D2.3 ICOS ecosystem: Technologies, requirements and state of the art

Document Identification			
Status	Final	**Due Date**	30/04/2024
Version	1.0	**Submission Date**	11/06/2024

Related WP	WP2	Document Reference	D2.3
Related Deliverable(s)	N/A	Dissemination Level (*)	PU
Lead Participant	NCSRD	Lead Author	Georgios Xylouris (NCSRD)
Contributors	ATOS, NCSRD, PSNC, L-PIT, SUITE5, XLAB, ENG, UPC, SSEA ZETTA, RHT, BSC, CeADAR, SIXSQ, TUBS, NKUA, CRF, WSE	Reviewers	Carles Miralpeix i Llorach (FGC) Artur Jaworski (PCSS)

Keywords:

Cloud-Edge-IoT Continuum Management, Cognitive Cloud, Artificial Intelligence, Data Management over the continuum, Trustworthy

This document is issued within the frame and for the purpose of the ICOS project. This project has received funding from the European Union's Horizon Europe Framework Programme under Grant Agreement No. 101070177. The opinions expressed and arguments employed herein do not necessarily reflect the official views of the European Commission.
The dissemination of this document reflects only the author's view and the European Commission is not responsible for any use that may be made of the information it contains. This deliverable is subject to final acceptance by the European Commission.

(*) Dissemination level: **(PU)** Public, fully open, e.g., web (Deliverables flagged as public will be automatically published in CORDIS project's page). **(SEN)** Sensitive, limited under the conditions of the Grant Agreement. **(Classified EU-R)** EU RESTRICTED under the Commission Decision No2015/444. **(Classified EU-C)** EU CONFIDENTIAL under the Commission Decision No2015/444. **(Classified EU-S)** EU SECRET under the Commission Decision No2015/444.

Document Information

List of Contributors	
Name	Partner
Alex Volkov	ATOS
Francesco D'Andria	ATOS
Marc Fabián	ATOS
Alex Barceló	BSC
Anna Queralt	BSC
Francesc Lordan	BSC
Andrés L. Suárez-Cetrulo	CeADAR
Jaydeep Samanta	CeADAR
Ricardo Simon Carbajo	CeADAR
Sebastián Cajas Ordóñez	CeADAR
Marina Giordanino	CRF
Gabriele Giammatteo	ENG
Marie Claire Ciampini	FBA
Kalman Meth	IBM
Iman Esfandiyar	L-PIT
Marta Łakomiak	L-PIT
George Xilouris	NCSRD
Nikos Dimitriou	NCSRD
Anastasios Giannopoulos	NKUA
Konstantinos Skianis	NKUA
Panagiotis Gkonis	NKUA
Panagiotis Trakadas	NKUA
Artur Jaworski	PSNC
Dariusz Dziel	PSNC
Marcin Kotliński	PSNC
Jose Castillo Lema	RHT
John White	SIXSQ
Khaled Basbous	SIXSQ
Konstantin Skaburskas	SIXSQ
Lionel Schaub	SIXSQ
Nabil Abdennadher	SIXSQ
Rosalia Davi	SSEA
Konstantinos Latanis	SUITE5
Admela Jukan	TUBS
Fin Gentzen	TUBS
Jasenka Dizdarevic	TUBS
Marc Michalke	TUBS

List of Contributors	
Name	Partner
Jordi Garcia	UPC
Montserrat Farreras Esclusa	UPC
Xavi Masip-Bruin	UPC
Ignasi Garcia-Milà Vidal	WSE
Izabela Zrazinska	WSE
Aleš Černivec	XLAB
Daniel Nikoloski	XLAB
Hrvoje Ratkajec	XLAB
Ivan Paez	ZSCALE

Document History			
Version	Date	Change editors	Changes
0.1	30/04/2024	George Xylouris (NCSRD)	First TOC
0.2	20/05/2024	George Xylouris (NCSRD)	First round of contributions – edits
0.3	06/06/2024	George Xylouris (NCSRD)	Last round of contributions - edits
0.4	11/06/2024	Carmen San Román (ATOS)	Quality assurance check
1.0	11/06/2024	Francesco D’Andria (ATOS)	Final version to be submitted

