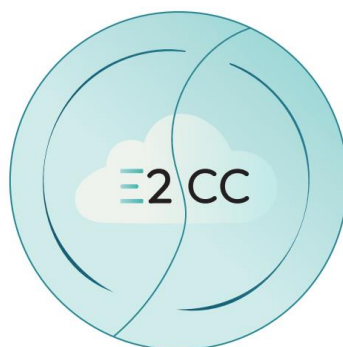




D7.2 Architecture V2

Document Properties

Contrat number	DOS0235220/00 – DOS0235222/00
Level of spread	Public
Nature	Report
Authors	Patrice Calegari (Bull), Emmanuel Zychla (Bull), Lionel Bouzon (Bull), Miguel Felipe Silva Vasconcelos (IRIT), Patricia Stolf (IRIT), Georges Da Costa (IRIT), Tarek Menouer (CGI), Gwladys Auffret (La-croix), Patrice Bleuze (Eviden/Atos), Aline Brouard (Armadillo), Benoit Vaugon (Ningaloo), Alain Warde (Bull), Valentin Menu—Cerri (Eviden:Atos), Paolo Milanesi (Institut Imagine), Antonio Rausell (Institut Imagine), Denis Galiana (Eviden/Atos), Thomas Cadeau (RYAX), Thomas Ropars (UGA), Rachid Zarouali (NBS System), Yann Pincivy, Damien Monguillon (NBS System), Mathieu Lagiet (NBS System), Hafeda Bakhti (Eviden/Atos), Cheickna Sylla (LAAS-CNRS), Pascal Berthou (LAAS-CNRS), Philippe Owezarski (LAAS-CNRS)
Date	23/06/2026
Status	Final



This project was financed by French State in the context of France 2030.

History of Changes

Version	Date	Authors (Organisation)	Description of changes
0.1	24/04/2026	Patrice Calejari (Bull)	Draft creation with context and overview
0.2	30/04/2026	Emmanuel Zychla, Lionel Bouzon (Bull)	ALUMET
0.3	04/05/2026	Miguel Felipe Silva Vasconcelos, Patricia Stolf, Georges Da Costa (IRIT)	MojitO/S
0.4	07/05/2026	Tarek Menouer (CGI)	Container Orchestration + Resource Management
0.5	11/05/2026	Gwladys Auffret (Lacroix)	IoT & Water network
0.6	11/05/2026	Patrice Bleuze (Eviden/Atos)	KMS KMIP
0.7	12/05/2026	Aline Brouard (Armadillo) Benoit Vaugon (Ningaloo)	IA/MLOps
0.8	18/05/2026	Alain Warde (Bull)	OneBSM
0.9	19/05/2026	Valentin Menu—Cerri (Eviden/Atos) Paolo Milanesi, Antonio Rausell (Institut Imagine)	Computer Vision platform AI for rare diseases
0.10	21/05/2026	Denis Galiana (Eviden/Atos)	IAM
0.11	22/05/2026	Thomas Cadeau, Andry Razafinjatovo (RYAX)	Ryax solution
0.12	25/05/2026	Thomas Ropars (UGA)	Anomaly detection
0.13	08/06/2026	Rachid Zarouali, Yann Pincivy, Damien Monguillon (NBS System)	Container runtime + Confidential computing
0.14	08/06/2026	Hafeda Bakhti (Eviden/Atos)	ZTO
0.15	08/06/2026	Valentin Menu—Cerri (Eviden/Atos), Thomas Cadeau (RYAX)	Add schemas for computer vision and Ryax solution
0.16	10/06/2026	Cheickna Sylla, Pascal Berthou, Philippe Owezarski (LAAS-CNRS)	Add schema and motivation for the tracing/monitoring functional architecture
0.17	15/06/2026	Patrice Calejari, François Exertier (Bull)	Review of document and roadmap table.
1.0	19/06/2026	Patrice Calejari (Bull)	First complete version
1.1	19/06/2026	Patrice Calejari (Bull)	Updated complete version
1.2	23/06/2026	Patrice Calejari (Bull)	Final version

Table of Contents

DOCUMENT PROPERTIES.....	1
HISTORY OF CHANGES	2
TABLE OF CONTENTS.....	3
1. EXECUTIVE SUMMARY	5
2. CONTEXT.....	6
PROJECT	6
WORK PACKAGE OVERVIEW	7
GENERAL OVERVIEW OF THE SOFTWARE STACK.....	7
3. COMPONENTS' SPECIFICATION	9
E2CC CORE PLATFORM	9
<i>OneBSM.....</i>	<i>9</i>
<i>KMS KMIP.....</i>	<i>11</i>
<i>Zero Touch Onboarding (ZTO).....</i>	<i>13</i>
<i>Anomaly detection.....</i>	<i>15</i>
<i>Distributed tracing for detection and mitigation of anomalies and attacks.....</i>	<i>17</i>
<i>E2CC Orchestrator.....</i>	<i>20</i>
<i>Container Orchestration.....</i>	<i>21</i>
<i>RYAX solution.....</i>	<i>23</i>
E2CC MODULAR PLATFORM.....	26
<i>ALUMET.....</i>	<i>26</i>
<i>MojitO/S.....</i>	<i>29</i>
<i>Resource Management.....</i>	<i>30</i>
<i>Container Runtime.....</i>	<i>32</i>
<i>Confidential computing.....</i>	<i>33</i>
<i>AI MLOps.....</i>	<i>37</i>
AI/MLOps Application: Anomaly detection and correction in time series.....	39
AI/MLOps Application: Design and development of new automatic indexing algorithms	40
<i>IAM solution.....</i>	<i>41</i>
Identity Federation.....	43
Token-based authorization	44
Account provisioning & deprovisioning.....	44
CLOUD-EDGE APPLICATIONS	46
<i>AI-analytics solution.....</i>	<i>46</i>
<i>IoT & Water Network.....</i>	<i>48</i>
<i>AI for rare diseases.....</i>	<i>50</i>
4. ESTIMATION OF THE SCHEDULE AND DURATION OF ACTIVITIES.....	52
5. SUMMARY	53
6. ACRONYMS.....	54

Table of Figures

Figure 1: General Architecture Overview.....	8
Figure 2: Example of OneBSM Graphical User Interface (GUI) view	10
Figure 3: OneBSM software layers	10
Figure 4: OneBSM interactions.....	11
Figure 5: HSM with embedded KMS Architecture Overview	13
Figure 6: ZTO architecture	14
Figure 7: Anomaly detection architecture.....	16
Figure 8: Anomaly detection model.....	16
Figure 9: Overall architecture of the monitoring system implemented.....	19
Figure 10: Container Orchestration	22
Figure 11: RYAX architecture.....	24
Figure 12: ALUMET Input/Output Overview	26
Figure 13: ALUMET Architecture Overview.....	27
Figure 14: ALUMET Client configuration.....	28
Figure 15: ALUMET Server configuration	28
Figure 16: Mojito/S structure.....	29
Figure 17: Resource management	31
Figure 18: Confidential computing architecture.....	34
Figure 19: Confidential computing workflow	36
Figure 20: Anomaly detection in time series architecture and workflow	40
Figure 21: Authentication flow using OIDC.....	44
Figure 22: Securing interactions.....	45
Figure 23. Architecture schema for the visite product.....	47
Figure 24: Architecture schema for the Ecosystem of Lacroix stack linked to E2CC platform.	49

1. Executive Summary

The document provides a comprehensive overview of the architecture for the E2CC (Eco Edge to Cloud Continuum) project, which aims to establish a robust platform for cloud-edge orchestration. The architecture is designed to support continuous data processing and storage, addressing the demands of modern IoT applications while ensuring competitiveness in European industries. This deliverable describes the key architectural components along a three-layers structure:

- **Abstraction Layer:** This layer serves as an API to facilitate seamless integration of cloud and Edge infrastructure products.
- **Orchestration and services Layer:** It enables the development and deployment of applications, ensuring that services can be efficiently managed and scaled. It includes services focused on AI/MLOps, security, and decarbonization, which can be utilized by applications or the central platform itself.
- **Vertical Applications:** These consist of specific use cases that operate on top of the platform, leveraging the services provided by the underlying layers.

After providing a general overview of this architecture, each software component within the architecture is described in terms of its functionalities, technical specifications, software dependencies, and expected timelines for development and integration. The resulting development timeline of this architecture is also provided.

Overall, the architecture of the E2CC project is structured to provide a flexible, secure, decarbonated, interoperable, and efficient framework for integrating cloud and Edge services. It aims to facilitate innovative applications and ensure that European industries can effectively leverage data in a rapidly evolving digital landscape. The architecture is now definitive, but it might evolve a little bit during the implementation and integration phases, if issues or possible improvement are detected.

2. Context

Project

E2CC aims to provide solutions that facilitate the continuous processing and storage of data from IoT devices to the cloud, shifting from cloud-centric approaches to a comprehensive Edge to Cloud continuum. E2CC will create standardized abstraction layers to optimize infrastructure usage and implement DevSecOps development chains for secure Cloud adoption. The project aims to advance MLOps and AI development, emphasizing trust and eco-responsibility, while adapting machine learning approaches to new infrastructures. Furthermore, it will enhance resilience against evolving threats and provide a comprehensive decarbonisation strategy to improve clients' carbon footprints.

The E2CC project aims to create a standardized end-to-end integration layer that enables interoperability and establishes a continuum from Edge to Cloud. It will provide the essential technological foundation for interconnecting with Cloud providers, encompassing functions such as cybersecurity, decarbonisation, orchestration, and platform management. The main initiatives of the project include:

- The provision of Bare Metal as a Service (BMaaS) solutions based on high-end enterprise servers, aimed at facilitating cloud adoption for critical and memory-intensive applications, while ensuring seamless integration into the Cloud infrastructure according to standards or guidelines that currently do not exist.
- The development of Edge servers and an associated abstraction layer that can monitor, secure, orchestrate, standardize, and optimize the use of infrastructure through Edge to Cloud services.
- The provision of MLOps and AI technologies tailored to the Edge to Cloud continuum, enabling better support for AI-based applications while improving data privacy, accuracy/trust, and performance/energy efficiency.
- Identity and Access Management (IAM) for users and objects, as well as data protection and threat detection/response in Edge-Cloud environments, will be provided through innovative solutions such as prescriptive IAM, decentralized identity management and "as a service," container security for DevSecOps, and the Data Sanctuary.

Work package overview

The work package 7 manages a coherent E2CC architecture and integration framework, enabling the construction of the overall E2CC continuum platform solution from the building blocks that will result from work packages 1 to 6. This work package also aims at demonstrating the overall E2CC approach on selected use cases. Overall, this work package includes the deliverables shown in Table 1. This deliverable (D7.2) provides a detailed overview of the E2CC architecture, the identification of key components with their interactions, their specifications and the expected timing of their development. It will serve as a reference base for the effort of work packages 1 to 6.

Number	Deliverable name	Short description	Due
D7.1	Architecture V1	Global overview of E2CC's architecture's components	M12
D7.2	Architecture V2	Revised overview of E2CC's architecture's components	M24
D7.3	E2CC platform integration V1	Intermediate implementation and integration description of components in the E2CC platform	M30
D7.4	E2CC platform integration V2	Final implementation and integration description of components in the E2CC platform	M36
D7.5	Use cases Proof of Concept	Final deliverable on results obtained by use cases on the E2CC demonstrator	M54

TABLE 1: WP7 DELIVERABLES

General overview of the software stack

E2CC project aims at providing a platform for the cloud-edge orchestration. To achieve this goal, we propose the architecture described in Figure 1. This architecture is structured in 3 different layers:

- An E2CC core platform built from:
 - An abstraction layer (API) to easily integrate Cloud and Edge infrastructure products,
 - an orchestration layer enabling the development and deployment of applications.
- E2CC optional services addressing AI/MLOps, security, and decarbonization, which can be used (on demand) by applications or by the core platform itself. Some services could include some hardware, such as an HSM for the security services.
- Vertical applications (use cases) running on top of this platform. Some more use cases (not exhibited here) might be added in the future from new collaborations or from accompanying projects.

The first 2 layers (The core platform and the services) make the 2ECC modular platform.

The 3 layers (including the use cases templates that can be provided for new clients/partners with the same needs) make the complete E2CC Platform

The resulting software stack that will be developed in the E2CC project will include several software components tackling specific aspects of this architecture. These components will address the features required in each of the 3 layers described above. We provide in this deliverable a description of each of these components and their interactions at each layer, along with the description of the planned effort that will be undertaken during the E2CC project. The intent of this global overview is as such to complement the other workpackage deliverables which will focus in more details on different aspects of this project.

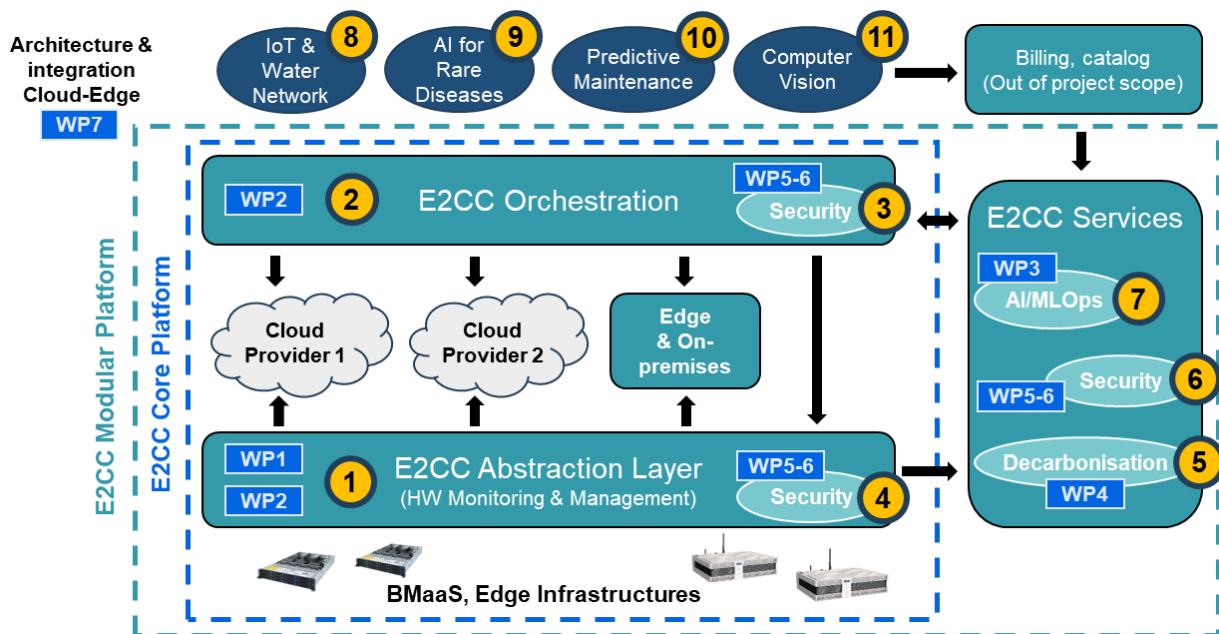


FIGURE 1: GENERAL ARCHITECTURE OVERVIEW

The numbers , etc. are used in the remaining of this document to locate the E2CC components within this general architecture.

