 NFR_UC2_03 NFR_UC3_05 NFR_UC4_05
Heterogeneity and adaptability	The system should support the use of heterogeneous nodes, protocols, and systems, whereas adapt to different conditions	Interoperability IoT Device Virtualized Functions IoT Device Management	NFR_UC1_12-14 NFR_UC2_08 NFR_UC2_12 NFR_UC3_11-15 NFR_UC4_11-12 NFR_UC4_16
Efficiency	The system should be efficient in bandwidth usage and energy consumption	Compute Continuum Network Management Federated Resource Management IoT Device Management	NFR_UC1_07-08 NFR_UC2_05 NFR_UC3_07-08 NFR_UC4_07-08

Table 10. System Requirements by category

Category	Description	NEPHELE related feature	Use Case
Computational Power	The system shall have enough computational power and virtualization capabilities to run computationally demanding algorithms	Cloud and Edge Synergetic Orchestration Compute Continuum Network Management Federated Resource Management Cloud continuum	SR_UC1_01 SR_UC1_06 SR_UC2_01 SR_UC3_01 SR_UC4_01
Internet Connectivity	The system shall have a stable and highly performing internet connection	Compute Continuum Network Management Ad-hoc networking	SR_UC1_02 SR_UC1_03 SR_UC2_02 SR_UC2_07 SR_UC3_02 SR_UC4_02 SR_UC4_03 SR_UC4_09
Sensors & Hardware Requirements	The system shall have a predefined number of IoT devices with/or a predefined feature, a predefined number of GPUs, storage capacity and servers/gateway with predefined capabilities	Cloud and Edge Synergetic Orchestration Cloud continuum	SR_UC1_04 SR_UC1_06 SR_UC1_07 SR_UC1_09 SR_UC2_03 SR_UC2_05 SR_UC2_09 SR_UC2_10 SR_UC2_11 SR_UC3_03 SR_UC3_05 SR_UC4_05 SR_UC4_07 SR_UC4_08
Virtualization	The system shall support virtualization and containerized applications	Generic/Supportive Functions	SR_UC1_05 SR_UC2_04 SR_UC3_04 SR_UC4_06
AI models and data	The system shall provide relevant data and pretrained AI models for the use cases	Generic/Supportive Functions	SR_UC1_08 SR_UC2_06 SR_UC2_08 SR_UC3_06 SR_UC4_04

6. NEPHELE Architectural Approach

6.1. Application Graph Specification

A HDA is represented as an application graph consisting of application nodes and interconnection links. The nodes of the graph represent application components or consumed third-party services illustrating part of the business logic of the application. All of them are appropriately linked together in an overlay network, delivering the full application functionality in a synergetic way [30].

Declarative statements accompany the application graph in the form of tags (reflecting a value range for a metric or a set property) or simplified expressions. These statements address nodes and/or links or the whole application and aim to specify the application's objective and constraints regarding its deployment and operation to deliver the required results. An application's objective refers to an optional high-level goal set to pursuit when the application is deployed in allocated infrastructural resources and executed along its lifecycle (i.e., energy efficiency, real time response, high proximity to the users etc.) [30]. An application's constraints pertain:

- to measurable requirements for each node and overlay link mostly specifying quantity and quality demands regarding required resources (i.e., computational power needed for an application component to run, overlay link bandwidth required between two application nodes, Quality of Service level expressing the value range for a set of network metrics, required for an overlay link etc.);
- to properties enabling or setting required characteristics or functionalities for nodes and/or links (i.e., a secure overlay connection, autoscaling support for an application component etc.); and
- to restrictions applying to single node or link or a group of nodes or links, expressed as simplified expressions (i.e., a required exclusive access to an application component, locality specifications for a node or a group of nodes, a collocation demand for a group of nodes etc.). The constraints set are further characterized as hard and soft if they should be satisfied during the application's deployment and operation or if they should be treated in a best effort manner to be satisfied, respectively. The application's optional objective set is soft by nature since during deployment and operation there will be a pursuit for optimizing the associated objective function in the best possible way [30].

In Figure 18, a visual representation of an indicative application graph is provided. In Figure 18, the main application components that compose the application graph are depicted, along with their constraints in terms of requirements, properties, and restrictions. This intent-based declarative definition of the distributed application and associated overlay network is extended in Figure 19 where the application graph is augmented with the inclusion of services that illustrate specified requirements, properties, and restrictions. Third-party services chain structure is revealed while network-oriented functionalities are decomposed to network services required to be activated for the illustration of each functionality (e.g., activation of a network firewall, enforcement of a routing policy, activation of network observability mechanisms). Furthermore, each component in the augmented application graph is tagged with the administration authority (AA) that is responsible for the deployment and operation of each application component. In the computing continuum various providers (compute infrastructure providers, network infrastructure providers, services providers, edge and far edge infrastructure providers) collaborate to make accessible the offerings from each different administrative domain to the application developer and achieve through a negotiation and observability framework a synergetic deployment and orchestration of distributed applications not running under the control of a single authority. In Figure 20 the augmented application graph is partitioned per authority responsible for the deployment and operation of a group of nodes, while connection between authorities is represented with virtual links between Administrative Authorities service offering connection points. The latter are considered as the negotiation and communication points that illustrate bilateral agreed policies towards making it possible for a distributed application, partially operating in different administrative domains, to efficiently meet its objective and satisfy its constraints.

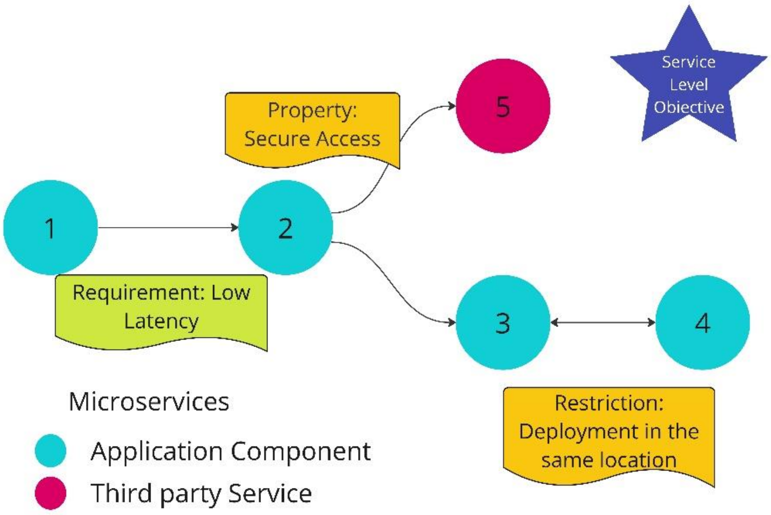


Figure 18. Distributed application graph representation (high level view) [30]

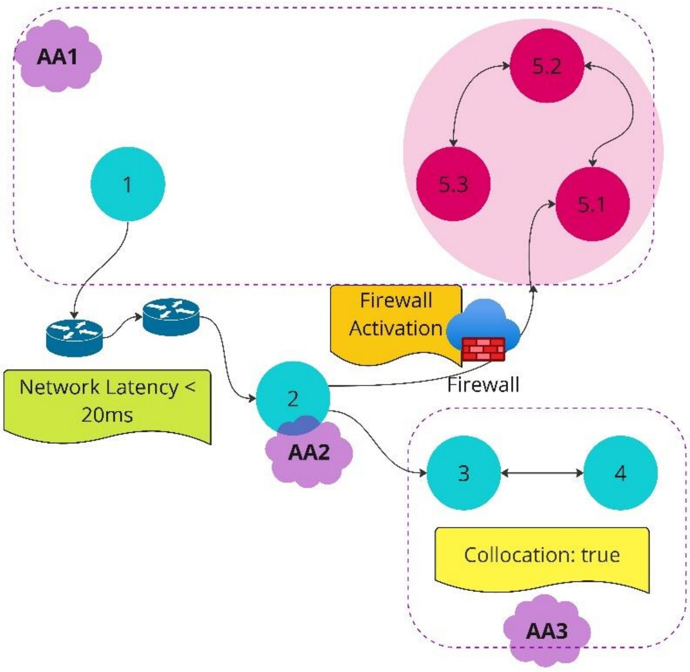


Figure 19. Distributed application graph representation (detailed view with the inclusion of services and partitioned per authority) [30]

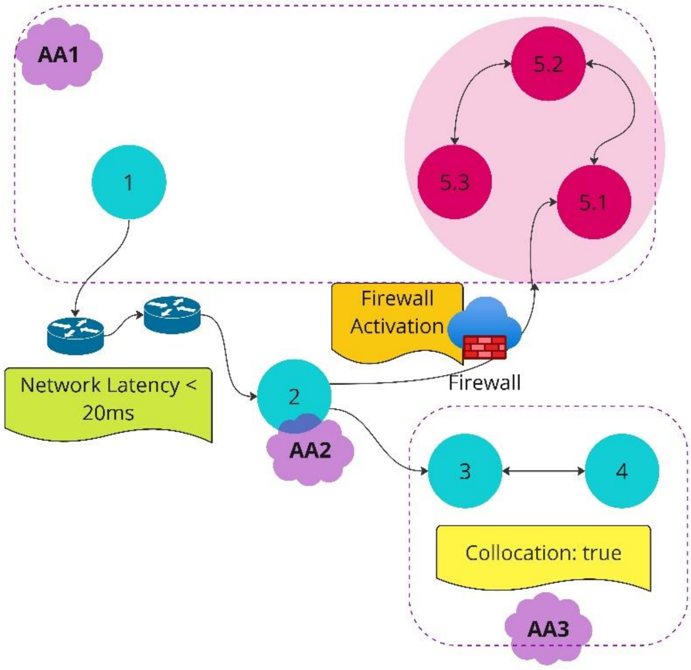


Figure 20. Distributed application graph representation (detailed view with the inclusion of services) [30]

As already noted, the VOs and cVOs constitute part of the application graph. Each (c)VO can be accompanied by a set of requirements, properties, and restrictions. In Figure 21, an indicative IoT application is depicted, where the IoT devices are represented by their virtual counterparts in the edge part of the continuum, while further services are provided through the application business logic (running at the edge or cloud part of the continuum). This application is decomposed in an application graph, as depicted in Figure 22.

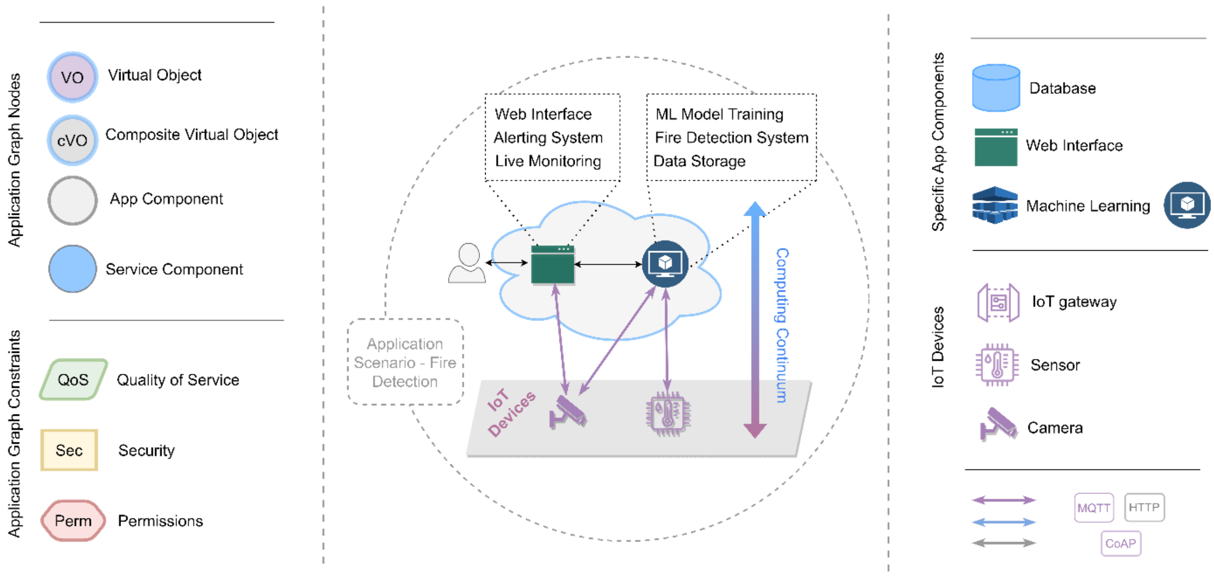


Figure 21. Indicative IoT application

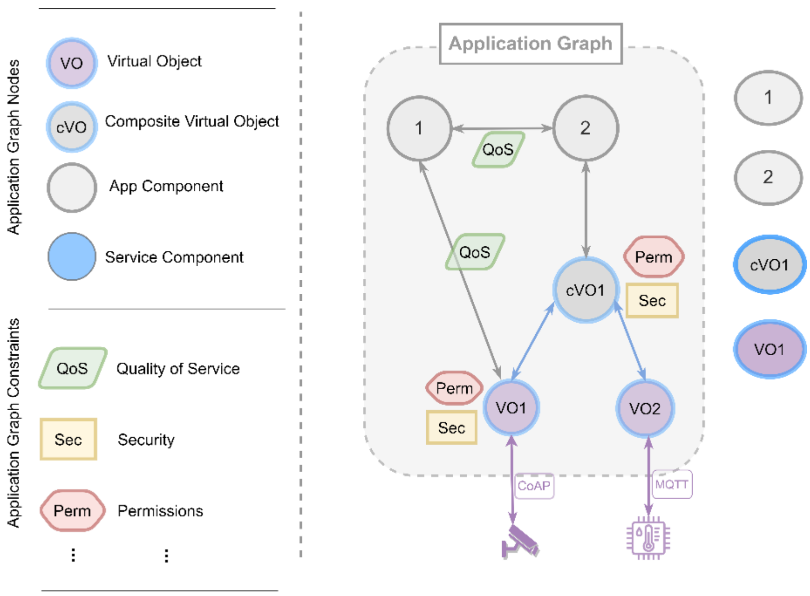


Figure 22. (c)VOs as part of a distributed application graph

6.2. Conceptual Architectural Approach

The proposed architectural approach for supporting intent-driven orchestration of distributed applications in the computing continuum is depicted in Figure 23. The top-level entity is the Meta-orchestrator that is responsible for accommodating a deployment request for a distributed application. The deployment request includes the desired intent on behalf of the application provider in the form of declarative statements (objectives, constraints detailed as requirements, properties, restrictions). The set of declarative statements is formulated in the form of a Service Level Agreement (SLA) that must be supported during the application lifecycle [30].

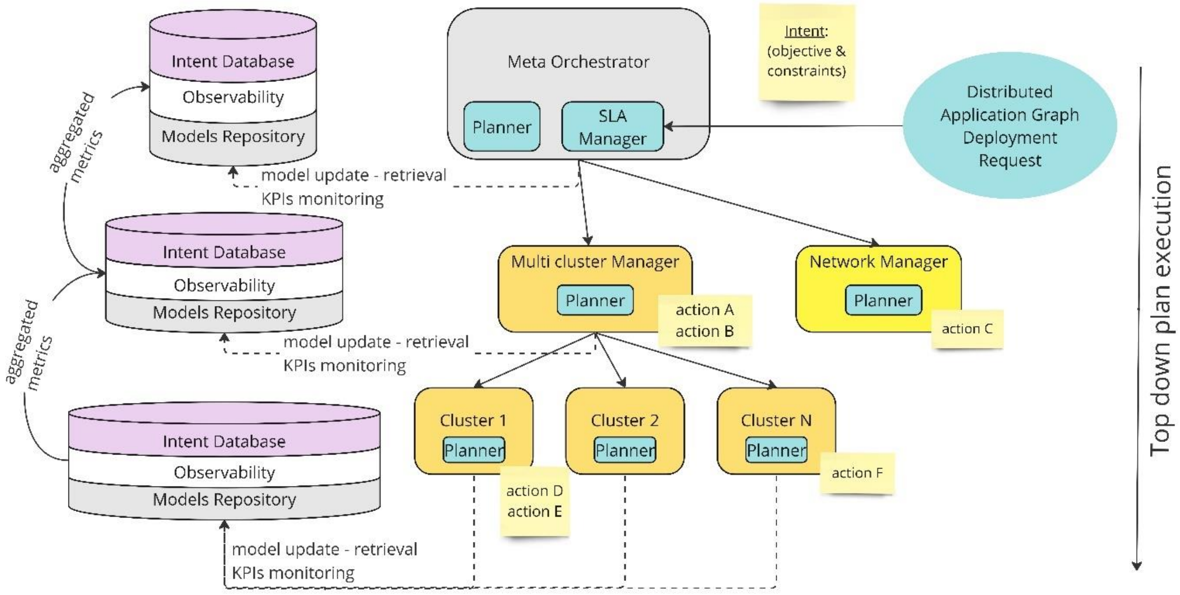


Figure 23. NEPHELE conceptual architecture [30]

The main innovative characteristic of the proposed approach is that it jointly considers the interaction with a Multi-cluster Compute Manager and a Network Manager. These entities must collaborate -under the supervision of the Meta-orchestrator- to support the deployment and the runtime management of the distributed application based on the defined SLA. Different business scenarios can be considered, depending on the interaction between application providers, cloud/edge computing providers and network providers (e.g., telecom operators). The existence of all or part of the stakeholders may be applicable. For instance, if the networking part is considered as a commodity, both the computing and networking part may be managed by the same provider (e.g., the case where a multi-Kubernetes cluster is available where inter-cluster networking mechanisms are applied). In another case, the multi-cluster manager may belong to an edge/cloud computing provider and the network manager to a telecom provider. In each case, a "system of systems" is created, and management responsibilities are assigned to different entities [30].

Continuous monitoring probes are activated for examining the state of the application graph in terms of the agreed SLA for achieving the desired intent. In case of deviations from the desired state, re-configuration actions take place and appropriate requests are triggered and sent to the Multi-cluster Compute Manager and/or the Network Manager.

Hierarchical decision-making processes are applied based on the specification of orchestration control loops and relevant optimization process through a set of planners. To properly implement planning per hierarchical level, it is important to disassemble the high-level intent to a set of metrics that can be monitored at each level of the hierarchy. For instance, an intent for high availability and efficiency can be transformed to an SLA for managing 10 requests per second by a horizontally scalable application component with a service provision time that is less than 50ms. This request can be further translated in the enforcement of a specific scaling rule by the compute managers, as well as the provision of high priority to specific traffic flows in a network link (associated with a virtual application graph link) by the network manager. For the scaling part, metrics such as the average CPU and/or memory usage per container can be examined. For the network management part, the delay, jitter, and packet loss metrics can be examined. Similarly, an intent for privacy can be translated in the deployment of a private 5G/6G network slice by the network manager. The translation rules from an intent to specific actions are made available to the intent database that exists in each level of the hierarchy [30].

To support the continuous monitoring of the various metrics, the development of cloud-native observability tools in the various levels is required. Data fusion of information coming from various signals (e.g., resource usage metrics, QoS metrics, software traces, logs) must be considered [21]. Aggregation of metrics and production of composite metrics (e.g., Key Performance Indicators (KPIs)) is also needed, where aggregated metrics can be provided to higher levels in the hierarchy for assisting decision making processes.

The collected data at each level is provided as input to the Planner component to guide any re-configuration in the runtime management of the distributed applications. Decision making can take place in any level in the hierarchy, while enforcement of actions can be applied in local level or in the lower levels of the hierarchy (e.g., the planner in a parent node can guide the operation of planners in the children nodes). Decision making is supported by the developed models that can be based -among others- on machine learning techniques (e.g., Reinforcement Learning, Multi-agent Reinforcement Learning, neural networks), rule-based management systems, and linear programming solvers. Specifically, the trained models can be available to the planners during deployment time (see the Models repository in Figure 23) and be further trained based on the data collected during the runtime phase of the distributed application. The models can be used for reactive planning (e.g., reaction to an event based on a rules-based management system), opportunistic planning (e.g., consume output of a solver with an objective to reach as much as close as possible to the desired objective) as well as proactive planning (e.g., based on forecasting mechanisms) [14]. Outcomes of the models in one level can provide input to models in higher level [30].

Network Resources Management

Enabling the deployment of cloud native, micro-service-based applications over a 5G and beyond (B5G) network requires to deal with the administrative domain separation between, on the one side, application orchestration and, on the other side, network orchestration, as the former is the responsibility of application service providers and cloud application developers, while the latter is operated by network/telecom providers. In this section, we provide details for the intent-driven network resources management in the case where this is supported by a network/telecom provider. In this scenario, the Operating Support System (OSS) is specifically conceived for simplifying and automating the management of distributed applications onto B5G infrastructures, by mostly hiding the complexity of the 5G environment to application developers and providers [30].

The interaction between the OSS and the meta-orchestrator can be done through the specification of open Application Programming Interfaces (APIs). It should be noted that various initiatives are considering this approach, such as the work in ETSI Multi-Access Edge Computing (MEC) for the specification of traffic management APIs [31] and the 3GPP Common API Framework for 3GPP Northbound APIs [32].

In this respect, we provide details where the proposed approach can be implemented by an open-source OSS [33] (see Figure 24). The OSS is designed according to a highly modular architecture where all the software services are state-of-the-art cloud-native software. The OSS architecture is organized in a suite of five main software services, grouped into two main modules: the North-Bound OSS (NB-OSS) and the South-Bound OSS (SB-OSS). The NB-OSS interacts with the meta-orchestrator. It manages network slice negotiations for distributed applications (Slicing Interface) and maintains metadata (e.g., coverage area served, operational capabilities, etc.) of one or multiple onboarded SB-OSS modules (North-Bound Core service). The SB-OSS is a chain of software services that can be selectively activated to gain access to various programmability levels, passing from a simple catalogue of available resources in case of no programmability, up to the complete terraforming of the physical infrastructure in case of full programmability.

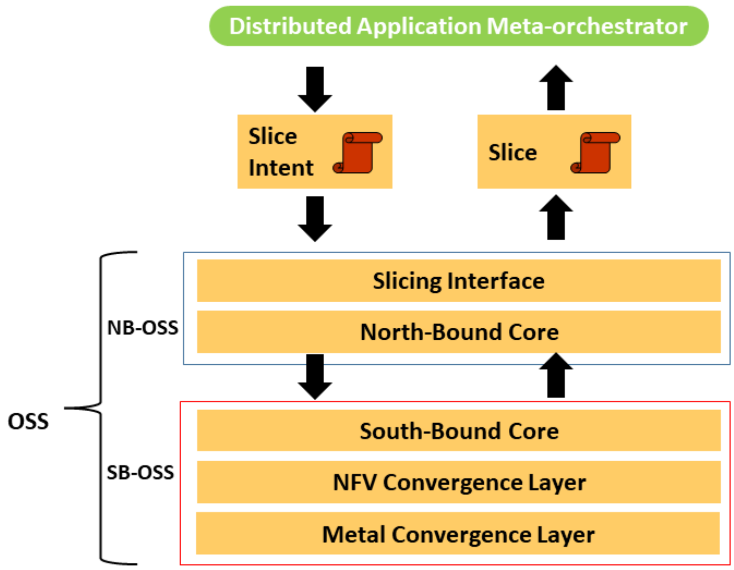


Figure 24. Network Operating Support System [30]

The SB-OSS includes three "chained" services: the South-Bound Core service, the NFV Convergence Layer (NFVCL), and the Metal Convergence Layer (MetalCL). The South-Bound Core service is the only mandatory element in the SB-OSS, and it is devoted to process the slice instantiation/modification/de-instantiation requests and related resources. If the NFVCL and the MetalCL services are available, the SB Core can request to them the setup or the change of new or existing network slices/services and of infrastructure resources. The NFVCL manages the lifecycle of NFV services to provide suitable connectivity to distributed application components in a fully automated and zero-touch fashion [30].

The MetalCL is the service dedicated to manage and terraform bare-metal resources (i.e., physical servers and hardware network equipment) to create IaaS/PaaS environments compliant with the 5G-platform needs.

The Slice Intent mechanism is responsible for requesting the creation of an application-aware network slice, considering the set of declared computational, networking (in terms of network services) and QoS requirements on behalf of the application provider. The deployment procedure of the network infrastructure to support the distributed application has three phases: (i) the transformation of the objectives and constraints of an intent into a network slice intent submitted to the Northbound API of the OSS, (ii) the realization of the application slice by the OSS, and (iii) the deployment of the target application over the created slice by the meta-orchestrator.

The slice intent can describe at which locations the application components should be deployed, what policies and network requirements should be granted for these components for them to interact with each other or with external entities, and to meet desired performance criteria. The requirements for the network connectivity among two individual components are detailed in terms of delay, jitter, packet loss and throughput. The slice intent includes also User Equipment (UE) related metadata, considering the interaction between an application component with a UE (e.g., based on a specific QoS Class Identifier) [30].

Compute Resources Management

We consider the management of compute resources across the continuum where a set of clusters are registered to a multi-cluster manager. The proposed approach is generic and can be applied in any type of orchestration technology; however, in our case we focus on the management of Kubernetes clusters. To support multi-cluster management, various open-source solutions can be adopted, such as Open Cluster Management[8], Karmada[9], Liqo[10], Razee[11]. Such tools can provide unified views of the resources managed by the various clusters. As detailed in the proposed approach, hierarchical decision making is supported where the planner of the multi-cluster manager coordinates the planners of the various clusters across the continuum. Global decision making takes place at the multi-cluster manager planner, while the rest of the planners can take local decisions for their clusters.

The multi-cluster manager planner manages the placement of the distributed application components across the various clusters, aiming to achieve global optimization objectives (e.g., energy usage or cost optimization) and to introduce distributed intelligence characteristics. In case of edge computing scenarios, especially when combined with high mobility patterns, the planner can guide live migration and workload offloading actions across the computing clusters to guarantee specific SLA parameters. It also supports the health check of the application based on the collection of monitoring feeds from the various clusters and service discovery mechanisms.

The planners in the various clusters coordinate local actions to achieve the desired intent state and introduce autonomy characteristics for local decision making. The supported actions may regard scaling decisions (e.g., autoscaling mechanisms taking advantage of machine learning techniques [34], lifecycle

[8] Open Cluster Management, https://open-cluster-management.io/
[9] Karmanda, https://karmada.io/
[10] LIQ, https://liqo.io/
[11] Razee, https://razee.io/

management of the application components (e.g., restart, self-healing actions upon specific events), enforcement of security policies and trusted computing mechanisms, enforcement of load balancing policies and parameterization of the monitoring probes (e.g., activation of probes or configuration of the data collection frequency).

Data Management and Cloud-native Observability

Proper data representation schemas are essential for supporting intent-driven orchestration of cloud applications. They provide a common language for communication, facilitate automation, and enable the integration of diverse cloud resources and services through the exchange of information about the state of resources, dependencies, and other relevant factors. Different interpretations of the data schema may be present at the various levels of the hierarchy.

A relevant data schema to support the description of the intent and its mapping with metrics that are related to the defined requirements, properties and descriptions is provided in Figure 25. A requirement can be represented in the form of an expression and be applied both in an application component (source) or a link between two application components (source and target of the link). The expression includes a metric (KPI), an operator or a range and the corresponding value(s) (e.g., a latency value for a link that is less than 20 ms). In the case of properties, we follow a similar approach, however the value in this case can be a boolean or a compound metric (e.g., horizontal scaling support for an application component with a maximum of 10 replicas {autoscaling:true, max-pods:10}). In the case of restrictions, a rule-based expression is required (e.g., collocation for application component A and B) that can be interpreted by the planner, potentially through the support from an inference engine. The use of a Domain Specific Language (DSL) for a formal description of the requirements, the properties and the constraints that compose the intent can be considered; however, by recognizing that this may be hard in terms of expressivity for the end users [35].

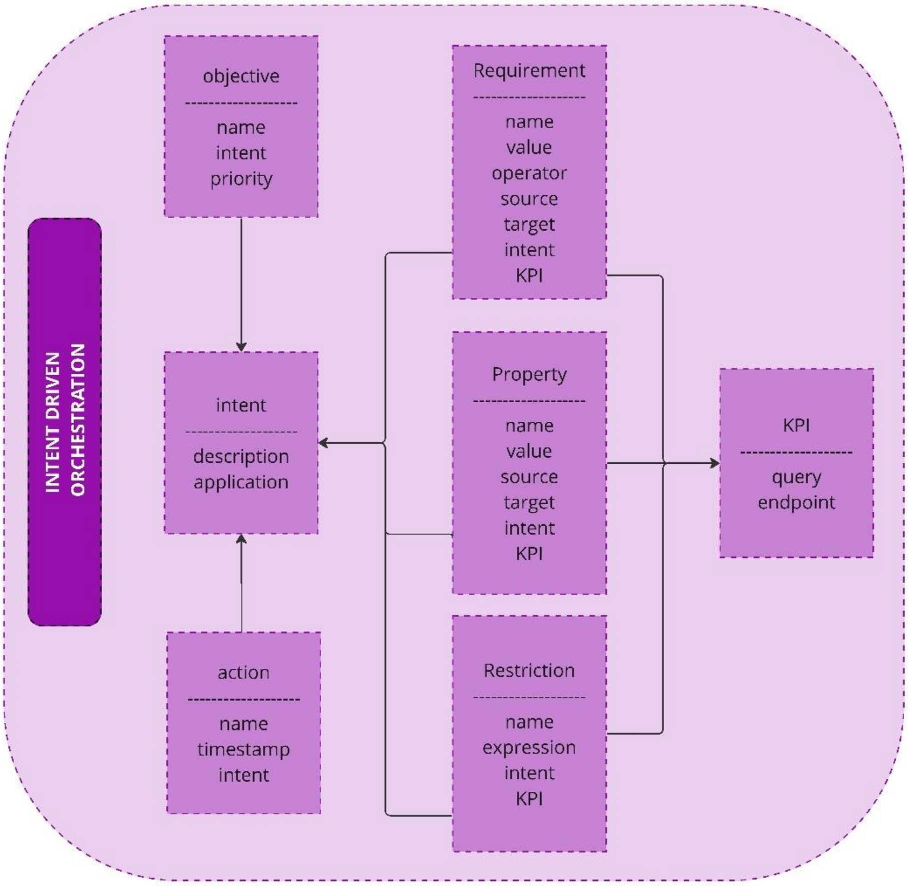


Figure 25. Intent representation scheme [30]

To be able to monitor and evaluate the status of the current state per level of the hierarchy based on the aforementioned metrics and expressions, modern observability solutions have to be adopted aligned with the emergence of cloud-native observability. In this way, queries can be applied by the various planners in the supported observability solutions, while the models in the relevant repository can be continuously trained and/or evaluated. To support cloud-native observability, a relational schema is proposed (see Figure 26 as part of the architecture, extending the work in [21].

The schema is divided into five main sections, namely Application, Infrastructure, Metrics, Traces and Logs. A central aspect represented in the schema is the graph capturing the application components and their inter-dependencies. An application is composed of independent components and each component includes different API endpoints. To represent application workflows, i.e., sequences of API calls between components (endpoints in specific), we use the 'Link' model to define an API call from a source endpoint to a destination endpoint. Components are mapped to infrastructure nodes to provide a real-time view of the application deployment. The schema is especially designed to include real-time observability signals to the application graph construct. Traces represent workflow instances and provide the trace id for the corresponding computation and communication spans, which record the duration of the components' execution (Spans) and inter-communication (CommSpans) accordingly. Metrics and Logs are recorded and attributed to specific endpoints and application components.

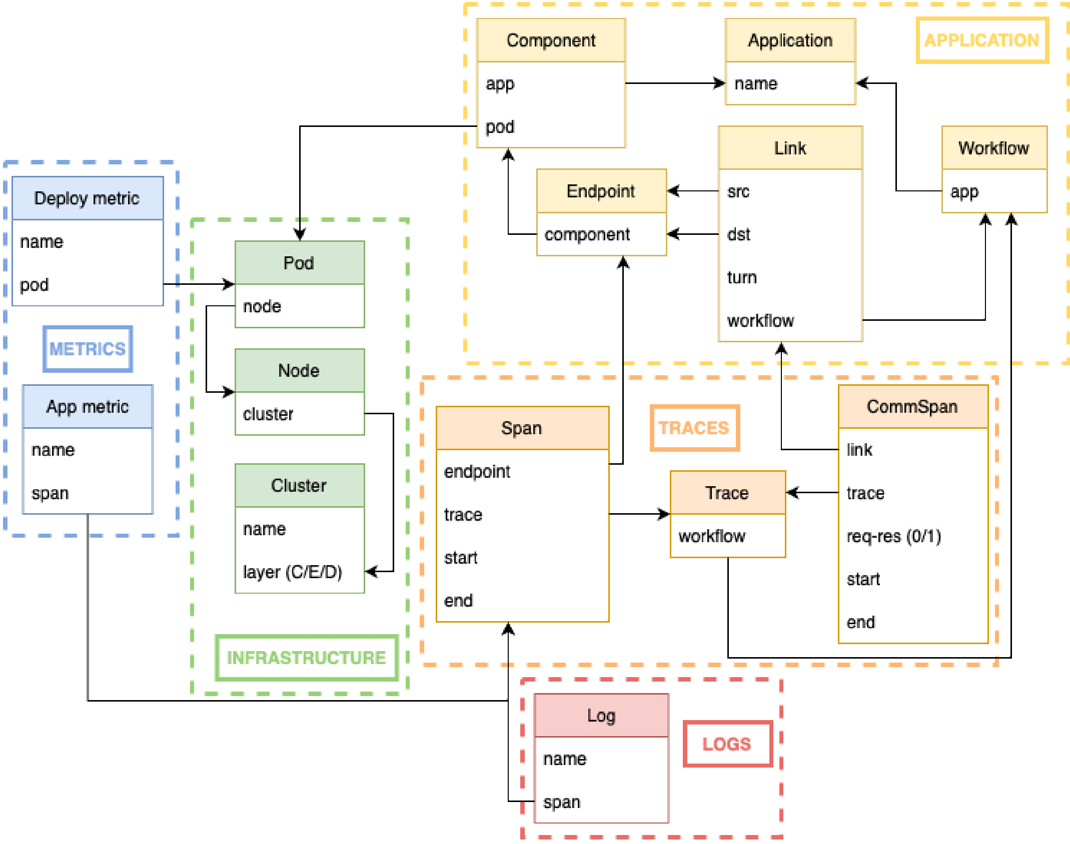


Figure 26. Observability database schema [30]

7. Conclusions

In this deliverable, we have presented the vision of NEPHELE and its contributions to different actors that participate in the NEPHELE ecosystem from a technical and business perspective. The contributions focus on providing an integrated environment for the next-generation HDAR management through two innovations: the first is an IoT and edge computing software stack for leveraging IoT devices' virtualization at the computing continuum's edge. The second is a synergetic meta-orchestration framework for coordinating cloud and edge computing orchestration platforms.

Regarding the IoT and edge computing software stack, we presented the VO as the virtual counterparts of IoT devices to provide a set of abstractions for managing any type of IoT device through a virtualized instance and the interfaces to interact with the computing continuum. We described the VOStack specification and the requirements to support interaction with both physical IoT devices and edge/cloud computing orchestration platforms. The multi-layer VOStack allows the integration of physical, logical, and orchestration aspects to provide a flexible platform for HDA deployment in different industry verticals.

The synergetic orchestrator framework addresses the complexity of managing and coordinating the resources and services of the HDAs while ensuring efficient and effective operation. We showed the challenges in the orchestration domain related to application lifecycle management, efficient resource allocation, interoperability, service level agreement, observability, and network orchestration to raise the need for scalable and efficient orchestration mechanisms. We presented the requirements supported by the different NEPHELE's architecture components, including the goals in terms of reaction time, scalability, high availability, performance monitoring, latency, error rate, saturation, and traffic.

The proposed architectural approach in NEPHELE will be validated, evaluated, and demonstrated through four use cases focused on disaster/emergency management, AI-assisted logistic operations in a port environment, energy management in smart buildings, and remote health care services. We described the scenario of each use case, the involved stakeholders, the technical challenges, the application scenarios, and the interaction between the application components through the NEPHELE architecture. We presented a general categorization of the data processing requirements, functional requirements, non-functional requirements, and system requirements for the use cases in the deliverable, and the reader can find the extended description in the Appendix.

Finally, we presented an overview of the NEPHELE reference architecture based on the VOStack and synergetic orchestrator requirements. We explained the application graph, which visually represents applications' components from different perspectives or levels. Furthermore, we presented the NEPHELE conceptual architecture that jointly considers the interaction between to entities, under the supervision of the meta-orchestrator, to support the deployment and the runtime management of the distributed application based on the defined SLA. The two entities are the Multi-cluster Compute Manager for the resource allocation across the computing continuum, and the Network Manager to deal with the administrative domain separation between, on the one side, application service providers and cloud application developers and, on the other side, network/telecom providers. The detailed description of the NEPHELE reference architecture is presented in Deliverable 2.2.

This deliverable will serve as a reference document to the other WPs and deliverables of the project since it gathers main information about the NEPHELE's system requirements and the use cases description.

References

[1] IoT Business News, “State of IoT”, https://iotbusinessnews.com/2023/05/25/34645-state-of-iot-2023-number-of-connected-iot-devices-growing-16-to-16-0-billion-globally-wi-fi-bluetooth-and-cellular-driving-the-market/ (accessed Jun 6, 2023)

[2] J-M. Fernandez, I. Vidal, and F. Valera, “Enabling the Orchestration of IoT Slices through Edge and Cloud Microservice Platforms,” in *Sensors*, vol. 19, no. 13, p. 2980, 2019.

[3] GSMA, “Network Slicing - Use Case Requirements,” 2018. [Online]. Available: https://www.gsma.com/futurenetworks/wp-content/uploads/2018/04/NS-Final.pdf

[4] H. Thompson, “Next-Generation Internet of Things and Edge Computing, Event Report from the Fireside Chat Workshop,” European Commission, CNECT-E4, 2021, [Online] Available: https://ccn.unistra.fr/websites/ccn/documentation/Territoire-Economie-Ecologie/NGIoTandEdgeComputingExecutiveReport-30032021v13pdf.pdf

[5] M. Nitti, V. Pilloni, G. Colistra, and L. Atzori, “The Virtual Object as a Major Element of the Internet of Things: A Survey,” in *IEEE Communications Surveys & Tutorials*, vol. 18, pp. 1228–1240, 2016.

[6] D. Spatharakis, I. Dimolitsas, G. Genovese, I. Tzanettis, N. Filinis, E. Fotopoulou, C. Vassilakis, A. Zafeiropoulos, A. Iera, A. Molinaro, and S. Papavassiliou, “A Lightweight Software Stack for IoT Interoperability within the Computing Continuum,” 5th International Workshop on Intelligent Systems for the Internet of Things, 19th Annual International Conference on Distributed Computing in Smart Systems and the Internet of Things, Pafos, Cyprus, 2023.

[7] L. Atzori, J. L. Bellido, R. Bolla, G. Genovese, A. Iera, A. J. Jara, C. Lombardo, and G. Morabito, “SDN&NFV Contribution to IoT Objects Virtualization,” in *Computer Networks*, vol. 149, pp. 200–212, 2019.

[8] M. Jarwar, S. Ali S, and I. Chong, “Microservices Model to Enhance the Availability of Data for Buildings Energy Efficiency Management Services,” in *Energies*, vol. 12, no.3, p. 360, 2019.

[9] M. Segovia and J. García, “Design, Modeling and Implementation of Digital Twins,” in *Sensors (Basel, Switzerland)*, vol. 22, no. 14, 2022.

[10] H. Kokkonen, L. Lovén, N. H. Motlagh, J. Partala, A. Gonz’alez-Gil, E. Sola, I. Angulo, M. Liyanage, T. Leppanen, T. Nguyen, P. Kostakos, M. Bennis, S. Tarkoma, S. Dustdar, S. Pirttikangas, and J. Riekki, “Autonomy and Intelligence in the Computing Continuum: Challenges, Enablers, and Future Directions for Orchestration,” ArXiv, vol. abs/2205.01423, 2022.