Quality Control		
Role	Who (Partner short name)	Approval Date
Deliverable leader	George Xylouris (NCSRD)	10/06/2024
Quality manager	Carmen San Román (ATOS)	11/06/2024
Project Coordinator	Francesco D’Andria (ATOS)	11/06/2024

Table of Contents

List of Tables

List of Figures

List of Acronyms

Abbreviation / acronym	Description
AFO	Adaptive Federated Optimization
AGIX	SingularityNET
AI	Artificial Intelligence
AIaaS	Artificial Intelligence as a Service
AODV	On-Demand Distance Vector Routing Protocol
AORP	Agriculture Operational Robotic Platform
API	Application Programming Interface
ARM	Advanced RISC Machine
AuthN	Authentication
AuthZ	Authorization
AWS	Amazon Web Services
BATMAN-adv	Better Approach to Mobile Ad-hoc Networks – advanced
BSD	Berkeley Software Distribution
CA	Certification Authority
CaaS	Container as a Service
CAN	Controller Area Network bus
CC	Cloud Continuum
CFG	Control Flow Graph
CI/CD	Continuous Integration / Continuous Deployment
CIDR	Classless Inter-Domain Routing
CLI	Command Line Interface
CO2	Carbon Dioxide
CP	Cloud Provider
CPU	Central Processing Unit
CRD	Custom Resource Definition
DAG	Directed Acyclic Graph
DC	Data Center
DDS	Data Distribution Service
DNN	Deep Neural Network
DNS	Domain Name System
DoA	Description of Action
DSO	Distribution Supplier Officer
DSR	Dynamic Source Routing protocol

Abbreviation / acronym	Description
DX.Y	Deliverable X.Y
ECIP	Edge Computing Infrastructure Provider
ECS	Amazon Elastic Container Service
ECU	Engine Control Unit
ELSA	Edge Lightweight Searchable Attribute-based encryption system
EMCS	Energy-Efficient Makespan Cost-Aware Scheduling
EMDS	Energy Management and Decision Support system
ESBN	Electricity Supply Board Networks
EU	European Union
EV	Electric Vehicle
FaaS	Function as a Service
FCOS	Fedora CoreOS
FedAvg	Federated Averaging algorithm
FedEL	Federated Evolutionary Learning
FedGNN	Federated Graph Neural Network
FedPer	Federated Learning with Personalization Layers
FedProx	Federated Optimization in Heterogeneous Networks
FedSage+	Subgraph Federated Learning with Missing Neighbor Generation
FedTL	Federated Transfer Learning
FGC	Ferrocarrils Generalitat Catalunya
FL	Federated learning
GB	Giga Byte
GDPR	General Data Protection Regulation
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
GPT-2	Generative Pre-trained Transformer 2
GPU	Graphical Processing Unit
GUI	Graphical User Interface
GW	Gateway
HPC	High Performance Computing
HPE	Hewlett Packard Enterprise
HSV	Hue Saturation Value
HTTP	Hypertext Transfer Protocol
HW	Hardware
IaaS	Infrastructure as a Service
IaC	Infrastructure as a Code

Abbreviation / acronym	Description
IAIMM	In-car Advanced Infotainment and Multimedia Management system
ICOS	Towards a functional continuum operating system
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers
IL	Intelligence Layer
ILP	Integer Linear Programming
IMU	Inertial Measurement Unit
IOS	iPhone Operating System
IoT	Internet of Things
IoTP	Internet of Things Provider
IP	Internet Protocol
IT	Information Technology
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
LAN	Local Area Network
LoRa	Long Range
LSABE-MA	Lightweight Searchable ABE with multi-authority
MADRL	Multi-Agent Deep Reinforcement Learning
MCC	Mobile Cloud Computing
MFCC	Mel-Frequency Cepstral Coefficients
ML	Machine Learning
MLMD	Machine Learning Metadata
MLOps	Machine Learning Operation
MNIST	Modified National Institute of Standards and Technology database
MQTT	MQ Telemetry Transport
MW	Mega Watt
NBI	North Bound Interface
NDN	Named Data Networking
NFV	Network Function Virtualization
NLP	Natural Language Processing
NN	Neural Network
NNI	Neural Network Intelligence
NP	Network Provider
NP-hard	Nondeterministic Polynomial time
OBS	One Big Switch
OCI	Open Container Initiative