3. Components' specification

For the specification of each component of the E2CC project and to provide a general overview of the planned activities, we use a template of the form:

Component name	The name used to reference the component or activity
Architecture location	Specify a number from the General Architecture Overview
Description	Description of the component or activity
Planned work	The work to be performed in terms of high-level steps
Software dependencies	Other software/components required for the final release/evaluation
Work start	Expected start month
Work end	Expected end month
Outcome	What will be delivered: code, report
Final demo/evaluation	What will be demonstrated
Code access	Planned type of code licence if applicable

E2CC Core Platform

These components will provide the internals of the cloud-edge orchestration capabilities.

OneBSM

Component name	OneBSM
Architecture location	
Description	Description of the component or activity: OneBSM is a centralized monitoring and management tool that uses standard protocols, dominantly Redfish, to communicate with the BMCs of supported servers (currently BullSequana SH, SA, EXR). Through one single portal, OneBSM allows to display information, inventory, metrics, health status, send alerts to email, as well as run FW update tasks on one or many servers, change boot settings, run power on/off/restart operations, remotely.
Planned work	N/A (Done)
Software dependencies	OS: Windows or Linux Windows: Microsoft C++ Runtime Linux: Net-Tools
Work start	01/01/2024
Work end	31/12/2025
Outcome	Deployable version of the software as well as the relevant documentation

Final demo/evaluation	The interface of the tool, as well as its multiple features and ability to monitor and manage the devices in particular in the testbed environment.
Code access	N/A.

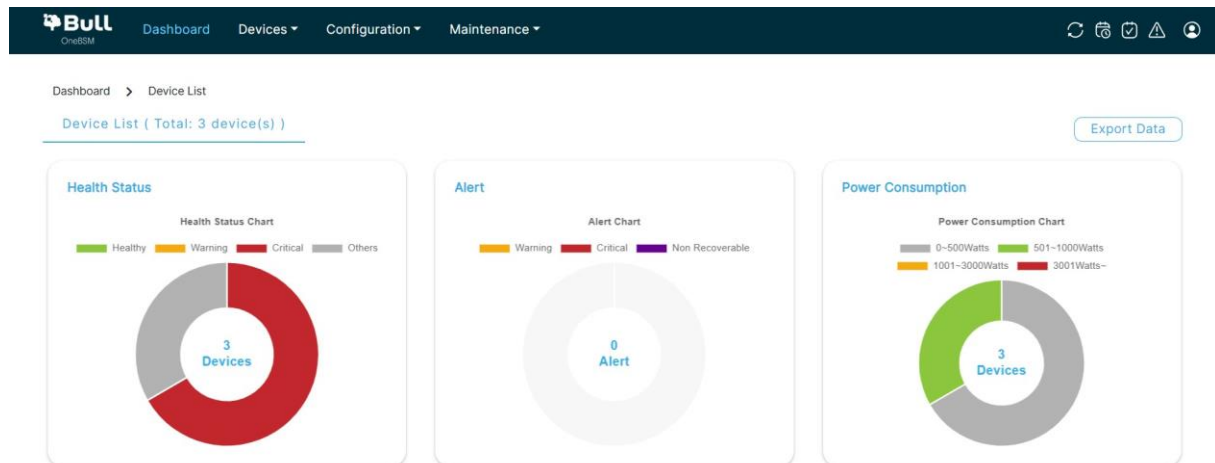


FIGURE 2: EXAMPLE OF ONEBSM GRAPHICAL USER INTERFACE (GUI) VIEW

Software infrastructure

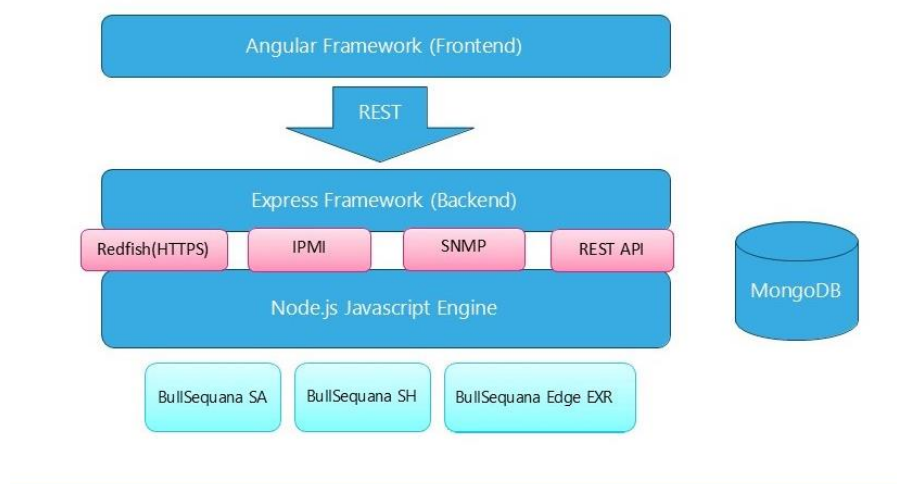


FIGURE 3: ONEBSM SOFTWARE LAYERS



FIGURE 4: ONEBSM INTERACTIONS

KMS KMIP

The component KMS KMIP is a software Key Management and encryption Server (KMS) enabling cryptographic APIs based on KMIP (Key Management Interoperability Protocol) for key management.

The shift to dynamic infrastructures such as the Cloud has generated a multiplication of secrets such as cryptographic keys used for data protection. Cryptographic keys management is the basis of all data security and follows a lifecycle of operations to ensure that keys are created, stored, used, and rotated securely.

The KMS with its proposed APIs will be available to all components that enable various functionalities within the infrastructure (agents, applications, operating systems, file systems, databases, virtualization system, deployment system) and hardware (servers/ computers, network devices, storage systems, and peripheral devices) composing the infrastructure.

Component name	KMS KMIP
Architecture location	  E2CC Orchestration (North bound) & E2CC Abstraction Layer (South bound),
Description	Description of the component or activity: The component is a software Key Management and encryption Server (KMS) enabling cryptographic APIs based on KMIP (Key Management Interoperability Protocol) for key management.

	The KMS with its proposed APIs will be available to all components that enable various functionalities within the infrastructure (agents, applications, operating systems, file systems, databases, virtualization system, deployment system) and hardware (servers/ computers, network devices, storage systems, and peripheral devices) composing the infrastructure.
Planned work	The work to be performed in terms of high-level steps: • Development of the Key Management System (KMS) implementing the KMIP Protocol • Development of a REST API interface for the administration of the product. • Development of an administration SAP (Single Page Application) WebUI consuming the administrative REST API above • Integration of the software KMS components into containers. • Integration of the KMS containers into a QEMU KVM virtual machine for the software appliance form factor. • Integration of the KMS containers inside the Eviden Proteccio Hardware Security Module (HSM) for the hardware appliance form factor
Software dependencies	Other software/components required for the final release/evaluation: Eviden Secured Linux Operating System TWOS Applications/Hardware able to consume cryptographic services through KMIP Protocol.
Work started	January 2024
Work end	Expected end month: December 2026
Outcome	What will be delivered: Report and prototype demonstration in a relevant environment
Final demo/evaluation	What will be demonstrated: The goal is to demonstrate some administration and cryptographic features provided by Eviden KMS/KMIP. The features to be demonstrated are: • Embedded form factor • KMS/KMIP embedded in Proteccio HSM • Virtual Machine form factor • Autonomous KMS/KMIP virtual appliance based on QEMU KVM. • KMS WebUI demo • Basic test with Seagate KMIP client • Interoperability tests with PyKMIP client • Encipherment of a data volume of a Synology NAS • Custom demo application storing individually ciphered documents in a MongoDB (community edition) database
Code access	Planned type of code licence if applicable: Software License Business Source Licence 1.1 (BSL). Code source available on github. Repository URL: https://github.com/Cosmian/kms

The following diagram in Figure 5 shows how the E2CC platform can be secured through the KMS and its HSM:

- A network (in blue) is dedicated to hardware administration communications (for the HSM Trustway, the NAS and Fortigate firewalls),
- A network (in double red lines) is dedicated to communications related to the management of cryptographic keys via the KMIP protocol
- Crypto keys are generated by the KMS installed on the Proteccio DataProtect Trustway KMS
- Data is encrypted during its communications as well as for storage on the NAS

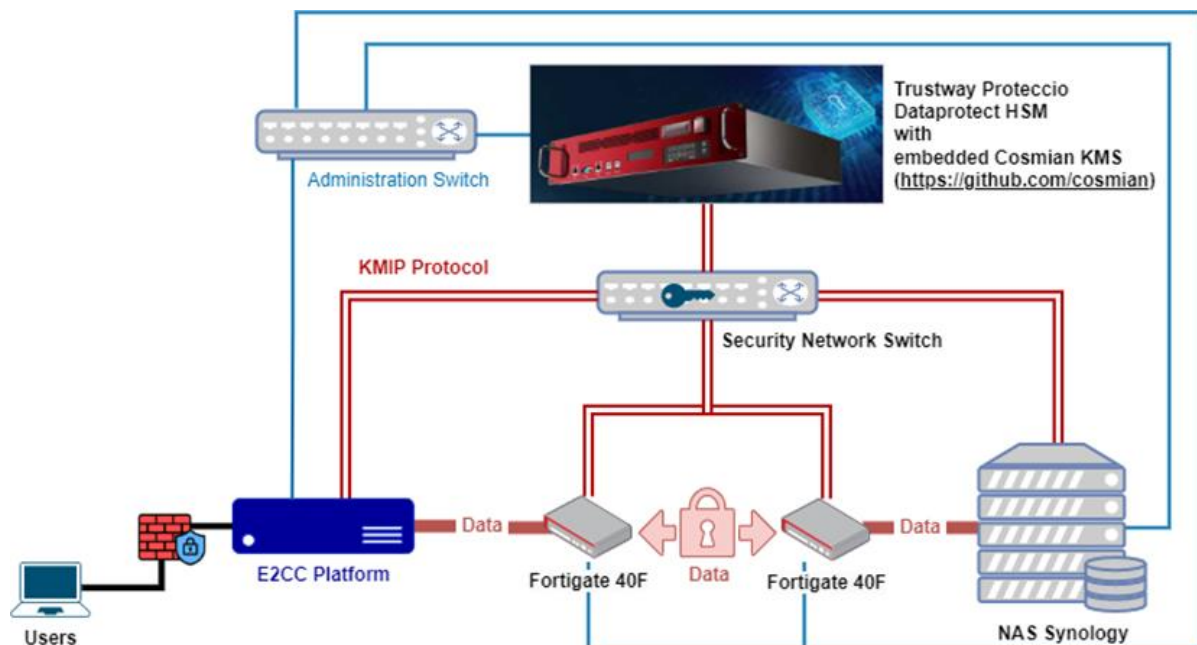


FIGURE 5: HSM WITH EMBEDDED KMS ARCHITECTURE OVERVIEW

Zero Touch Onboarding (ZTO)

One of the major concerns of industrial operators is to ensure the cyber-secure digital transformation of OT systems, while guaranteeing optimum continuity of service and the principles of functional safety in compliance with the legal and regulatory requirements in force (CyberResilience Act, IEC 62443, etc.). Building on IT/OT convergence, the Zero Touch Onboarding (ZTO) approach provides a reliable means of automating tasks, reducing costs and prevent provisioning errors.

Component name	Atos Eviden ZTO (Zero Touch Onboarding)
Architecture location	3 6

Description	Description of the component or activity: The component is a software suite enabling automatic and secure deployment of connected devices (IoT/IoTs). It relies on a PKI (Public Key Infrastructure) to deliver digital identities (X.509 certificates).
Planned work	The work performed in terms of high-level steps: • Development of standardized protocols like BRSKI (RFC 8995), EST (RFC 7030) • Based on X.509 certificates • Integration with secure element • Integration with PKI • Development of web interfaces for the administration and monitoring of the product • Integration of the ZTO components into containers • Setting up a demonstrator in a relevant environment
Software dependencies	A ZTO client is installed in devices that consume ZTO components to obtain digital identities for secure operations.
Work start	January 2024
Work end	March 2026
Outcome	What has been delivered Report and prototype demonstration using LaCroix device and cloud environment
Final demo/evaluation	What has been demonstrated: The goal was to demonstrate the automated and secure onboarding of a device in its operational environment. The features demonstrated are: • Onboarding of a device • Enrolment of a device with digital identity delivery • Renewal of the device identity
Code access	Proprietary Software License. Proprietary code source solution. Code cannot be shared.

Here is the overview of the generic architecture solution:

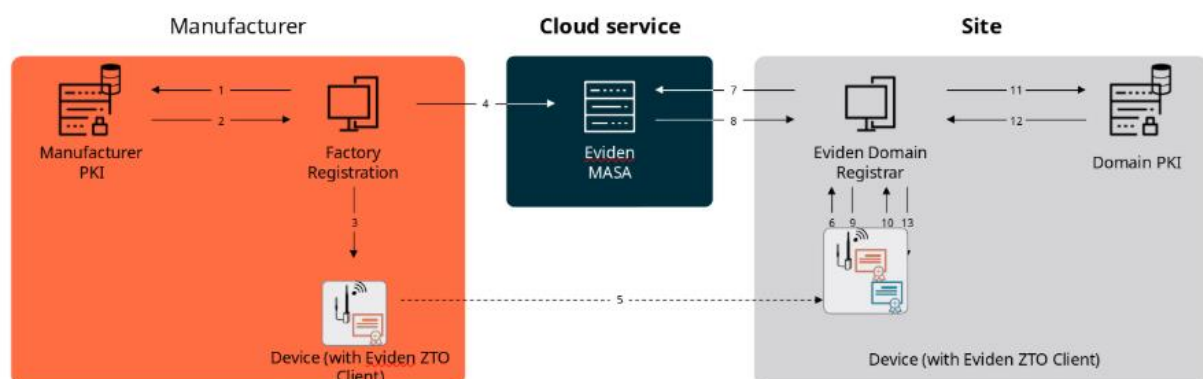


FIGURE 6: ZTO ARCHITECTURE

Anomaly detection

The anomaly detection component developed by UGA will be in charge of detecting anomalies at the software level for applications deployed in the cloud-edge continuum.