[11] M. Sánchez, E. Exposito, and J. Aguilar, “Implementing Self-* Autonomic Properties in Self-Coordinated Manufacturing Processes for the Industry 4.0 Context,” in *Computers in Industry*, vol. 121, p. 103247, 2020.

[12] S. Dustdar, V. Casamayor, and P. Kumar, "On Distributed Computing Continuum Systems," in *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 4, pp: 4092-4105, 2023.

[13] B. Costa, J. Bachiega, L. Rebouças, and A. Araujo, "Orchestration in Fog Computing: A Comprehensive Survey," in *ACM Computing Surveys*, vol 55, no.2, pp. 1-34, 2022.

[14] T. Metsch, M. Viktorsson, A. Hoban, M. Vitali, R. Iyer, and E. Elmroth, “Intent-Driven Orchestration: Enforcing Service Level Objectives for Cloud Native Deployments,” in *SN Computer Science*, vol. 4, p. 268, Mar. 2023.

[15] L. Bittencourt, R. Immich, R. Sakellariou, N. Fonseca, E. Madeira, M. Curado, L. Villas, L. DaSilva, C. Lee, and O. Rana, "The Internet of Things, Fog and Cloud Continuum: Integration and Challenges," in *Internet of Things*, vol. 3, pp. 134-155, 2018.

[16] J. Bachiega, B. Costa, L. Rebouças de Carvalho, M. Rosa, and A. Araujo, "Computational Resource Allocation in Fog Computing: A Comprehensive Survey," in *ACM Computing Surveys*, 2022.

[17] F. Guim, T. Metsch, H. Moustafa, T. Verrall, D. Carrera, N. Cadenelli, J. Chen, D. Doria, C. Ghadie, and R. Prats, "Autonomous Lifecycle Management for Resource-Efficient Workload

Orchestration for Green Edge Computing." in *IEEE Transactions on Green Communications and Networking*, vol. 6, no.1, pp. 571-582, 2021.

[18] M. Avgeris, D. Spatharakis, D. Dechouniotis, A. Leivadeas, V. Karyotis, and S. Papavassiliou, "ENERDGE: Distributed Energy-Aware Resource Allocation at the Edge," in *Sensors*, vol. 22, no. 2, p. 660, 2022.

[19] A. Ferrer, J. Marquès, and J. Jorba, "Towards the Decentralised Cloud: Survey on Approaches and Challenges for Mobile, Ad-Hoc, and Edge Computing," in *ACM Computing Surveys*, vol. 51, no. 6, pp. 1-36, 2019.

[20] T. Faisal, J. Ordońez, D. Lopez, C. Wang, and M. Dohler "How to Design Autonomous Service Level Agreements for 6G," in *IEEE Communications Magazine*, vol. 61, no. 3, pp. 80-85, 2023.

[21] I. Tzanettis, C.-M. Androna, A. Zafeiropoulos, E. Fotopoulou, and S. Papavassiliou, "Data Fusion of Observability Signals for Assisting Orchestration of Distributed Applications," in *Sensors*, vol. 22, no. 5, 2022.

[22] S. Clayman, A. Neto, F. Verdi, S. Correa, S. Sampaio, I. Sakelariou, L. Mamatas, R. Pasquini, K. Cardoso, F. Tusa, C. Rothenberg, and J. Serrat, Joan "The NECOS Approach to End-to-End Cloud-Network Slicing as a Service," in *IEEE Communications Magazine*, vol. 59, no. 3, pp. 91-97, 2021.

[23] *Zero-Touch Network and Service Management (ZSM): Terminology for Concepts in ZSM*, ETSI GS ZMS 007 V.1.1.1, ETSI, France, 2021.

[24] *Zero-touch Network and Service Management (ZSM); Reference Architecture*, ETSI GS ZSM 002 V1.1., ETSI, France, 2019.

[25] *Zero-touch Network and Service Management (ZSM); Closed-Loop Automation; Part 1: Enablers*, ETSI GS ZSM 009-1 V1.1.1, ETSI, France, 2021.

[26] S. Kuklinski, R. Kolakowski, L. Tomaszewski, L. Sanabria-Russo, C. Verikoukis, C. Phan, L. Zanzi, F. Devoti, A. Ksentini, C. Tselios, G. Tsolis, and H. Chergui . "MonB5G: AI/ML-Capable Distributed Orchestration and Management Framework for Network Slices," in *2021 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*, Athens, Greece, Sep 2021, pp. 29-34.

[27] N. Gist, K. Behrman, A. Deza, and G. Gheorghiu, *Python for DevOps: Learn Ruthlessly Effective Automation*. Sebastopol, CA, O'Reilly Media, 2019.

[28] H. Percival, and B. Gregory, *Architecture Patterns with Python: Enabling Test-Driven Development, Domain-Driven Design, and Event-Driven Microservices.* Sebastopol, CA, O'Reilly Media, 2020.

[29] L. Militano, A. Arteaga, G. Toffetti, and N. Mitton, "The Cloud-to-Edge-to-IoT Continuum as an Enabler for Search and Rescue Operations," *Future Internet*, vol. 15, no. 2, p: 55, 2023.

[30] A. Zafeiropoulos, E. Fotopoulou, C. Vassilakis, I. Tzanettis, C. Lombardo, A. Carrega, R. Bruschi, "Intent-driven Distributed Applications Management over Compute and Network Resources in the Computing Continuum," in *4th International Workshop on Real-life modeling in 5G/6G networks and beyond,19th Annual International Conference on Distributed Computing in Smart Systems and the Internet of Things*, Pafos, Cyprus, 2023.

[31] *ETSI Multi-Access Edge Computing (MEC) Traffic Management APIs*, ETSI GS MEC 015 V2.1.1, ETSI, France, 2020.

[32] N. D. Tangudu, N. Gupta, S. P. Shah, B. J. Pattan, and S. Chitturi, "Common Framework for 5G Northbound APIs," *2020 IEEE 3rd 5G World Forum (5GWF)*, Bangalore, India, pp. 275–280, 2020.

[33] R. Bolla, R. Bruschi, F. Davoli, C. Lombardo, and J.Pajo, "Multi-Site Resource Allocation in a QoS-Aware 5G Infrastructure," in *IEEE Transactions on Network and Service Management*, vol. 19, no. 3, pp. 2034–2047, 2022.

[34] A. Zafeiropoulos, E. Fotopoulou, N. Filinis, and S. Papavassiliou, "Reinforcement Learning-Assisted Autoscaling Mechanisms for Serverless Computing Platforms," in *Simulation Modelling Practice and Theory*, vol. 116, p. 102461, 2022.

[35] M. Bezahaf, E. Davies, C. Rotsos, and N. Race, "To All Intents and Purposes: Towards Flexible Intent Expression," *in 2021 IEEE 7th International Conference on Network Softwarization (NetSoft)*, Tokio, Japan, pp. 31–37, 2021.

A Lightweight Software Stack and Synergetic Meta-Orchestration Framework for the Next Generation Compute Continuum

Appendix: Use Cases Specification and Data Processing Requirements

The Appendix presents the details of the four use cases that will be used to validated, evaluated, and demonstrated the proposed architectural approach in NEPHELE. The use cases domains are disaster/emergency management, AI-assisted logistic operations in a port environment, energy management in smart buildings, and remote health care services.

1 Use case 1 in NEPHELE

1.1 Introduction and objectives

When a natural or human disaster occurs, time is critical and often of vital importance. Data from the incident area, containing the information to guide first response operations and improve the intervention effectiveness, should be collected as fast as possible and with the highest possible accuracy. The main objective is to rescue as many victims as possible in the shortest possible time whereas ensuring secure operations through risk assessment. To this aim, the rescue team needs to 1) deploy network infrastructure and devices for the mission, 2) map the area and locate and identify victims, and 3) assess the damages and comprehend the remaining or upcoming risks to prioritize rescue operations.

The high-level goal for this use case is to enhance situational awareness for first responders. To this aim, data collected in the area is of utmost importance. On the data coming from the Internet of Things (IoT) devices image recognition, AI-powered decision-making, path planning, and other technological solutions can be implemented to support rescue teams. Sensor data fusion can help to provide precise 2D/3D representations of emergency scenarios in real-time, integrating the inputs from multiple sensors, equipment, and actors. Furthermore, all the information that is being extracted from the heterogeneous data should improve the effectiveness of decision-making and emergency response, increasing safety and coordination.

Robotic platforms have features that are highly appreciated by first responders, such as the possibility to generate 3D maps of a disaster scene in a short time. Open-source technologies (i.e., ROS – Robot Operating System) offer the tools to aggregate sensor data from different coordinate frameworks. To achieve this, precise localization and mapping solutions are needed, together with advanced sensor data fusion algorithms. The envisaged real-time situation awareness is only possible through substantial research advancement with respect to the state of the art in cooperative localization, mapping, and perception in emergency environments. The ability to provide information from a single specialized device (e.g., drone streaming) has been demonstrated, whereas correctly integrating multiple heterogeneous moving data sources with imprecise localization in real-time is still an open challenge.

1.2 Case Study: Post-disaster in a container terminal

In this use case, the technologies and solutions will be tailored for a post-disaster scenario in a container terminal environment. In very complex container terminal operations, the risk of work accidents is inevitable and can happen at any time. As an example, in an accident in June 2022 at the port of Aqaba, in Jordan, up to 14 people have died and more than 250 injured after a container exploded as it was being loaded on a ship[12]. The container of 25-30 tonnes contained chlorine and dropped on the deck of the vessel as the wires of the quay crane snapped. Consequently, the container exploded spreading toxic gas around the port and vicinity. Similar accidents, occur quite frequently in conainer terminals due to several causes. A risk assessment on the data of accidents in one of the major container terminals in Indonesia[13] showed indeed that the container fell to the berth when loading and unloading have one of the highest risk values. Traffic accidents, work accidents, fires, environmental causes are among the other highly rated risks in container terminals.

There are several other causes that can lead to severe accidents and disasters. On the one side natural conditions such as heavy rain, storm, earthquakes, floods, and wind can cause containers stacks collapsing or vessels accidents in approaching the terminal. On the other side, workers accidents due to human factors especially due to negligence in operating vehicles and equipment can lead to traffic

---

[12] https://theloadstar.com/deaths-in-aqaba-port-explosion-after-crane-drops-container-carrying-chlorine/

[13] Budiyanto, M. A., & Fernanda, H. (2020). Risk Assessment of Work Accident in Container Terminals Using the Fault Tree Analysis Method. *Journal of Marine Science and Engineering, 8(6), 466. https://doi.org/10.3390/jmse8060466*

accidents being one of the biggest potential risks. Finally, damage of equipment occupies an important place in the common causes of accidents in a container terminal, as was the case of the cited accident in Jordan. All the mentioned causes and accidents are potential factors that make this study case of high interest for containers ports.

To better analyze and define the study case, in Figure 27 we summarize the main stakeholders, the location (physical or virtual) and the constraints, challenges, and risks are identified.

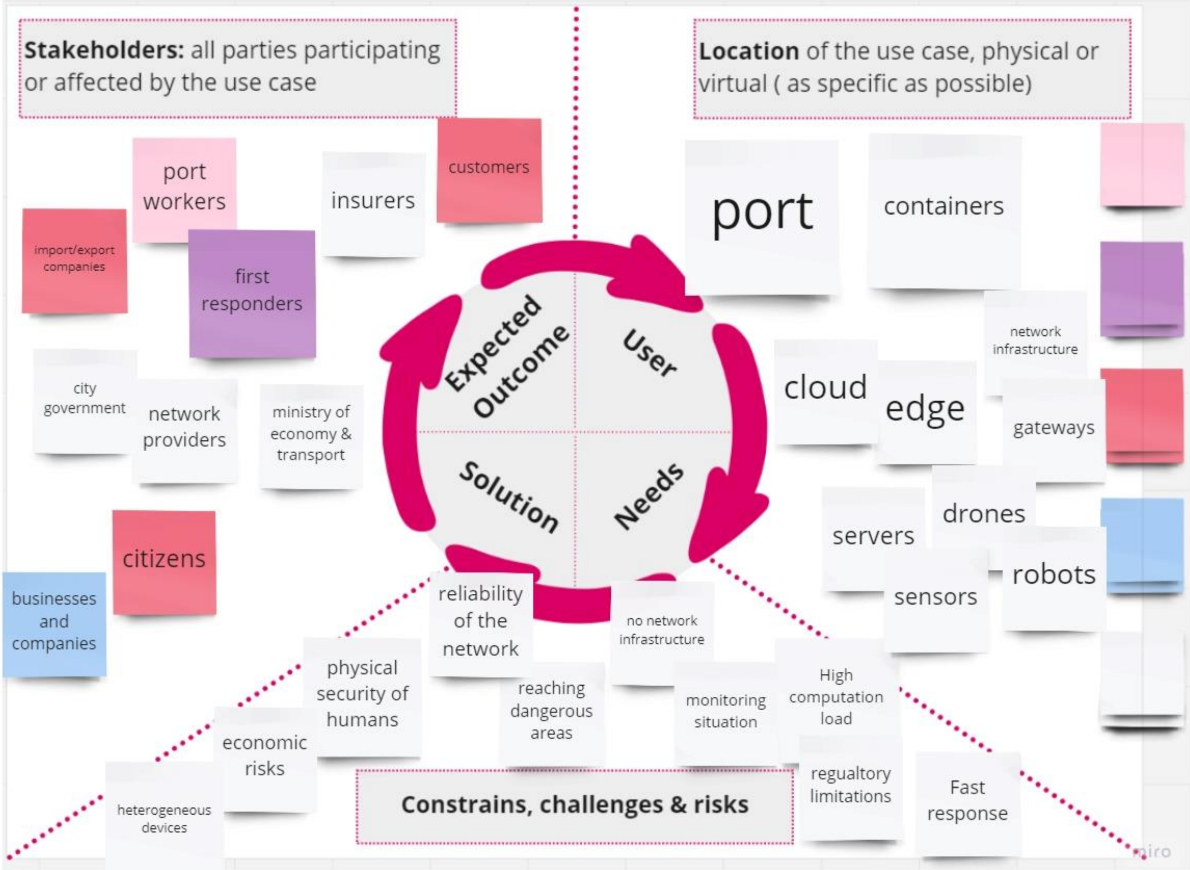


Figure 27. UC1 - Stakeholders, location, constraints, challenges and risks for the use case

Stakeholders

Several stakeholders are involved in the scenario in focus in this study case. These range from the port workers and the business/companies active in the area to the citizens and customers present in the port. All of them may be categorized as potential victims to be rescued or in general persons at risk. Besides them, the main actor in focus for this use case are the first responders. Other stakeholders in this use case may include the city government, the ministry of economy and transport, insurer companies and the general customer. While their general interest is to guarantee safe operations, to mitigate damages, and guarantee rapid rescue operations, they are not directly involved in the first response operations themselves. We will, therefore, in the development of this use case, consider them only for minor aspects such as regulatory aspects.

A firefighter brigade is an example of first responders as the main stakeholder in this use case. Firefighters are typically based in the container port and own a set of physical devices (robots, drones, and sensors). Besides the hardware, the firefighter brigade also defines the logic of a first response application to be deployed and executed over the NEPHELE platform. The application logic is represented as a HDA graph which will be available on the NEPHELE repository. The application logic will define the high-level goal and the KPI requirements for the application. The application graph will require the deployment of one or more VOs to represent IoT devices like robots or sensors and one or more application components supporting the operations with movement, sensing, and mapping capabilities. The VO description required by the HDA graph will be available on the NEPHELE Hyper-distributed Applications repository.

The robotic devices and drones will be ready to be used with some basic software components running. For instance, the robots already have ROS installed and correctly set up, with some basic ROS components already running. Once the network connectivity is established the VO configuration will also enable some device management features to start and configure components on the devices and orchestration of software components according to the specific task to be executed over time. First responders will then use the physical devices and the hyper-distributed application to guide them in their mission and benefit from the enhanced situational awareness offered thanks to the NEPHELE platform for the specific use case.

In Figure 28 we report a detailed view of the main users for the use case with a high-level analysis of the needs, functionalities and expected outcomes for each of them.

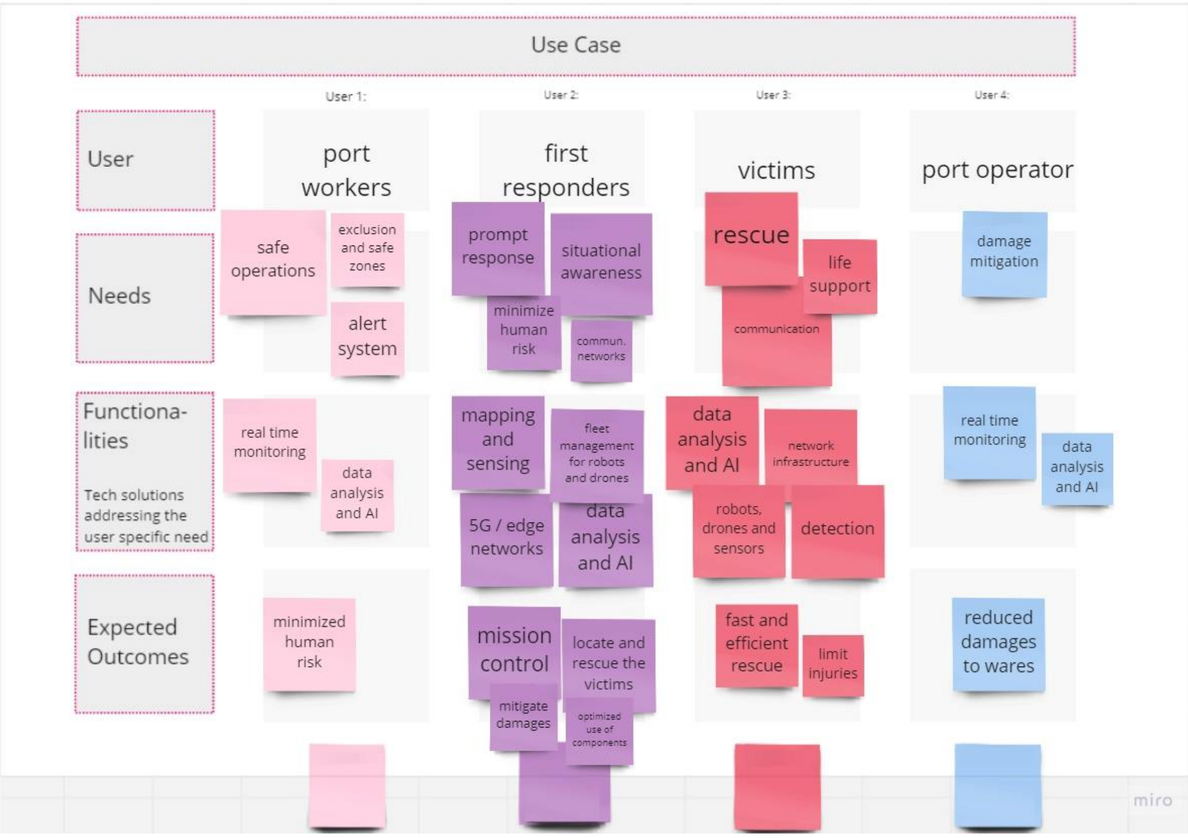


Figure 28. UC1 - Needs, functionalities and expected outcomes for the main users in the use case

Location

The main physical location for the study case is a container terminal. After an accident in a terminal, there might be victims, due to explosions or collapsed containers/equipment/buildings, that need to be rescued or helped, there might be high-risk areas as a consequence of collapsed/damaged containers carrying dangerous materials or due to gas/liquid leakage, there might be lack of networking infrastructure and any available map of the port area may be not usable (or not completely) as the landscape was modified. The implementation of first response operations in this scenario, will require the use of hardware and virtualized locations. The following hardware is used for the scope:

- **Mobile ground robots** that will be used to map the post-disaster area, to monitor the area and approach victims, to deploy sensor networks, to take samples of non-identified liquids leakage, to react to network disconnections by deploying additional nodes.
- **Drones** that will be used to fly over the area and send video streams and pictures from the integrated camera to identify objects/victims/leakages and make an early map of the area of interest.
- **Sensor networks** that are deployed by the ground robots in the post-disaster area are used to monitor the area for potentially dangerous situations through, e.g., gas detection, leakage detection, temperature detection, collapse detection and others.
- **Depth/Thermal cameras** mounted on the robots and the drones for risk assessment, risk prediction in the area, and victim identification and monitoring.

Moreover, networking and computation devices such as a 5G gateway, IoT gateways, edge servers and Wi-Fi routers will be used for this use case (see Figure 29). Part of the computation will also occur in virtualized environments at the edge and cloud using containerized application components.

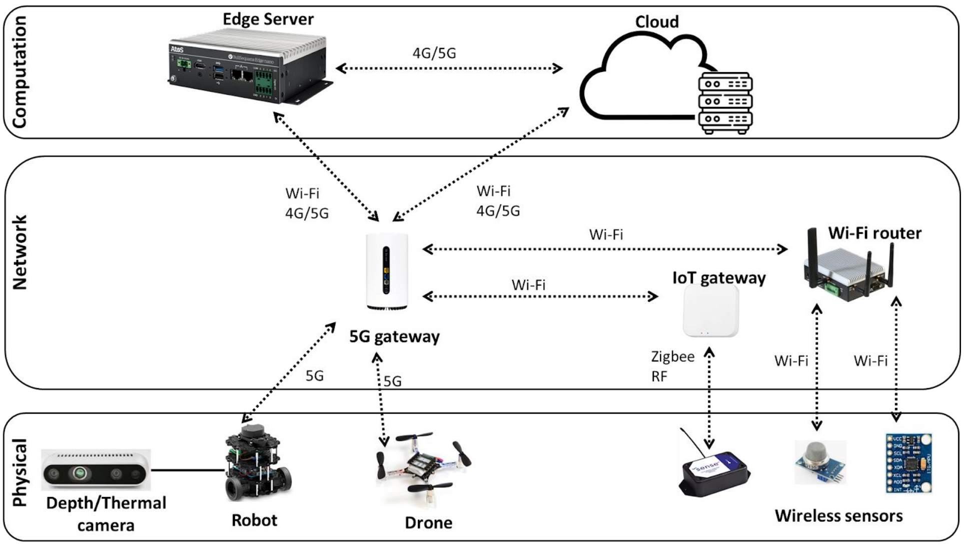


Figure 29. UC1 - Devices at the physical, networking and computation levels

Constraints, challenges, and risks

As reported in Figure 1, the container terminal post-disaster study case carries with it several inherent constraints, risks, and challenges as detailed next.

Risks

- Among the main risks we see the physical security of the humans present in the physical location where the event occurred.
- Economic risks are also linked to a port container environment not only for the direct consequence of the damages caused by the accident but also for the economic consequences for the involved companies and businesses for every minute of inactivity in the port.
- Physically reaching out to insecure areas might be dangerous for first responders.

Constraints

- In a post-disaster scenario typically no network infrastructure is available or not reliable.
- Regulatory limitations may limit the intervention in a post-disaster area. We think here for instance of the use of drones and the use of 5G frequencies that are regulated by national and international regulations.

Challenges

- Fast response is required to reduce the consequences of the disaster both in terms of victims and economic damages.
- To perform advanced analysis of the available data requires a high computation load which cannot be provided by simple physical devices used by the first responders when entering an area.
- Network coverage should be reliable during the whole time of intervention.
- The devices used by first responders are very heterogeneous in nature, such as small sensors to detect gas leakage or vibration, cameras, drones, and complex ground robots. These devices differ in hardware, software and communication protocols used.

1.3 Technical requirements and challenges

There are several technical requirements and challenges for this use case. One of them is the heterogeneity of devices and time strong requirements. Data should be transmitted, filtered, and processed at different levels of the compute continuum to guarantee short delays while maintaining full knowledge of the situation. Therefore, communication technologies and protocols should guarantee low latency. Devices are heterogeneous in terms of CPU, memory, sensors, and energy capacities, some of the hardware (HW) and software (SW) components are use-case specific, while others are common to multiple scenarios (see Figure 30). Different complementary application components can be run on top of the same devices but exploit different sets of data, services, and application components. The network is dynamic because of link fluctuations, energy depletion of devices and device mobility (which can also be exploited when controllable) and this should be dealt with. How to use VOs, where to deploy edge computing for what application in such a context is a tremendous challenge that NEPHELE can address. The orchestration of VOs and their performance are highly related to the hardware that manages them.

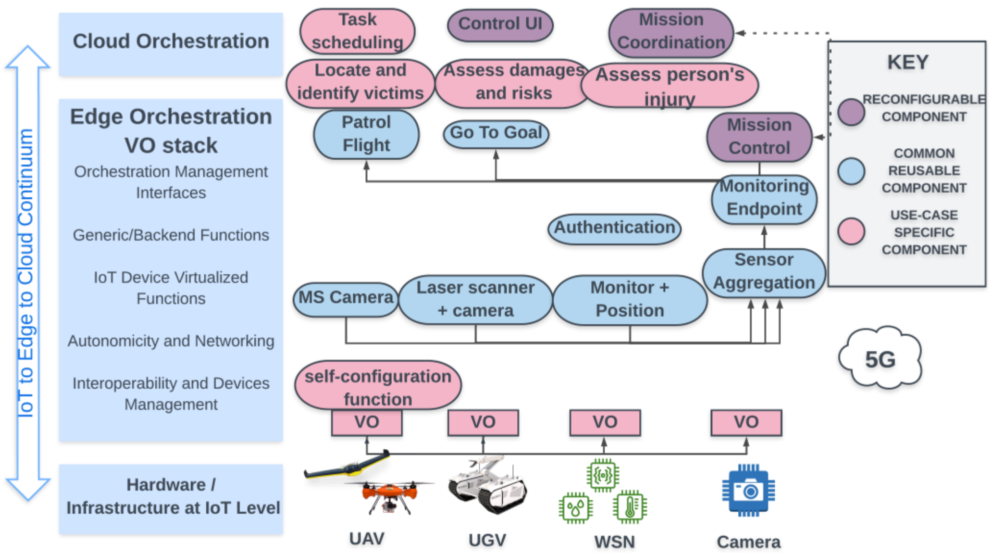


Figure 30. UC1 - VO-Stack mapping to emergency/disaster recovery application scenarios

To best face the needs of the operations in the use case above and offer solutions to reach the overall goal for the solution we can summarize the following main technical requirements and challenges.

- **Orchestration of software components**: given the application graph, a dynamic placement of software components should be enabled based on service requirements and resource availability. This will require performance and resource monitoring at the various levels of the continuum and dynamic components redeployment.
- **Device Management**: some application functionalities can be pre-deployed on the devices or at the edge. The device management should also enable bootstrapping and self-configuration, adding and removing devices on the fly, supporting hardware heterogeneity, and guaranteeing self-healing of software components.
- **Low latency communication**: communication networks to/from disaster areas towards the edge and cloud should guarantee low delays for fast operation in first under mobility conditions and possible disconnections.
- **Dynamic multi-robot mapping and fleet management**: coordination, monitoring, and optimization of the tasks allocation for mobile robots that work together in building a map of unknown environments or executing tasks in a collaborative manner.
- **Computer vision for information extraction**: AI and computer vision enable people/object detection, position detection and localization from image and video data.
- **Smart data filtering/aggregation/compression**: a large amount of data is collected from sensors, robots, and cameras in the intervention area for several services (e.g., map building, scene, and action replay). Some of them can be filtered, others can be downsampled or aggregated before sending it to the edge/cloud. Smart policies should be defined to also tackle the high degree of data heterogeneity.

1.4 First response operations for the study case

The first response operations can be split into the following four main sub-problems.

1. **Deployment of network infrastructure and application software**: The network infrastructure should ensure that the disaster area is covered in terms of connectivity for the duration of the intervention, whereas wireless sensor networks, robots and drones should be deployed for monitoring and risk assessment. The operation base for the first response is placed in a safe zone near the disaster area to host an edge server, a 5G/Wi-Fi gateway and any radio system antenna needed to provide low-latency communication and high computational power close to the ground. Several technologies could be integrated if they are available such as Wi-Fi, cellular, or satellite. In its minimal deployment, the network infrastructure should include a 5G or WiFi network gateway, edge servers for the area and a wireless sensor network for monitoring and risk assessment. Once the networking infrastructure is set up, a drone will be set in place for an aerial view of the area, whereas ground robots are installed in the disaster area. They will be used to map the area from the ground with greater detail of analysis as done with the drone and to deploy the wireless sensor network in the areas of interest.

   The application software to be deployed in the involved devices includes all the components in the application graph and the VO for the hyper-distributed application. The same hardware can be used for different tasks that use different application components over time. Initially, basic components will be deployed at different levels of the continuum. However, with every new task being assigned, different software components should be enabled at different levels of the computing continuum.

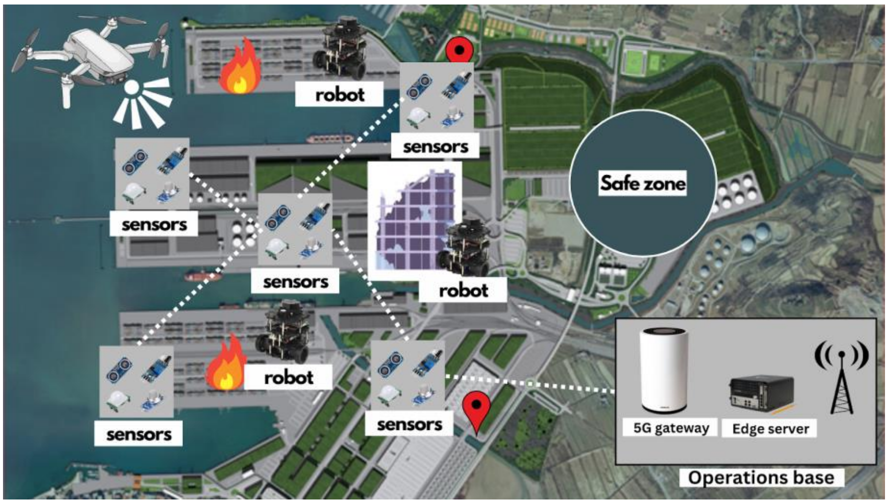


Figure 31. UC1 - Initial deployment of network infrastructure, robots, drones, and sensors in selected areas of the disaster area

2. **Reaction to the dynamic environment:**Once the network infrastructure and the wireless sensor networks are deployed, communication with all devices is to be guaranteed over time. Due to varying conditions and mobility, disconnections can occur. The overall application should be able to monitor these conditions and react accorgingly using and deploying additional devices. To this aim, data about battery level status and connectivity is continuously collected and analysed. Whenever a disconnection risk is detected a redeployment of devices is triggered. In case of additional wireless sensor needed, ground robots will be sent to deploy them to guarantee safe operations in dangerous areas and free the first responders for other tasks.
3. **Data collection and analysis:** Data will be collected by the physical devices and sent to the higher levels of the continuum for further analysis. Some initial filtering, aggregation or elaboration may also occur on the physical device itself to limit the communication burden. The data can be either images, video streaming, sensing data or device management/monitoring data. The data will be used for tasks like object/victim detection, risk prediction and monitoring, network conditions monitoring. The analysis will be supported by pre-trained AI models and functions offered by the VO-stack.
4. **First response operations**: These include all the tasks that are performed for the first response operations once all networking, software and hardware components are in place. These include mapping the area using drones or ground robots, approaching victims or objects, picking/placing objects, taking samples of unidentified liquid leakages, take smart decisions for mission control and optimization. All these operations are based on the analysis of the available data and in case of triggered operations (see points 2 and 3 above).

1.5 First response application components

The HDA will have a classic three-tier architecture with a presentation tier, an application tier, and a data tier.

- **Presentation tier:** the application will offer a front-end for visualization and mission control by the end-user. A mission-specific dashboard will provide real-time situational awareness (i.e., 2D/3D maps with the location of robots, victims, and threats) to take well-informed and confident decisions. The dashboard integrates data coming from heterogeneous sensors and equipment (i.e., drones, mobile robots, sensors) and will be accessible through a web browser or a graphical user interface (GUI) remotely. The dashboard will enable the user to interact with the application tier to take mission decisions and analyse historical data for enhanced situational awareness.

- **Application tier:** the inputs and requests coming from the presentation tier are collected and application components are activated to execute mission tasks. At this level, all application components for supporting the application logic in this use case are included. Some of these components will run directly on the IoT devices, some on the edge and some on the cloud through the interaction with the VO. New data can be produced, and old data accessed from the data tier.
- **Data tier:** this includes a storage element for storing processed data such as images and videos or historical data about the mission. The data produced by the IoT devices (drones, robots, sensors) will be compressed, downsampled and/or secured before being stored for future use by the application tier. The data can be stored on the VO data store or on specific storage components for the application and is to be transmitted from the physical devices to the corresponding VO with low latency.

Several application components will be required for the application. Besides the graph descriptor for the HDA and the VOs descriptors for the hardware elements (robots, drones, and sensor networks), the following software components will be required:

- Interactive GUI for the presentation tier.
- APIs to trigger main application tasks such as GUI update, map the environment, localize victims on a map, perform risk assessment, perform historical analysis of robot actions, send robot/drone to an object/victim on the map, take liquid samples, and pick and place for devices deployment.
- Mission control software component able to list tasks based on data collected by the robots and sensors.
- Fleet management software component to enable and control multiple robots moving around simultaneously.
- SLAM (Simultaneous Localization and Mapping) software component to enable mobile robots to navigate autonomously the area and map it using multiple robots and map merging functionalities.
- Locate and identify victims in unknown areas, assess the victims' injuries and monitor their health based on sensor data analysis and computer vision.
- Risk assessment component to identify and classify the areas based on dangerous and risk elements using sensor data analysis and computer vision (e.g., liquid leakage).
- Storage and replay software component for historical analysis of robot actions and performed tasks.
- Pick and place software components for a mobile robot to deploy sensor devices and take liquid samples.
- Trajectory planner for robot movements for an optimized mission.
- Data aggregation, data filtering and compression for sensor and camera data.
- Monitoring components to verify the status of network connectivity, sensor, robots and drones' status and trigger actions in case of low energy or disconnections.

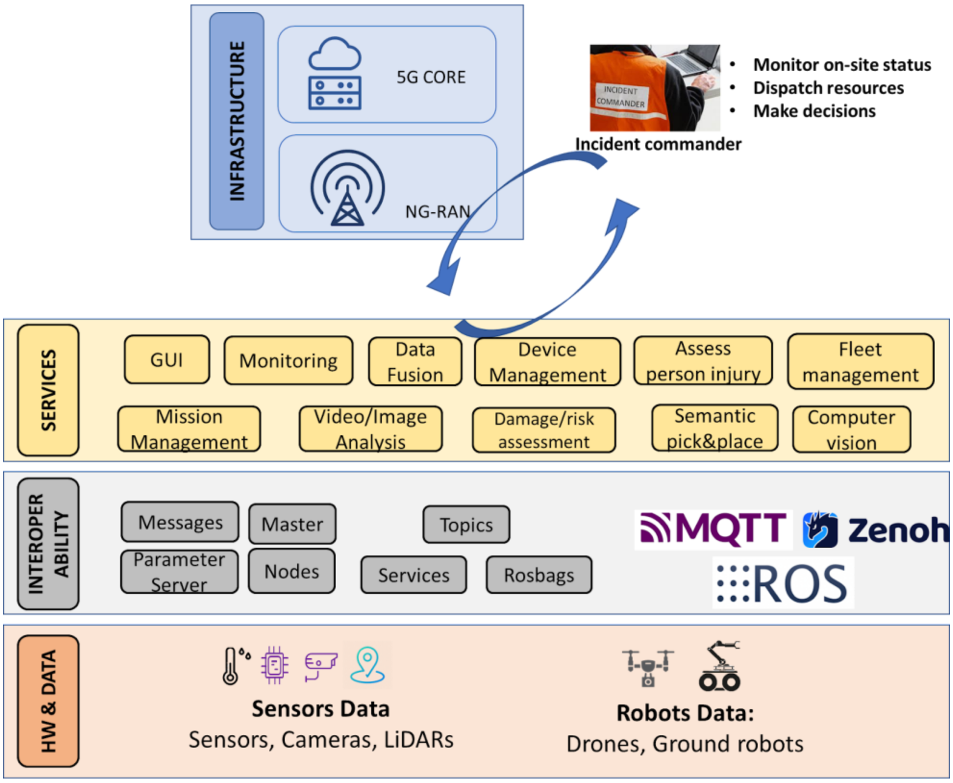


Figure 32. UC1 - Overall hardware, services, and interoperability vision for the application

The overall picture including services, hardware and interoperability layers of the study case is reported in Figure 32. An interoperability layer for communication among robots and sensors is based on ROS, whereas communication protocols like Zenoh or MQTT can be used for communication towards the VOs and among the application components. For each physical device and subtask in the application, different software components will be used and deployed either on the physical device, at the edge, or on the remote cloud. We will make use of application graphs representations in the next subparagraphs to classify the application components and services that will be implemented for different application aspects in this use case. Three different levels of detail are used to show: i) the high-level view of a specific scenario for the application with the involved devices and overall objectives; ii) the application graph reporting the logical application components, the VOs/cVOs and the conditions and requirements for them to communicate among each other and with the identified VOs/cVOs, and iii) the service graph where the single services and the links among them are shown.

Mapping

This scenario refers to the application components and services needed to map a given area using ground robots and/or drones through cameras and lidars. The resulting map and its graphical representation will be used to give the first responder commander a graphical overview of the area and by this enhance his situational awareness. Drones will be used to make an aerial 3D map of the area and detect the condition of buildings and containers, the location of people in safety areas, fires, liquid leakages and their progress, and other potential risks. Ground robots will be used to map the area from the ground in 2D with greater detail of analysis as done with the drone. A VO should be deployed for each ground robot and drone at the edge of the network. A network connection fulfilling data rate and latency requirements for video streaming is required between drone and NEPHELE through the corresponding VO to send the videos and process them. The communication between physical devices, the virtual counterparts at the VO and the other application components is enabled through the Zenoh protocol. Some services will be running on the physical devices, whereas others on the edge and cloud continuum and will have to be configured through the VO. The storage and analysis of the collected data may be demanding, reason for which having the edge/cloud data storage support is important. Nonetheless, robots and drones should have local storage to save the video in case they lose connection and should be sent when the connection is recovered. A fleet management service, a GUI with alerting, monitoring, task control, replaying and dashboards services through a web interface complete the application graph.

In Figures 33, 34 and 35 we represent respectively the high-level view, the application graph, and the service graph for the mapping application scenario.