Abbreviation / acronym	Description
OCM	Open Cluster Management
OLM	Open Learner Models
OLSR	Optimized Link State Routing protocol
OS	Operating System
OSI	Open Systems Interconnection model
OT	Operation Technology
OTE	Optimal Trustworthy EdgeAI
P2P	Peer to Peer
PDP	Programmable Data Plane
PLONK	PLONK is a cloud native stack for application developers: Prometheus; Linux; OpenFaaS; NATS; Kubernetes
PoI	Point of Interest
PoP	Point of Presence
PV	PhotoVoltaics
QoE	Quality of Experience
QoS	Quality of Service
RAM	Random Access Memory
RDD	Resilient Distributed Dataset
REST	Representational State Transfer
RGB	Red Green Blue
RL	Reinforcement Learning
ROS2	Robot Operating System 2
RSAM	Railway Structural Alert Monitoring system
RTK	Real-Time Kinematic positioning
SBI	South Bound Interface
SDK	Software Development Kit
SDN	Software Defined Networking
SFC	Service Function Chaining
SFV	Sensor Function Virtualization-based
SGX	Intel Software Guard Extensions
SLA	Service Level Agreement
SLO	Service Level Objective
SNR	Signal To Noise Ratio
SplitFL	Split Federated Learning
SQL	Structured Query Language
SSOT	Single Source of Truth
StaSA	Service–Time-Aware Scheduling Algorithm

Abbreviation / acronym	Description
SW	Software
TEE	Trusted Execution Environment
TPU	Tensor Processing Unit
TSO	Transmission Supplier Officer
UC.X	Use Case X
UI	User Interface
URL	Uniform Resource Locator
USB	Universal Serial Bus
US.X	User Story X
VM	Virtual Machine
VPN	Virtual Private Network
WPX	Work Package X
XR	Extended Reality
YAML	YAML Ain't Markup Language

Executive Summary

The ICOS project aims to cover the range of challenges that arise when addressing the Cloud-Edge-IoT continuum paradigm, proposing an approach that embeds a well-defined set of functionalities, culminating in the definition of an IoT2cloud operating system. ICOS’s mission is to design, develop and validate a meta operating system for the Cloud-Edge-IoT continuum.

The deliverable "The ICOS ecosystem - Technologies, requirements and state of the art" presents a detailed exploration of the Intelligent Cloud-Oriented Services (ICOS) ecosystem, addressing its underlying technologies, requirements, and current advancements. This document is meticulously structured to guide stakeholders through the various facets of ICOS, from its fundamental concepts to practical applications and state-of-the-art solutions.

Beginning with an introduction that outlines the purpose, intended audience, and structure, the document progresses to updating key ICOS concept definitions. It then describes the ICOS ecosystem comprehensively. Following this, detailed user stories illustrate various scenarios of ICOS implementation, such as cloud continuum realization, IoT device onboarding, application deployment, dynamic resource reconfiguration, data access and processing, and security monitoring.

The document continues with use case scenarios that tackle specific ICOS challenges across domains like agriculture, railway monitoring, in-car infotainment systems, and energy management. Each use case is elaborated through summaries, infrastructure details, architectures, user stories, requirements, and key performance indicators (KPIs). Additionally, the outcomes of the first ICOS Open Call are presented, showcasing innovative projects that leverage ICOS technologies for diverse applications.