Component name	Anomaly detection
Architecture location	
Description	The anomaly detection component will be in charge of detecting problems (with a focus on performance issues) for applications deployed in the cloud-edge continuum. The component will integrate information coming from different levels (hardware, OS, application) to improve the accuracy of anomaly detection. In the Edge context, the goal will also be to have a solution that can work in a fully distributed environment with limited amount of monitoring data to avoid depending on network performance and reduce the energy consumption of the anomaly detection mechanisms
Planned work	• Analysis of the state-of-the-art machine learning techniques for anomaly detection, focussing on Multivariate Time Series • Design of a solution to automatically detect anomalies using data coming from different levels (hardware, OS, software), with the goal of having a solution that is accurate and easy to use for operators (simple configuration) • Improve the solution to minimize the need for coordination between distributed elements of the infrastructure, and to minimize the amount of data to collect to make accurate detection • Run experiments with realistic workloads and anomaly scenarios to validate the approach
Software dependencies	Classical monitoring tools for virtual machines and containers such as cAdvisor and Prometheus
Work start	March 2025
Work end	June 2028
Outcome	We will deliver code and reports. The reports will evaluate the efficiency of the proposed methods in terms of detection accuracy and efficiency
Final demo/evaluation	Experiments evaluating the ability of the tools to automatically detect injected anomalies under realistic scenarios
Code access	The tool will be published as Open Source

The tool developed by UGA analyses metrics in real time to raise alerts in case of anomalies. The proposed approach is flexible and scalable: a dedicated model is deployed for each

component to be monitored in the system. This makes the model of each component light-weight and simple to train. The global architecture is illustrated by the figure below.

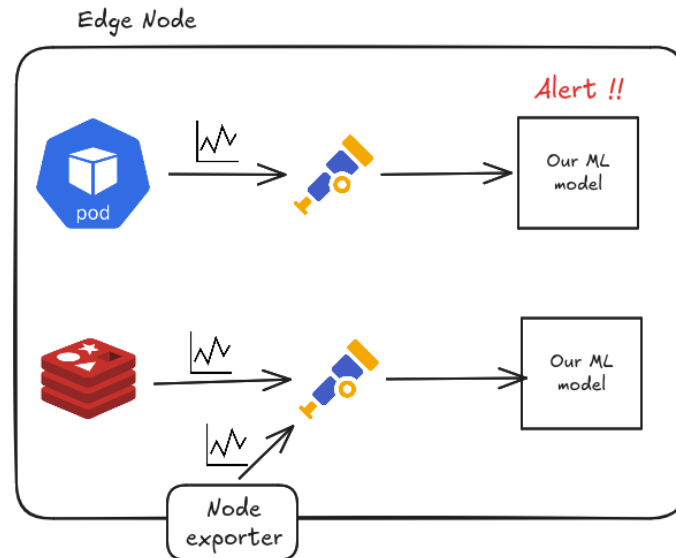


FIGURE 7: ANOMALY DETECTION ARCHITECTURE

Our model takes multiple time series as input coming from different sources. Based on the collected metrics, the model predicts the expected future values and if the future values diverge significantly from the expected ones, an alert is raised. The core of our approach is a model based on a Variational Autoencoder that takes advantage of control variables to make accurate predictions. The way of model works is illustrated by the figure below.

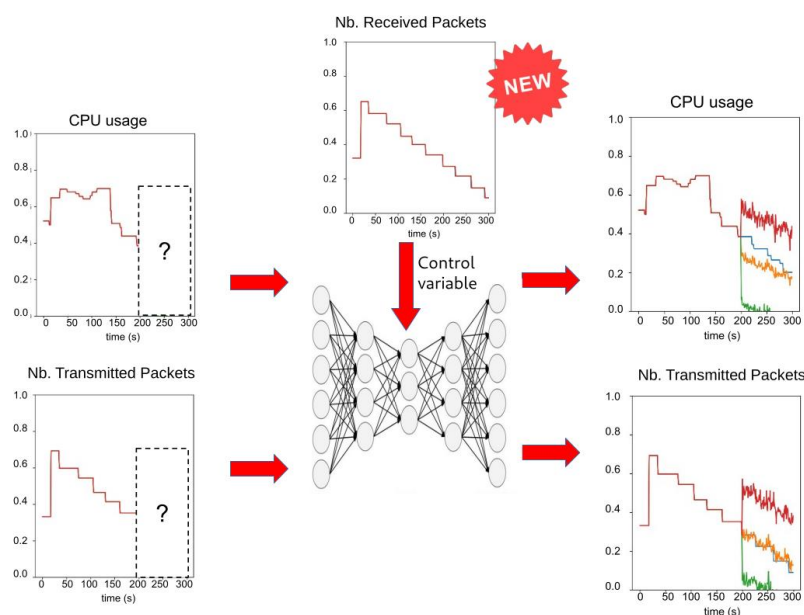


FIGURE 8: ANOMALY DETECTION MODEL

Distributed tracing for detection and mitigation of anomalies and attacks

The distributed tracing component developed by LAAS-CNRS aims at collecting and modelling all data exchanges and events at all levels of the cloud architecture. Based on this model, anomalies and attacks have to be detected and mitigated for enhancing cloud resilience.

Component name	Distributed tracing
Architecture location	
Description	The distributed tracing component aims at collecting and modeling all data exchanges and events at all levels of the cloud architecture. A model representing formally all these interactions will be designed, a priori combining DAG and GNN. Based on new algorithms, sometimes leveraging machine learning techniques, the sources of performance decrease or even denial of services will be detected, analysed, and mitigated. This includes misconfigurations but also anomalies and attacks.
Planned work	• Experiment techniques for distributed tracing, especially focusing on OpenTelemetry: identify its lacks for further proposing an extension (potentially leveraging federated learning) • Design of a model for representing all components interactions and communication at all levels of the cloud infrastructure • Design of algorithms for detecting erroneous, non-optimal and malicious behaviours • Assess the principles of distributed tracing and analysis with realistic workloads and various scenarios to validate the approach
Software dependencies	OpenTelemetry, Kubernetes, operating systems, Docker, applications
Work start	March 2025
Work end	May 2028
Outcome	We will deliver the components of the system as open-source code. The design of the tracing/monitoring system as well as the evaluation methodology and results will be reported in scientific articles and in the project deliverables.
Final demo/evaluation	Evaluation by means of simulation, and experimentally first on SLICES-FR, and further on the E2CC experimental platform.
Code access	The code will be published as Open Source

The overall architecture of the monitoring system, modelled in Figure 10, forms the basis of the experiments conducted on the Kubernetes cluster. This diagram illustrates the end-to-end telemetry chain, highlighting the integration of the various tools and the nature of the network flows circulating within it.

The test application deployed on our cluster is OpenTelemetry Astronomy Shop (otel-demo) [OTEL-DEMO]. It is a platform that simulates an e-commerce site composed of a dozen microservices written in Go, Python, Java, Node.js, and C++. All components natively embed the OpenTelemetry SDKs and APIs, ensuring the continuous and automatic emission of observability signals.

This infrastructure is structured around three main pillars, which will be discussed in the following subsections

- System stimulation and disruption: the generation of realistic user traffic via Locust is coupled with the injection of controlled failures directly onto the execution nodes using Chaos Mesh.
- Collection and processing: the interception of trace and metric signals by the application SDK is followed by their filtering and enrichment through the addition of Kubernetes metadata, operations handled by a local OpenTelemetry agent deployed as a Daemon-Set.
- Export and analysis: asynchronous routing of telemetry data occurs to dedicated storage solutions—namely Jaeger, Prometheus, and MinIO—for analytical use via the Grafana interface.

This overview maps the exact path of the information, from the application to its persistence, and provides an optimal understanding of the observability mechanisms detailed in Figure 9 on next page.

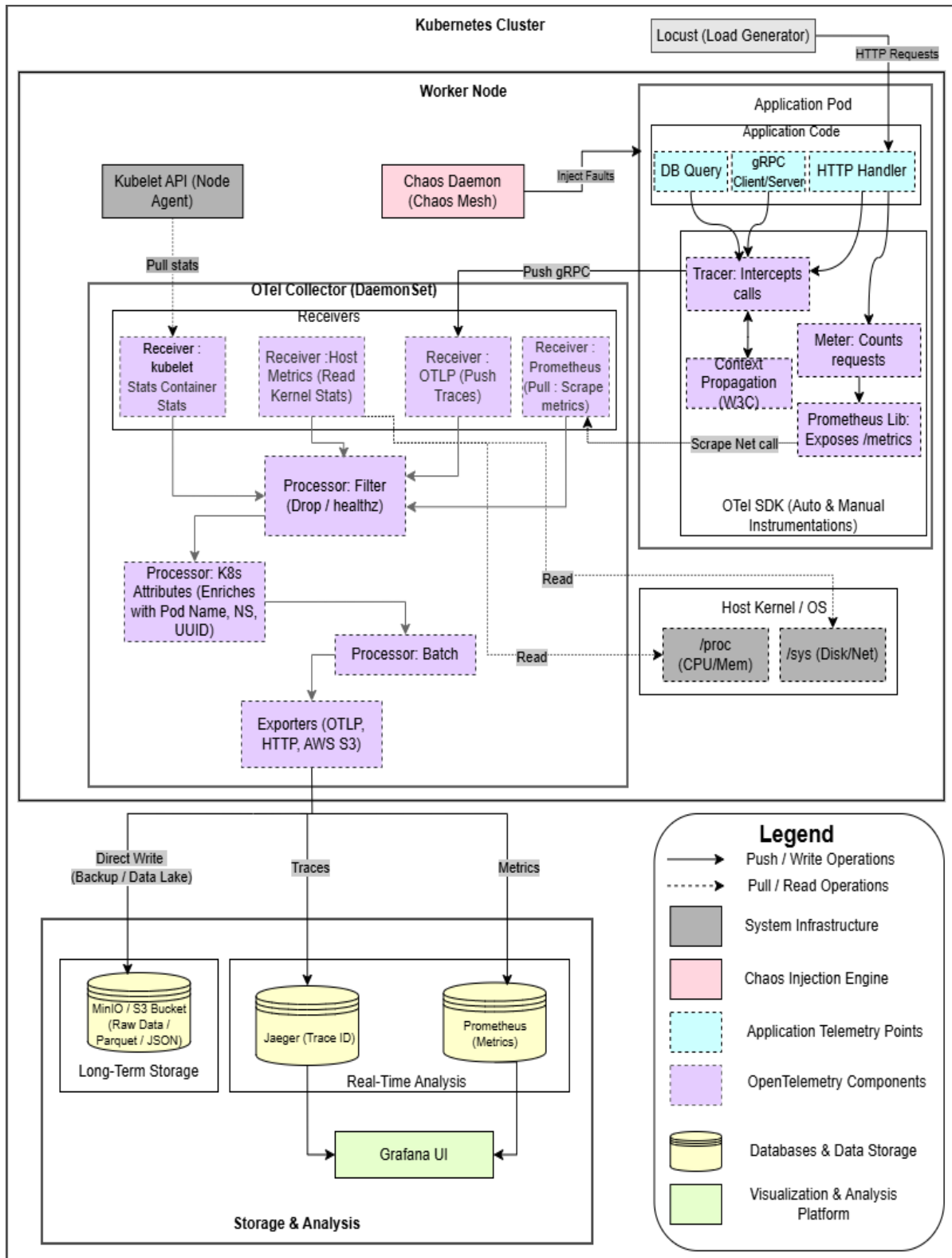


FIGURE 9: OVERALL ARCHITECTURE OF THE MONITORING SYSTEM IMPLEMENTED

E2CC Orchestrator

Component name	E2CC Orchestrator
Architecture location	
Description	The E2CC Orchestrator is responsible for managing and optimizing the deployment, execution, and lifecycle of applications across cloud and edge infrastructures. It ensures efficient resource allocation, dynamic scaling, service-level compliance, and automated remediation. The orchestrator enables seamless workload execution by leveraging (smart) scheduling, policy/service level objective enforcement, and real-time monitoring and observability mechanisms. The Resource & Workload Orchestration module dynamically allocates computing, storage, and networking resources, optimizing workload execution based on real-time conditions, application requirements, and policies. The Service Deployment & Lifecycle Management module automates the deployment, scaling, and termination of applications, supporting containerized and virtualized workloads while enforcing deployment constraints. The Policy & Compliance Engine ensures adherence to predefined Service Level Objectives (SLOs), security policies, and energy-efficiency strategies by continuously evaluating system behaviour and triggering corrective actions. The Monitoring & Observability module provides real-time telemetry, logging, tracing, and AI-driven anomaly detection to enhance visibility into the performance and health of orchestrated workloads. The Smart Placement module implements automated remediation strategies, such as load balancing, fault recovery, and energy-aware scheduling, to ensure resilience and efficient resource usage. By integrating these modules, the E2CC Orchestrator enables intelligent and adaptive application management, ensuring seamless execution, optimized resource utilization, and high availability across cloud and edge environments.
Planned work	Design and implement the core orchestration logic, including workload scheduling, resource management, and policy enforcement. Integrate real-time monitoring and observability features. Develop APIs and interfaces for user interaction via CLI and GUI. Validate performance, security, and scalability through iterative testing and refinement.
Software dependencies	Kubernetes, Container orchestration, RYAX, Open Cluster Management (OCM), Prometheus, OpenTelemetry, ALUMET, Mojito/S, KEPLER, Grafana, AI-based analytics engines, cloud and edge infrastructure SDKs, authentication and security frameworks (Keycloak).
Work start	September 2024
Work end	December 2027

Outcome	Source code, documentation, deployment scripts, integration reports, performance evaluations.
Final demo/evaluation	Demonstration of full application lifecycle management, including deployment, scaling, monitoring, and policy enforcement across cloud and edge infrastructures. Showcase automated remediation in case of failures or policy violations.
Code access	Open-source (Apache 2.0).

Container Orchestration

Component name	Container Orchestration
Architecture location	
Description	Containerization is a modern virtualization technique that streamlines the management of applications on computing resources. It enables users to run multiple operating systems on a single physical server, with each operating system functioning as an independent computer. Containerization has rapidly become an efficient method for delivering advanced computing services. Kubernetes, a container orchestration system, is widely used in industrial fields. Among its components, the container orchestrator (scheduler) is responsible for assigning containerized applications to various heterogeneous computing resources. In this project, the CGI France Research & Innovation team proposes a new Kubernetes (K8s) container management solution tailored for the cloud-to-edge continuum. Our solution is based on the Portfolio principle, which involves deploying multiple container orchestration methods simultaneously. The goal is to implement a generic solution capable of: • Optimizing container orchestration performance • Optimizing the orchestration of independent and dependent containers • Optimizing container orchestration in pre-emptive and non-pre-emptive modes according to container priority
Planned work	• Analysis of the state of the art in container scheduling strategies. • Design of new scheduling strategies tailored to various execution contexts. • Implementation or prototyping of the different scheduling strategies. • Comparative evaluation of the scheduling strategies based on multiple usage scenarios.
Software dependencies	Kubernetes and other tools that can be used to optimize container orchestration.
Work start	January 2024

Work end	December 2027
Outcome	We will deliver a scientific paper that explains the principle of our solution.
Final demo/evaluation	This point will be discussed later.
Code access	This point will be discussed later.

The container orchestration solution will be developed by CGI France as shown in the figure below.