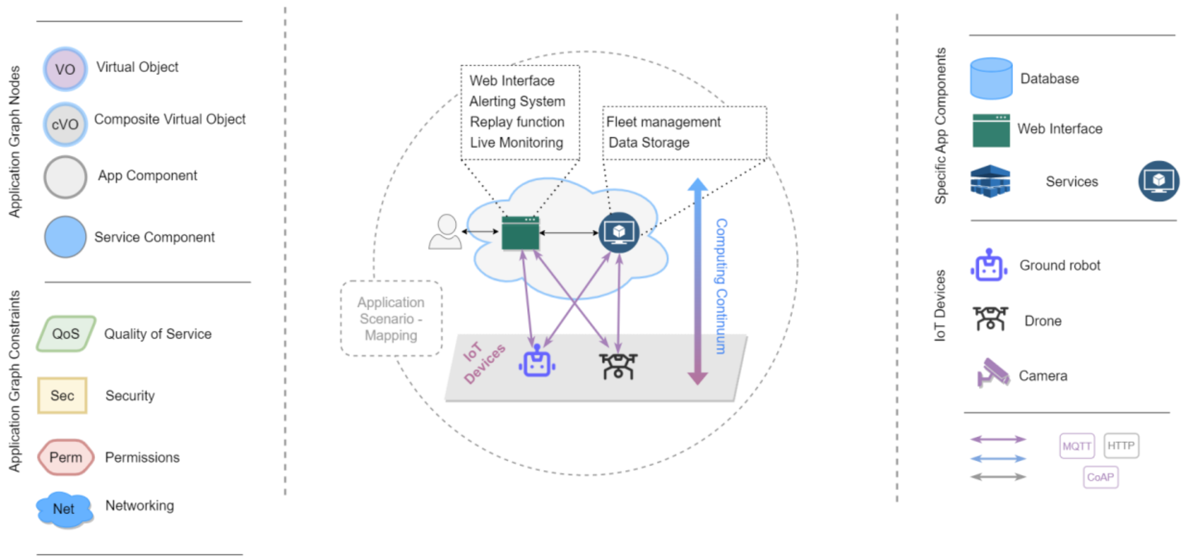


Figure 33: UC1 – Mapping application scenario – High level

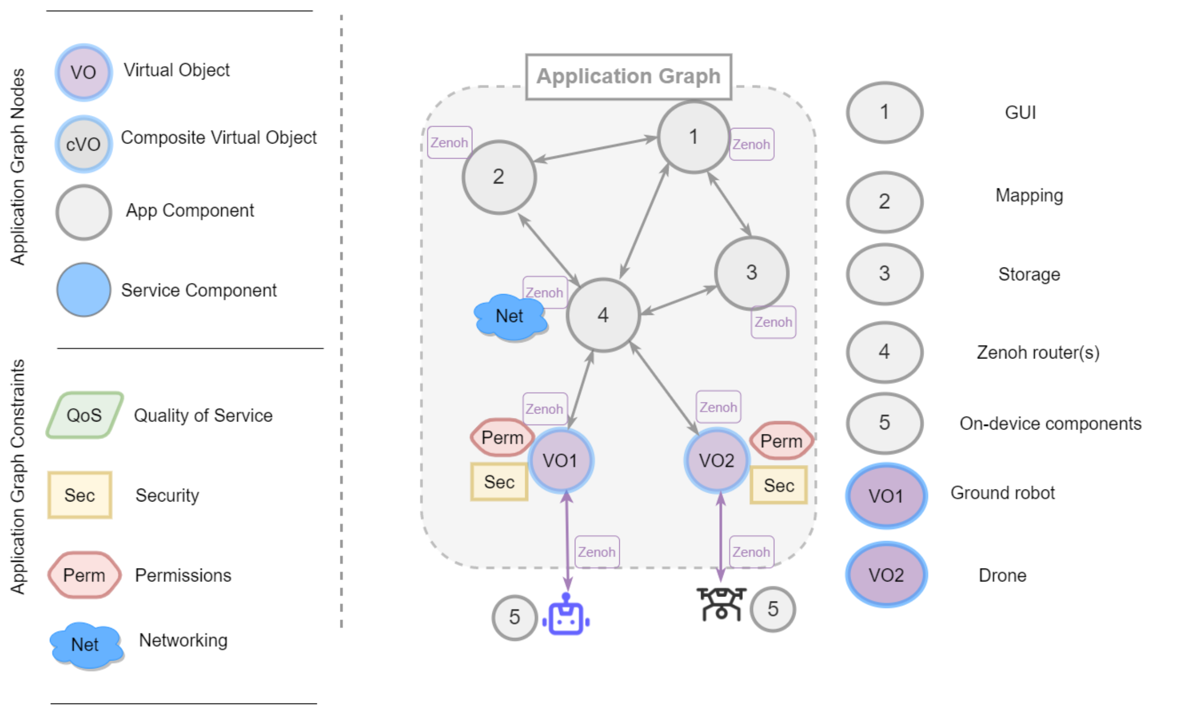
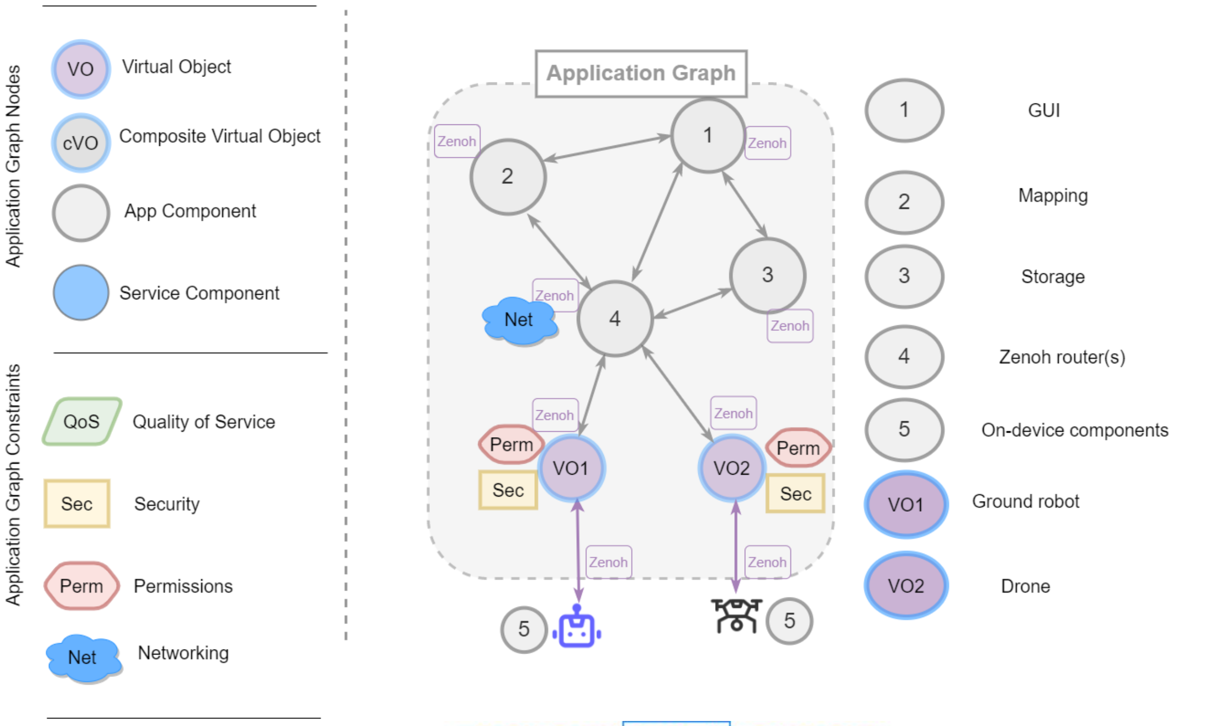


Figure 34: UC1 – Mapping application scenario – Application graph



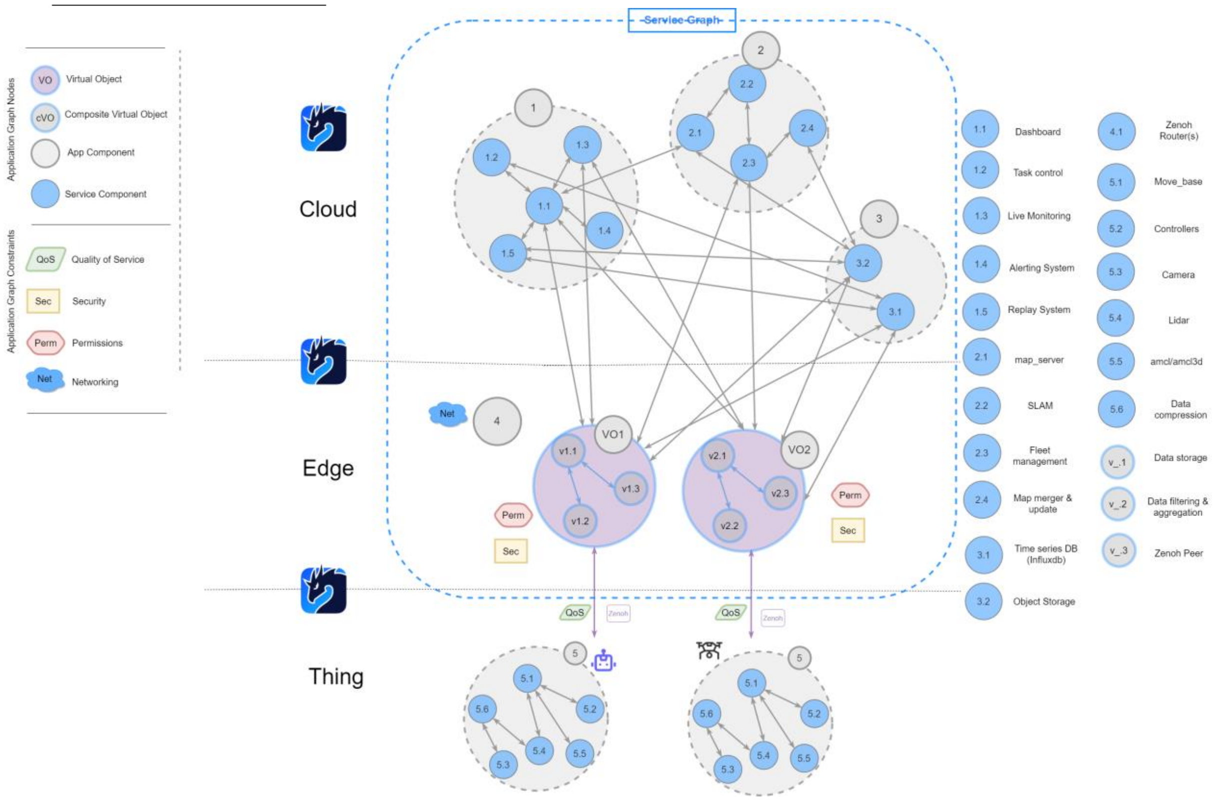


Figure 35: UC1 – Mapping application scenario – Service graph

Victim Detection and Injury Assessment

This scenario refers to the application components and services to detect victims and assess their injuries in the post-disaster area using ground robots and/or drones. Like the previous application scenario, a VO should be deployed for each ground robot and drone at the edge of the network, a network connection fulfilling data rate and latency requirements for video streaming is required between drone and NEPHELE through the corresponding VO, storage is required and computation at the edge are required. The Zenoh protocol will be used also in this case for communication between the application components and services running either on the physical devices or on the edge and cloud continuum. A trajectory planner for the ground robots and drones used is needed. Additionally, services are required for object/person detection and assessment of their injury. Using AI-supported algorithms, the map of the area can be enhanced with a graphical add-on about the detected information. By this, the first responder using a GUI will experience an enhanced situational awareness. Managing multiple robots for this task also requires a fleet management service for the application. Alerting, monitoring, task control, replaying and dashboards services through a web interface complete the application graph.

In Figures 36, 37 and 38 we represent respectively the high-level view, the application graph, and the service graph for the victim detection and injury assessment application scenario.

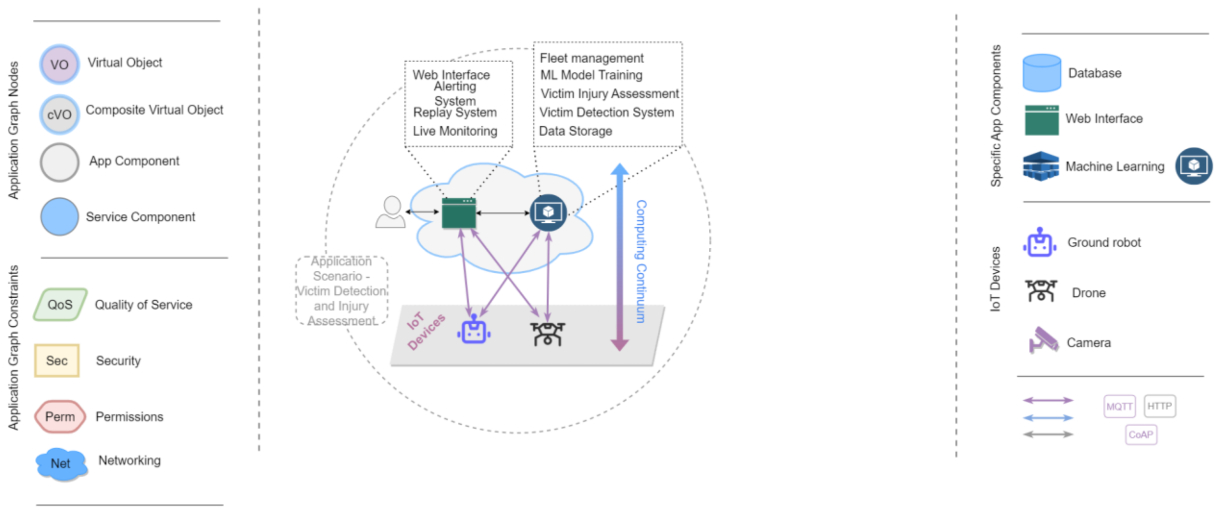


Figure 36: UC1 – Victim detection and injury assessment application scenario – High level

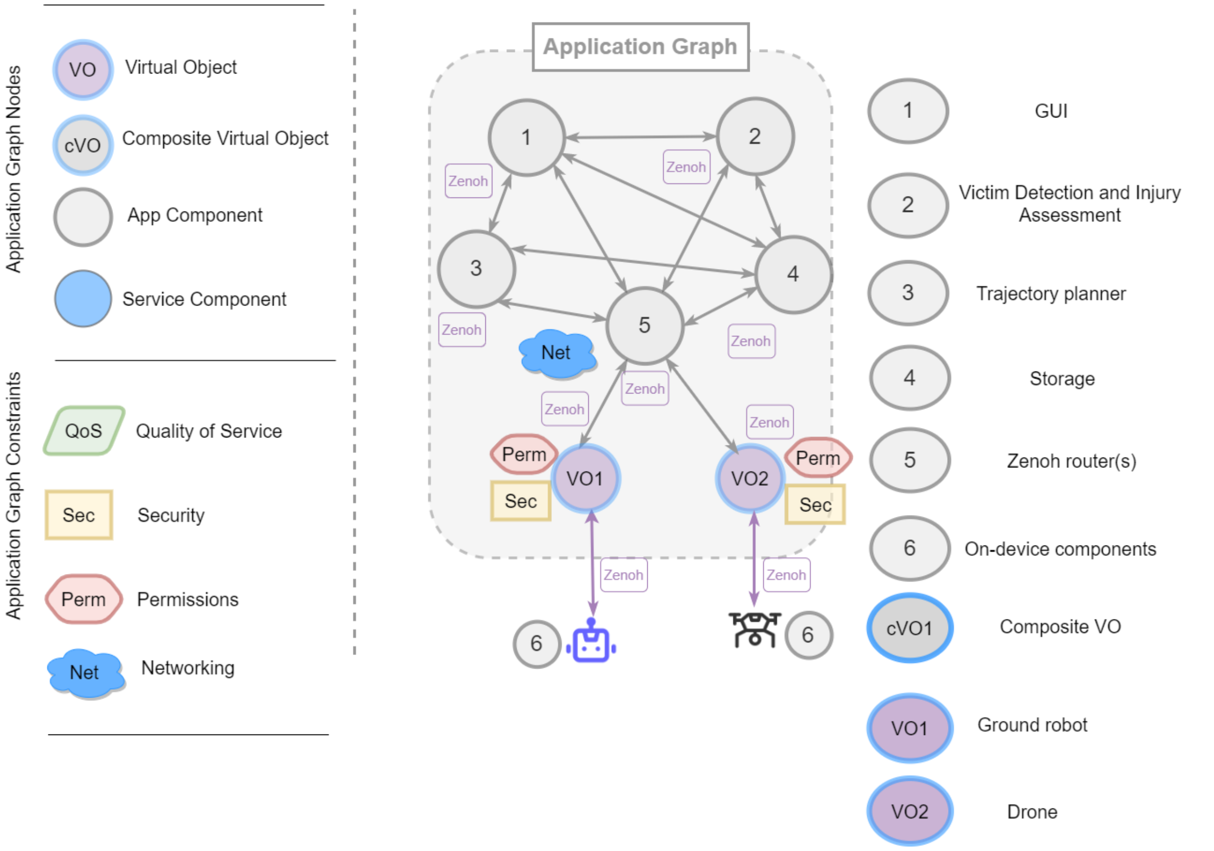


Figure 37: UC1 – Victim detection and injury assessment – Application graph

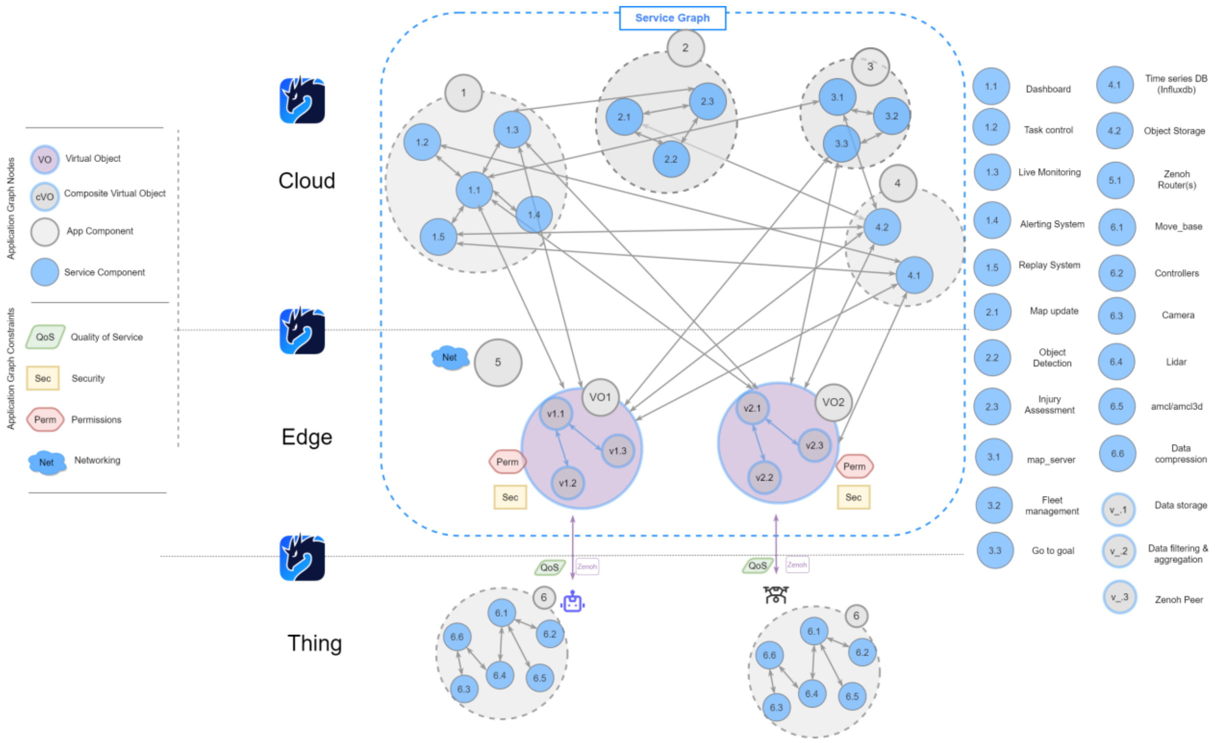


Figure 38: UC1 – Victim detection and injury assessment – Service graph

Risk Prediction

This scenario refers to the application components and services to predict possible risks in a post-disaster area using ground robots, drones, and sensor nodes. Once the wireless sensor network is deployed, the sensed data is sent to the edge of the network using the appropriate gateway or the access point depending on the network configuration and technology. Multi-hopping is available for sending data from the physical sensors deployed to ensure the connectivity and monitoring of all sensitive areas to the VOs when the application allows it. Robots and drones are used to further monitor the area and identify risks using their cameras. Liquid leakages are monitored over time to verify their movement and detect potential risks. A trajectory planner for the ground robots and drones used is needed, whereas AI-supported algorithms are used to detect risks to be shown on the GUI. Similar requirements as for the previous applications exist in terms of VOs, networking, communication protocols, fleet management, and storage. Alerting, monitoring, task control, replaying and dashboards services through a web interface complete the application graph.

In Figures 39, 40 and 41 we represent respectively the high-level view, the application graph, and the service graph for the risk prediction application scenario.

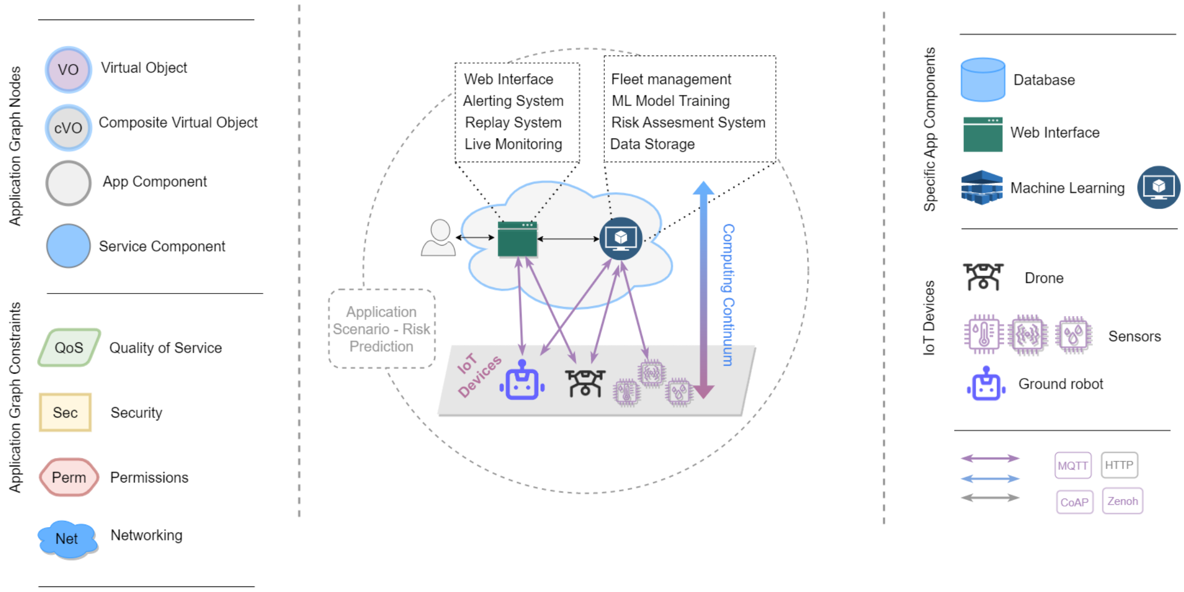


Figure 39: UC1 – Risk prediction application scenario– High level

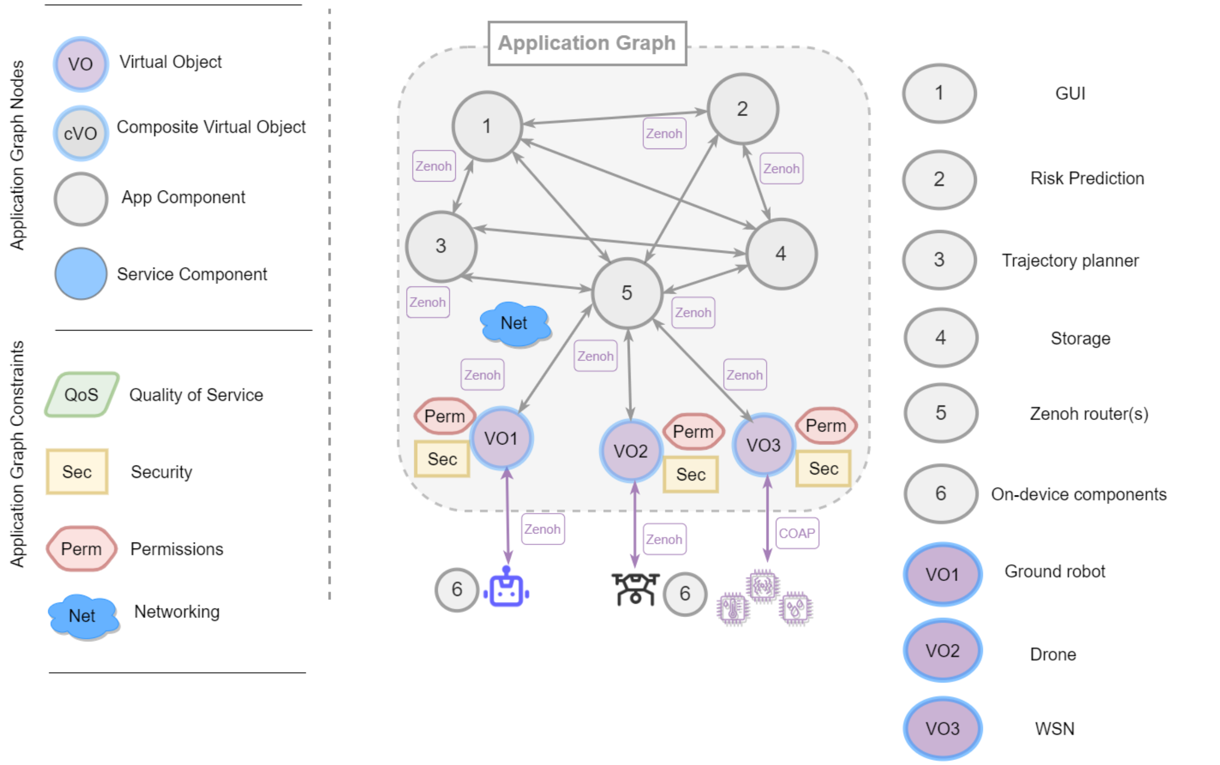


Figure 40: UC1 – Risk prediction– Application graph

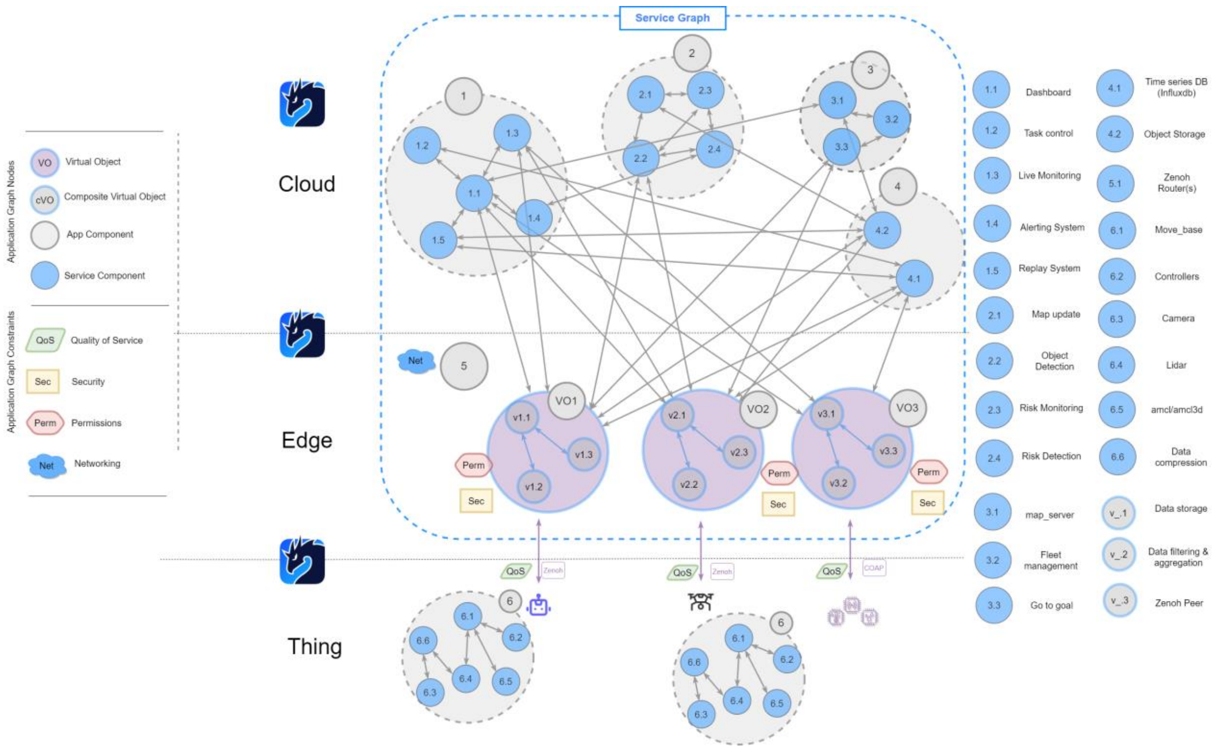


Figure 41: UC1 – Risk prediction – Service graph

Device deployer and liquid sampler

This scenario refers to the application components and services to physically deploy sensor nodes in a certain area or take liquid samples using ground robots equipped with a manipulator. A single VO should be deployed for the robot with the manipulator. Whenever a wireless sensor node is to be deployed in selected areas of the port, the ground robot with manipulator will be used. The selected areas for deployment are the most pertinent places according to the physical conditions of the port and the areas that have been identified as needing monitoring (e.g., based on the built maps). Sensors could be gas detectors, temperature, air quality, microphones, cameras, motion detectors, seismic detectors, and infrared sensors and thus may request different network capacities in terms of bandwidth, latency, etc. A gateway should be placed near the sensor network to send data from physical sensors to the Internet and a VO should be deployed for the wireless sensor network gateway at the edge of the network. The sensor network is pre-configured before the deployment, and it must ensure connectivity with the corresponding VO guaranteeing the needed bandwidth and latency. A network connection is required between robots and NEPHELE through the corresponding VO to receive instructions for the deployment of the wireless sensor network and to report feedback on the executed tasks. After the first deployment, sensors are continuously monitored to prevent disconnections. Data sent to NEPHELE through the VO should, therefore, also include information on the sensor/robot status itself (sensor temperature, battery level). Once a disconnection, or in general a need for additional devices is detected, the deployment of additional sensors can be triggered.

The same technology and hardware will be used in case some non identified liquid leakage was detected with the camera. A robot with a manipulator can be sent to take samples of the liquid for further analysis. This will avoid this risky operation to be performed by first responders directly. Fleet management, trajectory planner, storage, alerting, monitoring, task control, replaying and dashboards services through a web interface complete the application graph.

In Figures 42, 43 and 44 we represent respectively the high-level view, the application graph, and the service graph for the device deployer application scenario.

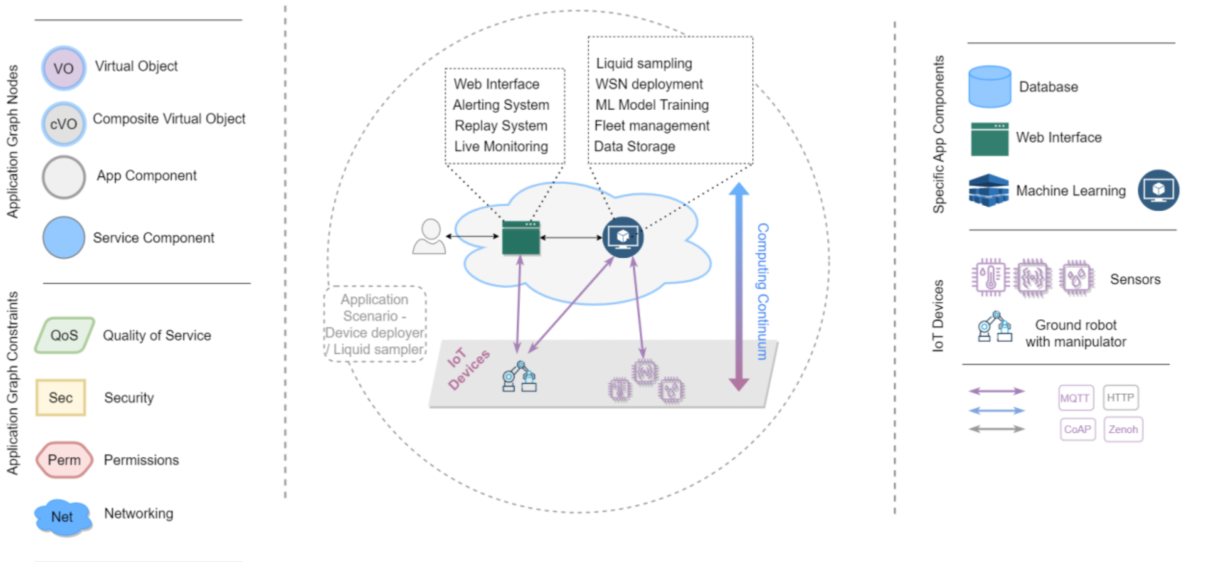


Figure 42: UC1 – Device deployer application scenario – High level

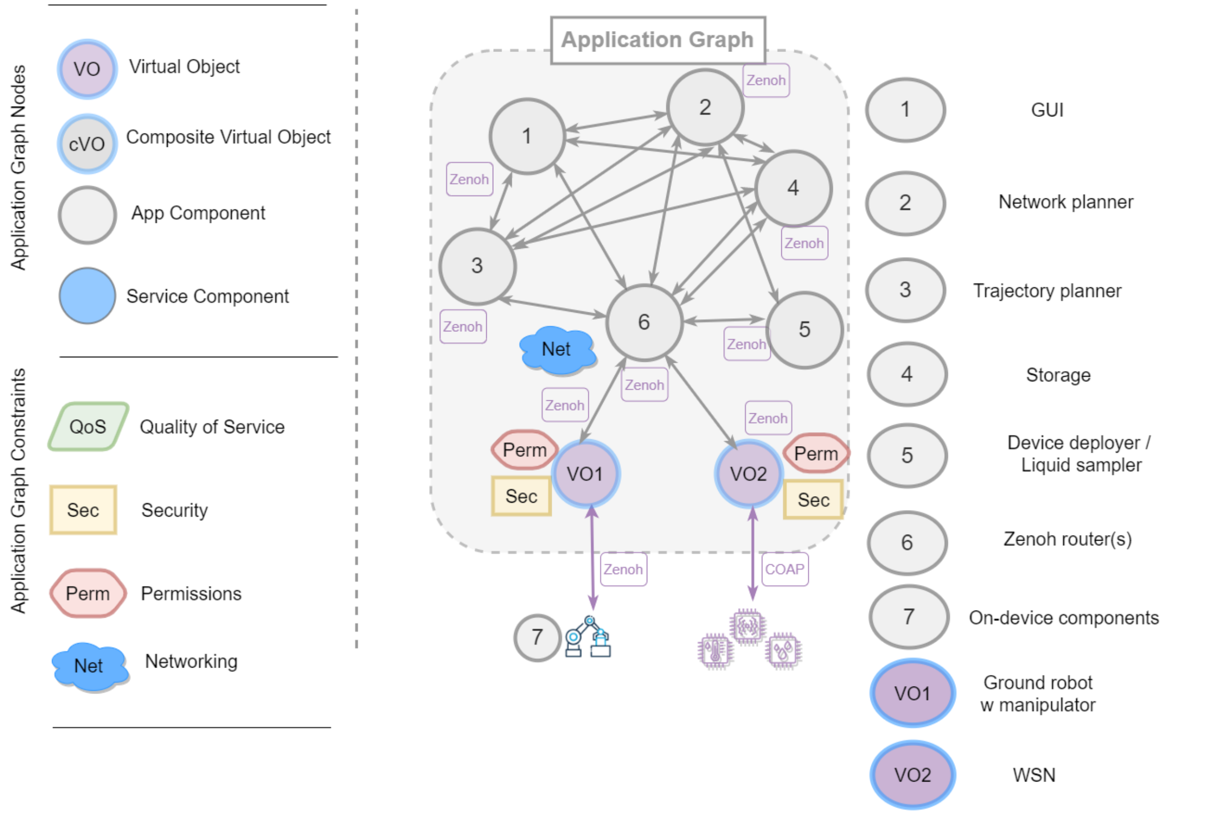


Figure 43: UC1 – Device deployer – Application graph

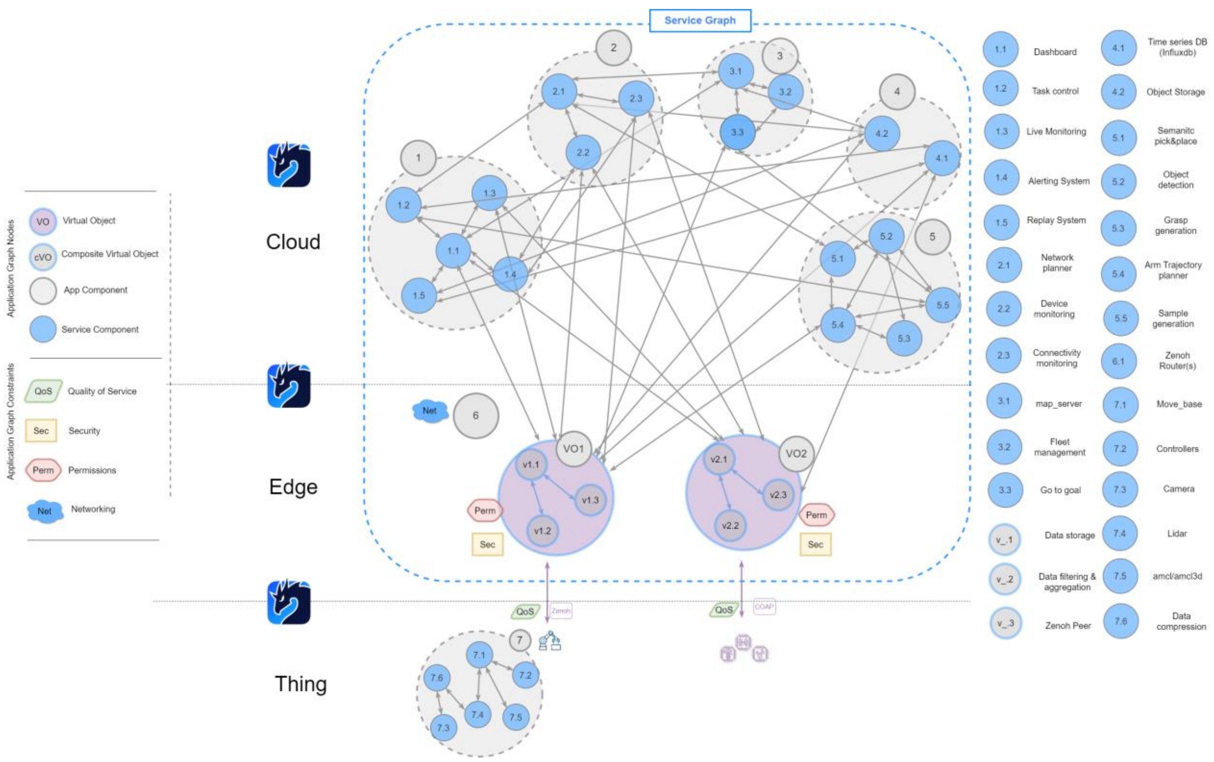


Figure 44: UC1 – Device deployer – Service graph

Network and device monitoring

This scenario refers to the application components and services to monitor the network connectivity for the IoT devices deployed in the post-disaster area. Network devices, sensors and robots are continuously monitored to prevent risks of device and network disconnection. To this aim, networking and device information are sent through the VO or directly (depending on the device) to the application components monitoring the network status. Similar requirements as for the previous applications exist in terms of VOs, networking, communication protocols, and storage. Alerting, monitoring, task control, replaying and dashboards services through a web interface complete the application graph.

In Figures 45, 46 and 47 we represent respectively the high-level view, the application graph, and the service graph for the network monitoring application scenario.