Updates on ICOS requirements are provided, differentiating between functional and non-functional aspects to ensure a comprehensive understanding of what the system necessitates for successful deployment. A brief review of MetaOS architectures situates ICOS within the broader context of cloud and edge orchestration systems. The document also discusses the selection and justification of technologies used in ICOS, covering frameworks, orchestration, network services, data processing, security, and trust. The conclusions summarize the main discussions and outline future directions for the ICOS ecosystem, making this document an invaluable resource for researchers, developers, and stakeholders involved in intelligent cloud-oriented services.

1 Introduction

This document updates the information of D2.1 deliverable [1]. Citing the original challenges for ICOS from D2.1 [1], ICOS set forward the following challenges for the design and specification of ICOS platform:

- Challenge 1: Enabling technology agnostic operation in a heterogeneous continuum infrastructure.
- Challenge 2: Facilitating an on-demand ad-hoc and AI-assisted development of the continuum infrastructure.
- Challenge 3: Considering reliability and trust-by-design.
- Challenge 4: Creating an open platform facilitating resources, models, data and services sharing, promoting EU innovation and new business models in the continuum arena.

The ICOS motivation, is to provide a MetaOS operating system that will facilitate the orchestration and operation capability of multiple systems i.e., Cloud, Edge and IoT towards the realisation of a Cloud Continuum capable of optimised and secure execution of distributed applications.

The document updates the definitions, the ICOS ecosystem and its stakeholders, and refines User Stories, Use Cases and elicited requirements as described in D2.1 [1]. In detail, it refines the ICOS User Stories that provide high level descriptions of the main functionalities and operations ICOS supports. The changes are based on the feedback received during implementation of the ICOS components included in the first release of the ICOS software suite.

Furthermore, it refines the Use Cases which are used for the evaluation of ICOS (i.e., UC1: Agriculture Operational Robotic Platform; UC2: Railway Structural Alert Monitoring system; UC3: In-car Advanced Infotainment and Multimedia Management system; UC4: Energy Management and Decision Support system). In addition, D2.3 presents the accepted projects awarded from the Open Call. Requirements are elicited from both Use Cases and Open Call Projects. Specific constraints are detailed, providing associated requirements along with measurable non-functional requirement key performance indicators.

ICOS Requirements Elicitation is the result of analysis and a series of activities carried out within the different actors and stakeholders of the project's value chain. To provide a set of requirements, User Stories and ICOS Use Cases were studied and the results from mentioned analysis are detailed in a set of tables. Similarly to D2.1[1] the MoSCoW method was followed to reduce the requirements elicitation extension. The document introduces updated view on the non-functional requirements with a direction of refining the non-functional requirements and the definition of measurable KPIs for each one. The initial list of requirements from D2.1 [1] was updated with the focus to deliver information on the ones that have already being satisfied/measured and the means to validate them.

Finally, the original state-of-the-art survey that is available in D2.1 [1], has been replaced by a section that provides information about the technology selection used for implementation and justification for each decision. A summary of the research in MetaOSes and the Cloud Continuum is also provided.

1.1 Purpose of the document

This deliverable presents the outcomes of the work carried out by the activities T2.1 – Ecosystem identification: Baseline technologies; T2.2 – Compute continuum requirements definition and T2.3 – AI, data management and trust/security requirements.

This document’s purpose is twofold, on one hand to discuss the Use Cases and User Stories that will define the ICOS development in terms of requirement elicitation and on the other hand to survey the technologies and frameworks that are relevant for the implementation of ICOS and satisfaction of the elicited requirements. The first version of the deliverable (D2.1 [1]) was aligned to iteration 1 (IT-1) of the project and was submitted at M6. The current deliverable (D2.3) is the second version (submitted at M20), and is aligned to iteration 2 (IT-2).

1.2 Indented Audience

This document is intended for a diverse group of stakeholders involved in the ICOS ecosystem. Primarily, it targets researchers and developers working on cloud, edge, and IoT technologies who need an in-depth understanding of the ICOS framework, its requirements, and current advancements. Additionally, it is aimed at project managers and decision-makers in organizations implementing or considering the adoption of ICOS solutions, providing them with the necessary technical and strategic insights.