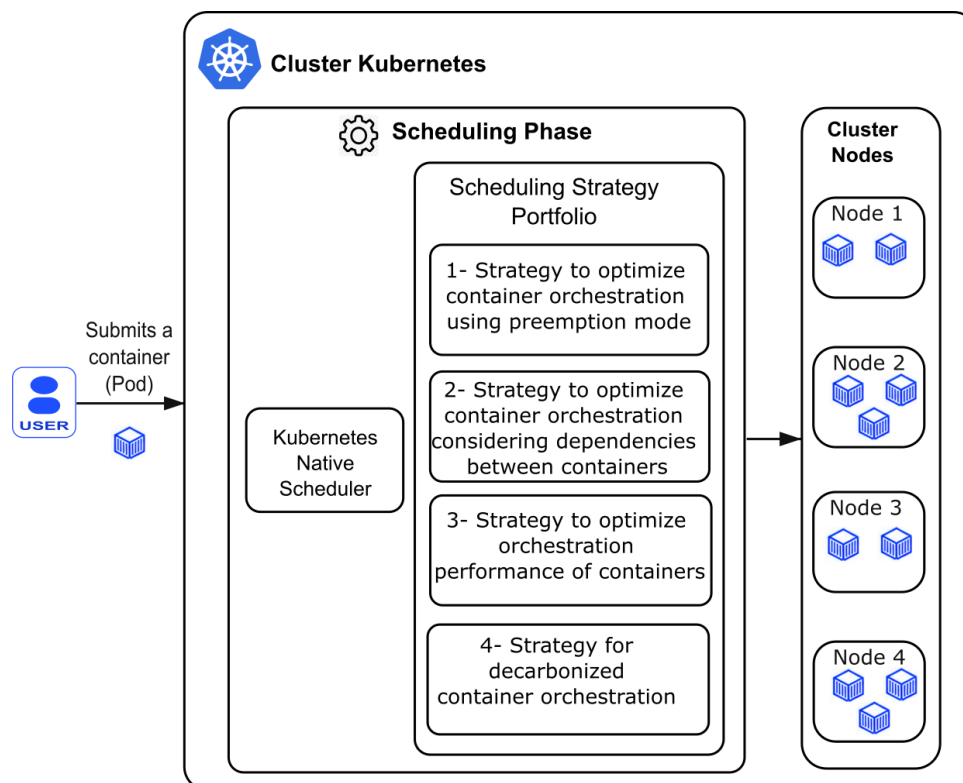


FIGURE 10: CONTAINER ORCHESTRATION

Figure 10 presents the main orchestration strategies that will be developed within this project. When a user submits a container (pod), the Kubernetes scheduler analyses its characteristics and selects the most appropriate orchestration strategy. If the user does not request a strategy included in our scheduling strategy portfolio, the native Kubernetes scheduler is applied by default. However, if a specific strategy is selected, the corresponding scheduling mechanism is used to identify the most suitable node within the cluster for efficient container deployment.

The strategy portfolio includes:

- Strategy to optimize container orchestration using pre-emption mode,

- Strategy to optimize container orchestration considering dependencies between containers,
- Strategy to optimize orchestration performance of containers,
- Strategy for decarbonized container orchestration

It is important to note that the last strategy, focused on decarbonization objectives, will also be leveraged to optimize the performance of the Resource Management component (architecture component #5) developed within the project.

RYAX solution

Component name	RYAX
Architecture location	
Description	Ryax is an open-source, AI-driven workflow and compute management platform. It enables the execution of data analytics and AI workloads (ETL, ML, LLM pipelines) and aims at optimizing resource utilization across CPU, GPU, and RAM on heterogeneous infrastructure (cloud, on-premises, edge, HPC). Within E2CC, Ryax addresses two complementary challenges. First, applications targeting the edge-cloud continuum must be composable and deployable across heterogeneous infrastructure: varying architectures, tiers, and hardware constraints — without requiring deep expertise in distributed systems. Second, once an application is deployed, placement decisions across available resources are not neutral: a cluster may be cheaper, another faster, another greener. A scheduler that treats all feasible candidates as equivalent leaves significant value on the table. To address these challenges, Ryax contributes two components to the project: • A workflow abstraction layer that allows developers to express applications as modular DAG workflows composed of typed, independent components, annotated with resource requirements and deployment constraints, and compiled into multi-architecture container images or serverless functions using Nix; • A multi-criteria placement policy that extends the existing Ryax scheduler with deterministic ranking of feasible candidates by predicted runtime and estimated cost, using a configurable performance/cost trade-off. This policy is opt-in and preserves all existing safety guarantees. It is designed to be extended with energy and carbon objectives in subsequent project phases.

Planned work	• Implementation of a multi-architecture build pipeline (x86_64, ARM64) based on the Nix package manager. • Design and implementation of an opt-in multi-criteria placement policy for the Ryax runner. • Extension of the placement policy to incorporate energy consumption and carbon emission objectives. • Design of a federated resource management architecture supporting intermittently connected edge clusters. • Evaluation of scheduling policies across multiple usage scenarios (cost-sensitive, energy-sensitive workloads).
Software dependencies	Ryax (open-source) Kubernetes / K3s Nix package manager Prometheus (monitoring)
Work start	January 2024
Work end	December 2027
Outcome	Ryax will deliver open-source implementations of the workflow abstraction layer and the multi-criteria placement policy, integrated into the Ryax platform.
Final demo/evaluation	A demonstration will be conducted on a testbed spanning cloud and edge nodes, showing end-to-end workflow deployment with multi-criteria placement decisions across heterogeneous resources. Evaluation will compare placement policies across runtime, cost, and (in later phases) energy and carbon metrics.
Code access	The contributions will be released as open-source, integrated into the Ryax platform repository.

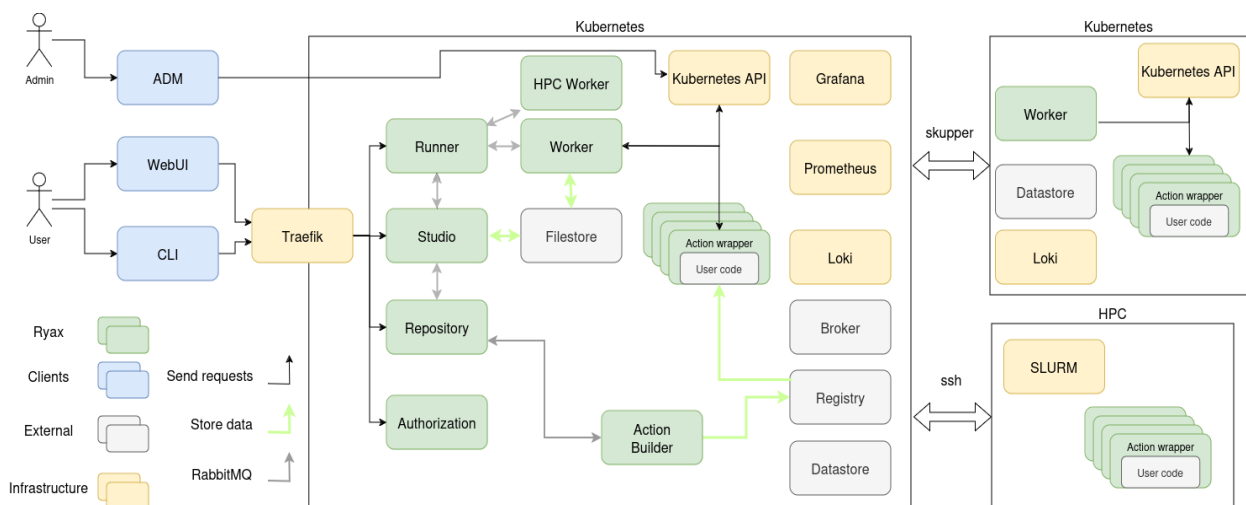


FIGURE 11: RYAX ARCHITECTURE

Figure 11 shows the architecture of Ryax, built as a set of independent micro-services, each responsible for a single functional scope. Services communicate asynchronously through a RabbitMQ message broker, and each service owns its data in an isolated datastore, avoiding shared state and tight coupling between services.

Each micro-service follows Domain-Driven Design principles, organized into three layers: a Domain layer defining core business objects in pure Python, an “Application Layer” orchestrating business logic, and an Infrastructure layer handling all interactions with external systems.

The platform groups its services into three main functional areas. Connectivity services handle user authentication, API exposure, and the connection to external compute sites through a Worker connector model. Thanks to the work carried out in WP2, multiple Workers can be deployed simultaneously across different infrastructures, enabling Ryax to orchestrate workloads across heterogeneous environments combining cloud, on-premises, and edge resources. Storage and monitoring services manage execution data, metrics, logs, and container images. Orchestration services form the core of the platform, covering workflow definition, scheduling, execution lifecycle management, and action building.

The entire platform runs on Kubernetes, which abstracts the underlying infrastructure. External Workers extend the reach of the platform to remote or edge sites, operating autonomously and synchronizing execution data with the central filestore when needed.

E2CC Modular Platform

These components, tight to the cloud-edge orchestration components, provides services to application developers.

ALUMET

The ALUMET component is a generic in-band tool for collecting energy consumption metrics in the IT world. Figure 12 shows the high-level data workflow of ALUMET.

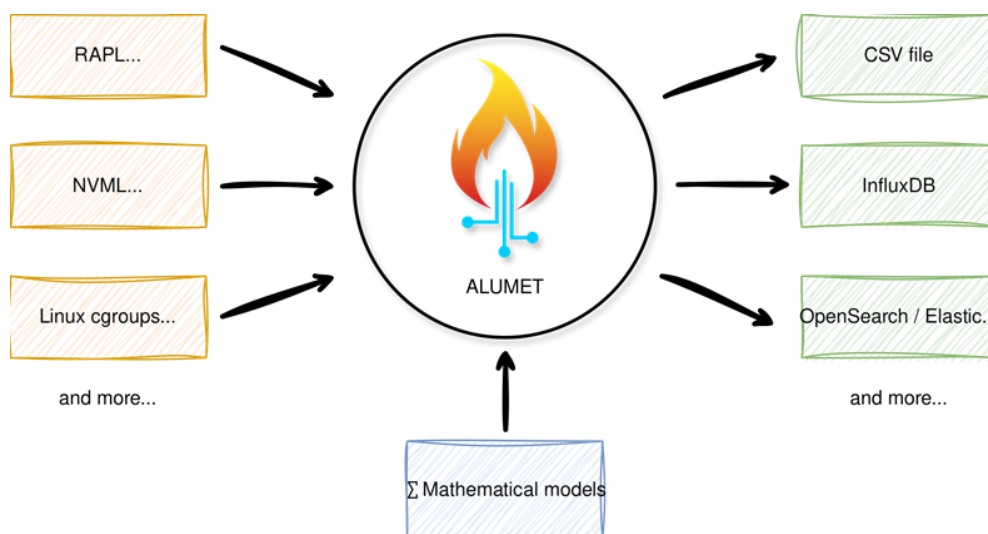


FIGURE 12: ALUMET INPUT/OUTPUT OVERVIEW

Component name	ALUMET
Architecture location	5
Description	Alumet is a novel framework designed to offer an extensible standard for energy measurement in a widerange of scenarios. With Alumet, it becomes possible to build measure-made measurement tools without the need to redevelop everything from scratch. A generic pipeline enables the gathering, transformation and export of any type of measurement. A plugin system allows to support new environments and new models without modifying the framework, thus making it ready for future innovations.
Planned work	Monitor in-band mode following components: • CPUs (x86 and ARM) • GPUs (Nvidia, AMD, and Intel) • memory Measure in-band mode the energy consumed by

	• HPC jobs • Kubernetes pods • Virtual Machines
Software dependencies	ALUMET is developed in RUST. It is a self-supporting binary and must run on a Linux OS, it can be containerized. No dependencies on other software.
Work start	September 2024
Work end	December 2026
Outcome	ALUMET will be delivered as: • Linux package (rpm or deb) • Container • Hem chart
Final demo/evaluation	Measure the energy consumed by applications on E2CC platform.
Code access	EUPL 1.2

In Figure 13 below a typical ALUMET deployment is shown on a Kubernetes cluster. ALUMET client is deployed on each worker node to collect metrics that are sent to ALUMET server. ALUMET server gathers all metrics and sent them to a Prometheus server.

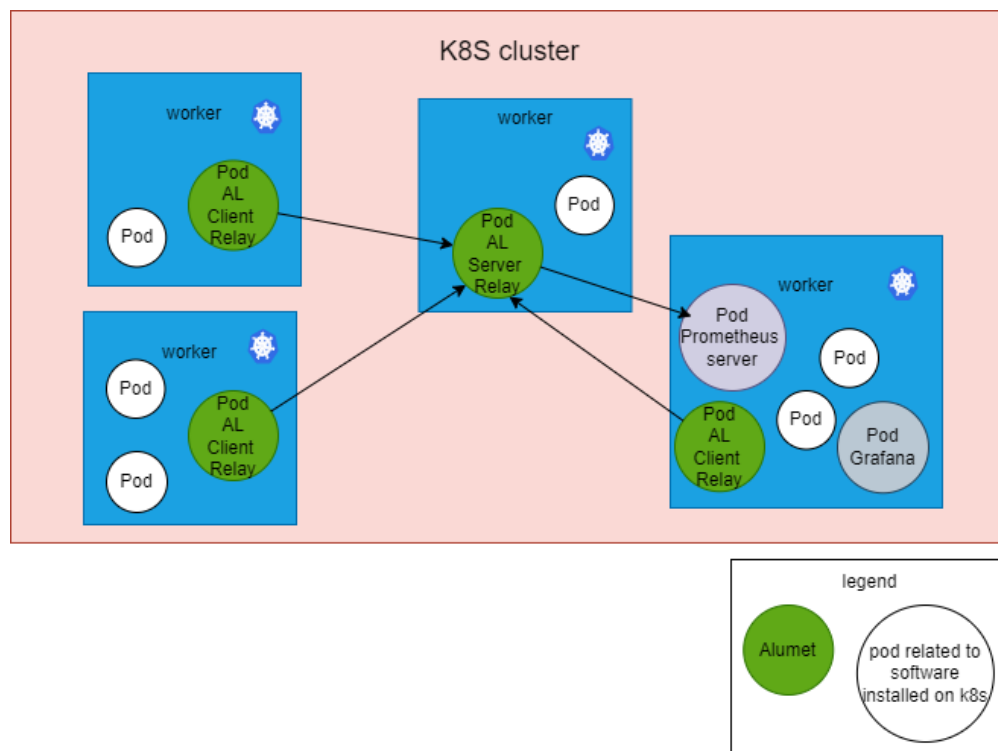


FIGURE 13: ALUMET ARCHITECTURE OVERVIEW

The ALUMET client configuration is described in Figure 14. ALUMET client collects all required metrics for measuring energy consumption of Kubernetes pods.