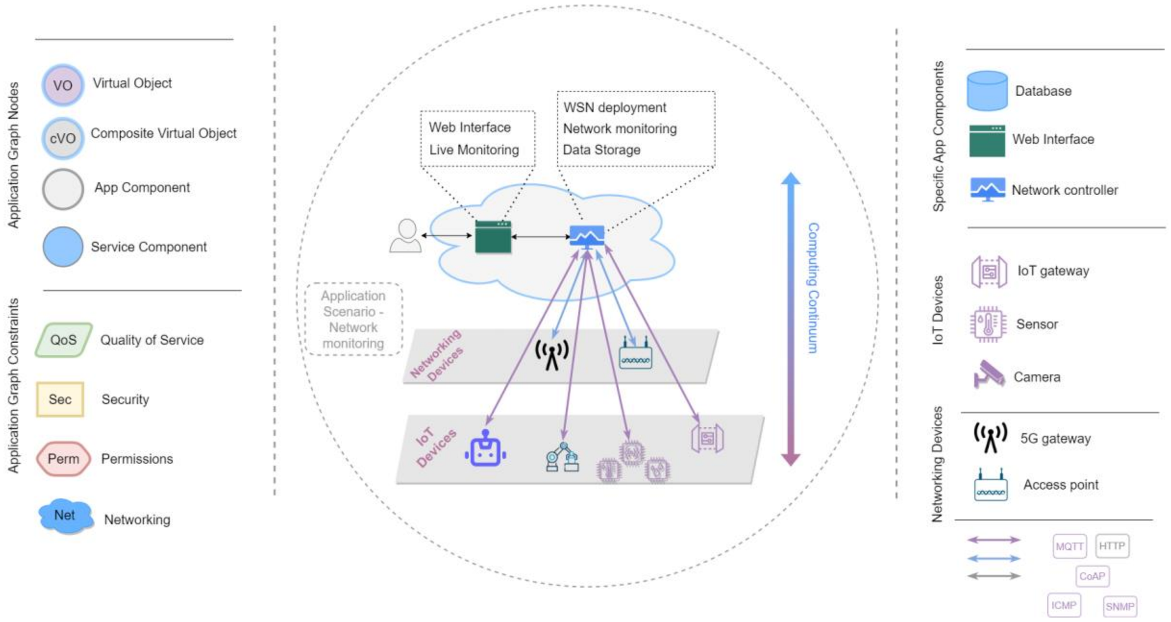


Figure 45: UC1 – Network monitoring application scenario – High level

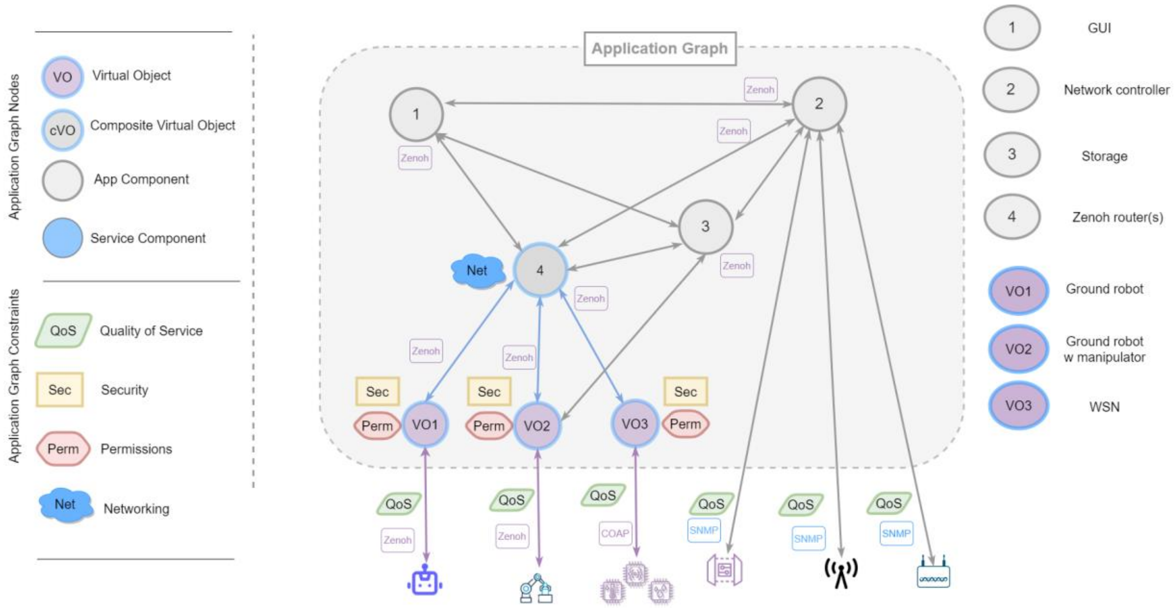


Figure 46: UC1 – Network monitoring – Application graph

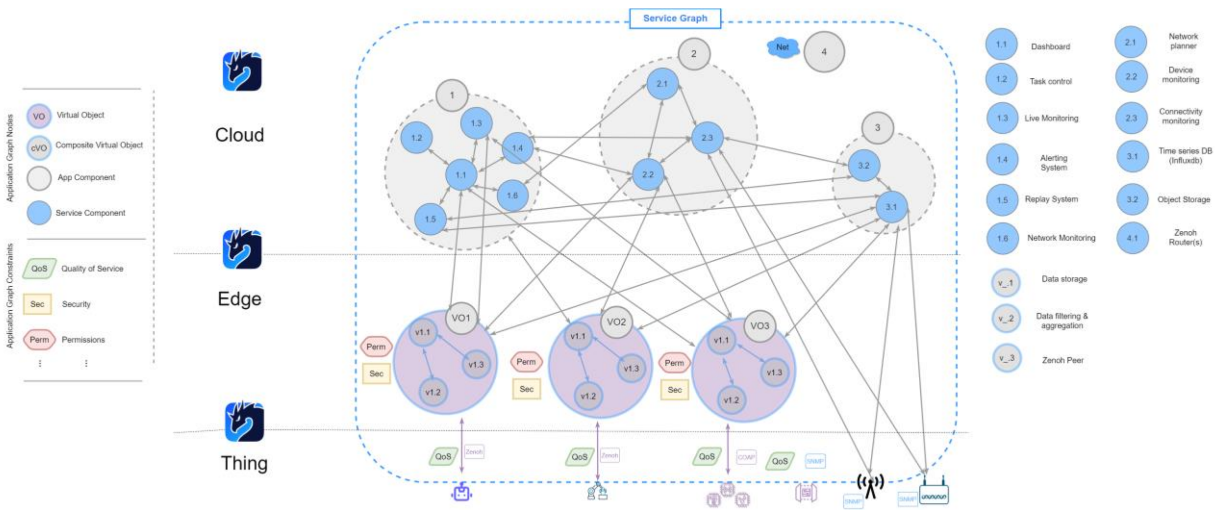


Figure 47: UC1 – Network monitoring – Service graph

1.6 Use case demonstration

The testing and demonstration of this use case will be performed first in simulation and later in a field trial scenario. The simulation will validate the methodology and concepts and will mimic the test that will be done at the end of the project in a field trial. This is expected to happen in a real port environment with tranship containers such as the Luka Koper port in Slovenia and an emulated post-disaster scenario. The hardware that will be used are those available in the labs of INRIA and ZHAW and integrated according to the needs of the use case.

1.7 Data processing requirements

The use case described requires several data types to be collected from several sources and sent to the edge for further elaboration or use. As already described earlier, we expect to use cameras and sensors to map and monitor the area of interest with the scope to identify victims to be rescued and identify further risks and damages in the area. Video streams, pictures and sensed values will be collected for further elaboration to enhance the situational awareness of the incident manager. Also, historical data about the actions taken and tasks accomplished by robots in a mission will be stored for future replay. At the same time, object recognition and victim recognition will typically be based on AI) algorithms based on pre-trained models on datasets. Finally, device monitoring data will be collected for mission control and coordination. The produced data is expected to be predominantly digital with sizes varying from bytes (e.g., sensed values) to GBs for video transmissions. The exact format of these data will be determined during the implementation phase.

Data-processing requirements typically fall into two classes: system-oriented and user-oriented. System-oriented requirements measure the amount of information that your systems process. By contrast, user-oriented requirements measure the impact of data-processing services on the user. Service-level agreements reflect these expectations of performance.

In Table 11 we report the main data processing requirements (DPR) for this UC.

Table 11. UC1 - NEPHELE's data processing requirements

ID	Type	Requirement	Description
DPR_UC1_01	User-oriented	Rapid response	Due to the need for timely intervention the system should guarantee low latency in communication.
DPR_UC1_02	User-oriented	Situational awareness	Data fusion from different sources/devices should be supported to enhance situational awareness for first responders.
DPR_UC1_03	System-oriented	Concurrent data sources	The number of concurrent data sources varies from case to case, but we imagine the following ranges: • 1-10 ground robots • 1-3 drones • 10-50 sensors • 1-10 cameras
DPR_UC1_04	System-oriented	Dynamic Workloads	A mix of static (sensing values) and dynamic workloads is expected (rescue victims if identified) as the use case is dynamic in its nature. A mix of light workloads (filter sensed values or send trajectory plans) and heavy workloads is expected (video streams and image analysis).
DPR_UC1_05	System-oriented	Storage	Collected data from sensors, cameras, robots/drones trajectory plans and executed tasks should be stored both in memory for fast use and in persistent storage to offer the possibility to replay executed actions/missions. The expected data sizes vary from bytes for sensed values to MBs/GBs for images and video data.
DPR_UC1_06	System-oriented	Bandwidth/ Latency	The requirements in terms of bandwidth and latency vary according to the different subtasks of the use case: **Bandwidth** • Disaster alert: < 1Mbps • First responder connectivity: ~10Mbps • Monitoring sensor networks: < 1Mbps • Video streaming: > 10Mbps • Drone/robot fleet navigation: ~100Mbps • Digital twinning: 100Kbps **Latency** • Disaster alert: 10ms • First responder connectivity: ~100ms • Monitoring sensor networks: ~1000ms • Video streaming: < 300ms • Drone/robot fleet navigation: 20ms • Digital twinning: 20ms (E2E)

DPR_UC1_07	System-oriented	Privacy and security	Although it is an emergency, personal data such as videos and images from victims should be guaranteed to be secured and adopt privacy standards in transmission and storage.

1.8 Use case requirement analysis

The following subsections provide an analysis of functional, non-functional and system requirements for the use case.

Functional Requirements

Table 12 presents a description of the functional requirements for the UC1, as well as some comments on how the requirement will be addressed.

Table 12: UC1 - Functional requirements

ID	Description	How to address	Priority
FR_UC1_01	Map an unknown post-disaster area	Use autonomous ground robots to map the area	High
FR_UC1_02	Detect and locate victims	Use cameras on robots/drones for person detection using AI models	High
FR_UC1_03	Detect and locate specific objects	Use cameras on robots/drones for object detection using AI models	High
FR_UC1_04	Assess victim's injury	Use cameras to analyze status using AI models	Medium
FR_UC1_05	Assess damages to infrastructure	Use cameras to evaluate current damages to infrastructure	Medium
FR_UC1_06	Detect if some dangerous gas is in the area	Use wireless sensor networks for gas detection	High
FR_UC1_07	Detect dangerous increase in temperature or pressure	Use wireless sensor networks for temperature/pressure detection	High
FR_UC1_08	Detect increasing fire/smoke in an area	Use wireless sensor networks for smoke/fire detection	High
FR_UC1_09	Detect if containers in the port are at risk of collapsing	Use cameras to identify the position of containers and predict the danger of collapse	High
FR_UC1_10	Deploy sensor network in unknown area	Use mobile ground robots to autonomously deploy sensors in the area	High
FR_UC1_11	Collect live data-streams from devices	Cameras from drones and robots, sensors will produce data streams for the system	High
FR_UC1_12	Record/store data for replay function	Use storage services to store maps, actions and video streams	Medium
FR_UC1_13	Allow the user to choose mission tasks	A graphical user interface is needed for human interaction	High
FR_UC1_14	Trigger alerts to the user	A graphical user interface is needed for human interaction	High
FR_UC1_15	Get and show the status of robots, drones and sensors	A graphical user interface is needed	Medium

FR_UC1_16	Keep track of executed tasks	A graphical user interface is needed supported by a storage service	High
FR_UC1_17	Keep an updated map of the area	A global knowledge about the environment at robots' level. Communication between robots and VOs. Cameras and sensors to get information from the environment	Medium
FR_UC1_18	Access and use existing pre-trained AI models	If certain pre-trained AI models already exist, it would be possible to use them when analysing a single video frame or image	High
FR_UC1_19	Place bounding boxes at specific (interesting) images parts (object detection)	A specific algorithm for extracting regions of interest is used, which can be further processed and analysed	Medium
FR_UC1_20	Redeploy network nodes / use local storage in case of disconnections	Use mobile ground robots to deploy additional nodes in case of disconnections or risk of disconnections are predicted	Medium
FR_UC1_21	Allow historical analysis of actions	Store maps, actions and tasks performed for historical analysis	Medium
FR_UC1_22	Guarantee networking communication among robots/drones/sensor networks	Use opportunistic network protocols, access mechanisms and routing schemes to keep devices communicating	High
FR_UC1_23	Enable devices to be registered and described in the ecosystem	Define protocols and procedures to make devices register to the corresponding VO	High
FR_UC1_24	Enable robots to offload tasks horizontally	Horizontal offloading based on the battery level of the robots and the importance of the collected data	Medium
FR_UC1_25	Real-time monitoring	The system should be able to collect and process data in real-time, and provide real-time alerts and notifications if certain thresholds are exceeded or if certain conditions are detected	Medium
FR_UC1_26	Decision support	The system should provide decision-support tools that can help emergency responders and other personnel make informed decisions about how to respond to a disaster. This can include visualization tools that display sensor data in a way that makes it easy to understand, as well as decision support algorithms that can analyze sensor data and provide recommendations.	Medium
FR_UC1_27	Enable migration of nodes between networks	The system should be resilient to mobility and migration of nodes.	Medium
FR_UC1_28	Detect liquid leakage and its movement	Images and videos from drones can be used to identify liquid leakages and their extension over time	High
FR_UC1_29	Enable liquid leakage sampling	Ground robots with manipulator could approach the identified liquid and take a sample	Medium

Non-functional Requirements

Table 13 provides a description of the non-functional requirements for the UC1 and how they will be addressed.

Table 13: UC1 – Non-Functional requirements

ID	Description	How to address	Priority
NFR_UC1_01	Be robust in terms of the number of sensors available and alive	Ground robots are used to deploy/redeploy sensor nodes as needed, while the sensor network is preconfigured to guarantee connectivity and robustness	Medium
NFR_UC1_02	Ensure connectivity in post-disaster area	Use and deploy 5G IoT gateway or an alternative valid solution	High
NFR_UC1_03	Ensure privacy	AAA techniques will be implemented	Medium
NFR_UC1_04	Guarantee security of data storage and processing	Secured and authorised access to the system should be implemented.	Medium
NFR_UC1_05	Delegate computational calculation to the Edge and Cloud	Vertical offloading	High
NFR_UC1_06	Be resilient, efficient, lightweight and with flexible design	Use containerized applications and cloud-native principles ready for orchestration	Medium
NFR_UC1_07	Limit bandwidth usage	Implement smart data filtering/aggregation/compression policies	High
NFR_UC1_08	Be energy efficient	Task assignment and sampling frequency based on battery level in the device	Medium
NFR_UC1_09	Ensure enough resources for application	Use network slicing and orchestration techniques	High
NFR_UC1_10	Ensure very low latency and high bandwidth	Use 5G communications, edge computing and low-latency communication protocols	High
NFR_UC1_11	Ensure good precision and high confidence in object/person detection	Use pre-trained models specific for the study case	Medium
NFR_UC1_12	Adapt sensor sampling frequency to situation	Implement smart policies based on multiple parameters	Medium
NFR_UC1_13	Ensure communication protocol integration with VO-Stack	Implement communication protocols that enable robots/drones/sensors to communicate with the VO-Stack (e.g., MQTT, Zenoh)	High
NFR_UC1_14	Ensure semantics integration with VO-Stack	Implement semantics that enable robots/drones/sensors to communicate with the VO-Stack	High
NFR_UC1_15	Ensure dynamic placement, performance monitoring and dynamic redeployment of software components	Adopt service mesh approaches and orchestration solutions for components of the application graph	High
NFR_UC1_16	Enable routing and multihopping schemes	Enable IoT devices to forward through relaying nodes to reach the Internet	Medium
NFR_UC1_17	Support adding/removing devices on the fly, including their lifecycle	Enable orchestration of components required to support adding/removing devices. Enable orchestration of components on devices	High

System Requirements

A system requirement (SR) defines the configuration that a system must have to run smoothly and efficiently. A system requirement may refer to computational power, hardware capacity, etc. Failure to meet a system requirement can result in the installation or performance problems. Table 14 provides a description of the non-functional requirements for the UC1 and how they will be addressed.

Table 14: UC1 – System requirements

ID	Description	How to address	Priority
SR_UC1_01	Have enough computational power	To run the mission control and AI methods sufficient memory and processing power is needed	High
SR_UC1_02	Have internet connectivity	To offload computation if needed and datasets that are needed	High
SR_UC1_03	Have the possibility to use 5G frequencies	To interconnect the devices with the NEPHELE devices an IoT/5G gateway might be needed with permission to use the 5G national frequencies	Medium
SR_UC1_04	Have enough storage capacity	To store and process the data coming from various data sources. Cloud storage will be used	High
SR_UC1_05	Have virtualization capacities	The system will be virtualized and re-deployable in containers. Thus, the processing unit should have virtualization capabilities and be optimized for container virtualization.	High
SR_UC1_06	Have enough GPU capacity	It is likely that GPU capacity will be needed	Medium
SR_UC1_07	Provide robots which work autonomously	The robots work independently from the system. Once a task is assigned, they will perform it without contacting the system	High
SR_UC1_08	Have proper AI models for edge	This will be provided as part of the VO-stack	Medium
SR_UC1_09	Deploy on-demand wireless sensor networks	Robots will deploy sensor networks in the area as needed	Medium

1.9 NEPHELE’s innovation for the use case

Table 15 summarizes the requirements of the UC1 and discusses the limitations that NEPHELE aims at overcoming and thus facilitating the realization of the UC1. For each requirement, we provide a reference to its definition, which can be found in Tables 12, 13 and 14.

Table 15: UC1 – Requirements to demonstrate NEPHELE innovation

Requirement	Current limitation	Innovation
Reduce the computational load on the robots and perform timely actions with very low latency FR_UC1_024, NFR_UC1_005, NFR_UC1_006, NFR_UC1_008,	Simply offloading to the edge may not improve as the edge has also limited resources. Communication protocols from physical devices	With NEPHELE, network and computation resources can be dynamically allocated through network slicing techniques. Orchestration of application components will enable adaptation to

NFR_UC1_009, NFR_UC1_010, SR_UC1_001, SR_UC1_005	should guarantee low latency.	the current status and predict dynamicity factors in the allocation of resources.
Generate a collaborative map of the area using the information from drones and robots FR_UC1_001, FR_UC1_002, FR_UC1_003, FR_UC1_017	Data fusion from heterogeneous data sources is a challenging task due to different communication protocols, semantics and data formats.	With NEPHELE the supportive functions of the VO-Stack will enable the effective and efficient elaboration of data over the cloud continuum.
Identify objects and victims, predict dangerous situations FR_UC1_002, FR_UC1_003, FR_UC1_005, FR_UC1_006, FR_UC1_007, FR_UC1_008, FR_UC1_009, FR_UC1_011, FR_UC1_019, FR_UC1_028, FR_UC1_029, NFR_UC1_011, SR_UC1_007	AI models are cumbersome to train and require computational power. Privacy concerns arise for victims' identification.	NEPHELE's VO-stack offers AI models as VO-supportive functions tailored to the specific use case requirement. End-to-end privacy and security are guaranteed by NEPHELE.
Improve situational awareness for first responders with technology-supported mission control FR_UC1_001-027, NFR_UC1_001-015, SR_UC1_001-008	Lack of integration of different IoT devices and technologies. First responders need to use multiple technologies and specialized personnel.	NEPHELE offers the possibility to orchestrate distributed applications and resources over the Cloud continuum reaching out to IoT devices. The VO-Stack enables heterogeneous devices to interoperate and collaborate.

2 Use case 2 in NEPHELE

2.1 Introduction and objectives

Port of Koper is a multi-purpose deep-sea freight port located at the Northern end of the Adriatic Sea. Logistic and port services are provided by the company Luka Koper d. d. Total maritime throughput in 2020 topped over 19,5 million tons and the company has strengthened its position becoming the most important container port in the Northern Adriatic. Port terminals are equipped with state-of-the-art transshipment and warehousing equipment, such as ship-to-shore cranes, reach-stackers, forklifts, utility tractor rigs, etc. Port of Koper has excellent connections to road and railway network. Continuous monitoring and optimization of the traffic within the port poses a daily challenge (traffic congestion, unplanned road closures, etc.), which will be addressed in the use case.

The main objective of this use case is to optimize the routing of containers from the Container terminal yard or Depo area to different Container Freight Stations (CFS) within the port, where the cargo is stuffed/stripped, and vice-versa. This is one of the most important operations in the port. This objective will bring business value in terms of reduced routing times, lower CO_2 emissions, higher truck/forklift utilization, and service level agreements (e.g., times of delivery, compliance with goods sensitivity, etc.).

The exploitation of the VOStack layers will allow to exchange and aggregate data among the physical components involved in the use case (e.g., forklifts, trucks, cameras, sensors). The application of decentralized machine learning techniques at a VO level will satisfy requirements regarding security and low latency regarding a set of port operations (containers routing optimization, traffic detection and classification). The integrated meta-orchestration framework will allow the orchestration of the deployed microservices between the cloud and edge computing orchestration platforms ensuring the self-healing, portability, and elasticity of the complete solution (Figure 48).

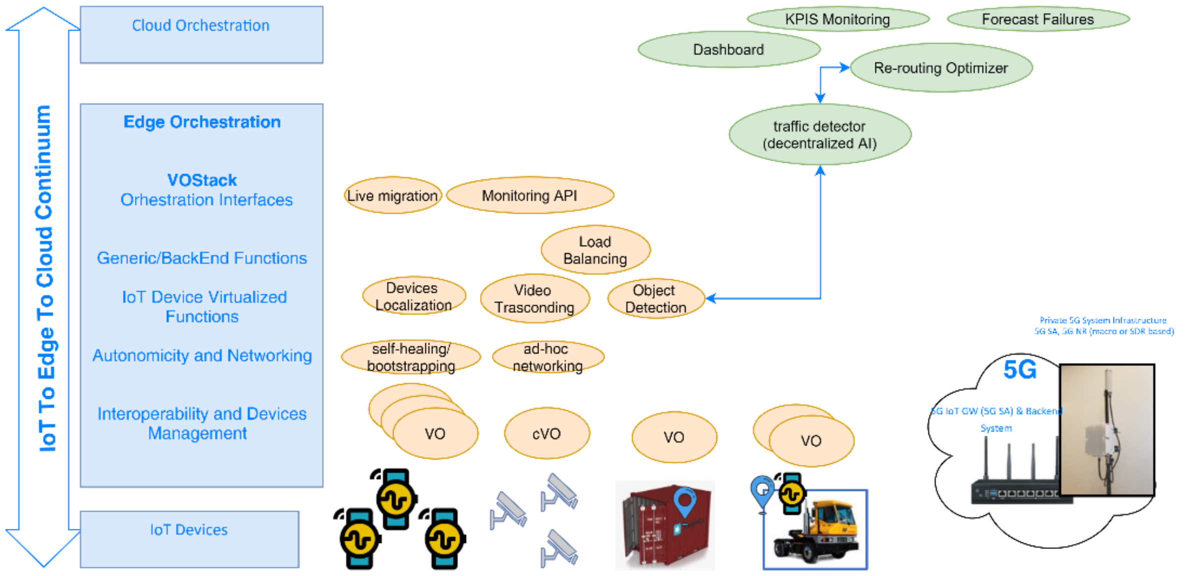


Figure 48: UC2 - AI-assisted Logistics Operations in port

2.2 Case Study: Containers routing optimization in the port

Freight forwarders place order to the Container Terminal to organize that the set of containers, both full (in import) or empty (for export), are timely delivered to the CFS in the port, where containers are loaded/unloaded of the cargo. Additionally, and in parallel, freight forwarders place orders to the General Cargo Terminal. On this basis, a common delivery plan is prepared, including a list of containers to be delivered to CFS (and vice-versa), delivery equipment requirements and staff accounted for the task. When the plan is set, the algorithms for the "container route optimization" should define the work order list sequence and the optimum number of trucks/forklifts, taking into consideration safety rules, priorities regarding vessel schedule, priorities about rail operations, cargo sensitivity, client ranking, terminal equipment availability, daily traffic in the port (road and rail), work on other terminals, etc.

In the Figure 49 we summarize the main stakeholders, the location (physical or virtual) and the constraints, challenges, and risks for this case study.

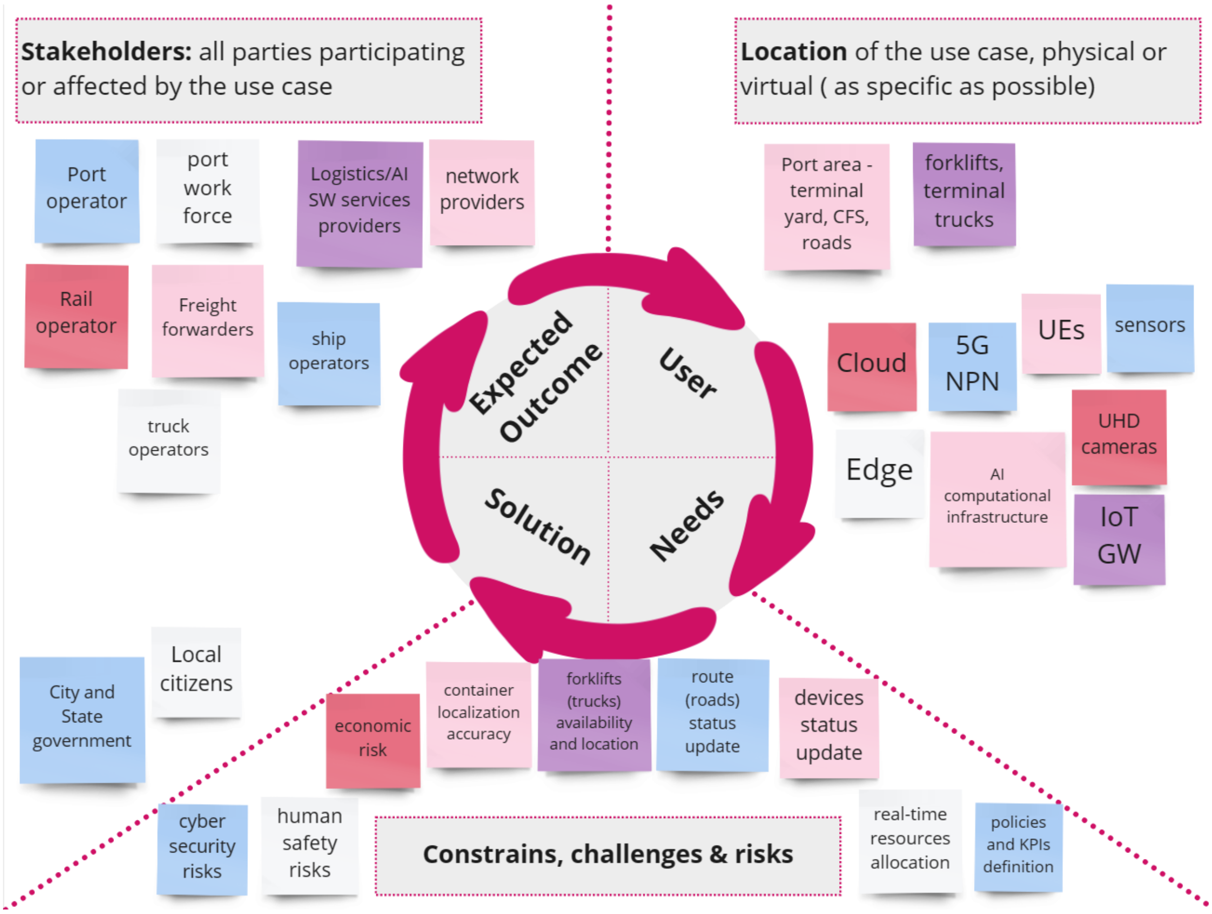


Figure 49: UC2 - Stakeholders, location, constraints, challenges and risks for the use case

Stakeholders

Several stakeholders are related to the use case scenario, its actors and its impact to the port and its neighborhood. Port operator is considered as a main stakeholder since its role as a main user of the solution explored within the use case. The port operator is therefore expected to gain certain economic benefits. Considering the latter, benefits are also expected for other stakeholders directly involved in the logistics process, i.e., freight forwarders, ship operators, rail operator, truck operators. All those will benefit out of the optimized business process(es) taking place in the port. The latter will also impact port work force in general, i.e., their productivity.

Since the use case requires specific software solutions, network, sensors, and other equipment, at least two additional stakeholders are required, i.e., software/service provider(s) and network provider/operator. On the other side, their economic benefits come from providing required services.

Specific interest in the use case is also related to local community, i.e., local citizenship and city/local government, as well as state government which all benefits due to the economic strength of the port. As well, optimized business processes in the port, as described, tends to reduce the greenhouse gas emissions thus contributing to greener environment and sustainable development.

In Figure 50 we report a detailed view of the main users for the use case with a high-level analysis of the needs, functionalities and expected outcomes for each of them.

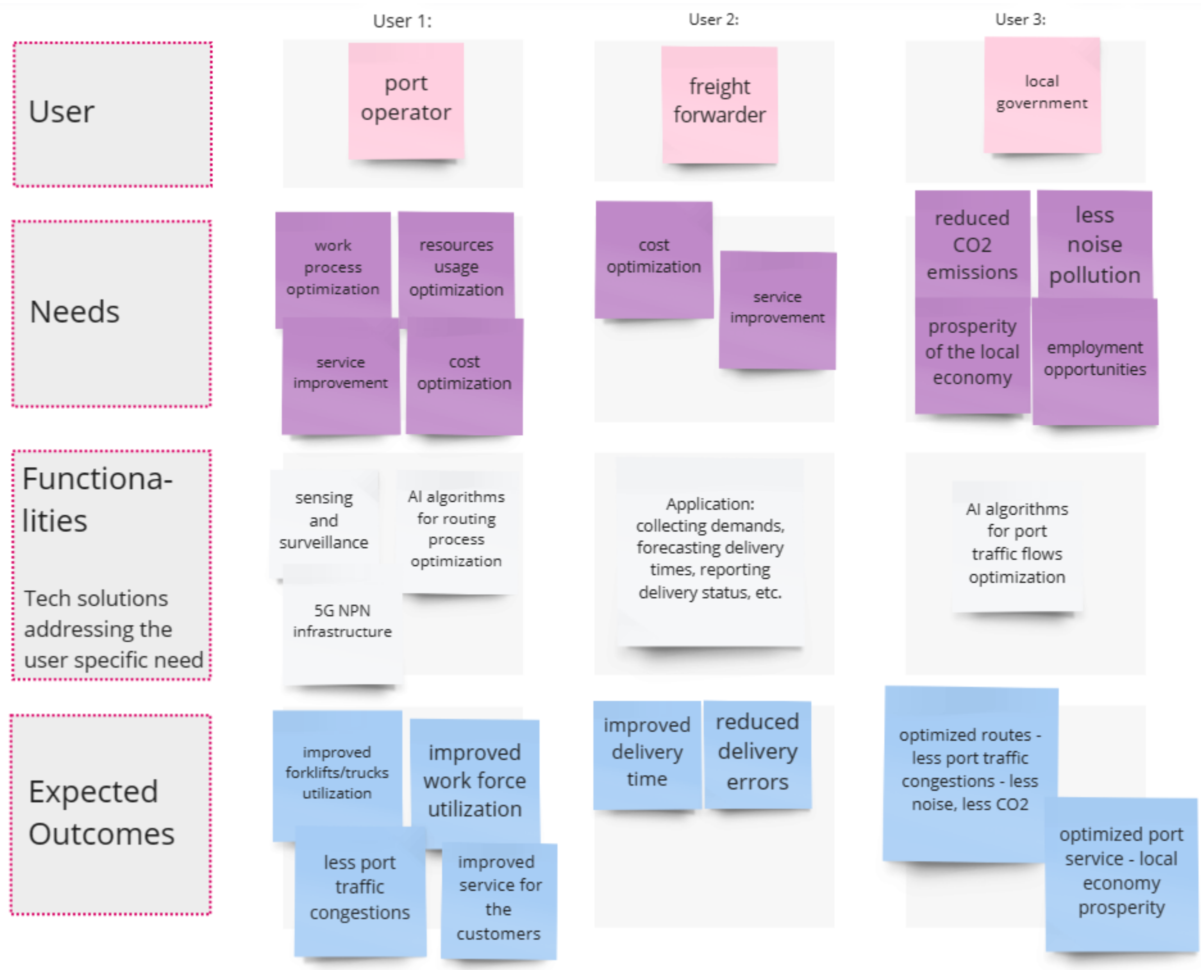


Figure 50: UC2 - Needs, functionalities and expected outcomes for the main users in the use case

Location

The physical location of the use case study is a container terminal where we are looking for optimizing the routing of containers from the container terminal yard (or depo area) to different container freight stations (CFS) thus reducing routing times, lowering greenhouse gas emissions, enhancing truck/forklift utilization and enhancing service level agreements (e.g., times of delivery, compliance with goods sensitivity, etc.). The implementation of the use case will require the use of hardware and virtualized locations. The following infrastructure is used for the scope:

- **Infrastructure as a Service (IaaS):** cloud infrastructure provided by UoM, required for the cloud components of the cloud-continuum based solution to run properly.
- **Edge Infrastructure as a Service:** edge infrastructure provided by LKOP, required for the edge components of the cloud-continuum based solution to run properly.

- **Portable Infrastructure as a Service:** mobile network solution (5G) provided by ININ, required for providing data connectivity between IoT devices (sensors) in the field/port, its corresponding VOs and application level of the solution.
- **IoT devices:** provided by LKOP and ININ, required for acquiring realtime data and status (e.g., location of terminal trucks) from the field. Now, following IoT devices are expected to be involved in the use case:
  - industry-grade 5G IoT gateway with additional computing capabilities (Far-Edge IaaS): serves as a gateway providing 5G connectivity to non-5G devices and enables far-edge components of the cloud-continuum based solution to run properly,
  - industry-grade UHD cameras,
  - GNSS sensors providing location, truck speed and other GNSS related information in real-time,
  - On-Board Units mounted on trucks/ forklifts.

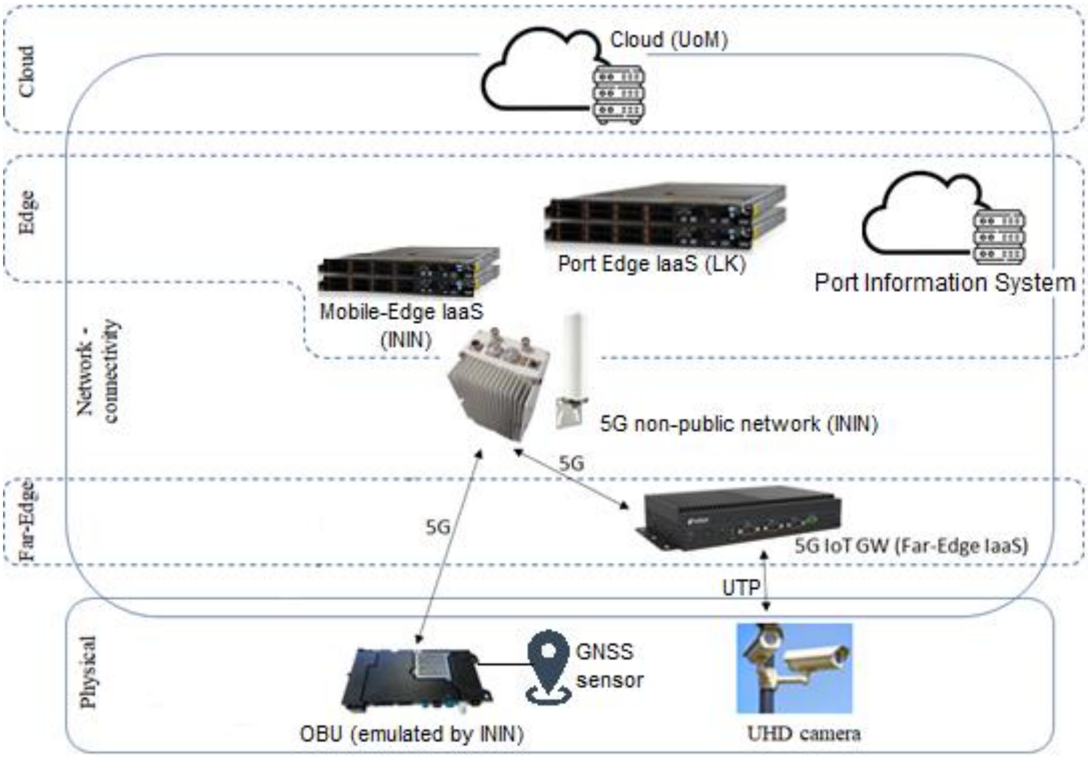


Figure 51: UC2 - Devices at the physical, networking and computation (cloud, edge, far-edga) levels

Constraints, challenges, and risks

As reported in Figure 49, containers routing optimization in ports use case includes several inherent constraints, risks, and challenges as detailed next.

Risks

- Cyber security risks are inherent to any ICT solution, however there are several options available to defend against it, one of them is using isolated non-public mobile network for data transmission within the port.
- Human safety risks.

- Economic risks are related with the effectiveness of the containers routing optimization process, i.e., in case the solution proves it is not capable meeting the requirements, the investment could be considered a financial flaw.

Constraints

- Policies and KPIs definition may not suit well the process changes introduced through the optimization process and may therefore be adapted.

Challenges

- Container localization accuracy could be a challenging in the environment with high stacks of steel containers, therefore affecting the containers routing optimization process due to not get correct input data.
- Forklifts/trucks availability and location are, similar as previous bullet, challenging due to radio propagation constraints within the environment where steel containers are loaded into high stacks.
- Route/roads status update is crucial for the optimal effect of the containers routing optimization process, while it will be challenging to have all details relevant to route/road status available at any time.
- Device status update is crucial for the effectiveness of the containers routing optimization as well, i.e., in case device reports invalid status, the outcome of the optimization process won't be optimal.
- Real-time resources allocation depends on correct resources status data available and related estimations on resources being occupied and released. The challenge is therefore related to collecting correct status data and to correct predictions of containers routing optimization process.

2.3 Technical requirements and challenges

Use case 2 main technical requirements and challenges are detailed as follows.

- **Radio network coverage (5G)**: the whole area where the containers routing optimization process takes place requires quality and stable radio network coverage providing data exchange between sensors in the field (stable UHD camera, drone camera, on-board units mounted on trucks/forklifts), applications running in far-edge, edge and cloud environment, as well as applications running on end users' devices.
- **Orchestration of software components:** containers routing optimization application components (as well as VO stack components) require certain conditions and availability of resources which need to be considered during the automated deployment. This will further require performance and resource monitoring at the various levels of the cloud-continuum and possibility of dynamic redeployment of certain components.
- **Device Management**: some application functionalities can be pre-deployed on the devices or at the edge. The device management should also enable bootstrapping and self-configuration, adding and removing devices on the fly, supporting hardware heterogeneity, and guaranteeing self-healing of software components.
- **Interface to data relevant for the business process**: containers routing optimization process requires data on freight forwarders demands and service level agreements stored separate databases.
- **Sensor data collection and aggregation**: data collected by sensor need to be properly stored in a secure place and available for further data processing.
- **Computer vision for information extraction**: since conditions relevant for containers routing optimization will be, among others, collected by cameras, algorithms for detecting relevant situations in video-stream or in still-pictures are required.
- **AI/ML supported data processing – containers routing optimization algorithm**: a key component of the system which considers all relevant data (sensors data, cameras data, freight

forwarders demand, service level agreements) and produces optimal schedules for freight/containers transportation within the port.

- **Providing feedback to port personnel and freight forwarders**: an application component providing outputs of the optimization algorithm – schedule and routes for truck/forklift drivers, schedules for freight forwarders, etc.
- **Analytics**: based on data collected and output data from the optimization algorithm, post-analytics should be available to evaluate the successfulness of the solution.

2.4 Phases of the containers routing optimization for the study case

The operations required for the use case can be split into the following subtasks.

- **Deployment of network infrastructure and application software**. The network infrastructure should guarantee data exchange in the port area (sensors, cameras, IoT GW, end-users' devices, etc.), including connectivity towards the internet. The application software to be deployed in specific devices includes all the components in the application graph and the VO/cVOs for the hyper-distributed application. The same hardware can be used for different tasks that use different application components over time. Initially, basic components will be deployed at different levels of the continuum.
- **Data collection.** According to the TRL level of the use case, some data will be collected by physical devices, and other data will be emulated to cover the whole spectrum of input data required. Data will be forwarded to the higher levels of the continuum. The data can be either images, video streaming, sensing data, telemetric data, device management/monitoring data or data extracted from the Port Information Systems (PIS).
- **Optimizing the routing of containers.** Data collected by physical devices, emulated data, and business process related data (freight-forwardes demands) will serve for optimizing the routing of the containers in the port, i.e., from the Container terminal yard or depo area to different Container Freight Stations (CFS) within the port.
- **Validation of the container routing optimization and improving the optimization process/algorithm.** For further improvement of the containers routing optimization process/algorithms, impact to the business process will be evaluated by comparing "optimized" containers routing process to the "manual" one.

Deployment of network infrastructure

The use case operational field is within the port area, specifically, it includes at least container terminal yard (or depo area), multiple container freight stations (CFS) and roads connecting them. The connectivity for field devices will be provided by 5G non-public network infrastructure deployed within the port area. All field devices used within the use case will therefore rely on 5G connectivity. Use case related data traffic will be routed among far-edge and edge servers both located in the port, and cloud instances located at UoM premises. The architecture and topology of the network infrastructure is expected to not change significantly during the duration of the use case development and testing/validating.

Deyployment of IoT devices in the port

Containers routing optimization use case will rely on video streaming provided by cameras and on various data provided by trucks' OBUs and PIS system.

Video cameras will be mounted on certain poles within the port, aiming at surveillance of roads and thus identifying traffic congestions, and at surveillance of parking areas identifying free parking lots for trucks enabling loading/unloading cargo.

There will be two types of OBUs available for the use case. OBUs that are already installed into the trucks have been provided by Continental within the Horizon 2020 5G-LOGINOV project. Multiple telemetry parameters related to each single truck are available by querying corresponding database, however, accessing data this way prevents these OBUs to be modelled by VO developed within Nephele project, while data can be anyway utilized, e.g., for emulating OBUs. This way, VO will be adjusted to extract data from the API provided by the OBU database via an appropriate southbound interface in order to emulate communication with IoT.

The other type of OBUs is not integrated to the vehicle as much as the previous one and therefore provides less data parameters, while on the other end, it can be modelled by VO. This OBU type will be connected to 5G IoT GW and the added GNSS sensors will provide information such as speed and location.

Collecting data from the port information system

Container routing optimization depends on many processes/operations taking place in the port, of which some are directly related to the container routing process, while others impact it since taking place within the same area and/or utilize the same resources, e.g, roads, transportation means, loading/unloading/warehousing facilities, various schedules, etc. Since the port information system stores data of multiple processes that may help improve the use case scenario, it will be identified through the use case development how much of these data will be useful to be considered in the container routing optimization algorithm. Data from the port information system are expected to be provided by an API exposing data required by the use case. At the moment, it is predicted the following data will be of interest: peak traffic hours, train schedules, containers routing schedules, time required for loading/unloading of cargo to the truck and its possible dependence on cargo/container/yard/warehouse type, etc.

Mapping the port area, identifying routes and other areas of the use case interest

Use case operational area will be described by a map with indicated routes suitable for trucks operation, areas of loading and unloading cargo and other areas that might be of interest for the process covered by the use case. Since Port of Koper/Luka Koper already has detailed maps of its area, these maps are expected to be utilized in the optimization algorithm. However, in case of potential issues/incompatibility, the other option is open street map which can also fulfil the requirements.

During the use case development, it is also expected that the map will be further enhanced by identifying additional areas of interest such as traffic congestion areas.

Emulations

Due to the complexity of the use case scenario and limited physical resources, extended scenario involving more trucks, video cameras detecting traffic congestions and parking spaces could be achieved only by emulating them, thus also demonstrating and validating Nephele approach to VO, cVO and data communication orchestration. “Field” trial version, using the available devices in the port, will be therefore augmented with the emulated version.

Optimizing routing of containers

Containers routing optimization application will finally take into account all available and relevant data collected from multiple sources and will propose optimal routes for each single container that needs to be routed/moved from terminal yard to specific container freight station or vice versa. Based on past statistical facts and temporal conditions, the algorithm will propose schedules and monitor its execution in order to adapt schedules for the containers which has not yet entered into the routing process. Through interations, it is expected the algorithm will also optimize itself.

2.5 Containers routing optimization application components

As discussed on the project’s proposal, the main objective of NEPHELE’s UC2 (Port of Koper) is to create a cloud-based application for scheduling the daily internal container deliveries, starting from the Koper terminal to internal warehouses of the port. In particular, the corresponding development activities will focus on an algorithmic/AI-based solution for allocating container transportation tasks to (Luka Koper LL-owned) vehicles in an optimal way, meanwhile being able to dynamically adapt the original deliveries schedule within the day, according to real-time observed travel delays or sudden disruptive events (e.g., vehicle failures, blocked routes or prolonged vehicle stay in a delivery/unloading bay).

The foreseen solution will comprise of a Logistics Scheduling Agent able to generate optimized delivery schedules in terms of (i) reducing the total deliveries execution times; (ii) optimizing vehicle usability and capacity exploitation; and (iii) reducing CO_2 emissions generation, if conditions allow. Therefore, the designed solution will consist of three (3) main application components, briefly described below:

- **Real-time adaptive logistics scheduling Agent:** This is the main application component of the optimization solution. The "agent" will constitute a pre-trained entity, which will be able to (i) communicate with the Port Information System (PIS), i.e., its corresponding ERP part; (ii) receive the daily list of containers that need to be transported to internal warehouses (backlog); and (iii) generate the optimized deliveries schedules (carrirers' tours inside the port facility). Moreover, after each step (e.g., completion of delivery or return of a vehicle in the terminal area for reloading), the originally defined routings (start of business day) may be rescheduled according to current traffic conditions and delays observed.
- **Backlog API handling freight data from terminal's ERP:** This Application Programming Interface (API) will be responsible for pulling the daily tasks (i.e., the first 100 containers that must be delivered each day) from the ERP system of the Luka Koper terminal, and creating a backlog list of remaining tasks, which will be the fed on the input of the scheduling/routing optimization agent.
- **Traffic delays API receiving real-time traffic data:** This API will run periodically or upon the realization of certain events (e.g., delivery completion or vehicle return). The API will act as a traffic information receiver who will direct this information to the scheduling agent, so that the second is able to update the daily deliveries accordingly. Traffic delays may be received either by making use of commercially available cloud services (e.g., TomTom/Google Maps), or from analyzing data (e.g., frames) from related infrastructure devices (e.g. machine vision cameras in crossroad points, if applied) within the (private use or public) roads of the port.

Figure 52 depicts the high-level scenario of the containers routing optimization process including application graph of data collecting and forwarding them to containers routing optimization application with the goal of processing them and reporting results (containers routing schedules) to the customer.

Data will be collected by video cameras, IoT sensors and queried from Port Information System. A VO would be deployed for every camera and every OBU/GW equipped with various sensors. Also, a VO will be deployed and adjusted to extract data related to port trucks from the port information system (i.e., OBUs not directly accessible). Considering certain requirements for data types provided through the VOs (e.g., video stream), additional network related configurations, such as one for QoS, will be applied. Based on the data types provided by each sensor, corresponding data processing component will be applied (e.g., image/video extractor and detector of certain scenes, sensors' data collector) either at the far-edge, edge and/or cloud. Pre-processed data and data from port information system will be feed into the route optimization engine which outcome will represent a ground base for deciding on containers routing schedule ("dispatch decision making" component). The latter will be then distributed to the customers. Since the final decision on considering and applying proposed schedules to physical containers routing process depends on the customer/dispatcher, the "containers routing optimization" process keeps collecting and processing data in real-time which also enables adapting containers routing schedules to be updated in real-time.

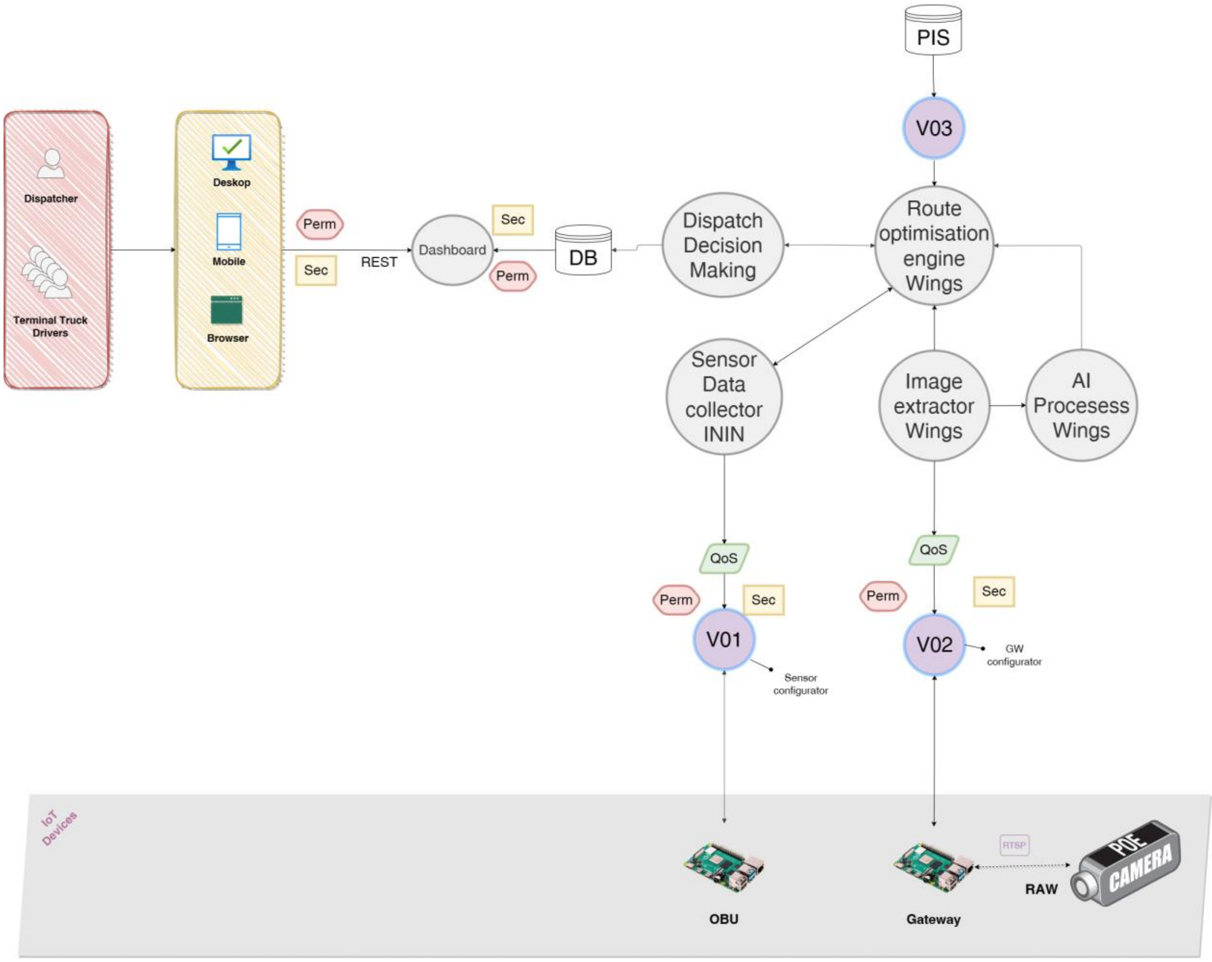


Figure 52: UC2 - Containers routing optimization application graph (high level).

2.6 Use case demonstration

The use case will be developed and first demonstrated as an emulated use case to involve more factors which impact the use case in its reality. To provide a realistic port environment and model it as accurately as possible (incl. traffic conditions within the port environment), SUMO[14], a well-established urban traffic simulator, will be employed (Figure 53). The SUMO simulation will provide device (truck, cargo container, etc.) location data to a Mininet emulation of a subset of the NEPHELE system (IoT, VO, and cVOs), allowing to test a variety of network protocols in a range of conditions.

[14] SUMO, https://www.eclipse.org/sumo/

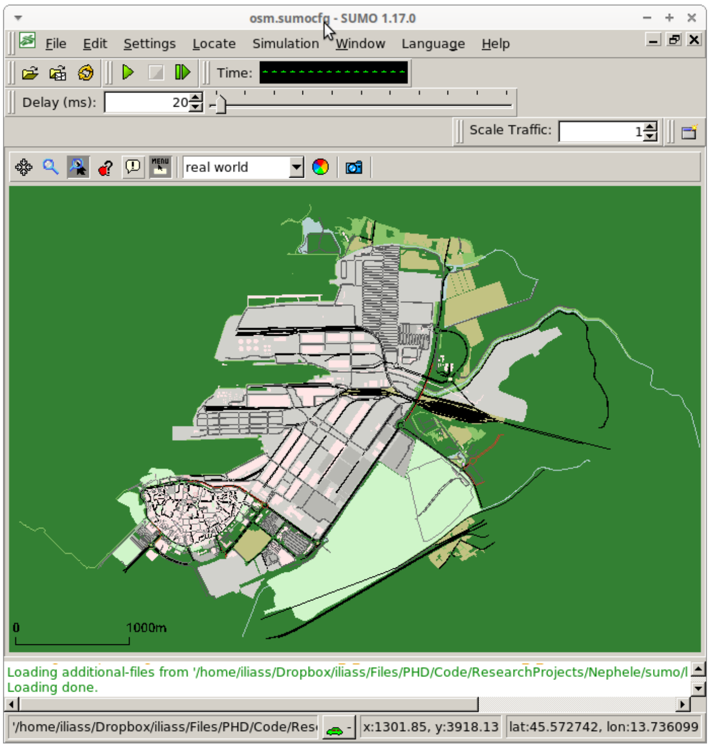


Figure 53: UC2 - Simulation of the port (Luka Koper/Port of Koper) in SUMO.

The simulation will validate the methodology and concepts and will mimic, among other things, the final tests that will take place at the end of the project. This version will be then deployed in reduced form to the field trial which will include physical devices deployed in the port (Luka Koper/Port of Koper) where final demonstration will be performed.

2.7 Data processing requirements

Data-processing requirements typically fall into two classes: system-oriented and user-oriented. System-oriented requirements measure the amount of information that your systems process. By contrast, user-oriented requirements measure the impact of data-processing services on the user. Service-level agreements reflect these expectations of performance.

In Table 16 we report the main data processing requirements (DPR) for the UC2.

Table 16. UC2 - NEPHELE's data processing requirements

ID	Type	Requirement	Description
DPR_UC2_01	User-oriented	Support multiple types of data sources	Data collection from different sources/devices should be supported to detect critical traffic conditions in the port (e.g., congestion, collision, accidents).
DPR_UC2_02	User-oriented	Concurrent data sources	Next to DPR_UC2_01, the solution will require adequate number of concurrent data sources of all types to provide expected results, i.e., optimized containers routing.
DPR_UC2_03	User-oriented	Rapid response	In order for the containers routing schedule to be updated in real-time and for the truck drivers to have ability to react in timely

			manner, use case requires rapid response from all components involved.
DPR_UC2_04	System-oriented	Storage	Data collected by sensors, video cameras, and emulation output data should be stored for real-time processing purposes and for later analysis. Expected sizes will be determined at later stages of the use case development.
DPR_UC2_05	System-oriented	Bandwidth in radio network	Bandwidth required for video streaming is >= 20 Mbit/s per single video stream. Bandwith required for sensors data (e.g., OBU) is 1Mbit/s per device.
DPR_UC2_06	System-oriented	Round Trip Time	RTT <= 20 ms
DPR_UC2_07	System-oriented	Packet Loss rate	Minimum packet loss rate required to assure the proper service operation (e.g., video streaming) should be < 10^{-4}.
DPR_UC2_08	System-oriented	Security, privacy	All sensitive data should be processed according to GDPR and local legislation, as well, the complete system should be designed considering best cyber-security practices.

2.8 Use case requirements analysis

The following subsections provide an analysis of functional, non-functional and system requirements for the use case.

Functional Requirements

presents a description of the functional requirements for the UC2, as well as some comments on how the requirement will be addressed.

Table 17: UC2 - Functional requirements

ID	Description	How to address	Priority
FR_UC2_01	Track a truck.	Use data collected by OBU.	High
FR_UC2_02	Map position of trucks in real-time.	Use data collected by OBUs.	High
FR_UC2_03	Navigate truck driver.	A GUI is needed.	Medium
FR_UC2_04	Detect traffic congestion.	Use cameras mounted on poles alongside port roads.	High
FR_UC2_05	Detect free parking lot at a container freight station.	Use cameras surveilling parking areas.	Medium
FR_UC2_06	Suggest de-tour (e.g., in case of traffic congestion detected en-route).	Use data available, a GUI is also needed to present de-tour route to the truck driver.	Medium

FR_UC2_07	Predict train schedules.	Use data provided by the port information system.	Medium
FR_UC2_08	Schedule containers loading and unloading to the truck.	All data available will be needed.	High
FR_UC2_09	Collect live data-streams from devices.	Cameras will produce data streams for the system.	High
FR_UC2_10	Distinguish between critical and non-critical network traffic.	Enable support of TSN capabilities on the 5G IoT gateway	High
FR_UC2_11	Ensure API specification and protocol integration between the VOs and the devices (i.e., trucks, cameras)	Address sensors, cameras, and other data sources.	High

Non-functional Requirements

Table 18 provides a description of the non-functional requirements for the UC2 and how they will be addressed.

Table 18: UC2 – Non-Functional requirements

ID	Description	How to address	Priority
NFR_UC2_01	Ensure privacy	AAA techniques will be implemented.	Medium
NFR_UC2_02	Guarantee security of data storage and processing	Secured and authorised access to the system will be implemented.	Medium
NFR_UC2_03	Delegate computational calculation to the Edge and Cloud	Vertical offloading.	High
NFR_UC2_04	Be resilient, efficient, lightweight and with flexible design	Adopt containerized and cloud-native principles ready for orchestration.	High
NFR_UC2_05	Be energy efficient	Task assignment and sampling frequency based on battery level in the device.	Medium
NFR_UC2_06	Balance processing load to meet required responsivness (latency, RTT) of the system.	Data processing services should be distributed along complete cloud-continuum.	High
NFR_UC2_07	Ensure uninterruped communications among use case components in the port area.	Use 5G non-public network deployed in the port area.	High
NFR_UC2_08	Ensure easy adding of new IoT devices to the network.	Use cloud-native (containerized application) software support for each single IoT device.	High
NFR_UC2_09	Ensure adequate resources for applications	Use network slicing and orchestration techniques for resource-optimization	Medium
NFR_UC2_10	Ensure very low latency and high bandwidth	Use 5G communication techniques, edge computing and low latency / TSN communication protocols	Medium

NFR_UC2_11	Support reactive multi-hop / ad-hoc routing schemes	Enable IoT devices to communicate through relaying nodes to reach to the Internet based on dynamic routing	High
NFR_UC2_12	Associate devices with VOs on-the-fly	Utilize resource discovery approaches for the IoT device-VO associations, e.g., based on clustering	High

System Requirements

Table 19 provides a description of the system requirements for the UC2 and how they will be addressed.

Table 19: UC2 – System requirements

ID	Description	How to address	Priority
SR_UC2_01	Have enough computational power	For the execution of AI/ML methods, sufficient memory and processing power is needed	High
SR_UC2_02	Have internet connectivity	To offload computation if needed.	High
SR_UC2_03	Have enough storage capacity	To store and process the data coming from various data sources. Cloud storage will be used	High
SR_UC2_04	Provide virtualization/containerization capacities	The system will be virtualized and re-deployable in containers. Thus, the processing unit should have virtualization capabilities and be optimized for container virtualization.	High
SR_UC2_05	Have enough GPU capacity	In case GPU capacity will be needed	Medium
SR_UC2_06	Have proper AI models for edge	This will be provided as part of the VO-stack	High
SR_UC2_07	Have enough bandwidth	Required for video streaming.	Medium
SR_UC2_08	Have proper input data	Provide access to the port information system and required data.	Medium
SR_UC2_09	Have proper IoT sensors	To be able to collect relevant data required for the use case process.	High
SR_UC2_10	Video cameras are surveilling targeted areas	To collect required video streams.	High
SR_UC2_11	Provide TSN capabilities on the IoT Gateway	The system will handle different traffic classes with different prioritization. Thus, the IoT gateway should utilize TSN-based schedulers, such as TAPRIO.	Medium

2.9 NEPHELE's innovation for the use case

Table 20 summarizes the requirements of the "Container Routing Optimization" UC2 and discusses the limitations that NEPHELE aims at overcoming and thus facilitating the realization of the Use Case. For each requirement, we provide a reference to its definition, which can be found in Tables 17, 18 and 19.

Table 20: UC2 – Requirements to demonstrate NEPHELE innovation

Requirement	Reference ID	Current limitation	Innovation
Efficiently collect relevant data required for containers routing process	FR_UC2_01 FR_UC2_09 FR_UC2_10 NFR_UC2_01 NFR_UC2_02 NFR_UC2_05 NFR_UC2_07 NFR_UC2_08 NFR_UC2_10 NFR_UC2_11 NFR_UC2_12 SR_UC2_01 SR_UC2_02 SR_UC2_04 SR_UC2_07 SR_UC2_08 SR_UC2_09 SR_UC2_10 SR_UC2_11	Although many data are already possible to be collected, it is still challenging to merge them together and use them as a unified input to AI/ML algorithms.	By applying VO-stack, a convergence layer can be established, enabling more efficient data collection and initial processing.
Efficiently process data within complete cloud continuum	FR_UC2_02 FR_UC2_03 FR_UC2_04 FR_UC2_05 FR_UC2_06 FR_UC2_07 FR_UC2_08 FR_UC2_11 NFR_UC2_01 NFR_UC2_02 NFR_UC2_03 NFR_UC2_04 NFR_UC2_05 NFR_UC2_06 NFR_UC2_07 NFR_UC2_09 SR_UC2_01 SR_UC2_02 SR_UC2_03 SR_UC2_04 SR_UC2_05 SR_UC2_06 SR_UC2_07	Balancing of data processing among different nodes of cloud continuum (cloud, edge, far-edge), while maintaining responsivness and energy efficiency is still challenging task.	The VO stack in cooperation with cloud continuum approach and distributed AI/ML techniques will allow for efficient and effective processing of data aiming at improving responsivness and energy efficiency of the system.
Optimize containers routing process in a port	NFR_UC2_04 NFR_UC2_05 FR_UC2_06 FR_UC2_07 FR_UC2_08	Although various solutions of optimizing routing processes within a port exist, they produce partially useful results	The solution provided in Nephele will be developed and tested in a specific environment and therefore adapted to its challenging

	FR_UC2_09 SR_UC2_08	from the customer's view. Thus, customers believe there is a room for further optimization of the process.	conditions, thus identifying insights on certain issues that might be overlooked in general solutions of such kind.

3 Use case 3 in NEPHELE

3.1 Introduction and objectives

The physical increase of cities and their population and the continuous advancement of technology motivate the need and the increasing popularity of the concept of smart buildings/cities. If we put the focus on sustainability and the need to reduce carbon emissions, it has become increasingly important that buildings are designed and operated in a way that minimizes energy consumption. Thus, energy efficiency is a crucial aspect of smart buildings and cities. The integration of cutting-edge technologies such as IoT, machine learning and edge computing can make them even smarter, more efficient, and more sustainable.

One of the keyways that smart buildings achieve energy efficiency is using sensors and automation systems. Sensors can be used to monitor environmental conditions like temperature, humidity, and light levels, and this data can be fed into automated systems that adjust heating, cooling, and lighting to maintain optimal conditions while minimizing energy consumption.

In this sense, smart buildings are prepared to monitor and control energy use in real time to guarantee the desired energy efficiency. By collecting energy use data and analyzing it in real time, building operators can identify areas where energy is wasted and take actions. This can be accomplished in a reduced amount of time, thereby achieving an efficient energy consumption.

With smart buildings, we can automatically adjust the use of heating, ventilation, and air conditioning systems, turn off lighting or other electrical devices when they are not in use, or even implement smart systems that allow us to dynamically adjust energy use based on the demand.

The integration of IoT to edge to cloud computing in smart buildings is thus important for energy efficiency reasons. By processing data closer to the source, edge computing can minimize the amount of data that needs to be transmitted to a centralized server, reducing energy consumption associated with data transfer and processing, too.

With the integration of the IoT, edge computing, and cloud computing, the possibilities for intelligent monitoring and remote energy management in these environments are expanding. This is where UC3, with focus on energy management in smart buildings/cities, comes into play. Led by ODINS and supported by SIEMENS and IBM, this use case aims to design, develop, produce, and market products that leverage the entire IoT to edge to Cloud Continuum, to better address strict Service Level Agreement (SLA) requirements for the development of smart energy solutions. By implementing an automation scheme that gathers real-time information from a variety of IoT devices, such as appliances, sensors, and HVAC systems, along with edge nodes that instantiate Virtual Objects (VOs), this approach avoids bottlenecks caused by placing all the intelligence in a centralized Smart Building/City monitoring and control system, delving into the technical challenges, applications, and benefits of this innovative energy management solution.

The objective of UC3 is to develop different advanced applications and services leveraging on the VOStack, to manage control actions of building equipment, providing the user with customized services for energy-efficient, well-being and comfort, covering security aspects, too. The aim of this use case is also to show some of the security features that NEPHELE will offer, such as secure and authenticated access, secure and distributed access sharing of data, as well as higher level applications such as detection of people or objects.

3.2 Case Study: Energy management in smart buildings/cities

In this use case, the technologies and solutions will be adapted for an energy management scenario focus on smart building, but that can even be exported to smart city scenarios.

In Figure 54 we summarize the main stakeholders, the location (physical or virtual) and the constraints, challenges, and risks for this case study.

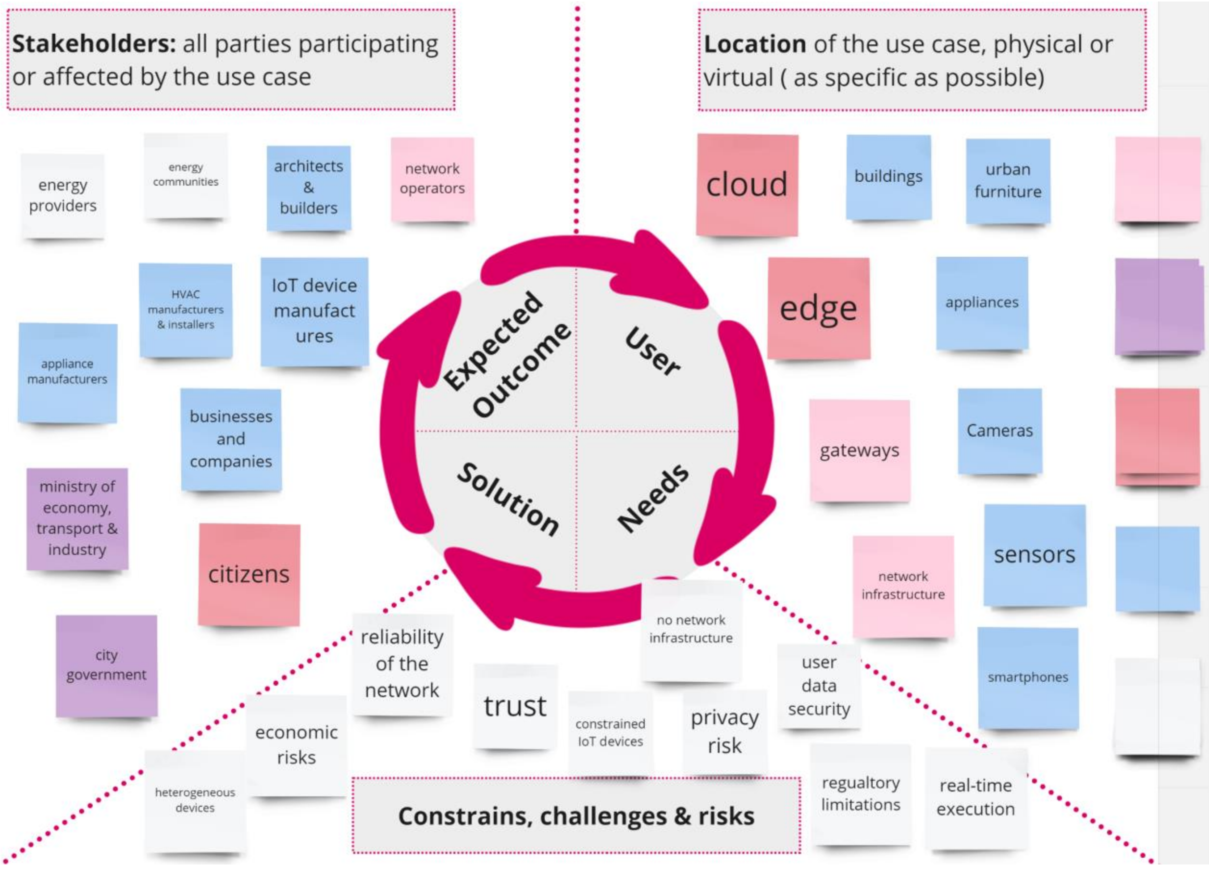


Figure 54: UC3 - Stakeholders, location, constraints, challenges, and risks for the use case

Stakeholders

Several stakeholders are involved in energy management solution in the context of smart buildings and cities. From the architects who design intelligent buildings to the citizens who finally live in them, there are different profiles that can benefit from the advantages that NEPHELE offers. Other examples are IoT providers for monitoring and actuation, household appliance manufacturers, network operators that connect the deployed systems and the energy operators, as well as governors, legislators and those responsible for building management as energy communities.

In Figure 55, we report a detailed view of the main users for UC3 with a high-level analysis of the needs, functionalities and expected outcomes for each of them.

Citizens are the main users of buildings and cities, and as such, they are the ones who end up choosing which solutions best suit their needs. Among the main features they are looking for are real-time monitoring and decision-making, AI-assisted information analysis systems, as well as security and control of access to information to reduce costs, improve efficiency as well as simpler tools.

Local governments, in charge of promoting the deployment of solutions that optimize the use of resources to improve the services offered to citizens, are looking for tools that allow them to monitor and analyse energy consumption in real time, as well as intelligent mechanisms that allow them to take assisted and automated decisions, while guaranteeing users the highest standards of privacy and information control. Among the advantages they seek in new technologies is edge computing, which allows optimizing the use of the network, distributed processing, and latency reduction, allowing real-time communications. AI-assisted decision-making allows governments to offer better and more efficient services, while authentication and authorization mechanisms allow sharing the minimum necessary information and controlling who and for what.

Building managers and energy communities seek to improve the use of resources and monitoring systems for production and consumption as a starting point for offering more advanced services. Among the fundamental aspects that must be offered are the security and privacy of user information, real-time monitoring and data analysis, and AI-assisted decision-making systems, to offer an intelligent management solution on an interface intuitive and accessible.

Another of the stakeholders that arise from this use case are security companies that can use security devices and cameras to control access to buildings or help with locating tasks when a person disappears. For these scenarios, in the first place, it is necessary to be able to process the images in a distributed way, analysing the data and monitoring other sensors to improve computational efficiency. In addition, exquisite care of sensitive information and the privacy of users is essential. For this, the proposed solution must offer security, distributed analysis tools, real-time access to data and assisted decision tools. In addition, interfaces for access to information and tools should be offered, as well as advanced object detection tools.

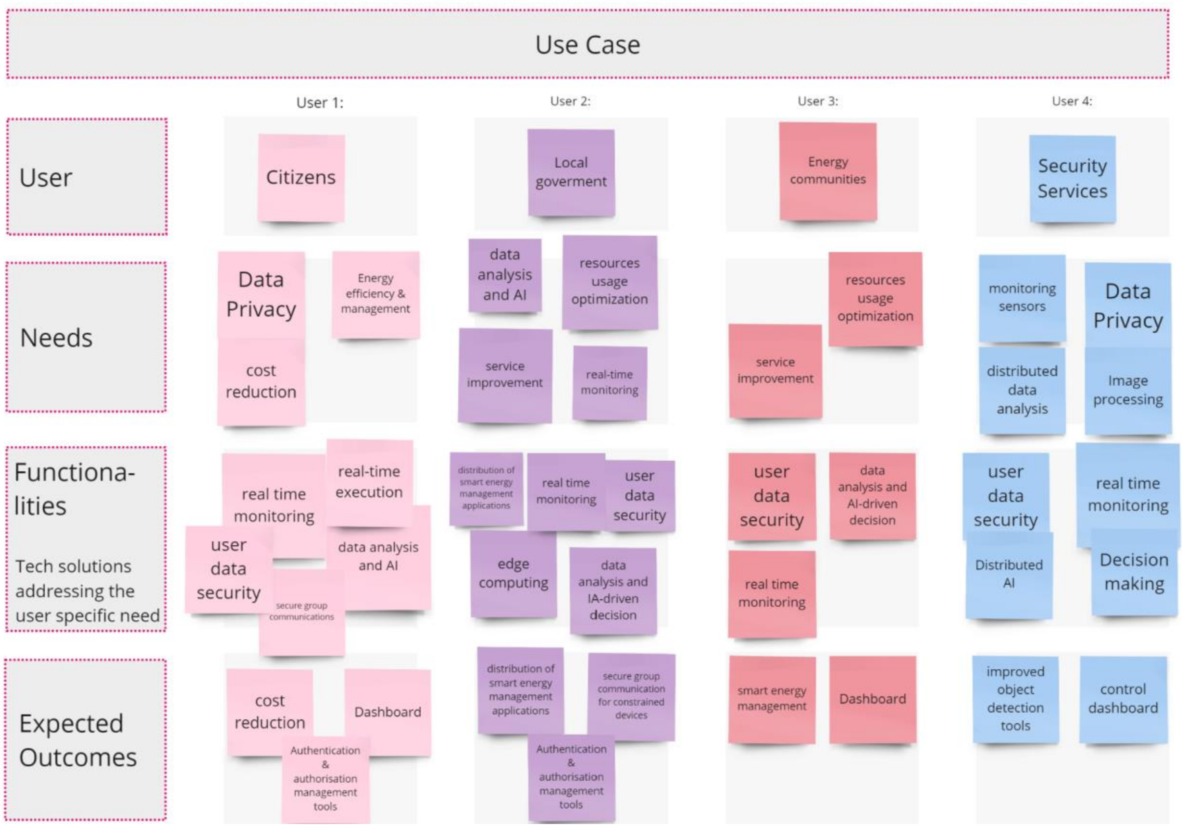


Figure 55: UC3 - Needs, functionalities and expected outcomes for the main users in the use case

Location

UC3 contemplates different areas of location, which can range from its adaptation to smart city (buildings, urban furniture, lighting systems, traffic lights, etc.) to smaller settings such as a building or offices. The different proposed applications contemplate not only the reduction of consumption at the level of ventilation systems but also the optimization of the use of IoT device connection interfaces or the improvement of image analysis systems thanks to the consideration of additional information provided by other sensors apart from the cameras.

Focusing on Figure 56 and regarding the physical layer, wireless sensors and actuators are used in smart buildings for obtaining the real-time measurements of the environmental conditions as temperature, humidity, and light levels. Similarly, other IoT elements such as smart plugs, relays, and HVAC systems, or even the configuration interfaces of the IoT devices themselves, offer the ability to act on energy consumption systems.

Regarding network infrastructure, the existing IoT and Edge/Cloud computing infrastructure is mainly composed by wireless microcontroller IoT devices communicating with Edge nodes and cloud platforms that offer higher compute and storage capacities and fewer power consumption constraints. At the top of continuous processing is the cloud tier that supports edge and IoT systems by offering more centralized and powerful control.

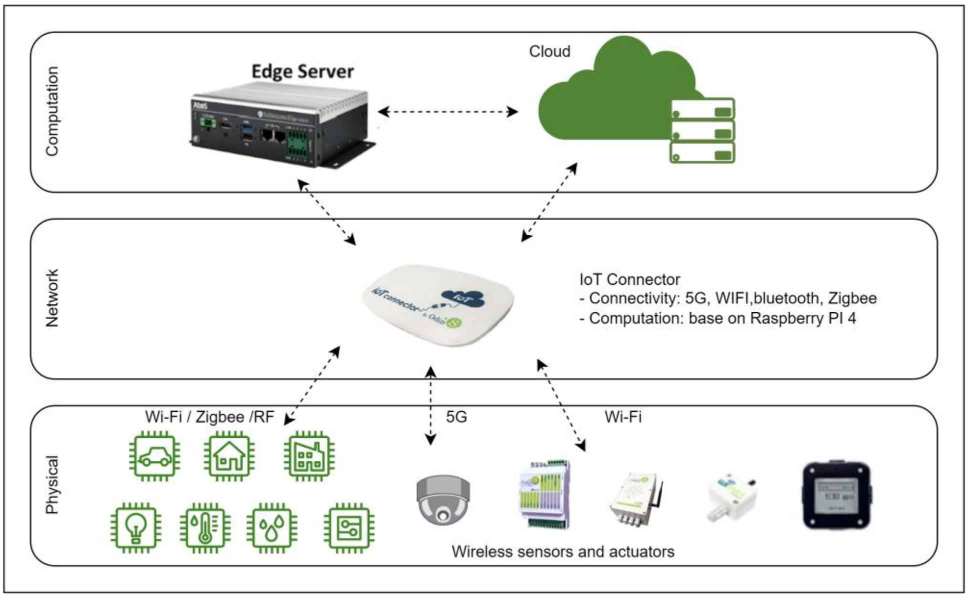


Figure 56: UC3 - Devices at the physical, networking and computation levels

Constraints, challenges, and risks

As reported in Figure 54, this study case carries with it several inherent constraints, risks, and challenges as detailed next.

Risks

- Privacy risk: When monitoring and processing information, it is essential to guarantee privacy and the proper use of data, both using the minimum information necessary and only using the information for which permission was given. The necessary security mechanisms must be designed and implemented so that users trust the solution.
- Economic risks: energy efficiency directly affects the money that users must pay for it, as well as the cost of associated services or that make use of energy. A reduction in consumption peaks has a reduction in the costs of the necessary power. Efficient management of the source of energy, consuming, for example, the cheapest sources at each moment, also has a direct impact or may, on the contrary, imply an extra cost.

Constraints

- To perform advanced analysis of the available data requires a high computation load which cannot be provided by simple and constrained physical devices.
- The privacy of the origin of the data and of the users must be always guaranteed.
- Regulatory limitations: The regulations and legislation of each country may limit the possibilities of monitoring and use of information as well as the mechanisms of exploitation and use of energy.

Challenges

- Guarantee the obtaining of data in real time and the performance of actions.
- Guarantee the security of the communications and the privacy of user and building data.
- Network coverage should be reliable and always available.
- The devices/sensors used in this use case, provided by different manufacturers, are very heterogeneous in nature. These devices differ in hardware, software, and communication protocols used which requires integrations efforts in the NEPHELE environment.
- Development of friendly high-level software for monitoring and maintenance of intelligent buildings.