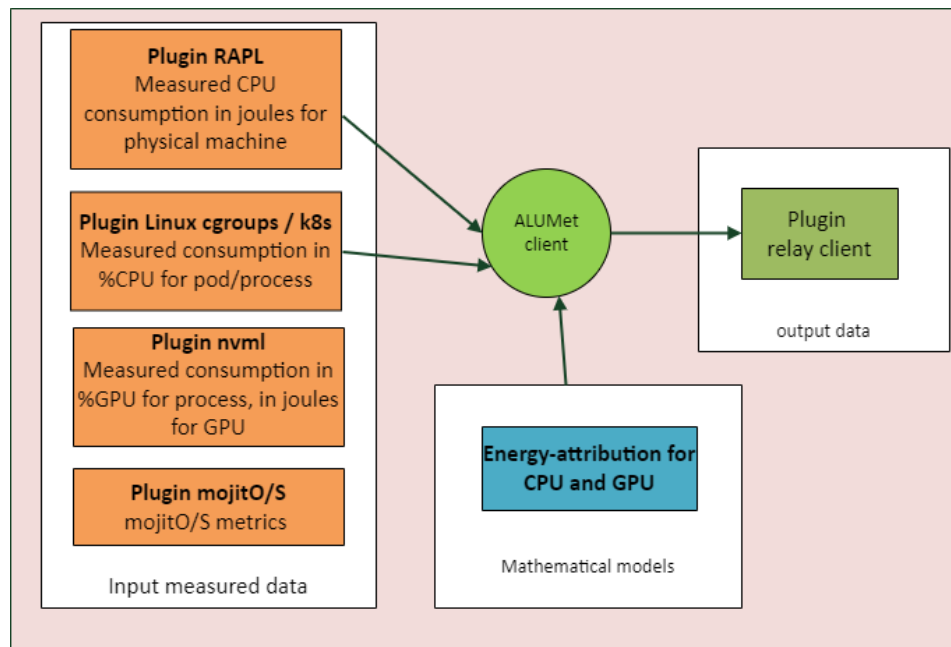


FIGURE 14: ALUMET CLIENT CONFIGURATION

The ALUMET server configuration is described in Figure 15. ALUMET server receives all metrics sent by ALUMET clients and send them to Prometheus server.

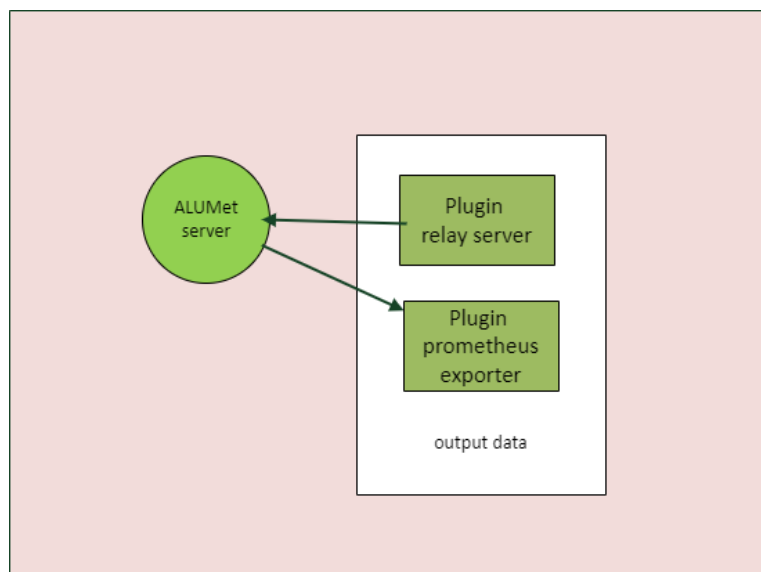


FIGURE 15: ALUMET SERVER CONFIGURATION

MojitoO/S

The component MojitoO/S (<https://gitlab.irit.fr/sepia-pub/mojitos>) is an open-source System, Energy and Network monitoring tools at the O/S level. Figure 16 shows the high-level data workflow of MojitoO/S with : Red for monitored hardware, Green for monitored values, and Blue for available output.

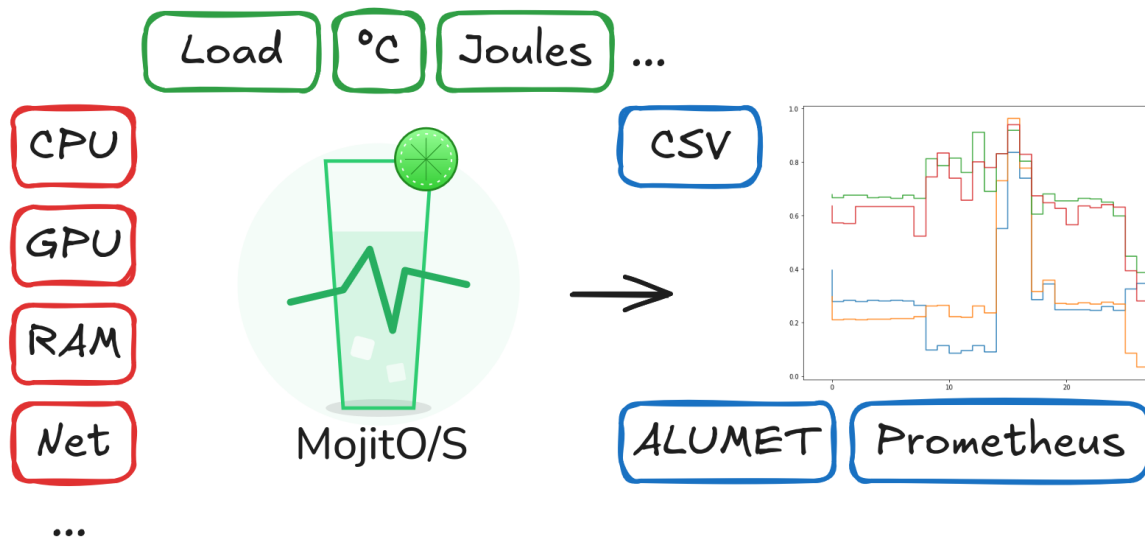


FIGURE 16: MOJITO/S STRUCTURE.

Component name	MojitoO/S
Architecture location	5
Description	MojitoO/S is a generic system level monitoring tool for single node systems. It helps to acquire RAPL, Network, Hardware Performance Counters, GPU, Temperature, ..., and supports a plugin system. Measures are synchronized for: machine learning usage; linking energy consumption and resource usage. It is aimed at HPC systems and have a low overhead, up to 1000Hz for most plugins. It can be used as a plugin for ALUMET and thus provide synchronized measures for many metrics with low overhead. It can also be used as a Prometheus plugin.
Planned work	Low level and low overhead energy measurements of CPU and GPU, along with their performance evaluation.
Software dependencies	MojitoO/S is developed in C and mainly need a classical Linux system. Depending on the hardware it needs low level libraries. Nvml for Nvidia systems is the only one which is not available as a default open-source solution. No dependencies on other software.
Work start	September 2024
Work end	December 2026

Outcome	MojitOS is already delivered as: • spack container • guix using guix-science channel
Final demo/evaluation	Measure multiple resources (energy, memory usage, processor usage) consumed by applications on E2CC platform through ALUMET.
Code access	GPLv3 or later

Resource Management

The resource management solution will be developed by CGI France.

Component name	Resource Management
Architecture location	
Description	The CGI France Research & Innovation team plans to propose a new resource management solution for a Kubernetes cluster deployed in an Edge Computing environment. Our solution addresses the issue of energy frugality, which is one of the major challenges for our future. It is becoming increasingly necessary for companies to optimize the management of their computing resources to reduce energy consumption and limit the environmental impact of their production activities. The solution proposed in this project aims to meet the energy constraint and the cost of resource usage. The objectives are to: • Reduce the costs of using Edge resources in a Kubernetes cluster • Minimize energy consumption and carbon footprint. We also plan to develop a new module that enables automatic detection of execution failures (anomalies) of applications on computing resources using Machine Learning (ML) techniques. The goal of this new module is to improve the Quality of Service (QoS) for users.
Planned work	• Analysis of the state of the art of existing solutions and available tools for measuring energy consumption in a Kubernetes cluster. • Design of a new solution adapted to the Kubernetes environment. • Implementation or prototyping of the proposed solution. • Evaluation of the proposed solution based on multiple usage scenarios.

Software dependencies	Kubernetes and classical monitoring tools for containers such as Prometheus, Grafana, Kepler, and OpenCost.
Work start	January 2024
Work end	December 2027
Outcome	We will deliver a scientific paper that explains the principle of our solution.
Final demo/evaluation	This point will be discussed later.
Code access	This point will be discussed later.

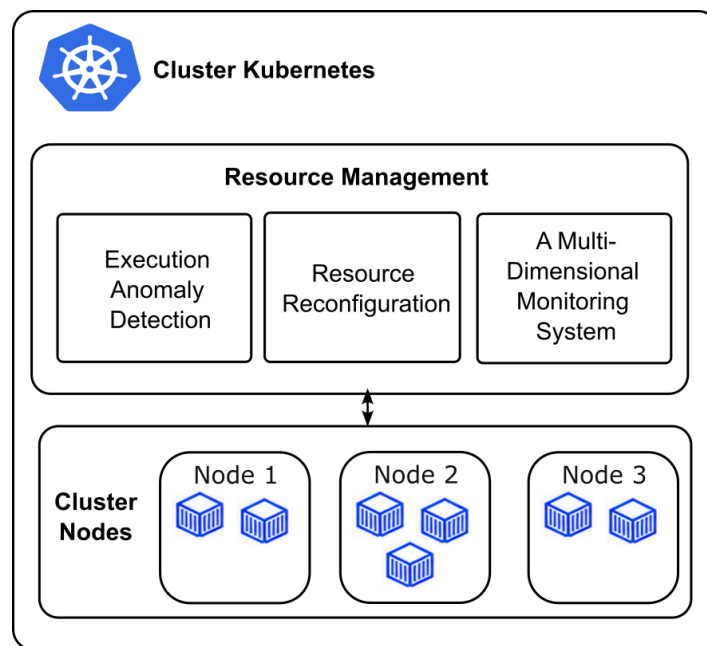


FIGURE 17: RESOURCE MANAGEMENT

To reduce the cost of computing resource utilization, as well as energy consumption and carbon footprint, our solution relies on several complementary modules, as illustrated in the figure above.

The first module, dedicated to resource reconfiguration, dynamically adapts the number of resources allocated to containers according to their actual consumption. This mechanism helps reduce resource waste while also decreasing energy consumption and carbon emissions.

Our solution also integrates a multidimensional monitoring system that supervises energy consumption and carbon footprint at multiple levels, including clusters, namespaces, and pods. This module provides a clear and comprehensive view of the resource and energy consumption of Kubernetes objects. In this context, Mojito/S and ALUMET, presented previously, will also be leveraged during the monitoring phase to collect and analyse energy-related metrics. The collected data will enable administrators to make informed decisions in order to better control and optimize the overall consumption and sustainability of their clusters.

In addition, our solution includes a module capable of automatically detecting execution failures and anomalies of applications running on computing resources by leveraging Machine Learning (ML) techniques.

Container Runtime

Component name	Container Runtime Operator
Architecture location	
Description	The runtime-operator is a multi-runtime Kubernetes operator that installs, configures, updates and lifecycle-manages alternative container runtimes — gVisor, Kata Containers, and KubeVirt — on bare-metal clusters. It is built with KubeBuilder v4 / controller-runtime and follows Cluster-API-style reconciliation patterns.
Planned work	• API & core controllers — CRDs, reconciliation loops, RuntimeClass plumbing, finalizers. • Provider plug-ins — implement gVisor, Kata, and KubeVirt behind the interface. • Per-node delivery — DaemonSet + Job installers, CRI-agnostic registration, KVM/virtualization node labelling. • Update strategies — Rolling, Canary, Blue-Green, Drain-and-Replace with state recovery and rollback. • Admission & policy — validating/mutating webhooks; optional Kyverno RuntimeClass-injection policies. • Observability — metrics, tracing, health checks, dashboards, alerts. • Production hardening & evaluation — CRD versioning (v1alpha1→v1beta1), load/chaos testing, air-gapped install, multi-cluster, end-to-end validation on a real cluster.
Software dependencies	Golang, KubeBuilder, controller-runtime; Kubernetes 1.30+; containerd and/or CRI-O on nodes; gVisor (runsc), Kata Containers (containerd-shim-kata-v2), KubeVirt operator; cert-manager (webhook certs); Prometheus / OpenTelemetry collector / Grafana (observability); cosign / Sigstore (image verification); Kind + an OCI registry (e.g. Zot) for local dev; Ginkgo/Gomega + envtest (testing); optionally Kyverno (policy).
Work start	July 2025
Work end	November 2026
Outcome	Code — the operator (controllers, providers, strategies, installers, webhooks), Helm chart and kustomize manifests, and per-runtime installer container images. Report — architecture documentation, deployment/operations guide, and an evaluation report.
Final demo/evaluation	On a bare-metal (or Kind) cluster: apply RuntimeConfigs for gVisor, Kata and KubeVirt → operator detects node capability, installs each runtime, registers RuntimeClasses → schedule pods onto each runtime → demonstrate a zero-/low-disruption version upgrade via an update strategy with automatic rollback on failure →

	show metrics/health and evaluate install success, update disruption, rollback correctness, and observability coverage.
Code access	The tool will be published as Open Source

Confidential computing

Component name	Confidential Computing
Architecture location	
Description	CRD-driven Kubernetes operator that installs, configures, updates and lifecycle-manages confidential-computing-capable container runtimes (Kata on SEV-SNP/TDX) and confidential VMs (KubeVirt launchSecurity) on bare-metal nodes. Acts as the <i>enabler / deployment layer</i> : detects per-node TEE capability, installs the measured Kata guest kernel+rootfs, registers RuntimeClasses, and provisions the node-side components required for remote attestation. It does not itself attest or hold keys — it ensures the expected, measured runtime is present so the external attestation/KBS flow can verify and release secrets.
Planned work	• 1. TEE detection — extend the node-label subsystem to detect AMD SEV-SNP / Intel TDX; emit • Confidential Kata provider — install CC kernel+rootfs, register a kata-cc RuntimeClass, gate scheduling to TEE-capable nodes. • KubeVirt confidential VMs • Attestation enablement — provision guest-components / attestation agent; reference a KBS/Trustee endpoint; guarantee measured images for KBS policy. • Observability + validation — CC-readiness conditions/metrics; end-to-end validation on SEV-SNP/TDX hardware.
Software dependencies	Kubernetes 1.30+, controller-runtime / kubebuilder; containerd or CRI-O with the Kata shim (containerd-shim-kata-v2); Kata Containers with a confidential guest (SEV-SNP/TDX kernel + rootfs); KubeVirt operator (confidential-VM path); an external Key Broker Service / attestation service (e.g. Confidential Containers <i>Trustee</i> /KBS + verifier); cert-manager (webhook certs), Prometheus (metrics), cosign / Sigstore (image verification); TEE-capable bare-metal nodes (AMD EPYC SEV-SNP / Intel Xeon TDX with the required firmware + host kernel).
Work start	November 2026
Work end	June 2028
Outcome	Code — operator extensions (TEE detection, confidential Kata provider, KubeVirt launchSecurity , attestation provisioning) plus

	CC runtime-installer container images. Report — architecture document, deployment guide, and an evaluation report.
Final demo/evaluation	On SEV-SNP/TDX hardware: apply a RuntimeConfig requesting confidential Kata → node is labeled TEE-capable → pod is scheduled into a measured TEE VM → successful remote attestation → KBS releases a secret/key; plus a KubeVirt confidential VM booting with launchSecurity . Evaluate isolation guarantees, attestation success rate, and the install/update lifecycle.
Code access	The tool will be published as Open Source

Figure 18 shows the runtime operator and its neighbours: the user, the Kubernetes control plane, the external KBS/attestation services, and the TEE worker nodes. Green boxes represent what already exists in the operator, orange boxes represent what is added by the Confidential Computing feature, and purple boxes represent the external systems that communicate with the operator. The operator installs and manages the runtime while key release and attestation live in the KBS.