3.3 Technical requirements and challenges

To best face the needs of the Use Case #3 operations and offer solutions to reach the overall goal for the solution we can identify the following main technical requirements and challenges.

- **Software component orchestration**: Considering the introduction done in Section 3.1, several components will be required to provide de applications and services that have been used along the use case. To work together, it is necessary to provide resource monitoring at several levels together with mechanisms to coordinate and orchestrate this cooperation in the continuum.
- **Device customisation and management:** Devices or edge nodes may be needed to be re-configured or updated. It is desired that an IoT device may be extended with a set of functions. An execution of a virtual function (e.g., provided in a form of Complex Event Processing rule or a Neural Network) on a device would turn it into an intelligent IoT device.
- **Device Interoperablity:** The access to VOs and their data should be provided via a standardized interface, e.g., W3C Web of Things. The functionality of IoT devices should be exposed via a standardized and semantically enriched interface at the VO level. This will **e**nable interoperability at the protocol and data level between diverse IoT devices and NEPHELE applications.
- **Control Access Management**: The access to resources, services or applications must be protected and controlled using access control policies that support distributed scenarios is required.
- **Identity Management:** the interaction between devices and services, the access to device data and more complex scenarios must be secured through advanced authentication mechanism that focus on privacy preserving mechanisms that allow controlled and limited disclosure and access of user or device attributes.
- **Data storage:** A distributed data storage system is necessary to store and share common information such as service public information, certifies, service policies or DIDs in the specific case of identity data.
- **Low latency communication**: real-time video requires high bandwidths and low latencies for quality streaming. Likewise, the management and monitoring of electrical consumption must be carried out over a stable connection with low latency to adjust the reaction of the management tools to the maximum.
- **Computer vision for information extraction**: Persons and objects detection, their position and location from picture and video data are all made possible by AI and computer vision.

- **Intelligent data filtering/aggregation/compression**: A large amount of data will be collected from sensors and cameras in the UC3 environment. Some of them can be filtered out, others can be reduced or aggregated before sending them to the edge/cloud. Smart policies need to be defined to also address the high degree of data heterogeneity.

3.4 Applications and services of the study case

This use case will provide several applications and services as detailed next.

Secure group communication

This first application is used to provide secure attribute communication between two entities in a secured way, with the aim of offering minimum disclosure technics base on the use of W3C verifiable credentials. In this case the communication is done from a device to another device, but the target objective could be also an application or a service. This kind of protocols and other secure group communication protocols (e.g., CP-ABE) are computationally expensive and, in most cases, cannot be adapted to constrained IoT devices due to their computational limitations. Thanks to the VO in Edge nodes, the possibilities of instantiating security functions are much wider.

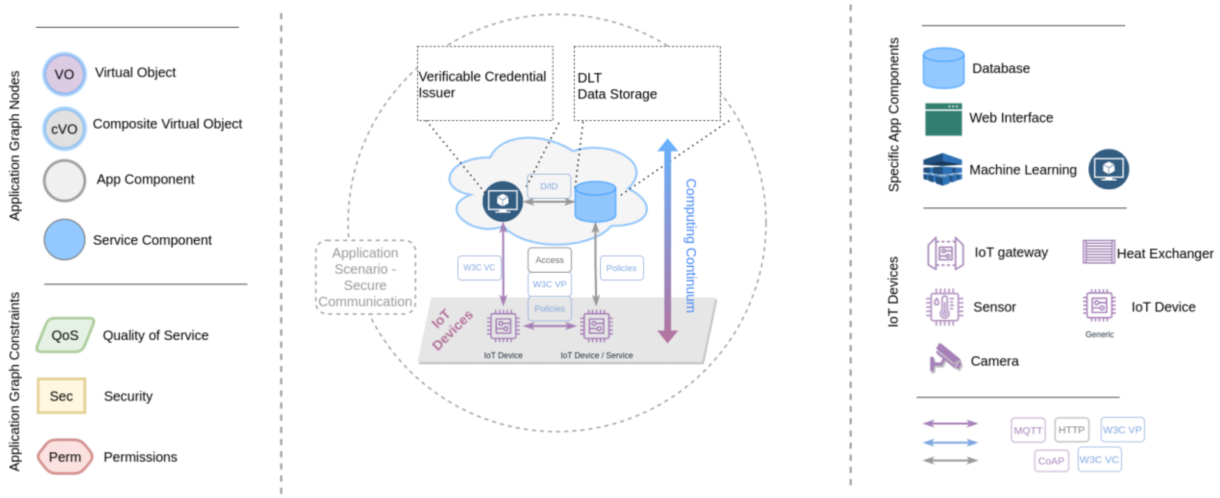


Figure 57: UC3 – Secure group communication – High-level

The translation of credential emission and use scenario, are very simple making use of Virtual Objects (VOs). For each device, a VO will be deployed in the edge, which provides more computation and storage capacity. VO1 deploys an identity agent component that interacts with App1 to obtain a new credential. App1 plays the role of credential issuer, signing the credential and sending it to the VO1, and after it, the app publishes the associated DID (Distributed Identifier) on the DLT (App2). With the obtained credential, VO1 generate a Verifiable Presentation, with the specific information required for the communication with VO2 and sends it to VO2, who can verify it and use the VO1's attributes included in the presentation.

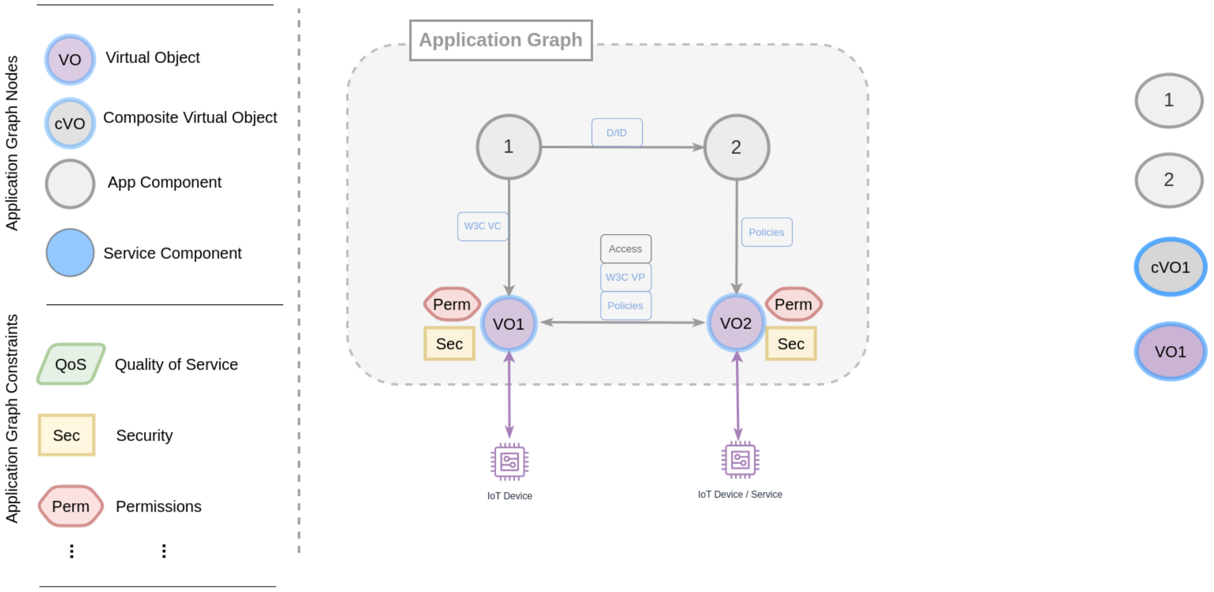


Figure 58: UC3 - Secure group communication - Application

Distributed complex decision making

This application scenario shows how VOs Edge nodes will perform intelligent energy saving actions not only based on the sensor measures collected by the IoT devices managed, but also the information and data coming from other nodes. This will drive the efficient use of renewable energy sources and the reduction of peaks in the energy consumption.

Figure 59 depicts a scenario where several temperature monitors and HVAC systems are controlled by the Smart Energy Balancer App (2) to avoid energy peaks. The objective is to control through the temperature and consumption sensors when the heating devices should be activated. To do this, decisions must be made based on multiple devices that must be able to coordinate. The system must also offer a management interface that allows the parameters to be met by the system to be indicated, such as the target temperature for each zone and the consumption limit established for the balancer.

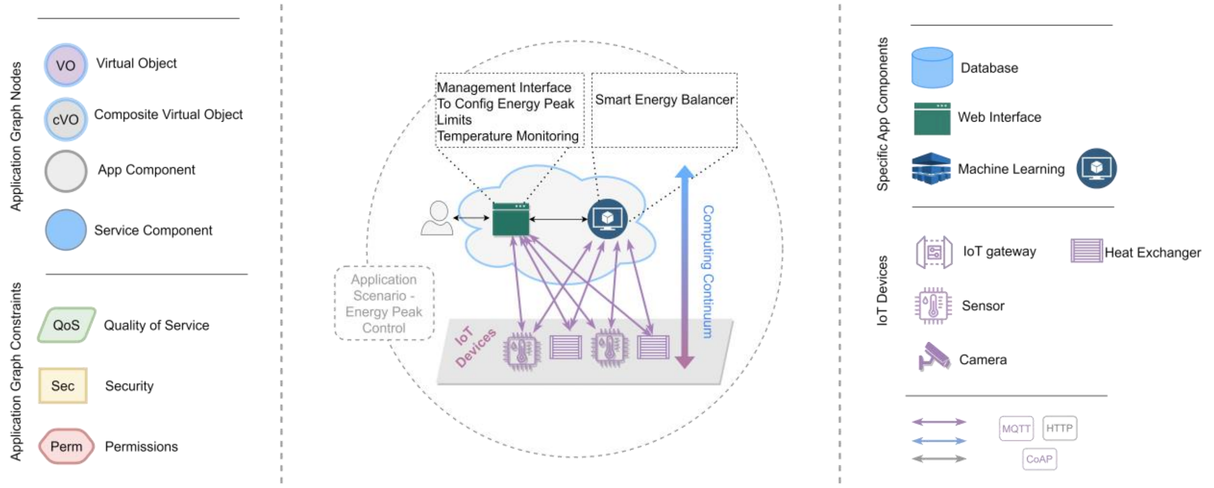


Figure 59: UC3 – Distributed complex decision making – High-level

Thanks to the virtual objects and the possibility of composing them into more complex ones, it is possible to design temperature sensor-HVAC pairs, which makes it possible to define more complex and intelligent virtual devices at a first level (cVO1 and cVO2), resulting in a system with HVAC with a thermostat. But on top of this level, it is possible to create a more advanced one that groups several intelligent air conditioning systems (cVO3) that interacts with the Smart Energy Balancer (App2), being able not only to work on temperatures but also to organize the switching on and off to limit the number of machines that are on at any given time, limit the power or prioritize one over the other depending on the complexity or intelligence that we want to implement on the composite system.

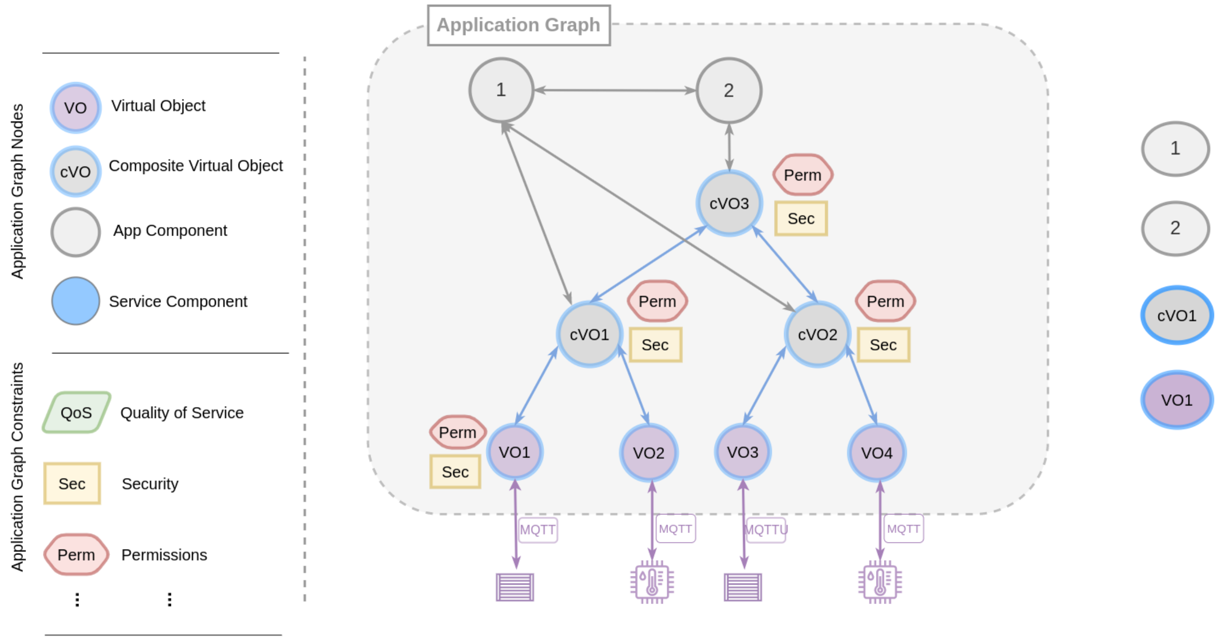


Figure 60: UC3 – Distributed complex decision making – Application

Distributed authorization scenarios

This application scenario shows where an access request to a resource is not decided exclusively by a centralized cloud platform but made by a back-end service leveraging in a Distributed Ledger Technology (DLT), that stores distributed access control policies, (e.g., distributed-XACML), employed at the edge nodes closest to the target resource to enforce access.

This application scenario shows, on the one hand, how policies can be configured by some entities in a distributed way using DLT, as well as on the other hand, how the policies are used in a distributed authorization process when a device tries to access a service using the application using the Service Access Control.

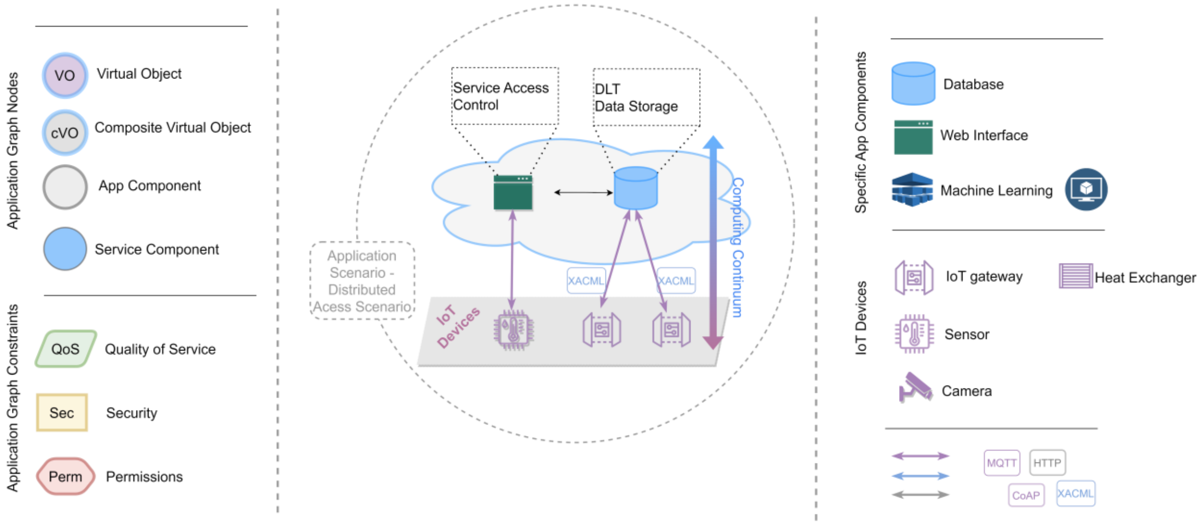


Figure 61: UC3 – Distributed authorization scenarios – High-level

The translation of this scenario to the field of Virtual Objects allows the handling of security policies and more complex and advanced credentials in the authorization processes. Application 1 offers an advanced access control system that retrieves access control policies from a DLT with several distributed peers (2 and 3). These policies can be updated on one of the DLT peers (3) which automatically updates and distributes the information to the others (2).

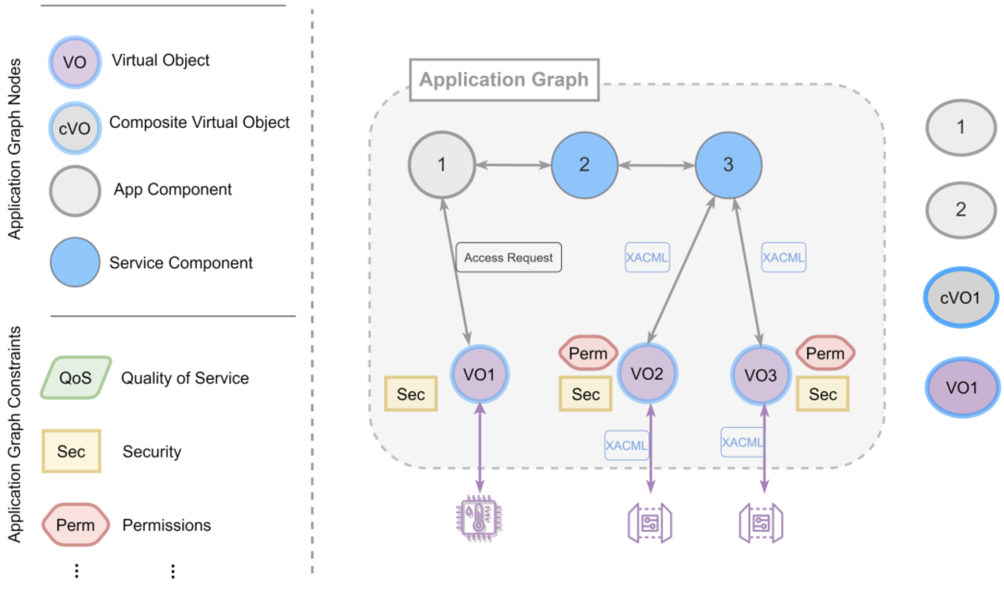


Figure 62: UC3 – Distributed authorization scenarios – Application

Object/Person detection

This scenario shows how to provide object and person detection through AI-assisted image processing tasks running in distributed Edge nodes. This is achieved from the data collected by video cameras deployed in the scenario for finding dangerous or suspicious objects, and getting the location of vulnerable persons, such as missing children or lost elders.

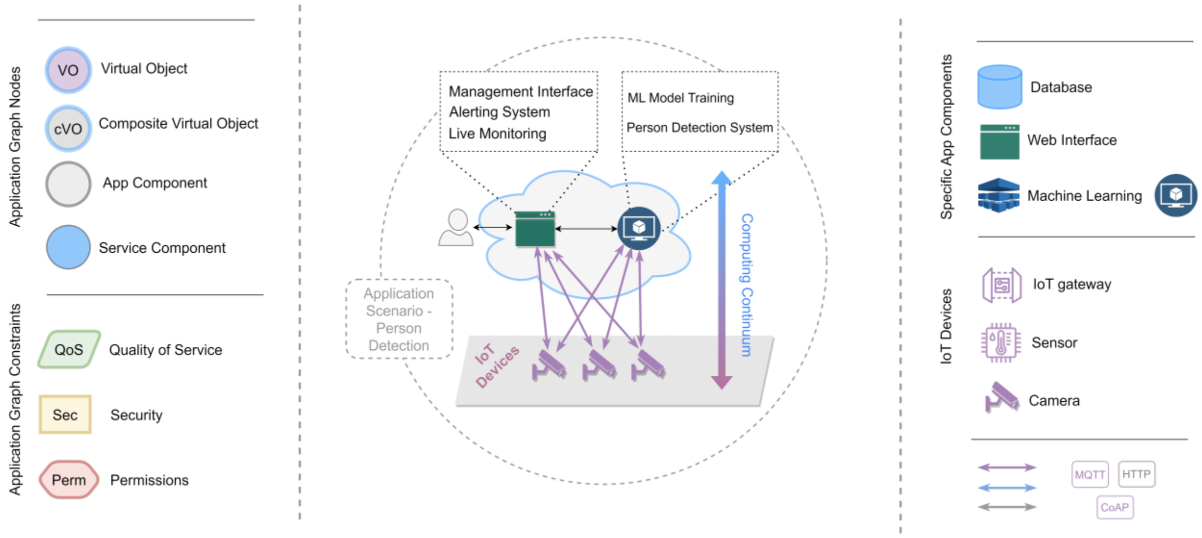


Figure 63: UC3 – Object/Person detection – High-level

The use of NEPHELE in this application case allows the definition of a complex virtual object (cVC1), which allows adding and processing several video streams at the same time, optimizing the processing and the required bandwidth, while allowing the application of quality control mechanisms. of service throughout all communication channels. cVO1 will offer an advanced interface to adapt to the people and object detection service based on ML/AI techniques deployed in 2. Application 1 will provide access to the management, monitoring and alert configuration interface that will act on the VOs of the cameras as well as it will allow to define the detection parameters in 2.

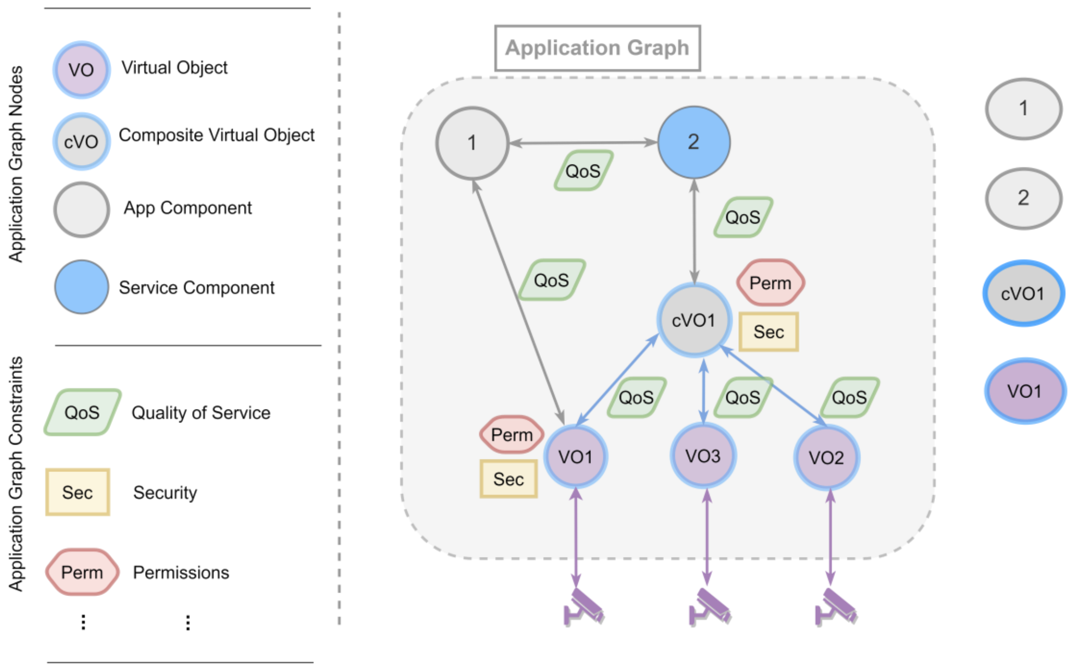


Figure 64: UC3 – Object/Person detection – Application

Communication radio offloading

This section shows how communication radio offloading for battery-powered devices or subscription base technologies can benefit from the use of virtual objects, they can be used to optimize the use of communication channels. Decentralized AI-assisted orchestration of VOs may avoid certain radio channels to save either IoT device battery or subscriber data, by offloading the communication flows to auxiliary technologies (e.g., switching from NB-IoT to WiFi access point).

In this application scenario, several devices are monitored to find out different parameters of energy consumption and available battery, as well as the available interfaces and the coverage they have. All this information is reported to the network monitoring nodes, which notify the distributed orchestration services of the changes. These should send instructions to the devices, indicating which interfaces they should use to properly manage network offloading.

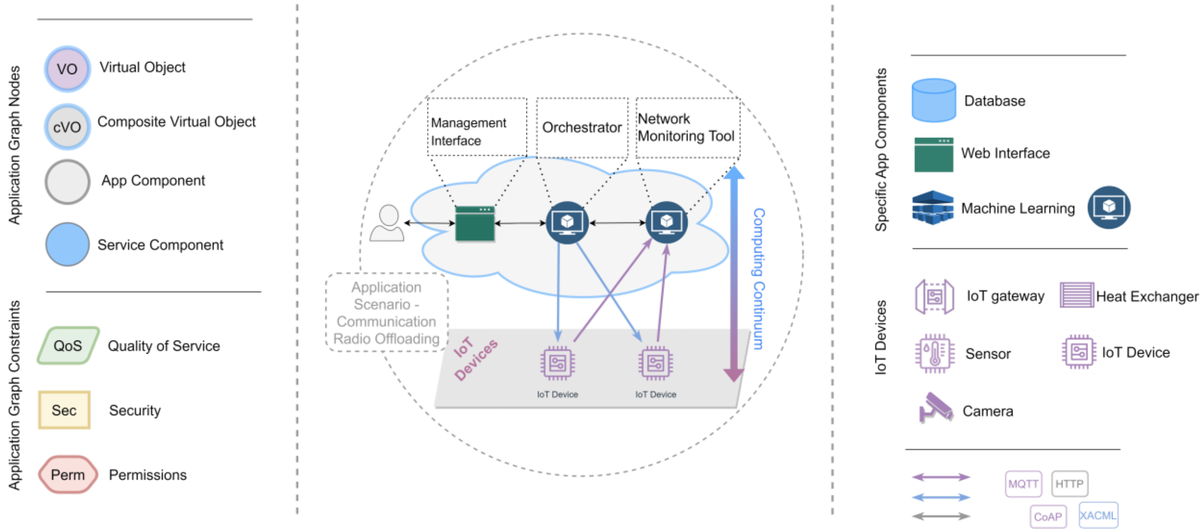


Figure 65: UC3 – Communication radio offloading – High-level

The use of virtual objects as a digital representation of the device allows communication to be established on the physical device in situations of unstable connectivity, even when communication is intermittent, acting as a cache for the instructions sent from the infrastructure, or also storing the record of the monitored data. In turn, the VO can become part of the distributed orchestration system and make local decisions based on the specific context of the device or devices it represents.

On the other hand, the scenario presents Application 1 with a management interface to configure the orchestration system and check the status of the system and the devices connected to it.

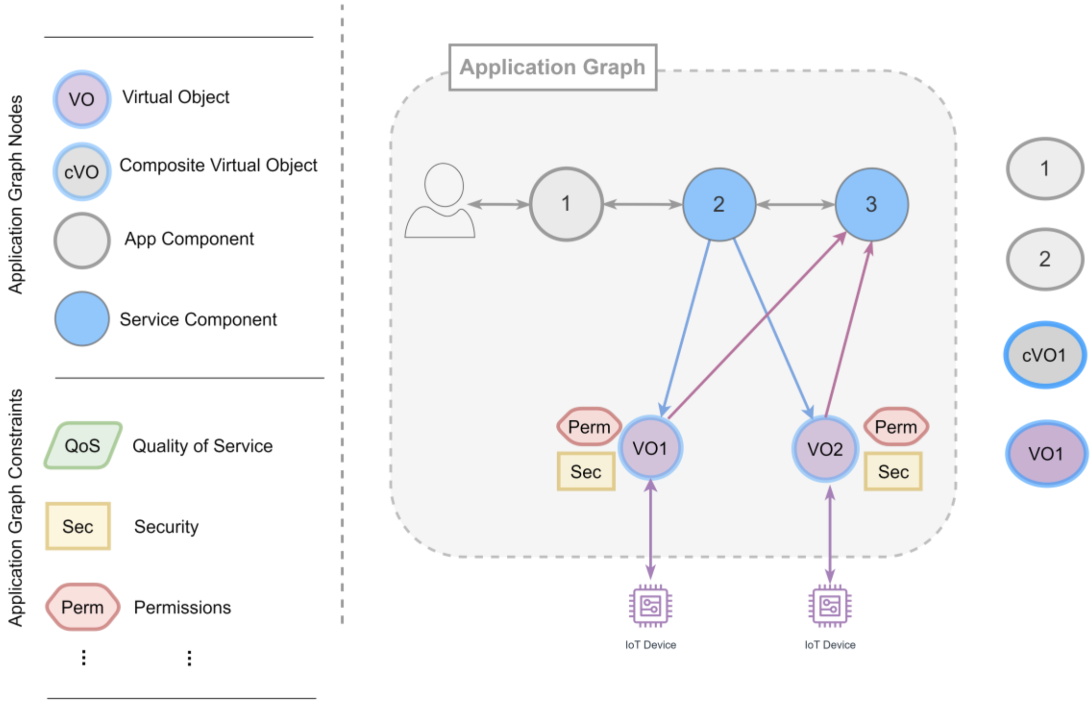


Figure 66: UC3 – Communication radio offloading – Application

Customizable IoT devices to support energy-efficiency and well-being in buildings

Building Automation Systems (BAS) has changed over time and thus need to be adapted. During their long lifetime their requirements change. For example, a high-energy efficient BAS turned not to be so efficient during the pandemic time or the post-pandemic time. Heating, ventilation, air quality and other building services designed prior to the pandemic may not be as energy-efficient nowadays. The reason for this is the occupancy of rooms, which in many cases has been changed. This application will demonstrate how the true presence in rooms can be easily determined and how that information can be used to make a BAS more energy efficient.

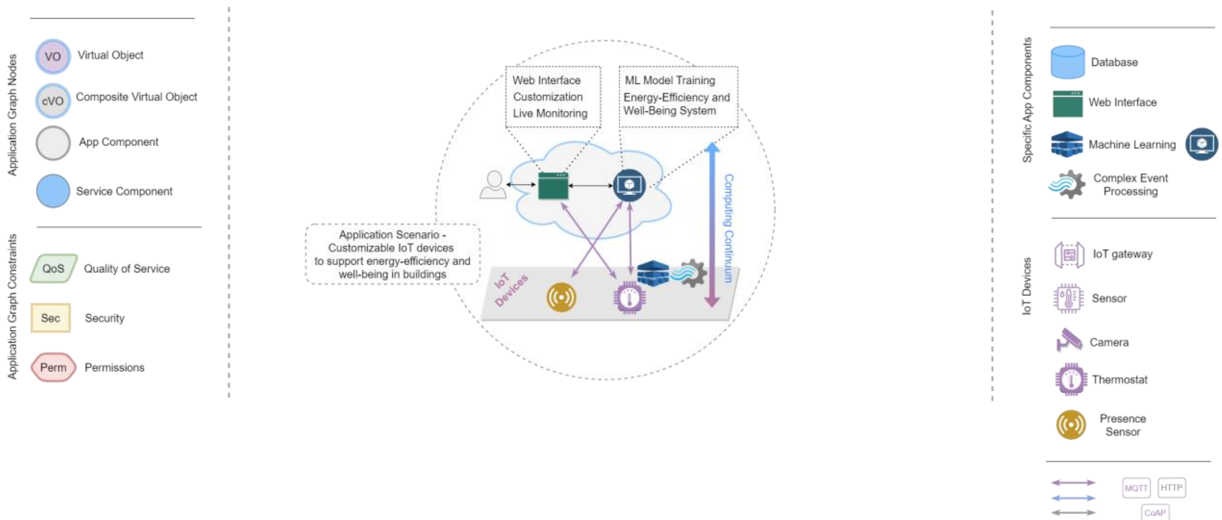


Figure 67: UC3 – True presence detection – Application – High-level

To achieve this, this application scenario will demonstrate two contributions of NEPHELE projects. First, new sensors need to be added and integrated into an existing BAS. This will be accomplished via the concept of the VO. In a plug and play fashion a VO needs to expose the functionality and data of a newly added sensor. Data access and semantics, which describes the device, should be standardized, e.g., with W3C standard Web of Things. Second, an existing device may need to be customized so that it can use the data from the new sensor. For instance, an existing thermostat will adapt its control based on the data from the new presence sensors. The thermostat can be customized to load and run a virtual function, provided in the form of Complex Event Processing rule or a Neural Network. The new function should be with no effort be exposed over an existing VO of that device. The concept of the VO will in this regard maintain the reality in terms of functionalities available in virtualized environments and play the role of the digital twin in the edge part of the continuum.

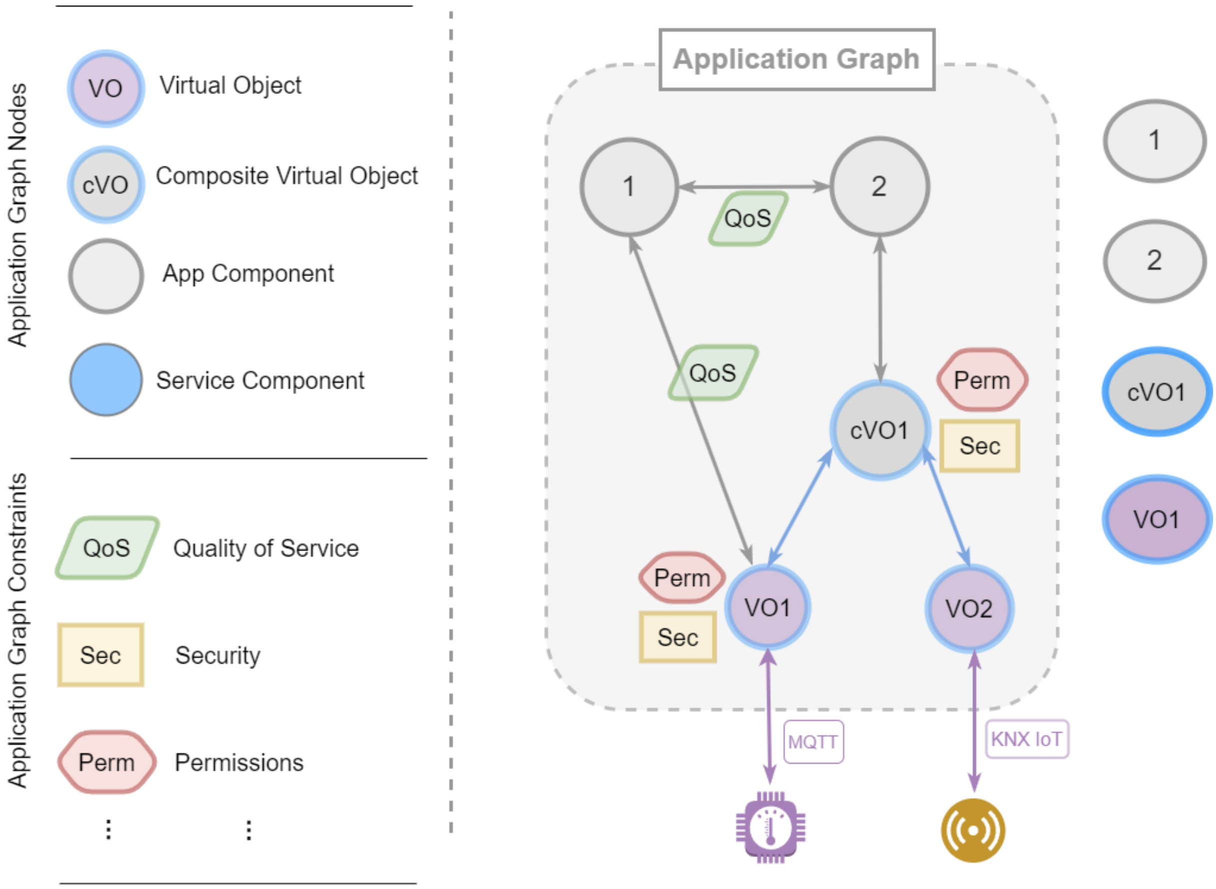


Figure 68: UC3 – True presence detection with customizable IoT devices – Application

Intelligence at both IoT device and VO level will be demonstrated, taking the advantage of decentralized AI and TinyML techniques. Decentralized AI can be applied for moving intelligence and learning at both VOs and IoT devices, while TinyML can support models that run on small, low-powered devices like microcontrollers and enable low-latency, low power and low bandwidth model inference at edge devices. This approach avoids bottlenecks caused by placing all the intelligence in a centralized Smart Building monitoring and control system. It also enables creation of complex virtual objects, which allows adding and processing several sensor data streams, optimizing the processing of data, and providing the added-value services such as for example the maintenance of energy-efficiency and well-being in buildings.

3.5 UC3 application components

The UC3 will have the next architecture tier:

- **Presentation tier**: this layer will offer interfaces which will allow, on the one hand, to monitor the state and the data of the use case, and on the other hand, to config or manage the application involve on it. It will be possible to have access to real-time data of heterogeneous devices (different manufacturers with different data representations) in this sense, the end-user or system administrator will take well-informed and confident decisions and perform actions over the Application level.
- **Application tier:** This layer is in charge of implementing and managing all the logic of the use case. You must first collect information and apply actions to and from the presentation layer. On the other hand, it must provide the logic, the processing and the actions of the different applications involved, which will allow the interaction with the VOs, the cVOs and the devices.
- **Data tier**: This layer encompasses different aspects of data management; from the moment it is generated in the devices to its processing, possible storage and/or secured before being stored for future use by the application tier. The data produced by the IoT devices (sensors) will be pre-processed and secured before being stored by the application tier. The data can be stored on the VO data store and is to be transmitted from the physical devices to the corresponding VO/cVO with low latency.

Regarding the application components, we highlight the following:

- An identity management component that also provides a credential management interface to ensure security and privacy with privacy-preserving mechanisms. Among others, it must provide verifiable credential issuer functionality to ensure security and privacy with privacy-preserving mechanisms. It must be compatible with distributed scenarios. In this sense, a distributed data storage system (DLT) can be used for these purposes.
- A Management Interface Component offers an interface that makes possible the configuration of the main aspects of most of the use case, for instance, configuring maximum power limits.
- Data monitoring component for accessing the real-time data of the use case as the current measurement of the temperature.
- Energy Analyzer Component, which acts as a power balancer. It will detect energy peaks and based on logic, such as Machine Learning (ML) training model, can execute actions to mitigate these situations.
- Orchestration Component for supporting add/remove/configure the devices. It must support hardware heterogeneity.
- Authorization framework to access the resources, services or applications of the use case. The configuration and storage of access control policies must be compatible with distributed scenarios. In this sense, a distributed data storage system (DLT) can be used for these purposes.
- Video Monitoring Component, for detecting events of the use case. For instance, to inform about objects/person detections or find dangerous or suspicious objects.