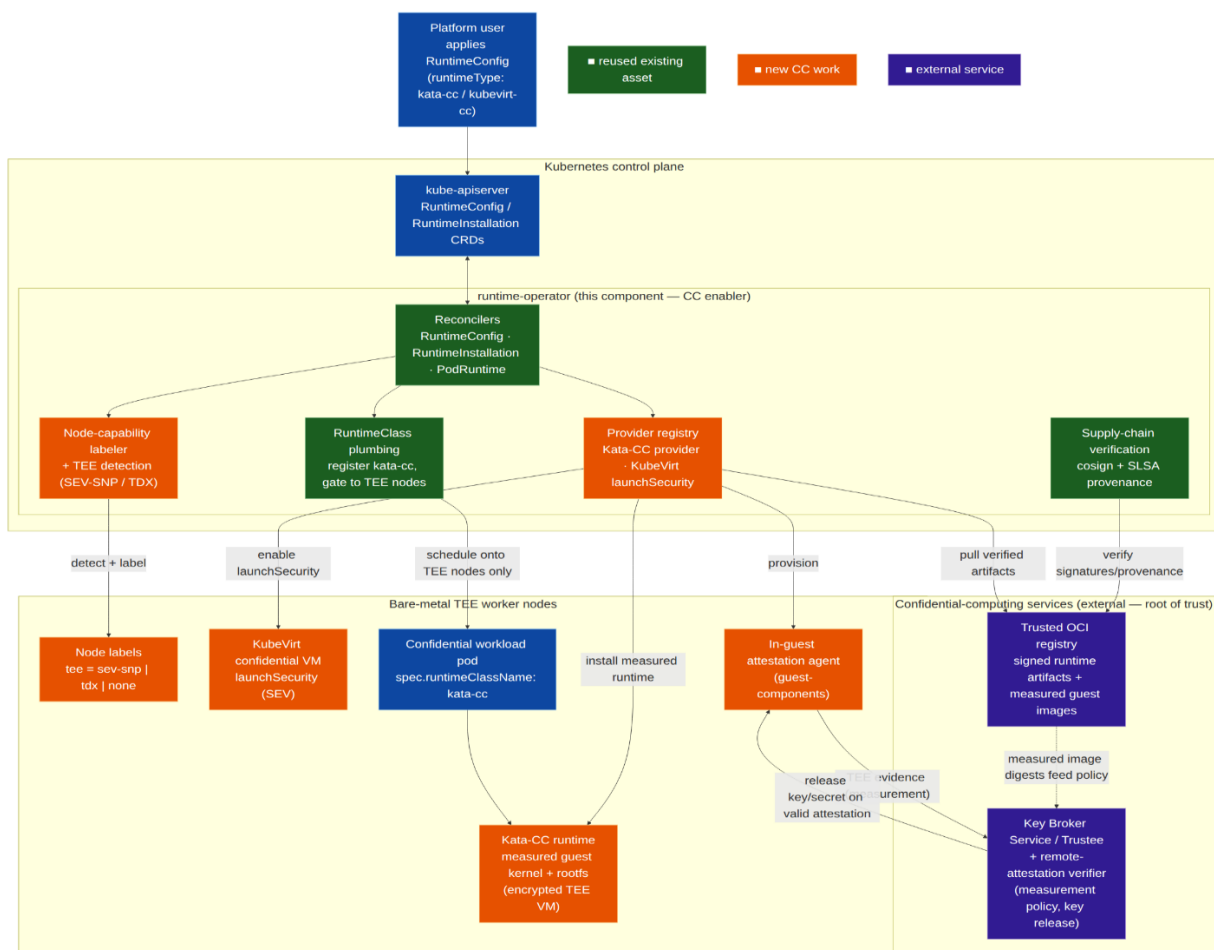


FIGURE 18: CONFIDENTIAL COMPUTING ARCHITECTURE

Figure 19, on next page, shows the workflow of a confidential computing use case:

1. Install phase (1–9): the operator detects TEE capability, labels the node, verifies artifacts with cosign, installs the Kata-CC runtime, and registers a RuntimeClass gated to TEE nodes.
2. Workload phase (10–16): a POD is scheduled into an encrypted TEE VM, the in-guest agent sends evidence to the KBS, and the KBS releases the secret only if the measurement checks out.
3. The operator does the setup; the KBS makes the trust decision.

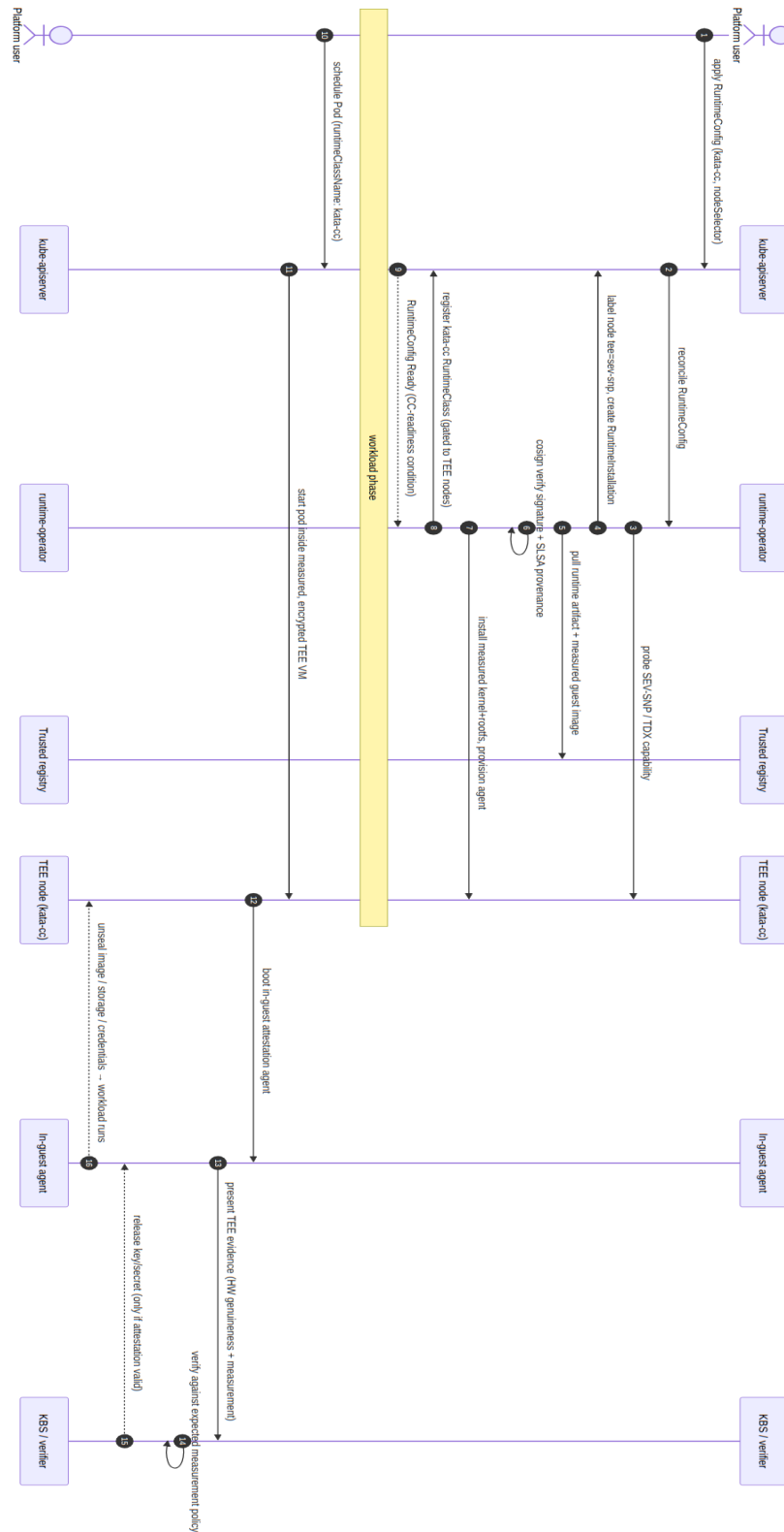


FIGURE 19: CONFIDENTIAL COMPUTING WORKFLOW

AI MLOps

The aim of MLOps component is to develop technologies enabling MLOps and AI on continuous Cloud-Edge computing environments with a high level of automation, interoperability, security, fairness and energy efficiency.

Component name	MLOps and AI enabled on continuum environments
Architecture location	
Description	The primary objective will be to develop a comprehensive solution for running efficient, scalable, federated, energy-optimized, and automated MLOps processes across Edge-to-Cloud infrastructures, meeting emerging requirements for producing reliable and greener AI systems. Considering the specificity and complexity of the Edge-to-Cloud continuum, Work Package 3 will enable: 1. Manage the complete lifecycle of machine learning models (and associated data and experimental design) (T3.1). 2. Create the necessary facilitators to ensure data confidentiality is preserved (T3.2). 3. Integrate the necessary mechanisms to ensure the security of the overall solution and its compliance with applicable regulations (T3.2). 4. Increase the robustness and resilience of MLOps processes and resulting neural networks against attacks (T3.3). 5. Ensure the reproducibility of results by producing detailed traces during MLOps operations and enabling future model scaling to meet new customer and user requirements. Implement mechanisms to ensure that models can be inspected or audited for critical applications, in line with the recommendations of the new regulations proposed by the European Commission (T3.3). 6. Reduce the amount of data required to train machine learning models, thereby increasing their energy efficiency. AutoML features will be enriched and extended to account for the carbon footprint of the entire MLOps workflow and the resulting model (T3.4). 7. Implement continuous learning so that models can be dynamically retrained using new data, overcoming issues such as domain shifts (T3.4).
Planned work	Task n° 3.1 MLOps for Continuum Computing Infrastructures Manage the complete lifecycle of Machine Learning models (and associated data and experimental design) • Ningaloo works on the use of neural networks to detect and fix time-series in a real time • Eviden build neural networks working on image detection in real time. The developed algorithms/models will be integrated in the computer vision platform, a video

	analysis platform that has multi-camera object tracking capabilities. • Armadillo works on post-processing of natural language processing (NLP) neural network to implement indexing in data base to facilitate search. Task n° 3.2: MLOps safe • Create the facilitators required to ensure that data confidentiality is preserved • Integrate the mechanisms necessary to guarantee the security of the overall solution and its compliance with current regulations Task n° 3.3: Robust, transparent and repeatable MLOps • Increase the robustness and resilience of MLOps processes and resulting neural networks against attacks • Ensure reproducibility of results, by producing detailed traces during MLOps operations and allowing models to be extended in the future to meet new customer and user requirements. Implement mechanisms to ensure that models can be inspected or audited for critical applications, in line with the recommendations of the new regulations proposed by the European Commission. Task n° 3.4: Eco-responsible AI • Reduce the amount of data required to train machine learning models, thereby increasing their energy efficiency. AutoML capabilities will be enriched and extended to account for the carbon footprint of the entire MLOps workflow and the resulting model. • Achieve continuous learning so that models can be dynamically retrained using new data, overcoming issues such as domain change.
Software dependencies	Libraries deep learning, NLP, Edge TPU (software), AI framework, Notebook (Jupyter)
Work start	January 2024
Work end	June 2028
Outcome	MLOps solution report
Final demo/evaluation	Study reports Federated learning study report Robustness, transparency and reproducibility evaluation report
Code access	• Both proprietary model and code and some code that are meant to be open source.

AI/MLOps Application: Anomaly detection and correction in time series

Component name	AI/MLOps Application: Anomaly detection and correction in time series extracted from sensor networks
Architecture location	
Description	Ningaloo will put into practice the generic MLOps utilities developed in the above component and apply them to the following use case: automatically compute correlation models between very large time series and use them in real time to detect anomaly and fix them. Since real world data are not always open, the whole deployment process will need to take care on data privacy. Security layers will be developed to maintain data and computation opacity regarding to other users of the platform. Compacity and efficiency of models and algorithms will be optimised to reduce needs of computation resources, memory and data transfers.
Planned work	• Formalisation of correlation models between time series • Definition and development of algorithms to automatically compute and maintain correlation up to date based on AI technologies and comparison with non-AI techniques. • Definition and development of algorithms to use correlation models to detect and fix anomalies in time series in real time. • Deployment of a secured end-to-end process to ensure confidentiality of user data. • Test effectiveness of developed algorithms on real world data and benchmark comparisons.
Software dependencies	AI framework
Work start	January 2024
Work end	June 2028
Outcome	Correlation model formalisation. Algorithms to compute correlation models between time-series, detect anomalies and automatically fix them. Test and benchmark reports.
Final demo/evaluation	POC on real world data
Code access	Both proprietary model and code and some code that are meant to be open source.