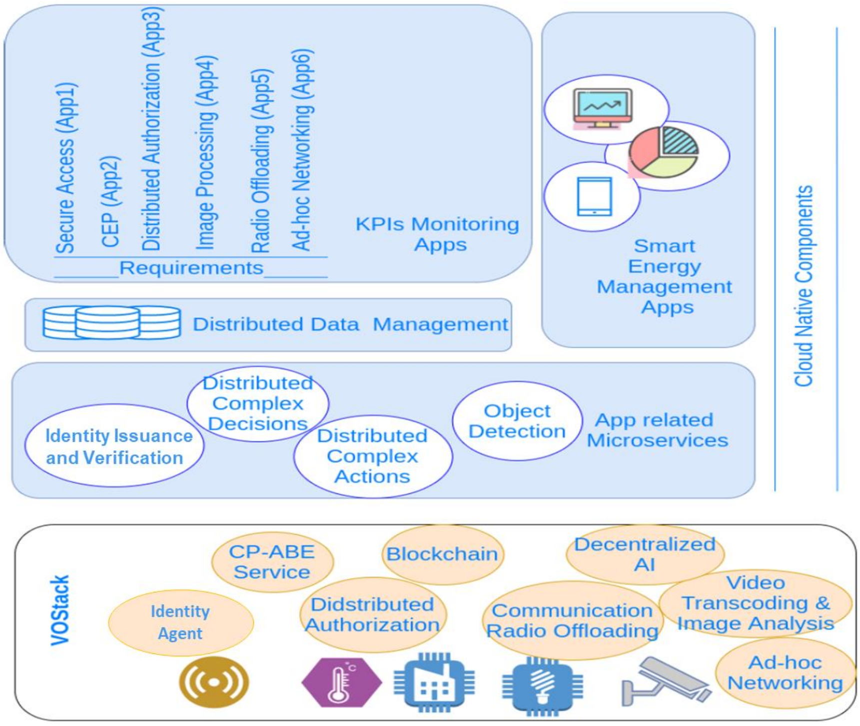


Figure 69: UC3 - Application services and VO Stack

3.6 Use case demonstration

The testing and demonstration of this use case will be performed first in simulation and later in a field trial scenario. The simulation will validate the methodology and concepts and will mimic the test that will be done at the end of the project in a field trial. This is expected to happen in PEANA, Enhanced Platform for IoT Smart Cities and Buildings provided by ODINS. This testbed is deployed on a smart building at Espinardo Campus in Murcia, Spain, thanks to a close collaboration of ODINS with the University of Murcia. The building offers several sensors and actuators such as: temperature, humidity, light, air quality sensors (CO2), occupancy detection, video cameras and smart sockets between others.

Figure 70. UC3 - Testbed (PEANA infrastructure)

3.7 Data processing requirements

The use case described requires several data types to be collected from several sources and sent to the edge for further elaboration or use. As already described earlier, we expect to use different IoT sensors and actuator to monitor and control several building and city applications. Between the most common devices to be used are power meters, temperature sensors, occupancy monitors and HVAC systems. In addition, video streams, pictures and sensed values will be collected for object and person detection based on AI algorithms that make use of pre-trained models on datasets. Also, device monitoring data will be collected for mission control and coordination. Finally, a record of the information processed, the decisions made, or the instructions received in the management processes will also be kept.

The produced data is expected to be predominantly digital with sizes varying from bytes (e.g., sensed values) to GBs for video transmissions. The exact format of these data will be determined during the implementation phase.

Data-processing requirements typically fall into two classes: system-oriented and user-oriented. System-oriented requirements measure the amount of information that your systems process. By contrast, user-oriented requirements measure the impact of data-processing services on the user. Service-level agreements reflect these expectations of performance.

In Table 21 we report the main data processing requirements (DPR) for the UC3.

Table 21: UC3 - NEPHELE's data processing requirements

ID	Type	Requirement	Description
DPR_UC3_01	User-oriented	Rapid response	Due to the need for timely intervention the system should guarantee low latency in communication
DPR_UC3_02	User-oriented	Multiple sources	Data fusion from different sources/devices should be supported to enhance person/object detection in different locations
DPR_UC3_03	System-oriented	Concurrent data sources	The number of concurrent data sources varies from case to case, but we can imagine the following ranges: • 2-10 devices • 1-10 HVACs • 10-50 sensors (temperature, energy consumption, air quallity) • 1-10 cameras
DPR_UC3_04	System-oriented	Dynamic Workloads	A mix of static (sensor values) and dynamic workloads is expected (different number of videostreaming) as the use case is dynamic in its nature.
DPR_UC3_05	System-oriented	Storage	Collected data from sensors, cameras, configuration information and executed tasks should be stored both in memory for fast use and in persistent storage to offer the possibility to replay executed actions/missions. The expected data sizes vary from bytes for monitoring values to MBs/GBs for images and video data.
DPR_UC3_06	System-oriented	Bandwidth/ Latency	The requirements in terms of bandwidth and latency vary according to the different subtasks of the use case: **Bandwidth** • Monitoring sensor networks: < 1Mbps • Video streaming: > 10Mbps

			Latency • Monitoring sensor networks: ~1000ms • Video streaming: < 300ms
DPR_UC3_07	System-oriented	Privacy and security	Personal data such as videos and images people and object detection should be guaranteed to be secured and adopt privacy standards in transmission and storage. In addition, other data such as building occupancy, device identity and other data that can be associated with people must be adequately protected.

3.8 Use case requirements analysis

The following subsections provide an analysis of functional, non-functional and system requirements for the use case.

Functional Requirements

Table 22 presents a description of the functional requirements for the UC3, as well as some comments on how the requirement will be addressed.

Table 22: UC3 - Functional requirements

ID	Description	How to address	Priority
FR_UC3_01	Implementing secure group communication protocol	Use of the VO in Edge nodes for instantiating security functions. Use of VC, VP and DIDs for Identity management purposes.	High
FR_UC3_02	Distributed complex decision making	VOs deployed in Edge nodes will perform intelligent energy saving actions, not only based on the sensor measures collected by the IoT devices managed, but also the information and data coming from other nodes.	High
FR_UC3_03	Distributed authorization scenarios	Through a Distributed Ledger Technology (DLT), that stores distributed access control policies, (e.g., distributed-XACML) and PDP/PEP components for authorization purposes	High
FR_UC3_04	Find dangerous or suspicious objects	Through AI-assisted image processing tasks running in distributed Edge nodes.	High
FR_UC3_05	Get the location of vulnerable persons	Through AI-assisted image processing tasks running in distributed Edge nodes.	High
FR_UC3_06	Communication radio offloading	Decentralized AI-assisted orchestration of VOs may avoid certain radio channels to save either IoT device battery or subscriber data.	High

FR_UC3_07	Get and show the status of sensors and actuators.	A management user interface is needed	Medium
FR_UC3_08	Keep track of executed tasks	A management user interface is needed	Medium
FR_UC3_09	Access and use existing pre-trained AI models for person and object detection	If certain pre-trained AI models already exist, it would be possible to use them when analysing a single video frame or image	Medium
FR_UC3_10	Access and use existing pre-trained AI models for optimization of video streams processing.	If certain pre-trained AI models already exist, it would be possible to use them when analysing several video streaming to improve	Medium
FR_UC3_11	Access and use existing pretrained AI models for energy peak mitigation and BAS	If certain pre-trained AI models already exist, it would be possilbe be use them when analysing and controlling energy consumtion and Building Automation Systems	Medium
FR_UC3_12	Allow historical analysis of actions	Store maps, actions and tasks performed for historical analysis	Medium
FR_UC3_13	Guarantee networking communication among video cameras and sensor networks	Use opportunistic network protocols, access mechanisms and routing schemes to keep devices communicating	Medium
FR_UC3_14	Enable devices to be registered and described in the ecosystem	Define protocols and procedures to make devices register to the corresponding VO	Medium
FR_UC3_15	Real-time monitoring	The system should be able to collect and process data in real-time and provide real-time alerts and notifications if certain thresholds are exceeded or if certain conditions are detected.	Medium
FR_UC3_16	Decision support	The system should provide decision-support tools that will be use to automate decisions (energy comsumption, network offloading, video streaming selection, ...) as well as help to make informed decisions about how to respond to diferent events. This can include visualization tools that display sensor data in a way that makes it easy to understand, as well as decision support algorithms that can analyze sensor data and provide recommendations.	Medium

Non-functional Requirements

Table 23 provides a description of the non-functional requirements for the UC3 and how they will be addressed.

Table 23: UC3 – Non-Functional requirements

ID	Description	How to address	Priority
NFR_UC3_01	Be robust in terms of the number of sensors available and alive	There must be a sufficient deployment of sensors, actuators, and cameras to ensure an adequate and effective deployment of measurement, detection or search depending on the application case.	Medium
NFR_UC3_02	Ensure privacy	AAA techniques will be implemented, specially in video processing scenarios, as well as	High
NFR_UC3_03	Provide mechanisms for reduce data disclousure	Privacy preserving mechanisms need to be provided to minimize data disclousure	Medium
NFR_UC3_04	Guarantee security of data storage and processing	Secured and authorised access to the system will be implemented.	Medium
NFR_UC3_05	Delegate computational calculation to the Edge and Cloud	Vertical offloading	High
NFR_UC3_06	Be resilient, efficient, lightweight and with flexible design	Use containerized applications and cloud-native principles ready for orchestration	Medium
NFR_UC3_07	Limit bandwidth usage	Implement smart data filtering/aggregation/compression policies, specially for video processing	Medium
NFR_UC3_08	Be energy efficient	Task assignment and sampling frequency based on battery level in the device	Low
NFR_UC3_09	Ensure enough resources for application	Use network slicing and orchestration techniques	Low
NFR_UC3_10	Ensure good precision and high confidence in object/person detection	Use pre-trained models specific for the study case	Medium
NFR_UC3_11	Adapt sensor sampling frequency to situation	Implement smart policies based on multiple factors and parameters to optimise sensor sampling	Medium
NFR_UC3_12	Ensure communication protocol integration with VO-Stack	Implement communication protocols that enable sensors and actuators to communicate with (to and from) the VO-Stack (e.g., MQTT)	High
NFR_UC3_13	Ensure semantics integration with VO-Stack	Implement semantics that enable sensors and actuator to communicate with the VO-Stack	High
NFR_UC3_14	Ensure dynamic placement, performance monitoring and dynamic redeployment of software components	Adopt service mesh approaches and orchestration solutions for components of the application graph	Low
NFR_UC3_15	Support adding / removing devices on the fly, including their lifecycle	Enable orchestration of components required to support adding / removing devices. Enable orchestration of components on devices	Medium

System Requirements

Table 24 provides the description of the system requirements for the use case 3 and how they will be addressed.

Table 24: UC3 – System requirements

ID	Description	How to address	Priority
SR_UC3_01	Have enough computational power	To run the mission control and AI methods sufficient memory and processing power is needed, including modelling trainning and video processing	High
SR_UC3_02	Have internet connectivity	To offload computation if needed and datasets that are needed	High
SR_UC3_03	Have enough storage capacity	To store and process the data coming from various data sources. Cloud storage will be used	High
SR_UC3_04	Have virtualization and replication capacities	The system will be virtualized and re-deployable in containers. Thus, the processing unit should have virtualization capabilities and be optimized for container virtualization and replication.	High
SR_UC3_05	Have enough GPU capacity	It is likely that GPU capacity will be needed, specially for ML trainning and video processing	High
SR_UC3_06	Have proper AI models for edge computing	This will be provided as part of the VO-stack	Medium

3.9 NEPHELE's innovation for the use case

Table 25 summarizes the requirements of this Use Case and discusses the limitations that NEPHELE aims at overcoming and thus facilitating the realization of the Use Case. For each requirement, we provide a reference to its definition, which can be found in Tables 22, 23 and 24.

Table 25: UC3 – Requirements to demonstrate NEPHELE innovation

Requirement	Current limitation	Innovation
Offer advanced security features (authentication and authorization) on limited devices DPR_UC3_07, FR_UC3_01, FR_UC3_02, FR_UC3_03, NFR_UC3_02, NFR_UC3_03, NFR_UC3_04, NFR_UC3_012	IoT devices often have limited computing, power, and connectivity capabilities.	With NEPHELE and the use of Virtual Objects, computing capabilities can be extended, and consumption requirements are made more flexible in order to associate advanced security functions associated with more demanding cryptography, storage or consumption with each device.

Advanced intelligent processing systems from multiple sources DPR_UC3_02, DPR_UC3_03, DPR_UC3_04, DPR_UC3_05, FR_UC3_09, FR_UC3_10, FR_UC3_11, FR_UC3_16, NFR_UC3_07, NFR_UC3_008, NFR_UC3_010, NFR_UC3_013	Artificial Intelligence models tend to be centralized and work on homogeneous sources	With NEPHELE and the use of Composite Virtual Objects, it is possible to extend ML/AI capabilities to process heterogeneous sources that optimize information processing, eliminating video streams from empty rooms or including more sources of information.
Distributed processing systems at the edge DPR_UC3_01, DPR_UC3_02, DPR_UC3_03, DPR_UC3_04, DPR_UC3_05, FR_UC3_04, FR_UC3_05, FR_UC3_08, FR_UC3_12, FR_UC3_14, FR_UC3_15, FR_UC3_16, NFR_UC3_01, NFR_UC3_04, NFR_UC3_05, NFR_UC3_06, NFR_UC3_07, NFR_UC3_08, NFR_UC3_09, NFR_UC3_12, NFR_UC3_15	Current work systems in the cloud are based on centralized processing systems that imply greater bandwidth and computing, as well as slower decision making	With NEPHELE the computing options are expanded, offering a distributed computing system at the edge, offering computing functions in each Virtual Object.
Optimization of network resources, processing and consumption DPR_UC3_01, DPR_UC3_02, DPR_UC3_04, DPR_UC3_06, FR_UC3_06, FR_UC3_07, FR_UC3_13,	IoT devices work individually and do not take advantage of information from the environment to optimize their operation.	With NEPHELE and the Composite Virtual Objects, the possibility of optimizing energy consumption, the use of network and computing resources is offered through a composite vision of several devices, which allows the establishment of advanced mechanisms for optimizing operation, use and management.

FR_UC3_15, NFR_UC3_05, NFR_UC3_06, NFR_UC3_07, NFR_UC3_08, NFR_UC3_09, NFR_UC3_11, NFR_UC3_14, NFR_UC3_15		

4 Use case 4 in NEPHELE

4.1 Introduction and objectives

The current ultrasound medical imaging processes are constrained by both the technical features of the local device and the knowledge of the (local) healthcare operator performing the examination. In fact, Electronic Health Record (EHR) processes are currently bound to on-premises dedicated hardware/firmware components to fulfil the need of a real-time or an almost real-time execution of the process. As a result, acquisition costs/capital expenses are very high and limit the degrees of flexibility in upgrading the hardware and, consequently, the types and number of functions that can be (locally) provided. Functions refer to those EHR processes that elaborate ultrasound data to provide the operator with additional qualitative information (often visualized over coloured overlay images over the black and white video) or quantitative data (spatial measures, pattern identifications, etc.).

The goal of this Use Case (UC) is to connect, and somehow to decompose and virtualize ultrasound medical imaging systems into the cloud-edge continuum to lose any barriers due to the hardware capabilities and localization of current physical systems.

As depicted in Figure 71, by exploiting and leveraging on 5G and IoT technologies, the idea is to transform the ultrasound acquisition hardware and the medical imaging viewers into smart wireless-connected "things", that can be "plugged and played" through the cloud-edge medical imaging application: the essential functions of the ultrasound system, with the sole exception of the probe and the input/output devices (such as monitors, keyboards, etc.), must be dematerialized and migrated to the cloud/edge.

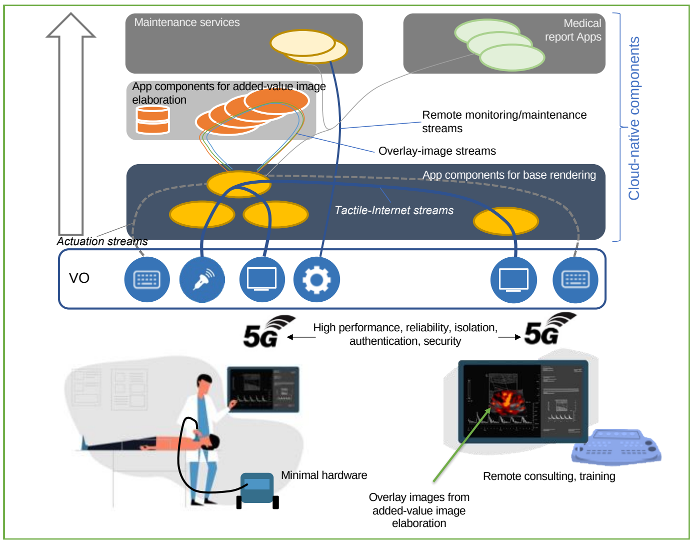


Figure 71: UC4 - Overall architecture of the e-Health use-case.

The ultrasound image processing currently involves a probe, image acquisition hardware, several local software functions devoted to the actual image processing and a monitor for displaying the images. The probe is a passive element cabled to the acquisition hardware in the device. The same device locally performs base and advanced image processing (typically using embodied GPUs) and renders the results on the local monitor. Since the image should have a high medical-grade resolution and that the whole imaging process is very complex, as it requires high volumes of data to be processed with strict latency (to react to human/operator-driven actions) and security constraints, present-day systems heavily rely on HW/FW tools. The decomposition of such systems into the cloud-edge continuum encompasses: a) the management through the Virtual Object stack of the connected physical acquisition and rendering devices, b) the possibility to "plug and play" physical systems (by means of their VOs) into different instances of the ultrasound medical app to execute visits even by involving remote operators (with their monitors and keyboards), c) the possibility of smartly manage (as-a-Service and at runtime) the processes for added value qualitative/quantitative analysis, medical reporting, and hardware maintenance.

As shown in Figure 71, the virtualized components, along with additional processes, will be deployed across the edge-cloud continuum, depending on the strictness of their time requirements. In more details, the base ultrasound image acquisition and visualization will become likely a tactile Internet application, and as such its proximity to the physical acquisition/rendering system will be crucial to provide the needed reactivity to the operator actions. On the other hand, the processes related to the medical reports' generation are less time-critical and can be deployed in the cloud. Between these two categories of applications, the edge-cloud continuum is accomplished with several services which still have strict latency requirements, but not as in the tactile Internet realm. The most relevant of these applications regard the overlay processes in charge of elaborating the raw images to identify known patterns or perform measurements, which are heavily based on Machine and Deep Learning (ML and DL) techniques.

Finally, the current system capabilities will be improved by pairing the physical components and, potentially, some of the virtualized functions, with a VO. Such digital counterparts will support and extend the capabilities of the IoT devices as well as helping with the interplay of physical and virtualized processes, for example, by adapting the image coding to the monitor resolution, providing data pre-processing, by managing caching, etc.

4.2 Benefits

There are several possible advantages considering the goal described in the previous section. In particular, the benefits can be grouped in different categories based on the possible beneficiaries.

From the point of view of the clinical device manufacture (e.g., ESAOTE) there is a dramatic cost reduction: no plastics, mechanical boards, spare parts. The focus is on the software transducers and high parallel computing network. There is also a dramatic reduction of transport and installation cost and service management and maintenance. An additional benefit could be a less environmental impact.

Instead, from the point of view of the clinical staff and the hospital there is the possibility to use always up-to-date equipment with the support of remote control and diagnosis. This solution also allows a space reorganization.

Finally, there are benefits regarding the reduced time for reporting and training for the clinical staff and the end-user (in this specific case: the patients).

4.3 Case Study: Ambulance in a Rural Environment (ARE)

In this UC, the technologies and solutions will be tailored for a 5G-enabled Ambulance in a Rural Environment (ARE) with the support of the mobility. Nonetheless, these can be adopted to a series of other similar scenarios where the mobility is involved, or where the environment has some connectivity limitations.

In Figure 72, we summarize the main stakeholders, the location (physical or virtual) and the constraints, challenges, and risks for this case study.

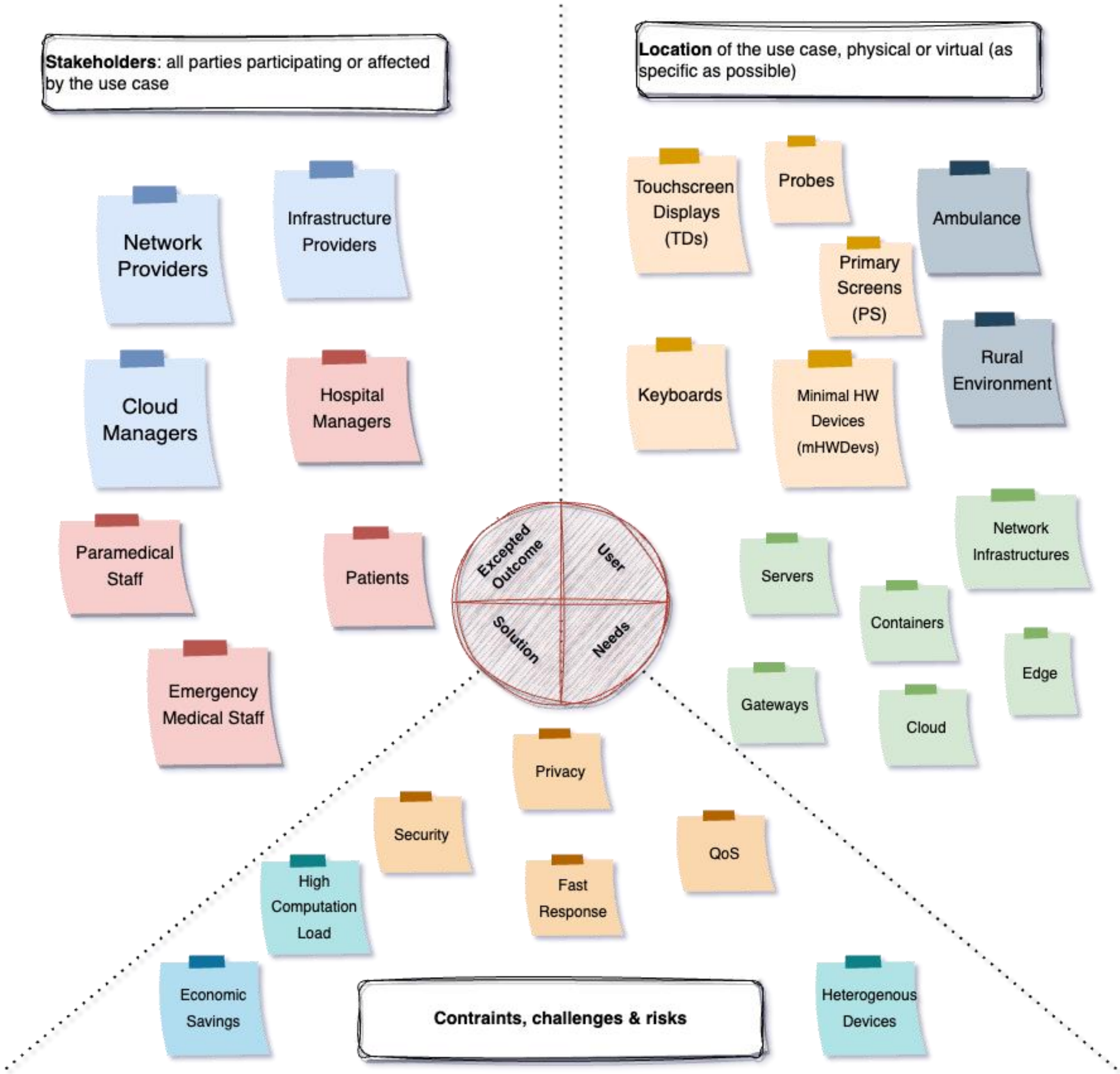


Figure 72: UC4 - Stakeholders, location, constraints, challenges, and risks for the UC.

Stakeholders

Several stakeholders are involved in an ARE scenario in focus in this UC. These range from the paramedical and the emergency medical staff to the network and infrastructure providers. All of them may be categorized as belonging to the medical staff or the ones involved to the infrastructure management. Besides them, the main actor in focus for this UC is the paramedical and emergency medical staff. For instance, the paramedical staff can use the dematerialized ultrasound system inside the ambulance consisting of the probe and the data processing part for local elaboration. Due to the 5G connectivity, the data obtained using the probe can be elaborated with further and advanced analysis in the cloud. Besides the hardware, the hospital defines the logic of a EHR application to be deployed and executed over the NEPHELE platform.

The application logic is represented as a Hyper Distributed Application (HDA) graph which will be available on the NEPHELE repository. The application logic will define the high-level goal and the Key Performance Indicator (KPI) requirements for the application. To run and deploy the HDA represented by the graph, some input parameters will be given. The application graph will require the deployment of one or more VOs/cVOs to represent IoT devices like probes, a Minimal HW Device (mHWDev) for Local Processing (LP), a touchscreen display, and one or more generic functions to support the application. These will support the EHR operations with scanning, processing, and displaying capabilities. The VO description required by the EHR HDA graph will be available on the NEPHELE Hyper-distributed Applications repository.

The data processing part for local elaboration will be ready to be used with some basic software components running. For instance, this component already has Operating System (OS) installed and correctly set up, with some basic applications already running. Once the network connectivity is established the VO/cVO configuration will also enable some device management features to start and configure components on the devices and orchestration of software components according to the specific task to be executed over time. The paramedical staff will then use the physical devices and the hyper-distributed application to guide them in their mission and benefit from the enhanced situational awareness offered thanks to the NEPHELE platform for the specific UC.

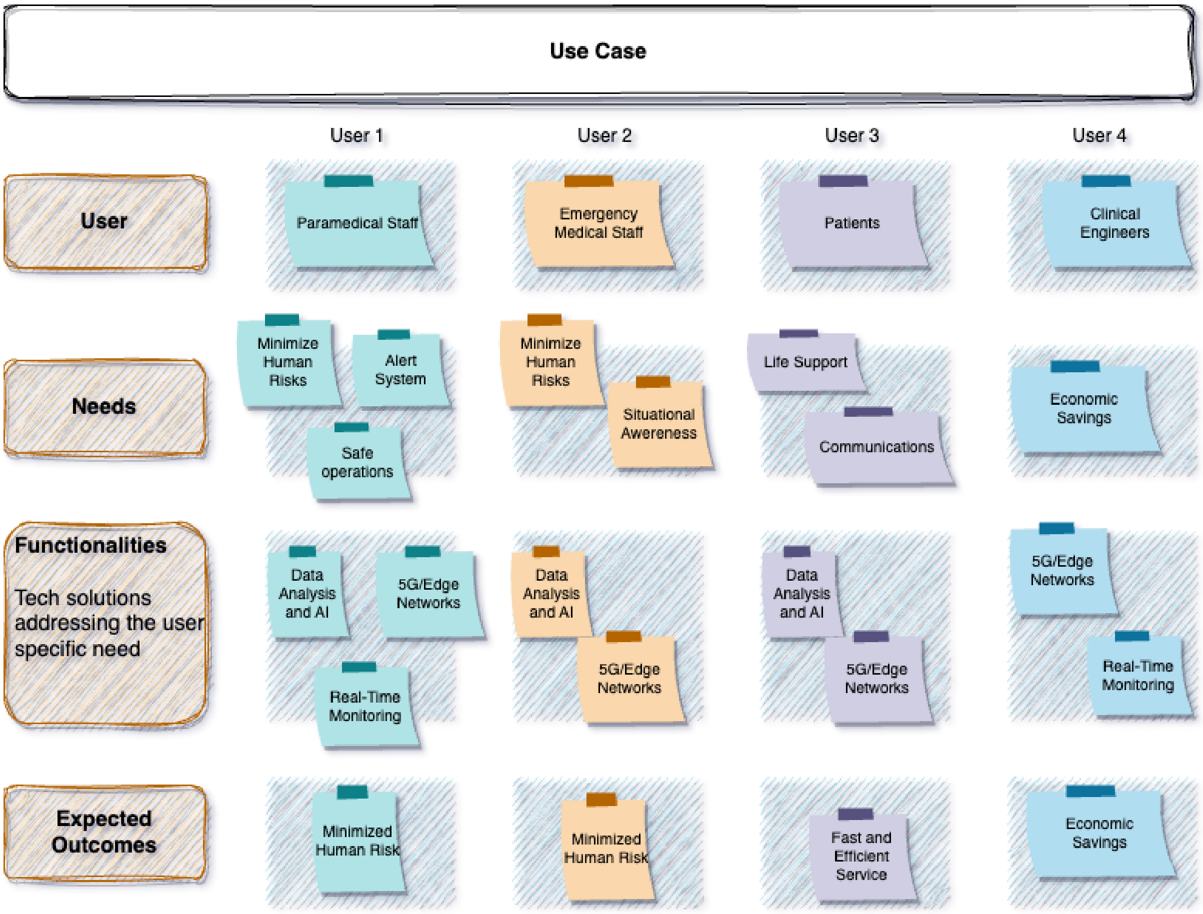


Figure 73: UC4 - Needs, functionalities and expected outcomes for the main users in the use case.

Location

The main physical location for the study case is the ambulance in a rural environment. The ambulance is connected using the 5G to the central hospital where the emergency medical staff can make remote support with advanced analysis useful as feedback for the paramedical staff. The rural environment increases the complexity of this scenario, adding some possible limitations for the connectivity. In this regards the data processing elaboration can integrate the remote advanced analysis ensuring in any case an answer even when communication is not enough.

The following hardware is used for the scope:

- The **probe** is a passive element cabled to the acquisition hardware in the device. It locally performs base and advanced image processing (typically using embodied GPUs) and renders the results on the local primary screen.
- A **Touchscreen Display** (**TD**) to control and configure the probe.
- A **Minimal HW device** (**mHWDev**) for applying preliminary and **Local Processing** (**LP**) of the collected data stream.
- A **keyboard** to control and configure the probe.
- A **Primary Screen** (**PS**) to visualize the analysed and processed image data.

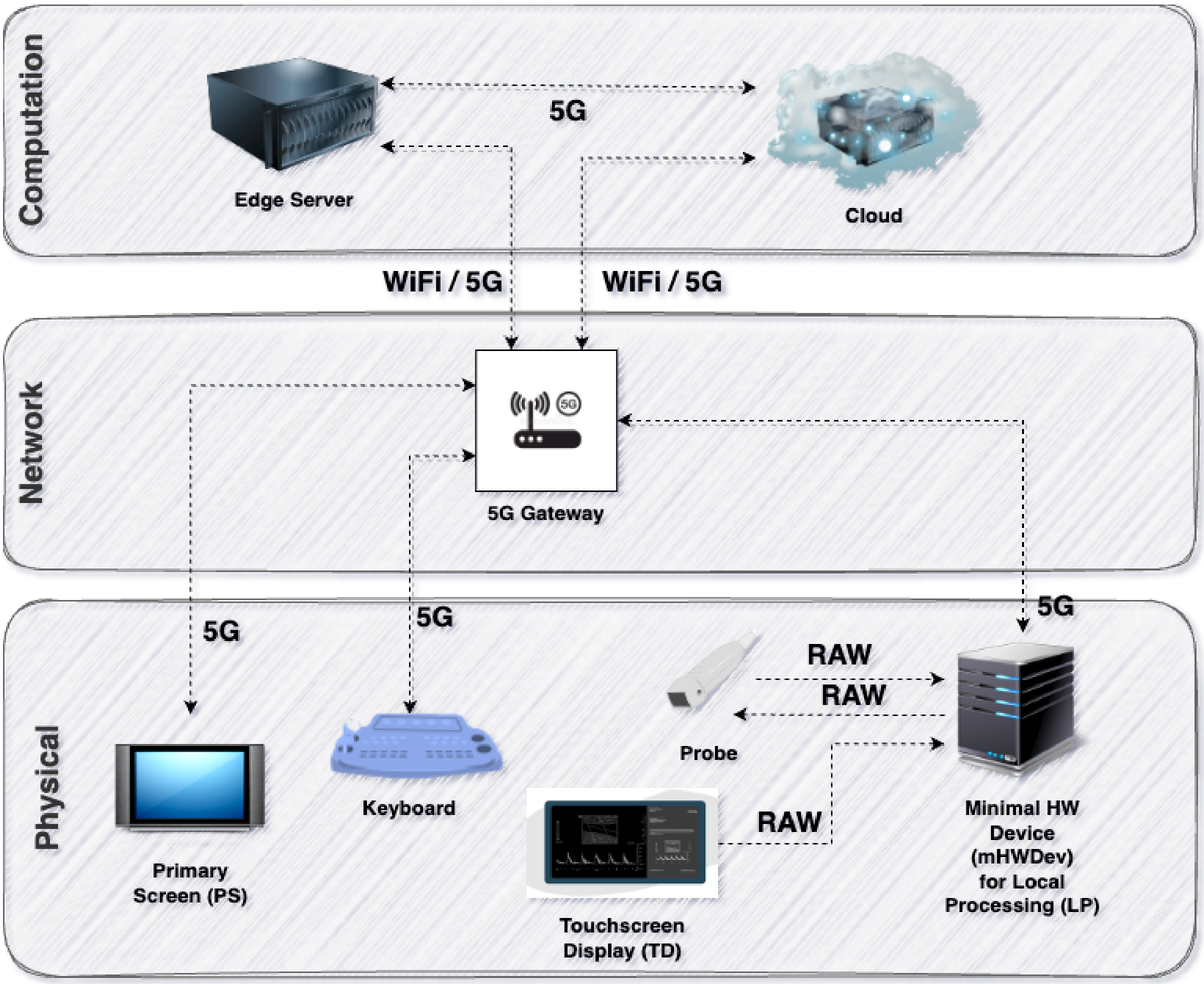


Figure 74: UC4 - Devices at the physical, networking and computation levels.

Constraints, challenges, and risks

As reported in Figure 71, the ARE study case carries with it several inherent constraints, risks, and challenges as detailed next.

Risks

- Ensure quick and safe intervention considering the security and the privacy of the patient data analysed in the edge / cloud part of the network.

Constraints

- In a rural environment scenario typically non network infrastructure is available or not reliable.
- Regularity limitations may limit the service in a rural environment. For example, the use of 5G frequencies that are regulated by national and international regulations.

Challenges

- Fast response is required to ensure efficient and effective intervention.
- To perform advanced analysis of the available data requires a high computation load which cannot be provided by the data processing devices located in the ambulance and used by the paramedical staff.
- Network coverage should be reliable during the whole time of intervention.
- The device used by the paramedical staff are very heterogenous in nature, such as probe, or touchscreen display. These devices differ in hardware, software and communication protocols used.

4.4 Technical requirements and challenges

The distribution of ultrasound medical system into different application components in the cloud-edge continuum poses several challenges related heterogeneous performance levels required by the different functions (falling from "Tactile Internet" requirements to ones generally provided by current cloud systems), and to the way data is treated. Data security is of paramount importance for medical processes and so is the need of a real-time or an almost real-time execution of the process.

To best face the needs of the EHR operations in the use case above and offer solutions to reach the overall goal for the solution we can identify the following main technical requirements and challenges.

- **Orchestration of software components**. Given the EHR application graph a dynamic placement of software components should be enabled based on service requirements and resource availability. This will require performance and resource monitoring at the various levels of the continuum and dynamic components redeployment.
- **Device Management**. Some application functionalities can be pre-deployed on the devices or at the edge. The device management should also enable bootstrapping and self-configuration, adding and removing devices on the fly, supporting hardware heterogeneity, and guaranteeing self-healing of software components.
- **Low latency communication.** Communication networks to/from a rural environment towards the edge and cloud should guarantee low delays for fast response under mobility conditions and possible disconnections.
- **High bandwidth for edge/cloud.** The data collected from the probe and after some pre-processing with the mHWDev should be sent to the edge/cloud for advanced analysis and to obtain additional diagnosis from remote and skilled operators.
- **Smart data filtering/aggregation/compression**. Large amount of data is collected from the probe. A part of this data can be filtered, other ones can be down sampled or aggregated before sending it to the edge/cloud using the mHWDev for LP. Smart policies should be defined to also tackle the high degree of data heterogeneity.

4.5 Applications and services of the study case

Real Time Cloud Elaboration

This scenario refers to the application components and services to provide real-time elaboration of the data collected the different devices (probe and keyboard). The collected data using the probe and some command sent with the keyboard are sent to the cloud for additional elaboration. The mHWDev is responsible to make preliminary elaboration.

Four different VOs should be deployed for the following IoT devices: PS, mHWDev, TD, and the Gateway (GW) that is used to send the command from the Keyboard to the cloud. The Keyboard and the Probe devices are not directly connected to the network, and, for this reason, a specific VO is not required.

A network connection fulfilling data rate and latency requirements for data streaming is required between the mHWDev and NEPHELE through the corresponding VO to send the data and process them. The communication between physical devices, the virtual counterparts at the VO and the other application components is enabled through the Zenoh protocol (some data communication can be integrated using HTTP and REST implementation). Some service will be running on the physical devices, whereas other on the edge and cloud continuum and will have to be configured through the VO.

The mHWDev is responsible to decrease the amount of data sent to the cloud for the processing. In addition, it includes local storage to save the data in case the connection is lost and should be sent when the connection is recovered.

In Figures 75, 76 and 77 we represent respectively the high-level view, the application graph, and the service graph for the Real Time (RT) Cloud Elaboration application scenario.

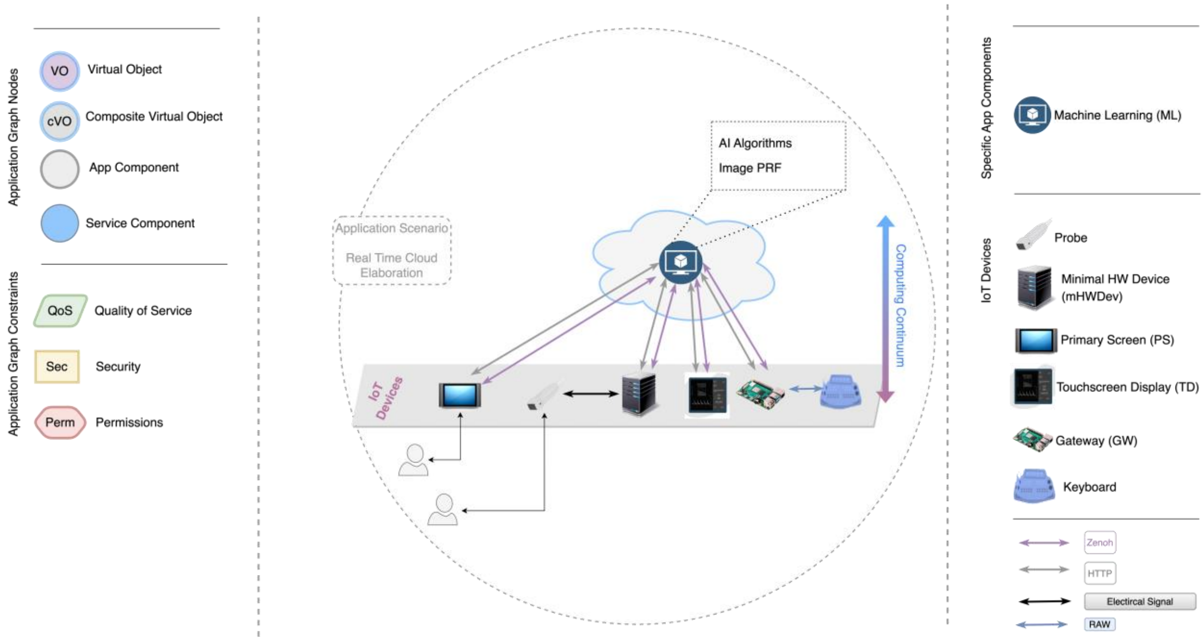


Figure 75: UC4 – RT Cloud Elaboration application scenario – High level.

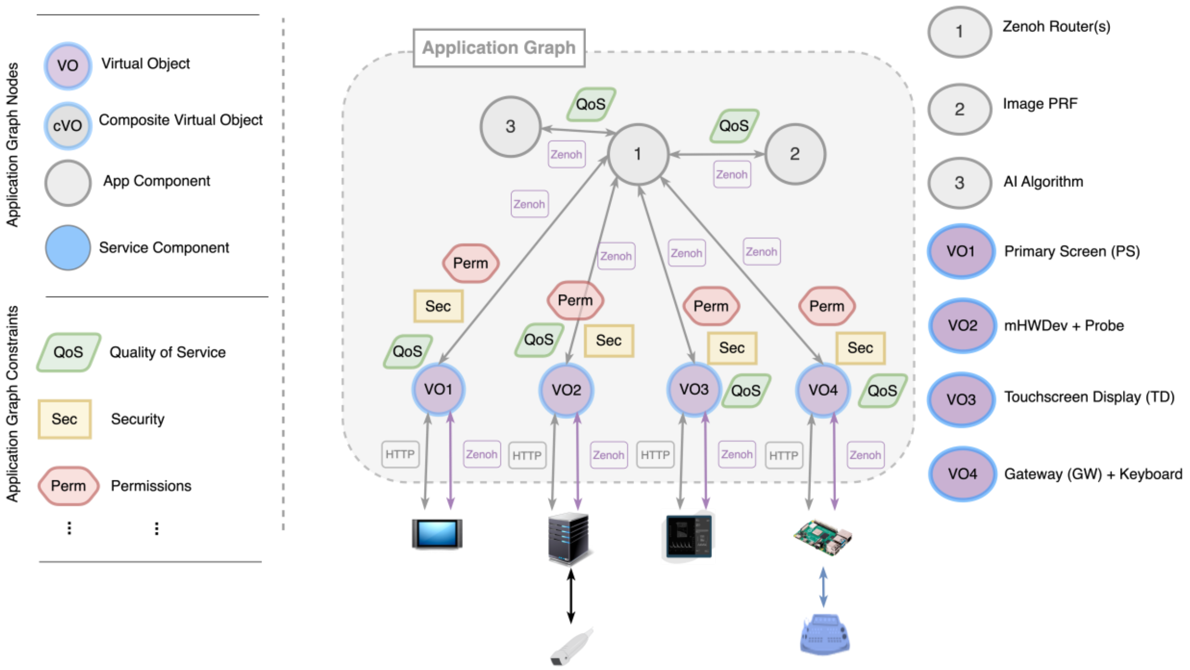


Figure 76: UC4 – RT Cloud Elaboration application scenario – Application graph.

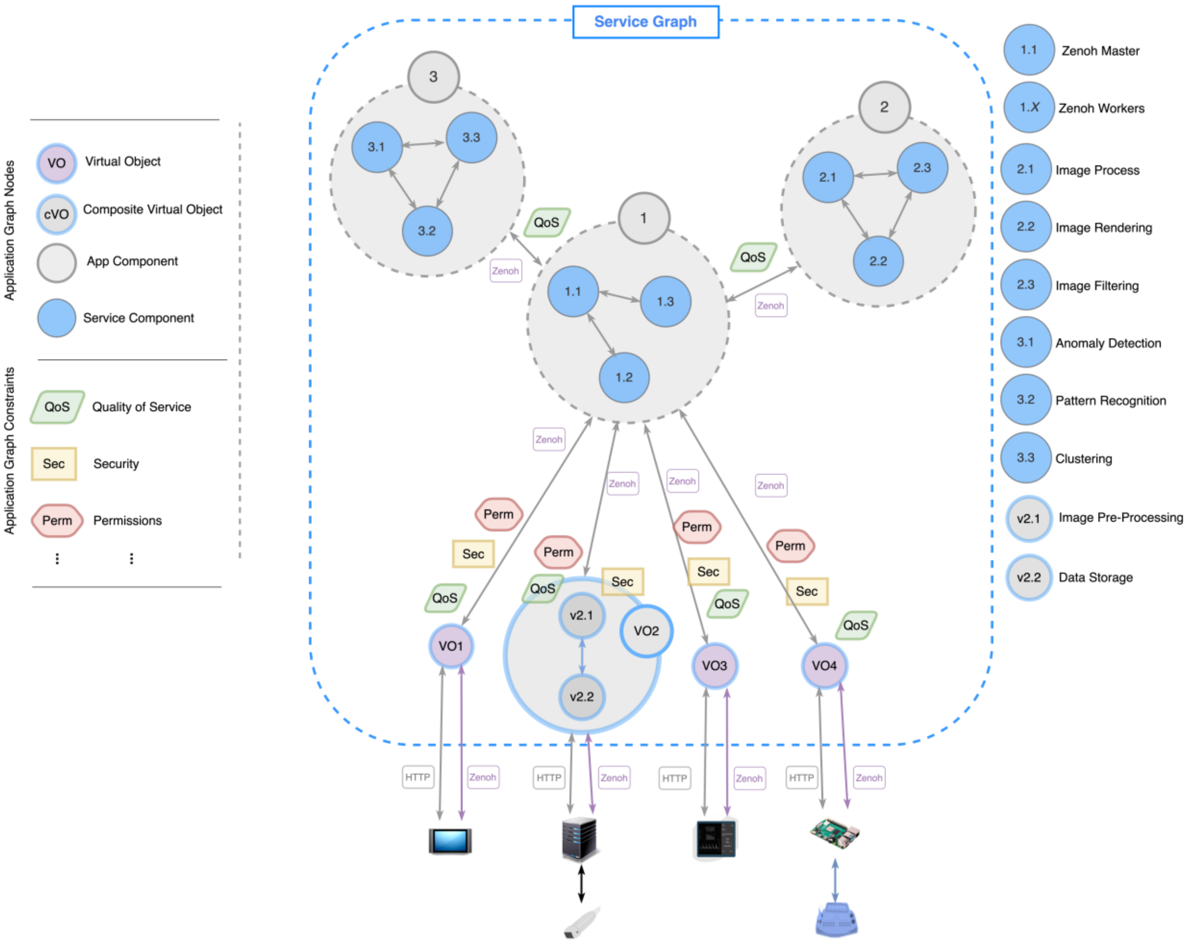


Figure 77: UC4 – Real Time Cloud Elaboration application scenario – Service graph.

Remote Support

This scenario refers to the application components and services to provide remote support for maintenance, tutorial, and training activities. The elaborated data is accessible in Real-Time using a dashboard on a Web Interface.

Like the previous application scenario, a VO should be deployed for the following IoT devices: PS, mHWDev, TD, and the GW.

A network connection fulfilling data rate and latency requirements for data streaming is required between the Minimal HW Data Processing and NEPHELE through the corresponding VO to send the data and process them. The communication between physical devices, the virtual counterparts at the VO and the other application components is enabled through the Zenoh protocol. Some service will be running on the physical devices, whereas other on the edge and cloud continuum and will have to be configured through the VO.

The Web interface includes a Dashboard that allows the following remote operations: monitoring, alerting, and replaying. In addition, with the Dashboard is it possible to make further remote elaboration for advanced analysis.

In Figures 78, 79 and 80 we represent respectively the high-level view, the application graph, and the service graph for the Remote Support application scenario.

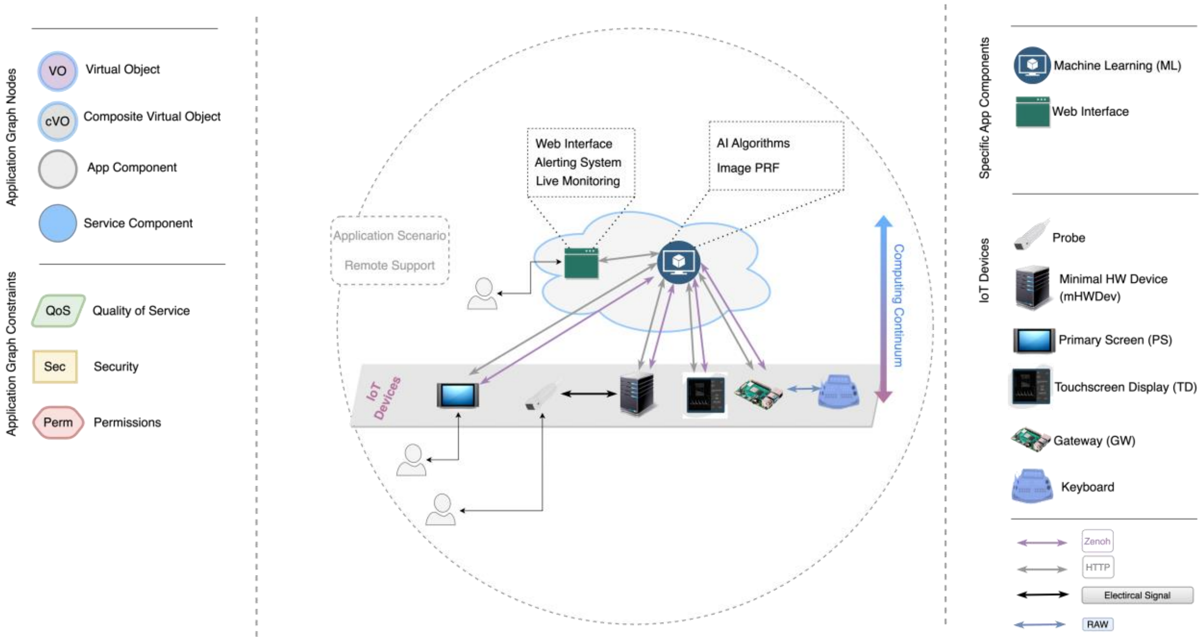


Figure 78: UC4 – Remote Support application scenario – High level.

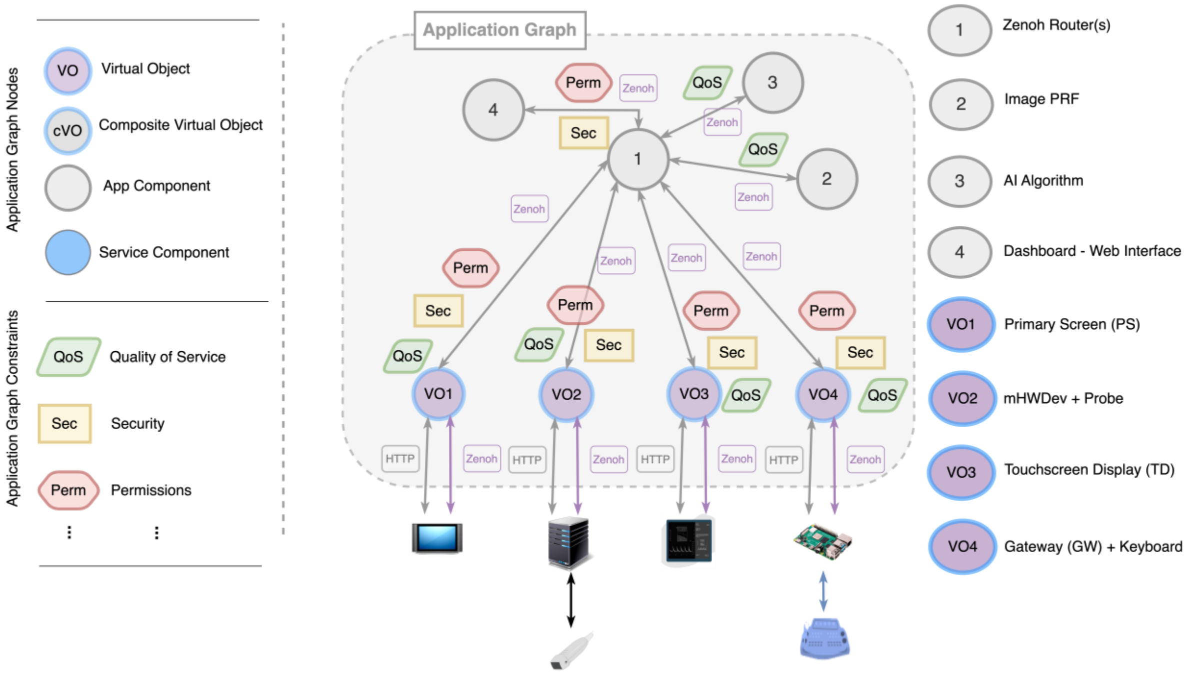


Figure 79: UC4 – Remote Support application scenario – Application graph.

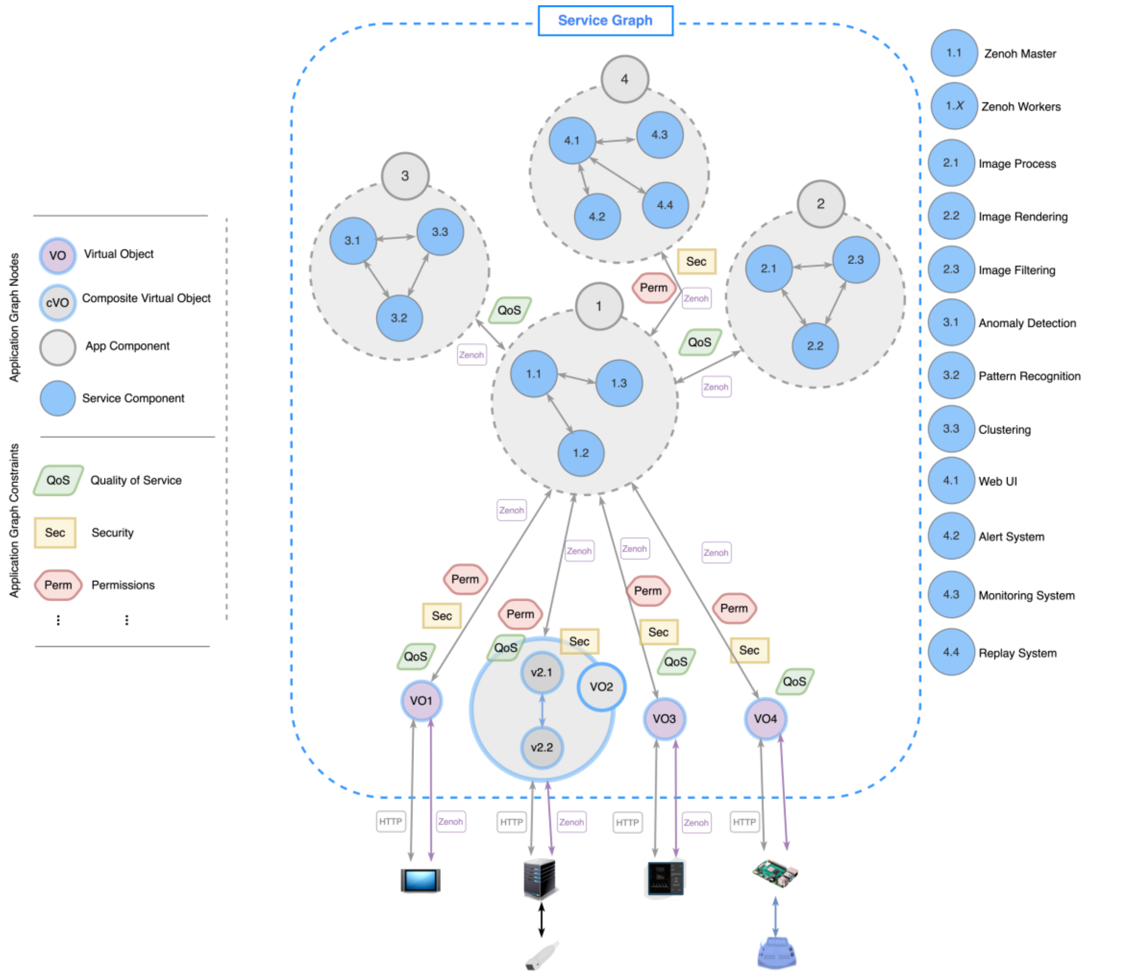


Figure 80: UC4 – Remote Support application scenario – Service graph.

Off-Line Remote Consultation

This scenario refers to the application components and services to provide a data storage needed to perform off-line consultation of the elaborated data using a Web Interface. The web interface makes possible to program further elaboration that can be useful for maintained, tutorial and training activities in a similar way of the previous application scenario.

A network connection fulfilling data rate and latency requirements for data streaming is required between the mHWDev and NEPHELE through the corresponding VO to send the data and process them. The communication between physical devices, the virtual counterparts at the VO and the other application components is enabled through the Zenoh protocol. Some service will be running on the physical devices, whereas other on the edge and cloud continuum and will have to be configured through the VO.

The Data storage includes a Time Series DB that allows to view the history of the elaborated data that can be used to simulate a stored cases useful to make additional analysis and elaboration.

In Figures 81, 82 and 83 we represent respectively the high-level view, the application graph, and the service graph for the Off-Line Remote Consultation application scenario.

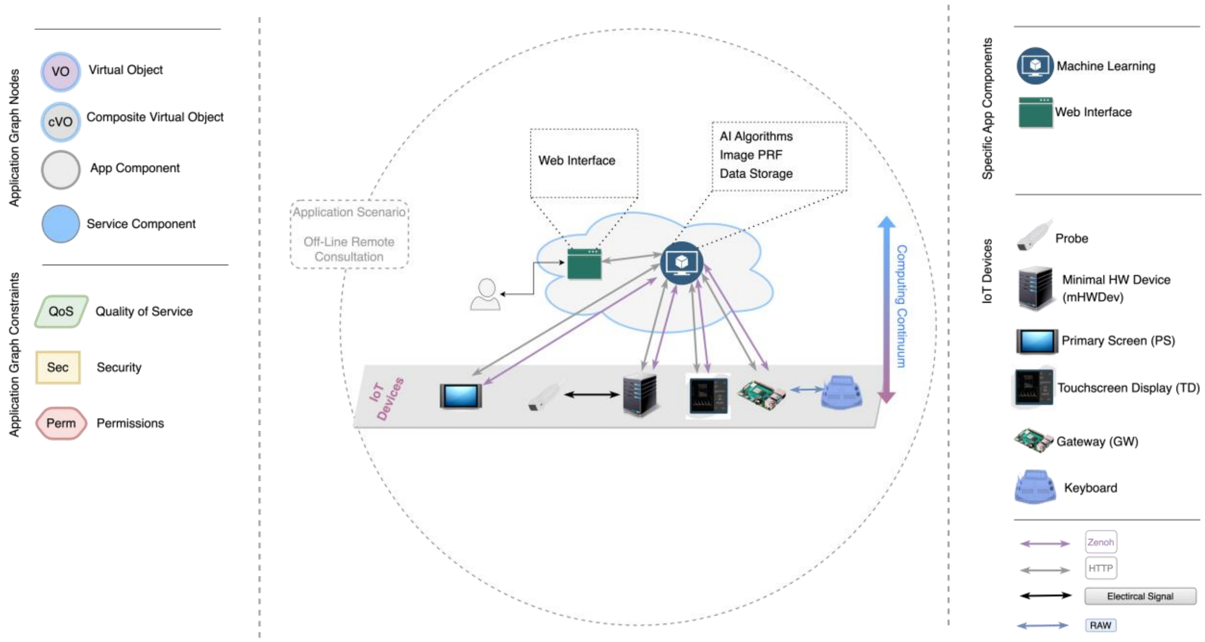


Figure 81: UC4 – Off-Line Remote Consultation application scenario – High level.

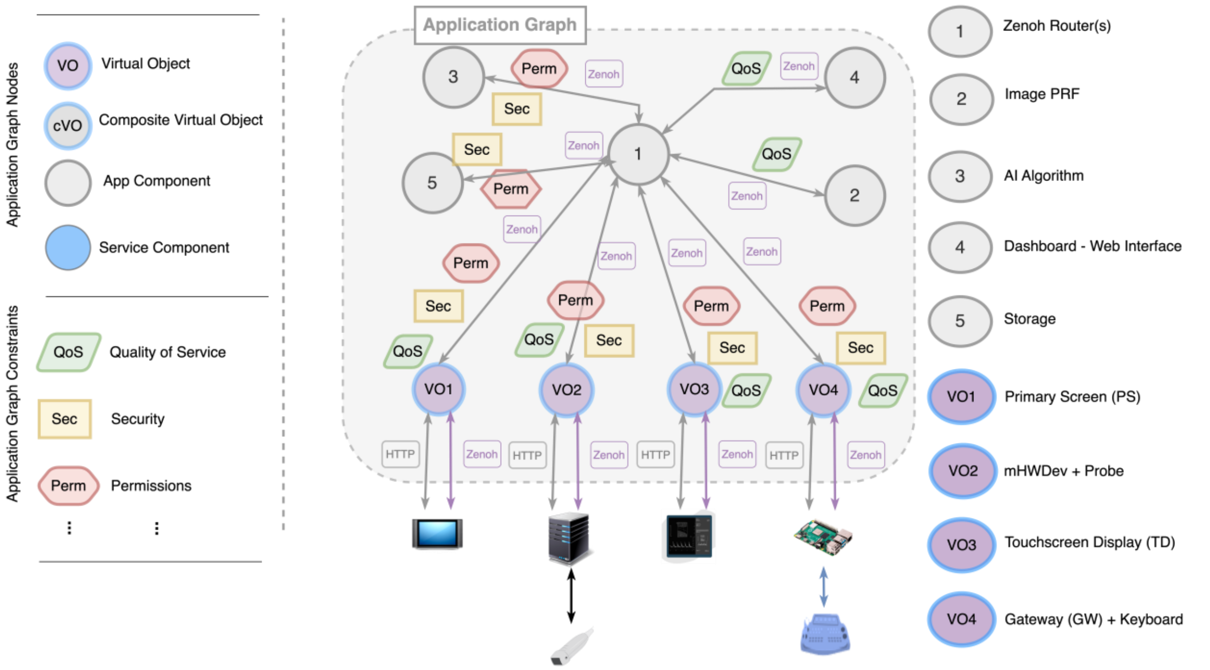


Figure 82: UC4 – Off-Line Remote Consultation application scenario – Application graph.

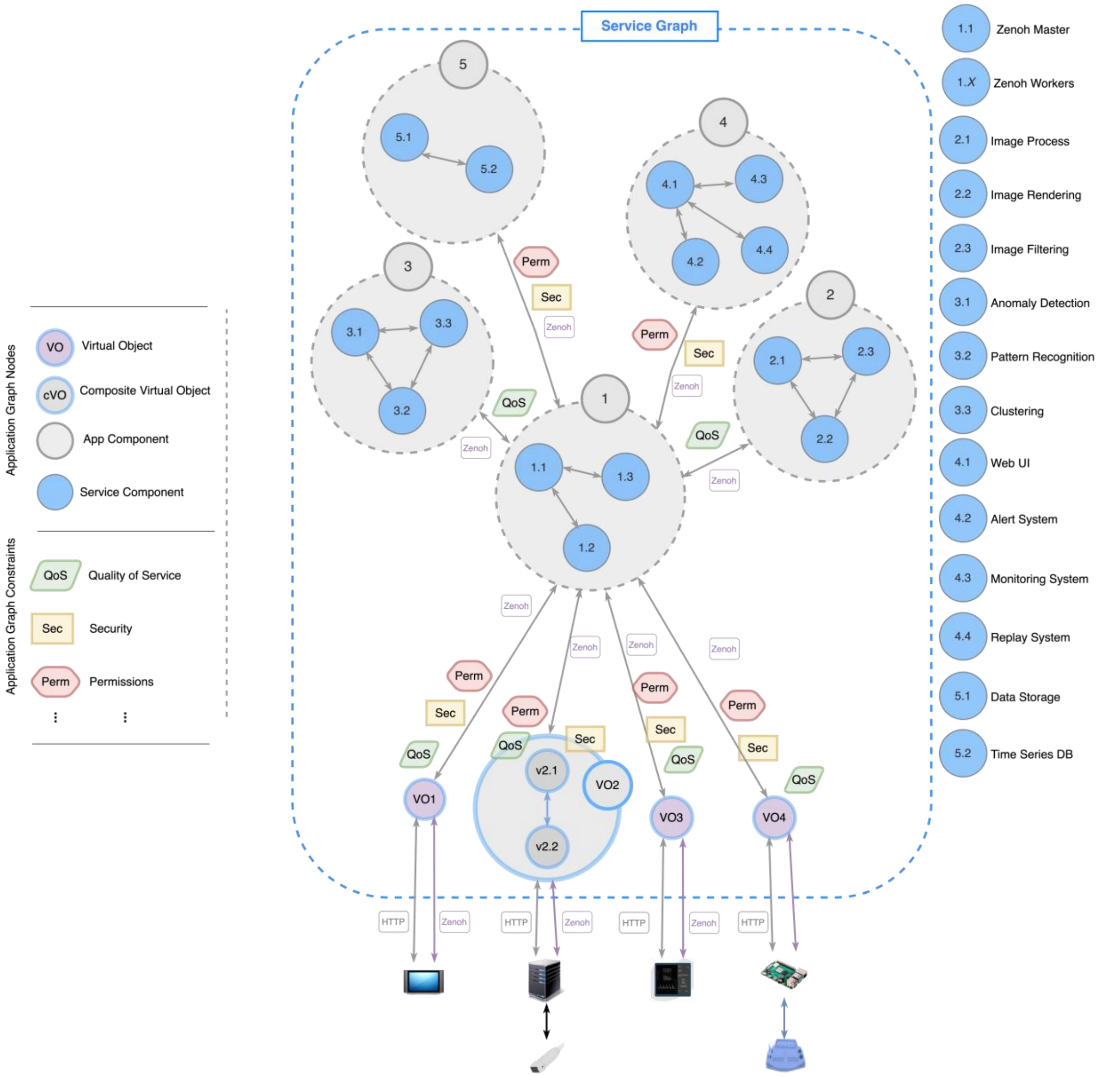


Figure 83: UC4 – Off-Line Remote Consultation application scenario – Service graph.

4.6 Phases of the operations for the study case

The EHR operations can be split into the following four main subproblems to be tackled.

- **Deployment of network infrastructure and application software**. The network infrastructure should guarantee communication in the area and towards the Internet. The application software to be deployed in the involved devices includes all the components in the application graph and the VO/cVOs for the hyper-distributed application.
- **React to the dynamic environment** which causes disconnections and requires monitoring (e.g., battery level status) and reconfiguration of devices. Once the network infrastructure is deployed, communication with all devices is to be guaranteed over time.
- **Data collection and analysis.** Data will be collected by the physical devices and sent to the higher levels of the continuum for further analysis.
- **Clinical intervention operations** that include collected data with probe, controlling and configuring the probe with the touchscreen display, take smart decisions for patients' diagnosis and optimization.

4.7 Application components

The EHR HDA will have a classic three-tier architecture with a presentation tier, an application tier, and a data tier.

- **Presentation tier:** the application will offer a frontend for visualization of the clinical situation of the patient. The dashboard integrates data coming from the probe together related to the intervention area and the patients' personal information. The dashboard will be accessible through a web browser or a graphical user interface (GUI) remotely and enable the Service Consumer to interact with the Application tier to see the clinical status of the patient and analyse historical data for further information collection and clinical awareness.
- **Application tier:** the inputs and requests coming from the presentation tier are collected and application components are activated to execute intervention operations. At this level, all application components for supporting the application logic in this UC are included. Some of these components will run directly on the IoT devices, some on the edge and some on the cloud through the VO/cVO. New data can be produced, and old data accessed from the Data tier.
- **Data tier:** this includes a storage element for storing processed images or historical data about the EHR intervention. The data produced by the probes will be compressed, down sampled and/or secured before being stored for future use by the application tier. The data can be stored on the VO data store and is to be transmitted from the physical devices to the corresponding VO/cVO with low latency.

4.8 Use Case Demonstration

The testing and demonstration of this UC will be performed first in simulation and later in a field trial scenario. The simulation will validate the methodology and concepts and will mimic the test that will be done at the end of the project. For what the request the HW: network and compute infrastructure provided by CNIT testbed used to support this UC, while ultrasound acquisition HW and medical imaging viewers provided by ESAOTE.

4.9 Data Processing Requirements

The data stream collected with the probe is pre-processed with the mHWDev and then sent to the edge/cloud infrastructure for further elaboration to enhance the clinical awareness of the patient. In addition, historical data about the intervention will be stored for future analysis. The produced data is expected to be predominantly digital with sizes varying from bytes (e.g., control/system data) to GBs for patients' data transmissions. The exact format of these data will be determined during the implementation phase.

Data processing requirements typically fall into two classes: system-oriented and user-oriented. System-oriented requirements measure the amount of information that your systems process. By contrast, user-oriented requirements measure the impact of data-processing services on the user. Service-level agreements reflect these expectations of performance.

In Table 26 we report the main Data Processing Requirements (DPR) for this Use Case (UC).

Table 26: UC4 - NEPHELE's data processing requirements.

ID	Type	Requirement	Description
DPR_UC4_01	User oriented	Rapid response	Due to the need for timely intervention the system should guarantee low latency in communication.

<table>
<tr><th>ID</th><th>Type</th><th>Requirement</th><th>Description</th></tr>
<tr><td>DPR_UC4_02</td><td>User oriented</td><td>Situational awareness</td><td>Data fusion from different sources/devices should be supported to enhance situational awareness for paramedical staff.</td></tr>
<tr><td>DPR_UC4_03</td><td>System oriented</td><td>Dynamic workloads</td><td>A mix of static (sensing values) and dynamic workloads is expected (rescue victims if identified) as the use case is dynamic in its nature.
A mix of light workloads (filter and pre-processed data by the mHWDev) and heavy workloads is expected (data streams and image analysis).</td></tr>
<tr><td>DPR_UC4_04</td><td>System oriented</td><td>Storage</td><td>Collected data from probes and executed interventions should be stored both in memory for fast use and in persistent storage to offer the possibility to replay executed interventions.
The expected data sizes vary from bytes for sensed values to MBs/GBs for images and video data.</td></tr>
<tr><td>DPR_UC1_05</td><td>System oriented</td><td>Bandwidth and latency</td><td>The requirements in terms of bandwidth and latency vary according to the different type of data stream:
<table>
<tr><th>Stream Type</th><th>Latency</th><th>Bandwidth</th></tr>
<tr><td>Data Streaming</td><td>5ms</td><td>10Mbps to 20Mbps</td></tr>
<tr><td>Control Data Streaming</td><td>10ms</td><td>100Kbps</td></tr>
</table></td></tr>
<tr><td>DPR_UC1_06</td><td>System oriented</td><td>Privacy and security</td><td>Personal data from patients should be guaranteed to be secured and adopt privacy standards in transmission and storage.</td></tr>
</table>

4.10 Use Case Requirements Analysis

The following subsections provide an analysis of functional, non-functional and system requirements for the use case.

Functional Requirements

A Functional Requirement (FR) defines a function of a system or component which describes a particular behaviour. Table 27 presents a description of the FRs for the UC, as well as some comments on how the requirement will be addressed.

Table 27: UC4 - Functional Requirements.

ID	Description	How to address	Priority
FR_UC4_01	Monitor patients' health.	Use IoT device to monitor status.	High
FR_UC4_02	Record/store data for historical database and future analysis.	Use VO storage to store patients' data.	Medium

ID	Description	How to address	Priority
FR_UC4_03	Get and show the status of IoT devices.	A GUI is needed.	Medium
FR_UC4_04	Keep track of the executed interventions.	A GUI is needed.	High
FR_UC4_05	Access and use existing pre-trained AI models.	If certain pre-trained AI models already exist, it would be possible to use them when analyzing the patients' data stream.	High
FR_UC4_06	Use local storage and sync data in case of disconnection.	Ad hoc configuration of the mHWDev to support local storage and synchronization mechanism with Edge/Cloud infrastructure	Medium
FR_UC4_07	Allow historical analysis of interventions.	Store data performed for historical analysis	Medium
FR_UC4_08	Guarantee networking communication among IoT devices.	Use opportunistic network protocols, access mechanisms and routing schemes to keep devices communicating.	High
FR_UC4_09	Enable devices to be registered and described in the ecosystem.	Define protocols and procedures to make devices register to the corresponding VO.	High
FR_UC4_10	Enable IoT Devices to offload interventions horizontally.	Horizontal offloading based on the battery level of the probes and the importance of the collected data.	Medium
FR_UC4_11	Real-time monitoring	The system should be able to collect and process data in real-time and to provide real-time alerts and notifications if certain threshold is exceeded or if certain conditions are detected.	Medium
FR_UC4_12	Decision support.	The system should provide decision-support tools that can help paramedical staff and other personnel make informed decisions about how to provide health intervention. This includes visualization tools that display processed data in a way that make easy to understand, as well as decision support algorithms that can analyze incoming data and provide recommendations.	Medium
FR_UC4_13	Enable migration of nodes between networks.	The system should be resilient to mobility and migration of nodes.	Medium

Non-functional Requirements

Unlike a functional requirement, which defines what the system should do, a non-functional requirement (NFR) specifies how the system works. Particularly, it defines criteria that judge the operation of a system, such as its performance, availability, etc. Table 28 provides a description of the non-functional requirements for the UC and how they will be addressed.

Table 28: UC4 – Non-Functional requirements

ID	Description	How to address	Priority
NFR_UC4_01	Ensure connectivity in rural environment.	Use and deploy 5G IoT gateway or an alternative valid solution.	High
NFR_UC4_02	Ensure privacy.	AAA techniques will be implemented.	Medium
NFR_UC4_03	Guarantee security of data storage and processing.	Security and authorized access to the system will be implemented.	Medium
NFR_UC4_04	Delegate computational calculation to the Edge/Cloud infrastructure.	Vertical offloading	High
NFR_UC4_05	Be resilient, efficient, lightweight and with flexible design.	Use containerized applications and cloud-native principles ready for orchestration	Medium
NFR_UC4_06	Limit network bandwidth usage.	Setup and configure the mHWDev for LP.	High
NFR_UC4_07	Be energy efficient.	Task assignment based on battery level of the probe.	Medium
NFR_UC4_08	Ensure enough resource for the application.	Use network slicing and orchestration techniques.	High
NFR_UC4_09	Ensure very low latency and high bandwidth.	Use 5G communication techniques, edge computing and low-latency communication protocols.	High
NFR_UC4_10	Ensure accurate analysis of the patients' clinical status.	Use pre-trained model specific the study case. Extensive use of historical data to identify patterns and ensure the most reliable diagnosis possible.	Medium
NFR_UC4_11	Ensure communication protocol integration with VO-Stack.	Implement communication protocols that enable the IoT devices to communication with the VO-Stack (e.g., MQTT).	High
NFR_UC4_12	Ensure semantic integration with VO-Stack.	Implement semantics that enable IoT devices to communication with the VO-Stack.	High
NFR_UC4_13	Ensure dynamic placement, performance monitoring and dynamic redeployment of software components.	Adopt service mesh approaches and orchestration solutions for components of the application graph.	High

ID	Description	How to address	Priority
NFR_UC4_14	Enable routing and multihopping scheme	Enable IoT devices to forward through relaying nodes to reach the Internet.	Medium
NRF_UC4_15	Support adding/removing devices on the fly, including the life cycle.	Enable orchestration of components required to support adding/removing devices.	High
		Enable orchestration of components on devices.	
NRF_UC4_16	Adapt probe setup based on current situation.	Implement smart policies based on multiple factors and parameters	Medium

System Requirements

A system requirement (SR) defines the configuration that a system must have to run smoothly and efficiently. A system requirement may refer to computational power, hardware capacity, etc. Failure to meet a system requirement can result in the installation or performance problems. Table 29 provides a description of the non-functional requirements for the use case (UC) and how they will be addressed.

Table 29: UC4 – System requirements

ID	Description	How to address	Priority
SR_UC4_01	Have enough computional power.	To execute the clinical operation and data processing sufficient memory and computation are needed.	High
SR_UC4_02	Have internet connectivety.	To integrate LP with AI, ML and DL analysis on edge/cloud infrastructure.	High
SR_UC4_03	Have the possibility to use 5G frequency.	To interconnect the devices with the NEPHELE devices an IoT/5G gateway might be needed with permission to use the 5G national frequencies.	Medium
SR_UC4_04	Have proper AI models for edge.	This will be provided as part of the VO-stack	Medium
SR_UC4_05	Have enough GPU capacity.	It is likely that GPU capacity will be needed.	Medium
SR_UC4_06	Have virtualized capacity.	The system will be virtualized and re-deployable in containers. Thus, the processing unit should have virtualization capabilities and be optimized for container virtualization.	High
SR_UC4_07	Have autonomy for local analysis.	The mHWDev should be configured and managed to guarantee a minimal level of data analysis for clinical report when the connectivity with the edge/cloud infrastructure is not possible.	High
SR_UC4_08	Have enough storage capacity.	The local storage capacity is needed for local analysis when the network connectivity is limited or absent. The remote storage capacity is needed to guarantee a historical database of all interventions that can also be used to improve the analysis of future interventions.	High

ID	Description	How to address	Priority
SR_UC4_09	Have enough network bandwidth.	A high network bandwidth can be necessary to increase the remote analysis with AI models. The local HW processing can be reduce this requirement with preliminary pre-processing.	Medium

4.11 NEPHELE's innovation for the use case

The exploitation of the VOStack layers will allow to exchange data and resources among the physical components involved in the use case (e.g., acquisition hardware, monitors, potentially other interactive HW input devices such as keyboards) and provide additional capabilities such as distributed data management and analysis based on ML and DL techniques, authorization, security, and trust based on security protocols and blockchain mechanisms, etc. Furthermore, the integrated meta-orchestration framework will allow the orchestration of data and resources between the cloud and edge computing orchestration platforms required for the proper and dynamic interplay of the functions in the application graph.

Table 30 summarizes the requirements of this Use Case and discusses the limitations that NEPHELE aims at overcoming and thus facilitating the realization of the Use Case. For each requirement, we provide a reference to its definition, which can be found in Tables 27, 28 and 29.

Table 30: UC4 – Requirements to demonstrate NEPHELE innovation.

Requirement	Reference ID	Current limitation	Innovation
Reduce the computational load on the mHWDev for LP and perform timely actions with very low latency and high bandwidth.	FR_UC4_06 FR_UC4_10 NFR_UC4_06 NFR_UC4_09 SR_UC4_01 SR_UC4_07 SR_UC4_09	Simply offloading to the edge may not improve as the edge has also limited resources. Communication protocols from physical devices should guarantee low latency and high bandwidth.	With NEPHELE, network and computation resources can be dynamically allocated through network slicing techniques. Orchestration of application components will enable adaptation to the status and predict dynamicity factors in the allocation of resources.
Integrate and enhance the local analysis with the mHWDev with the use of AI, ML and DL in the edge/cloud infrastructure.	FR_UC4_02 FR_UC4_03 FR_UC4_05 FR_UC4_08 FR_UC4_12 NFR_UC4_04 NFR_UC4_10 NFR_UC4_11 NFR_UC4_12 NFR_UC4_13 NFR_UC4_14 NFR_UC4_15	The balancing between the LP and the edge/cloud one to guarantee at the same time a lower bandwidth consumption and an accurate data analysis is a challenging task	With NEPHELE the supportive functions of the VO-Stack will enable the effective and efficient elaboration of data over the cloud continuum.

Requirement	Reference ID	Current limitation	Innovation
	SR_UC4_07 SR_UC4_08 SR_UC4_09		
Facilitate a diagnosis and patient report.	FR_UC4_01 FR_UC4_03 FR_UC4_04 NFR_UC4_16 SR_UC4_01 SR_UC4_02 SR_UC4_03 SR_UC4_04 SR_UC4_05	AI models are cumbersome to train and require computational power. Privacy concerns arise for patients' personal information.	NEPHELE's VO-stack offers AI models as VO-supportive functions tailored to the specific use case requirement. End-to-end privacy and security are guaranteed by NEPHELE.
Improve local analysis of the patients' clinical situations.	FR_UC4_04 FR_UC4_07 FR_UC4_08 FR_UC4_12 NFR_UC4_01 NFR_UC4_02 NFR_UC4_03 NFR_UC4_05 SR_UC4_01 SR_UC4_02 SR_UC4_03 SR_UC4_04 SR_UC4_05	Lack of integration of different IoT devices and technologies. Paramedical staff need to use multiple technologies and specialized personnel.	NEPHELE offers the possibility to orchestrate distributed applications and resources over the Cloud continuum reaching out to IoT devices. The VO-Stack enables heterogeneous devices to interoperate and collaborate.
Novel disruptive functionalities for augmenting medical analysis with the possibility to involve operators in different geographical positions (e.g., by connecting high-skill operators from remote hospitals).	FR_UC4_01 FR_UC4_03 FR_UC4_05 FR_UC4_08 FR_UC4_11 FR_UC4_13 NFR_UC4_09 NFR_UC4_10 NFR_UC4_11 NFR_UC4_12 SR_UC4_03 SR_UC4_04	Lack of integration of different IoT devices and technologies. Low latency and high bandwidth can be necessary to guarantee the support by operators in different geographical positions.	NEPHELE will enable cloud-edge powered AI exposed "as-a-Service" to support a better management and resource orchestration.

Requirement	Reference ID	Current limitation	Innovation
	SR_UC4_05 SR_UC4_06		