Distributed Edge / Cloud Architecture

Use case: anomaly detection in time series

Containerized federated learning system (Kubernetes) — edge acquisition, model aggregated in the cloud

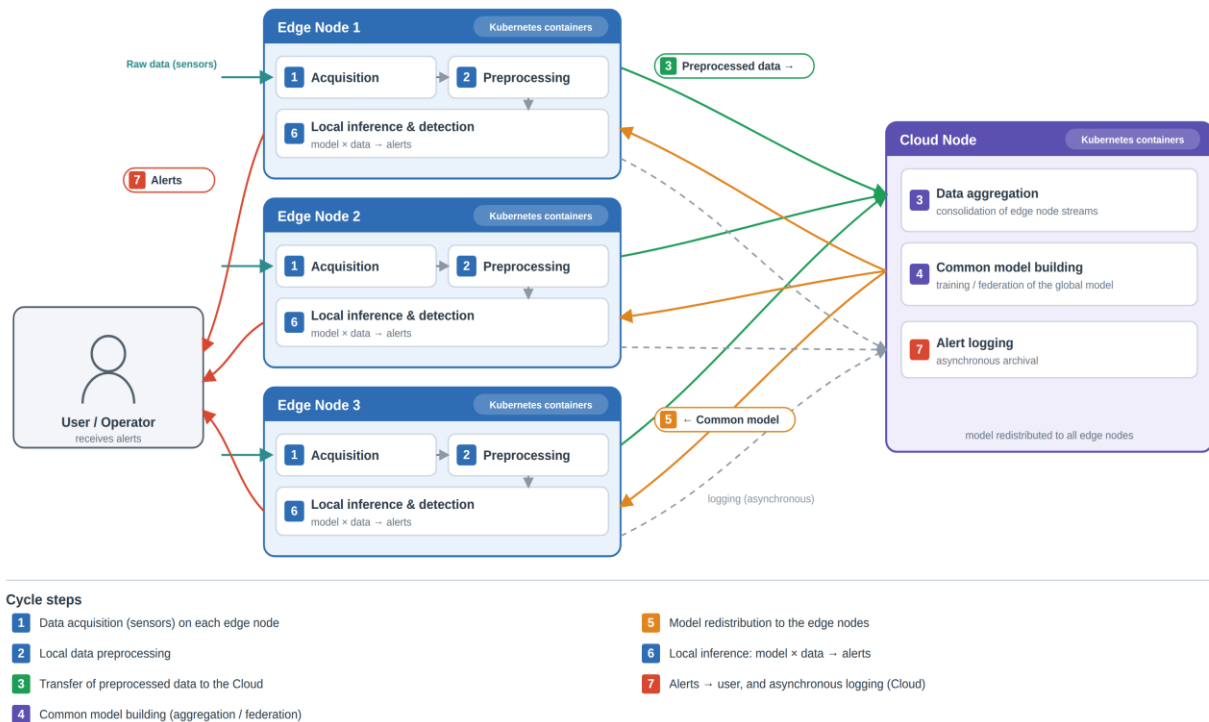


FIGURE 20: ANOMALY DETECTION IN TIME SERIES ARCHITECTURE AND WORKFLOW

AI/MLOps Application: Design and development of new automatic indexing algorithms

Component name	IA/MLOps Application: Design and development of new automatic indexing algorithms using a post-processing system for neural network outputs and application to the semantic indexing of illustrations
Architecture location	7
Description	In this project, we want to demonstrate that we can obtain a high-performance document or media indexing system by focusing on the exploitation of the results obtained at the output of small neural networks and in particular through the use of a thesaurus in order to reduce the size and training time of neural networks. In this context, we will compare the indexing systems obtained using a large neural network with a system using a small neural network for which training would be low cost, but the outputs would be exploited using algorithms and result exploitation tools such as a thesaurus. We also compare

	our results to the use of unsupervised methods. Our evaluation tools will focus on the ability to find documents or media similar to a first image in a database.
Planned work	• Formalization of document or media indexing models • Definition and development of algorithms to extract information to index a document or other media • Definition and development of algorithms to perform similarity searches between documents or media. • Deployment of a secure end-to-end process to ensure confidentiality of user data. • Test effectiveness of the developed algorithms on real data.
Software dependencies	Libraries associated to python, deep learning and NLP MLflow platform Query engine
Work start	January 2024
Work end	Decembre 2026
Outcome	Formalization of indexing model Formalization of similarity search model Method of Neural network and unsupervised methods used Method of post- processing algorithm used Test and evaluation report
Final demo/evaluation	POC on real world data
Code access	Both proprietary model and code and some code that are meant to be open source.

IAM solution

Component name	Atos Eviden Orbion (formerly IDaaS) Access & Governance
Architecture location	Core software on public Cloud (GCP) IAM Provisioning Gateway is an optional service on the E2CC platform 
Description	Evidian Orbion (formerly IDaaS) Access & Governance is a cloud-based IAM solution combining access management and identity governance. It provides centralized control of access to applications using SSO, MFA, and federation (SAML, OIDC, OAuth2). It automates the full identity lifecycle (joiner, mover, leaver) with account provisioning and deprovisioning. The solution enforces advanced security models based on multi-dimensional RBAC and least privilege principles.

	Users can request access via self-service workflows with configurable approval processes. It supports access certification campaigns and segregation of duties (SoD) for compliance. Built-in analytics and dashboards provide audit, reporting, and risk monitoring capabilities. It integrates with hybrid environments (cloud and on-prem) through standard connectors (AD, SCIM, SaaS) and a Provisioning Gateway to manage the identities of on-premises from the cloud without requiring network infrastructure changes. A GenAI assistant helps administrators perform tasks and configure policies. Its Kubernetes-based architecture ensures scalability, security, and high availability. Prescriptive IAM (WP5.2) complements the core IAM solution with an IAM event processing pipeline, from event observation to operational use. This pipeline combines several complementary building blocks: leveraging the Analytics module for event consolidation and contextualization, a Statistical AI module detects behavioural anomaly and combined with Symbolic AI for deterministic interpretation using business rules. Finally, a Workflow Runner automatically orchestrates the resulting actions.
Planned work	The work to be performed in high steps: • Implement centralized access control layer: SSO, MFA, and federation capabilities (SAML, OIDC, OAuth2). • Build identity lifecycle management workflows: provide automation for joiner, mover, leaver processes with provisioning and deprovisioning across target systems. • Design and enforce authorization model: implement multi-dimensional RBAC with least-privilege principles and policy enforcement. • Develop self-service access request portal: enable users to request access with configurable approval workflows. • Implement compliance and governance controls: access certification campaigns and segregation of duties (SoD) management. • Deliver audit, reporting, and risk monitoring capabilities; build dashboards and analytics for audit trails, reporting, and risk detection. • Implement connectors (AD, SCIM, SaaS) and support both cloud and on-prem systems, with the development of a provisioning Gateway, enable cloud-based identity governance to manage on-prem systems without network changes. • Integrate GenAI-powered administration assistant: provide AI-assisted operations for administration, configuration, and policy management.

	• Design and deploy scalable infrastructure: implement Kubernetes-based architecture ensuring scalability, resilience, and high availability. • Design, model & implement Statistical AI module detects behavioural anomaly • Combine it with a Symbolic AI module for deterministic interpretation using business rules. • Complement it with a Workflow Runner to automatically orchestrate the resulting actions.
Software dependencies	Dependent of public Cloud infrastructure for the Core IAM product. The IAM Gateway depends on the E2CC core services. E2CC applications (see integration below), leverage the authentication service of the IAM software, and may, optionally, be managed by provisioning connectors hosted by the IAM Gateway.
Work start	Janvier 2024
Work end	December 2025 Note: integration will be done in 2026.
Outcome	The software described in “Description” (and detailed in the corresponding Work Package delivery note) plus an integration with some applications of E2CC, as explained below.
Final demo/evaluation	The goal is to demonstrate the central management of users in the IAM software and the delegation of application authentication to the IAM software, based on Identity Federation. Optionally, the provisioning and deprovisioning of application accounts by the IAM software. Concerning Prescriptive IAM the goal is to demonstrate how we detect and handle some abnormal situations (e.g. a user performing an abnormal number of actions, connexions from different geographies in a very short timeframe, ...)
Code access	Proprietary Software License. Proprietary code source solution. Code cannot be shared.

The following schemas show how IAM can be used to secure the accesses to the application of the E2CC platform and centrally manage the identities and their profiles/rights in the different applications.

Identity Federation

The first level for securing the access to an application is to delegate its authentication to the Identity Provider (IdP) provided by the IAM platform. This can be done by leveraging the standard protocols SAML or OpenID Connect.

The following schema in Figure 21 shows a (simplified) authentication flow using OpenID Connect:

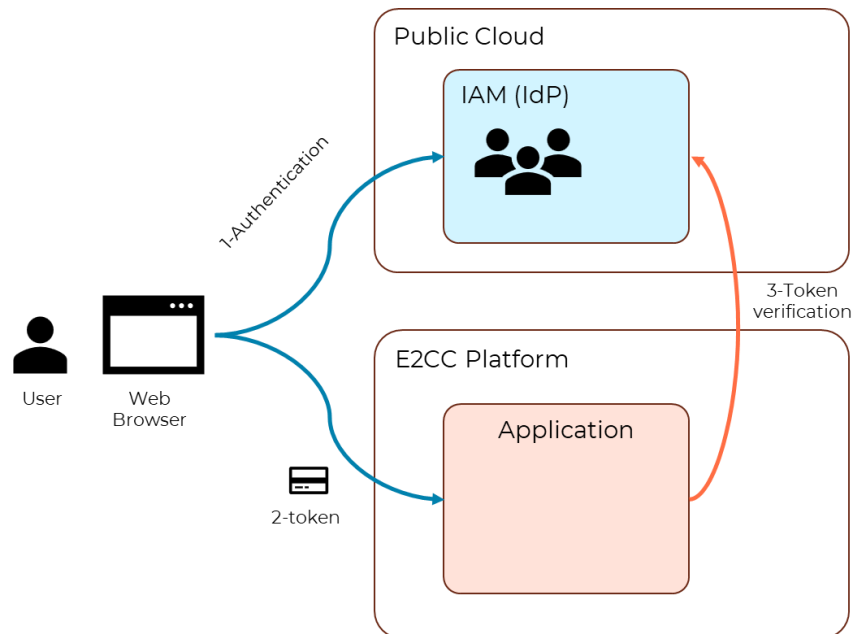


FIGURE 21: AUTHENTICATION FLOW USING OIDC

When users access an application:

- They are redirected (Web redirection 302) to the authentication UI of the IDP
- They provide their credentials, depending on their authentication method(s)
- The server provides a token which is sent to the application
- The application verifies the token and retrieves the user identity

Depending on the applications, an account can be created on demand, or it must have been created beforehand by an administrator of the application, who can manage the authorizations of the users inside the application.

Token-based authorization

The token-based authorization enhances the previous method by delegating the user authorization management: the application, instead of needing to assign authorizations to accounts, delegates this management to the IAM module.

Then, when the users authenticate, their authentication token is extended with authorizations (or claims). With this information, the application can dynamically get the users' authorizations from the token, instead of requiring a local administration in the application.

Account provisioning & deprovisioning

This other option of application integration can be used in complement of the token-based authorization or replace it.

Account provisioning is another method of delegating the management of the account of an application in an IAM software: users and authorizations are managed centrally in the IAM software, and a provisioning connector is used to create corresponding accounts in the applications and synchronize their authorizations. The provisioning connectors can also manage

the state of application account, for instance to deactivate accounts, and are able to terminate accounts (this is “deprovisioning”) when they are not needed anymore. [Note: when a user leaves, their accounts are usually deactivated during a retention period, until they get deleted to comply with regulations].

Provisioning connector role is to convert the generic authorization model to application specific model, leveraging their APIs, which can comply with different standard or be proprietary.

The IAM software we provide for this project is hosted in a public cloud, so the connectivity from the public cloud to the E2CC platform is not possible allowing inbound connections from the Cloud. We have implemented a more secure and less intrusive method, leveraging a provisioning Gateway that hosts the connectors. The provisioning Gateway leverages connections from the E2CC platform to the public cloud to retrieve and execute the provisioning orders issued in the IAM software.

The following schema in Figure 22 shows this securing use case. Note that, due to the innovative architecture of the IAM Gateway, the network connections are always initiated from the E2CC platform to the public cloud.

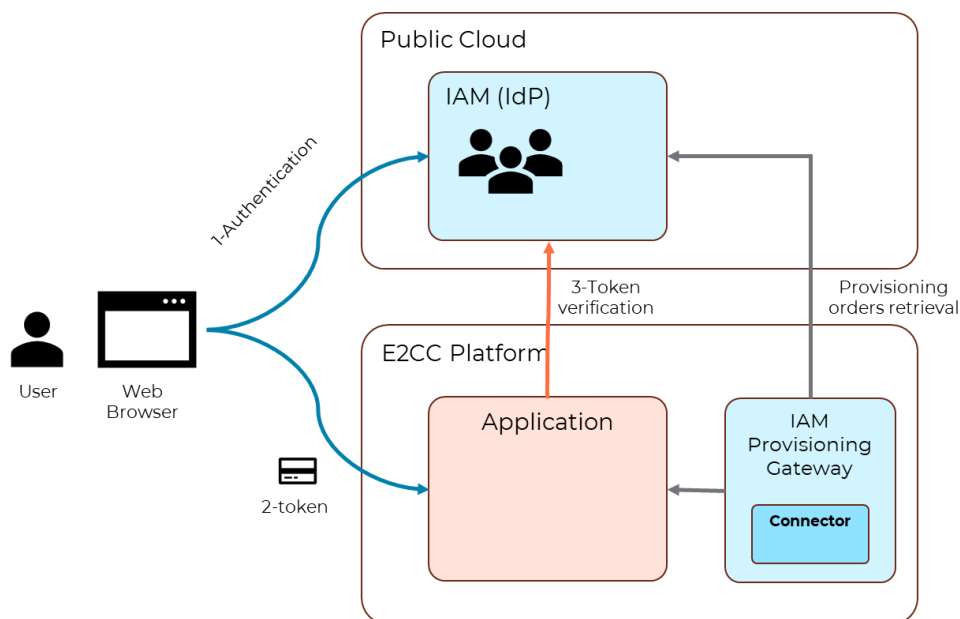


FIGURE 22: SECURING INTERACTIONS

Cloud-edge applications

These applications provide initial use-cases demonstrating the capabilities of our proposed E2CC platform.

To test the integrity of the E2CC platform, several applications will be developed. These applications will use supervised methods described in the MLOps component and unsupervised methods developed according to DevOps governance. This component describes the applications in their entirety, emphasizing the functional aspect.

AI-analytics solution

Component name	Computer vision
Architecture location	
Description	The objective of this component is to deploy Ipsotek's AI video analytics solution, VISuite, within the E2CC platform. VISuite is built on a Scenario-Based Rule Engine (SBRE) combined with a GPU-powered deep learning engine, enabling real-time detection, classification, and tracking of objects (people, vehicles) across large CCTV camera networks. The deployment targets an edge-to-cloud architecture: video processing and AI inference run as close as possible to the cameras (edge), while orchestration, metadata storage, and post-event analytics are handled on the cloud side. Compliance with applicable regulations (GDPR, AI Act) remains a core constraint, particularly regarding facial recognition and person tracking.
Planned work	• To validate the integration of VISuite into the E2CC platform, leveraging components from the various Work Packages and project partners' building blocks. • Set up the required infrastructures and datasets with first customers to validate edge-to-cloud processing capabilities (local inference + cloud-side upload of metadata and alerts). • Leverage VISuite's automatic calibration and rule templating mechanisms to accelerate large-scale deployments across distributed camera networks. • Demonstrate the ability to adapt and customise VISuite's AI models (object classification, detection, tracking) to customers' specific contexts, without exfiltrating data outside their environment. • Assess the performance gains achieved through the adaptation of models on local production data (edge), leveraging the MLOps framework of the E2CC platform.
Software dependencies	Other software/components required for the final release/evaluation

	From WP1: HW & SW BMaaS From WP2: Edge Agnostic Service Abstraction Layer, Trustworthy Edge HW From WP4: Carbon Management platform and services From WP6: Cloud-Edge DevSecOps framework & client/gate-way-side cloud encryption solution VISuite AI (Ipsotek): SBRE engine, deep learning engine, third-party integration API
Work start	RDI starts January 2024, FID starts January 2027
Work end	RDI ends December 2026, FID ends June 2028
Outcome	What will be delivered: code, report
Final demo/evaluation	Report on the deployment of VISuite within the E2CC platform and results obtained on real use cases (detection performance, edge-to-cloud scalability, regulatory compliance). API providing access to metadata stream and events generated by VISuite within the E2CC platform. Functional demonstrator on a real camera network illustrating the edge-to-cloud capabilities of the platform.
Code access	Proprietary (VISuite is a commercial Ipsotek product: integration components developed within the project may be subject to an open-source licence)

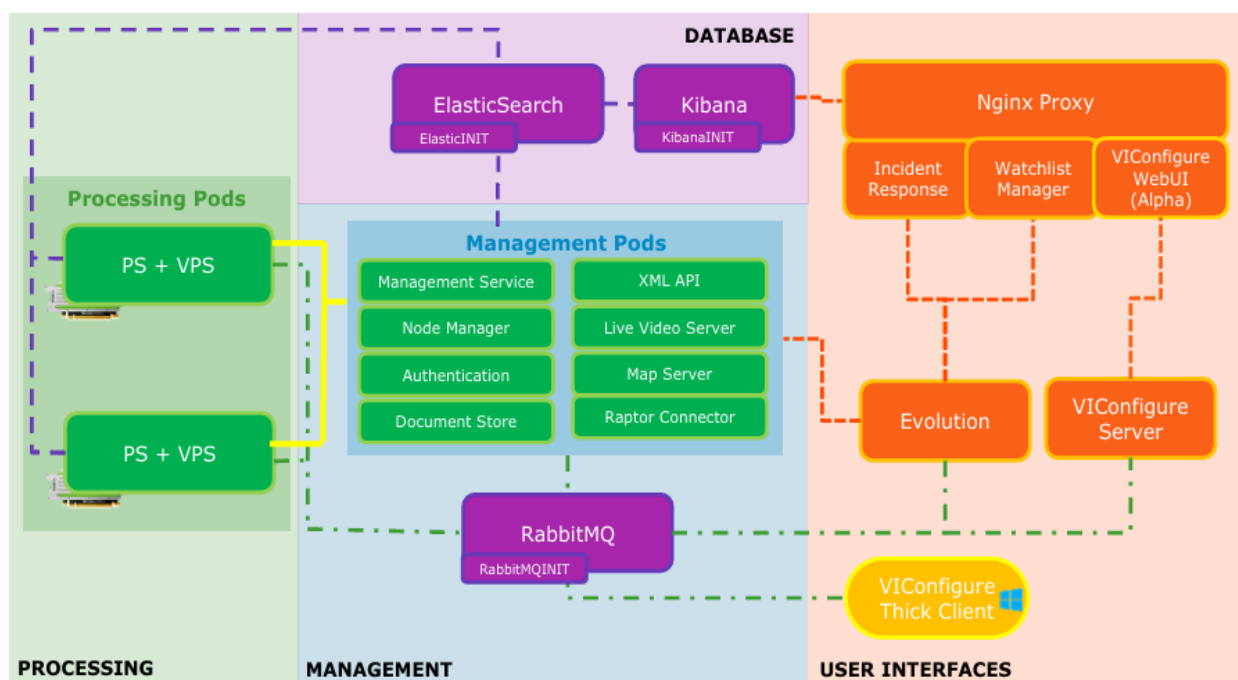


FIGURE 23. ARCHITECTURE SCHEMA FOR THE VISUITE PRODUCT

IoT & Water Network

The *IoT & Water Network* project is linked to the E2CC Edge-Cloud architecture and ensures the integration between Lacroix field equipment (Smart RTU, Smart Logger) and the E2CC platform components.

Component name	IoT & Water Network partnership (E2CC_IOTWN)
Architecture location	
Description	Connected E2CC platform on our platform Lx Connect (fleet management) and on our IoT products.
Planned work	- Platform Lx Connect deployment on E2CC platform through Kubernetes API - Implement MCP for a generic services component for AI Agent - Use Eviden PKI inside Lacroix IoT products - Use Identity & Access Management from E2CC to manage Lacroix users
Software dependencies	- E2CC orchestrator - PKI - IAM
Work start	January 2024
Work end	June 2028
Outcome	- Edge-Cloud integration software components (connectors, Kubernetes configurations, APIs) - Architecture and integration documentation (architecture diagrams, data flows, security and IAM description)
Final demo/evaluation	- Demonstration of an operational connection between an LX Connect Edge instance and the E2CC platform - Invocation of E2CC APIs (orchestration and internal services) - User identity and access management via E2CC IAM - End-to-end secured communication chain using PKI
Code access	Non-open-source code. Access restricted to project partners in accordance with E2CC and Lacroix intellectual property rules.

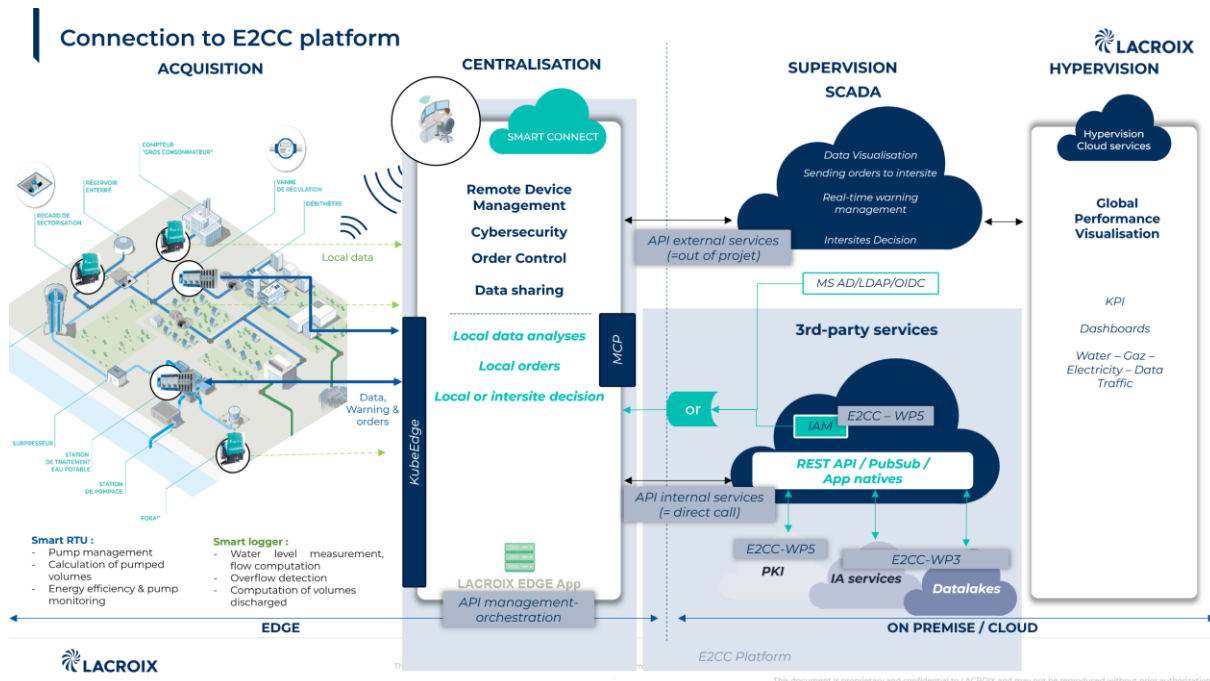


FIGURE 24: ARCHITECTURE SCHEMA FOR THE ECOSYSTEM OF LACROIX STACK LINKED TO E2CC PLATFORM.

Edge applications deployed on Lacroix Smart Connect platform handle data acquisition, local pre-processing, alarm management, and the dispatch of inter-site control commands. These Edge services are connected to the E2CC platform through REST APIs and publish/subscribe mechanisms, enabling data centralization, hypervision, and access to Cloud services such as artificial intelligence, data lakes, and performance dashboards.

The E2CC platform provides the core transversal capabilities of the system, including service orchestration (Kubernetes), identity and access management (IAM), secure communications through PKI, and the integration of advanced services (AI and multi-domain supervision). This architecture ensures scalability, security, and interoperability, which are required for the industrial deployment of IoT Water Network solutions.

AI for rare diseases

Component name	AI for rare disease partnership (E2CC-AI4RDP)
Architecture location	
Description	The AI for rare disease partnership (E2CC-AI4RDP) led by Institut Imagine is a use case project of the E2CC project. The research activities included in this project aim to improve diagnosis, prognosis, and treatment of diseases based on computational approaches. We are developing state-of-the-art solutions built on large-scale analysis of health big data coupled with artificial intelligence. Since the E2CC-AI4RDP project is a use case of the platform developed within the framework of the E2CC (Eco Edge to Cloud Continuum) project, the project is interdependent with the development of the E2CC program. In the context of WP7, we are developing the following components: 1) Development of a graph-neural network approach using multiplex biological knowledge graphs for the joint assessment of genetic variants within a genome. We are implementing a novel approach for the modelling of CNVs and SNVs as subgraphs in a multi-layer and multi-modal biological knowledge-based networks representing e.g. genomic interactions, gene regulation, co-expression, protein-protein interactions, etc. Notably, both coding and non-coding genomic regions are considered. Multi-modal clinical features are integrated as additional layers representing different types of associations: (i) between genomic elements and phenotypes, (ii) among phenotypes, and (iii) between phenotypes and diseases. Under such framework, assessment of pathogenic or benign CNVs and SNVs will be performed within a patient genome by modelling their combined effect as a subgraph classification task across the previous networks. Here, a supervised-learning approach based on Graph neural networks (GNNs) will be trained and tested on pathogenic and non-pathogenic variants from genetically characterized patients. The approach will provide pathogenicity scores for SNVs and CNVs allowing filtering and clinical prioritization. The bioinformatics pipeline and pathogenicity scores developed will be applied on specific disease cohorts, with a major emphasis on neurodevelopmental disorder (NDD) patients. 2) Development of uncertainty quantification and federated learning strategies for improved rare-disease diagnosis in a multi-site clinical setting. The first major objective of this use case is to explore uncertainty quantification techniques to guide alternative model selection for a given classification task, focusing on pathogenicity scoring of genetic variants in rare

	disease patients. This involves investigating the value of uncertainty estimates in identifying instances diverging from training sets, thus guiding alternative model selection. The second objective involves incorporating informative uncertainty estimates into federated learning approaches. Federated learning allows training and testing on protected datasets across multiple sites while respecting data privacy regulations. The goal is to adapt learning algorithms to consider uncertainty estimates from locally-trained models at each iterative aggregation in the global model as well as to derive client-specific models. The project's specific aims include evaluating different uncertainty estimation methods, such as conformal prediction and ensemble-based approaches, and integrating these into federated learning pipelines. 3) Distillation of single-cell foundation models. Single-cell foundation models are emerging as powerful tools for biological and biomedical research by enabling the integration and interpretation of large-scale multimodal single-cell datasets. Trained on millions of cells across diverse tissues and conditions, these models support applications such as cell type annotation, trajectory inference, perturbation prediction, and biomarker discovery. Due to their scale and complexity, efficient deployment requires adapted infrastructures supporting GPU acceleration, large-scale data processing, secure storage, and scalable orchestration. Such environments are essential to ensure performance, reproducibility, and integration into clinical and translational workflows. Deployed single-cell foundation models will also serve as teacher models in transfer learning and knowledge distillation frameworks, enabling the development of smaller and specialized models adapted to specific tissues, diseases, or clinical applications while reducing computational requirements.
Planned work	1) implementation of an end-to-end ML pipeline for single-cell omics data fully compatible with all E2CC building blocks; 2) implementation of a Federated Learning ML pipeline for clinical or omic data. 3) Development of a graph-neural network approach using multiplex biological knowledge graphs for the joint assessment of genetic variants within a genome.
Software dependencies	1) Weights&Biases, 2) Flower.
Work start	January 2026
Work end	June 2028
Outcome	1) papers describing the results, 2) git repositories.
Final demo/evaluation	Demo
Code access	Open-source code.

4. Estimation of the Schedule and Duration of Activities

Table 2 shows the schedule and duration of each E2CC component development.

	2024				2025				2026				2027				2028			
	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
E2CC Core Platform																				
OneBSM																				
KMS KMIP																				
Zero Touch Onboarding (ZTO)																				
Anomaly detection																				
Distributed tracing ...																				
E2CC Orchestrator																				
Container Orchestration																				
RYAX solution																				
E2CC Modular Platform																				
ALUMET																				
Mojito/S																				
Resource Management																				
Container Runtime																				
Confidential computing																				
AI MLOps																				
IAM solution																				
Cloud-edge applications																				
AI-analytics solution																				
IoT & Water Network																				
AI for rare diseases																				

TABLE 2: SOFTWARE COMPONENT ROADMAP

5. Summary

This deliverable D7.2 on architecture provides a detailed overview of the cloud-edge software stack that is developed in the E2CC project. The global architecture proposed is articulated around an orchestrator capable of deploying applications on a multi-provider cloud infrastructure plus edge. This orchestrator will propose a set of services to applications' developers focused on decarbonization, security and AI capabilities. To demonstrate the resulting capabilities of such a platform, different applications are being developed on this platform, each leveraging a subset of its services. The purpose of this deliverable is to show how all the components that are being developed within E2CC will be integrated and work together to provide the E2CC Platform. The description of the first implementation and integration of the components will be described in the next deliverable D7.3.

6. Acronyms

Acronym	Definition	Comment
AI	Artificial Intelligence	
ALUMET	Adaptive, Lightweight, Unified METrics	See page 27
API	Application Programming Interface	
BMaaS	Bare Metal as a Service	Model that allows to provide physical servers with cloud-like provisioning, automation, and API-driven management.
BMC	Baseboard Management Controller	A specialized microcontroller on a server motherboard that enables remote monitoring, management, and maintenance of the system, even when the server is powered off or unresponsive.
CPU	Central Processing Unit	
CSP	Cloud Service Provider	
CSV	Comma Separated Values	A text-based format
E2CC	Eco Edge to Cloud Continuum project	See https://e2cc.eu/
EUPL	European Union Public Licence	
GPU	Graphics Processing Unit	
GUI	Graphical User Interface	
HPC	High Performance Computing	
HSM	Hardware Security Module	
HW	Hardware	
IAM	Identity & Access Management	Framework of policies and technologies to ensure that users have the appropriate access to resources
IDaaS	Identity as a Service	
IoT	Internet of Things	
IPCEI-CIS	Important Project of Common European Interest, next generation Cloud Infrastructure and Services	French E2CC is one of the 19 projects that are parts of IPCEI-CIS
K8s	Kubernetes	Open source system for automating deployment, scaling, and management of containerized applications

KMIP	Key Management Interoperability Protocol	It defines message formats for the manipulation of cryptographic keys on a KMS
KMS	cryptographic Key Management Service, or Server, or System	
LLM	Large Language Model	Neural network trained on a large amount of text for natural language processing tasks
MLOps	Machine Learning Operations	paradigm that aims to deploy and maintain machine learning models in production reliably and efficiently.
NAS	Network-Attached Storage	
NLP	Natural Language Processing	
NVML	NVIDIA Management Library	
OIDC	OpenID Connect	Secured communication protocol
RAPL	Running Average Power Limit)	An Intel processor feature which allows real-time measurements of CPU and RAM.
REST	REpresentational State Transfer	REST APIs are sometimes referred to as RESTful APIs or RESTful web APIs.
SW	Software	
TEE	Trusted Execution Environment	A secure area within a device's main processor that ensures the confidentiality and integrity of code and data.
VPN	Virtual Private Network	
WP	Work Package	Groups of partners working on a same topic within E2CC
ZTO	Zero Touch Onboarding	secured digital identity for IoT

TABLE 3 - ACRONYMS