Furthermore, this document serves as a resource for academics and students in the fields of computer science, information technology, and engineering, offering them detailed case studies and user stories that illustrate real-world applications of ICOS. Finally, it is also intended for industry professionals and technical consultants who are involved in the design, deployment, and optimization of cloud and edge computing systems, giving them a comprehensive guide to the state-of-the-art technologies and best practices within the ICOS ecosystem.

1.3 Structure of the document

The document provides a comprehensive overview of the ICOS ecosystem, detailing its technologies, requirements, and current advancements. deliverable is structured. The introductory section outlines the document's purpose, audience, and structure. Subsequent sections discuss the motivation and challenges of ICOS, update definitions of key concepts, and describe the ICOS ecosystem. Detailed user stories illustrate various implementation scenarios, followed by use case scenarios that address specific challenges. The document also updates ICOS requirements, both functional and non-functional, reviews MetaOS architectures, and justifies the technology selections made for ICOS.

The latter sections of the document delve into the specific use case scenarios, providing summaries, infrastructure details, architectures, user stories, requirements, and KPIs for each case. The focus then shifts to updates on ICOS requirements, distinguishing between functional and non-functional needs. A brief review of MetaOS architectures is included, followed by a detailed discussion on the technology selection process and its justification, covering areas such as application description frameworks, orchestration and management, network connection services, data processing and management, and security and trust. The document concludes by summarizing the key discussions and presenting final thoughts and future directions for the ICOS ecosystem.

2 ICOS Concepts definitions updates

The project has created a glossary where terms used in all deliverables are explained. The glossary is included in Section 3 of Deliverable D2.1 [1] and has been updated in D2.2 [2] as part of the explorations related to the ICOS architecture. This section serves as an addendum for new or refined definitions.

Table 1: Updated Definitions on ICOS concepts and artifacts

Domain
An infrastructure managed and controlled under the provisions of a single administration entity. Maybe further detailed depending on the complexity of the infrastructure topology, the underlying technologies and the NBI provided for third party management/orchestration.
Cloud Continuum (CC)
A group of resources (CPU, memory, IoT devices, network, storage, intelligence) managed seamlessly end-to-end that may span across different administrative/technology domains in multi-operator and multi-tenant settings.
ICOS Agent
The basic ICOS software component that needs to be installed on each node (infrastructure node) capable of executing external software and/or providing data and metadata.
ICOS Controller
The ICOS component/software that is responsible for peering with ICOS Agents for providing control, intelligence and lifecycle operations.
ICOS Element
Is the unitary resource available on a node (e.g., Computing, Storage, Network, Intelligence etc.). The basic, unitary ICOS Element is defined as the minimum set of resources defined as CPU/RAM/Storage (dynamically allocated by the native infrastructure management administration) encapsulated within HW or SW entities such as (VM, container or HW node).
ICOS Node
Any resource physical or virtual that is running either an ICOS Controller or Agent.
Restricted Capability Devices
IoT device or infrastructure element (that can be resource-restricted) controlled through API provided by ICOS Agent on ICOS Nodes. The IoT device or infrastructure element could also be directly accessed via its control interface (e.g., API) i.e., not necessarily through the ICOS Node.
ICOS Instance
A subset of the CC participating in the execution of a multi-component Application of a certain topology, resource requirements and constraints."
ICOS Shell
A client distribution that is used to access the ICOS. The ICOS Shell can run externally to the ICOS Instance.
ICOS Application Descriptor
It is provided by ICOS Shell. It is the ICOS Application descriptor containing metadata and allows the definition of application components and deployment within the Cloud Continuum

2.1 ICOS Ecosystem

The Deliverable D2.1 [1] presented the ICOS ecosystem as illustrated in Figure 1. There is no significant update in our view and the description is provided for making the document more complete.

The roles in the ICOS ecosystem are i) the End User; ii) the Application Developer; iii) the Application Integrator and iv) the infrastructure providers. No significant changes have been made on the definition of ICOS ecosystem beyond those laid out in D2.1 [1]. In order to avoid duplication of Role descriptions, reader is referred to the D2.1 document.

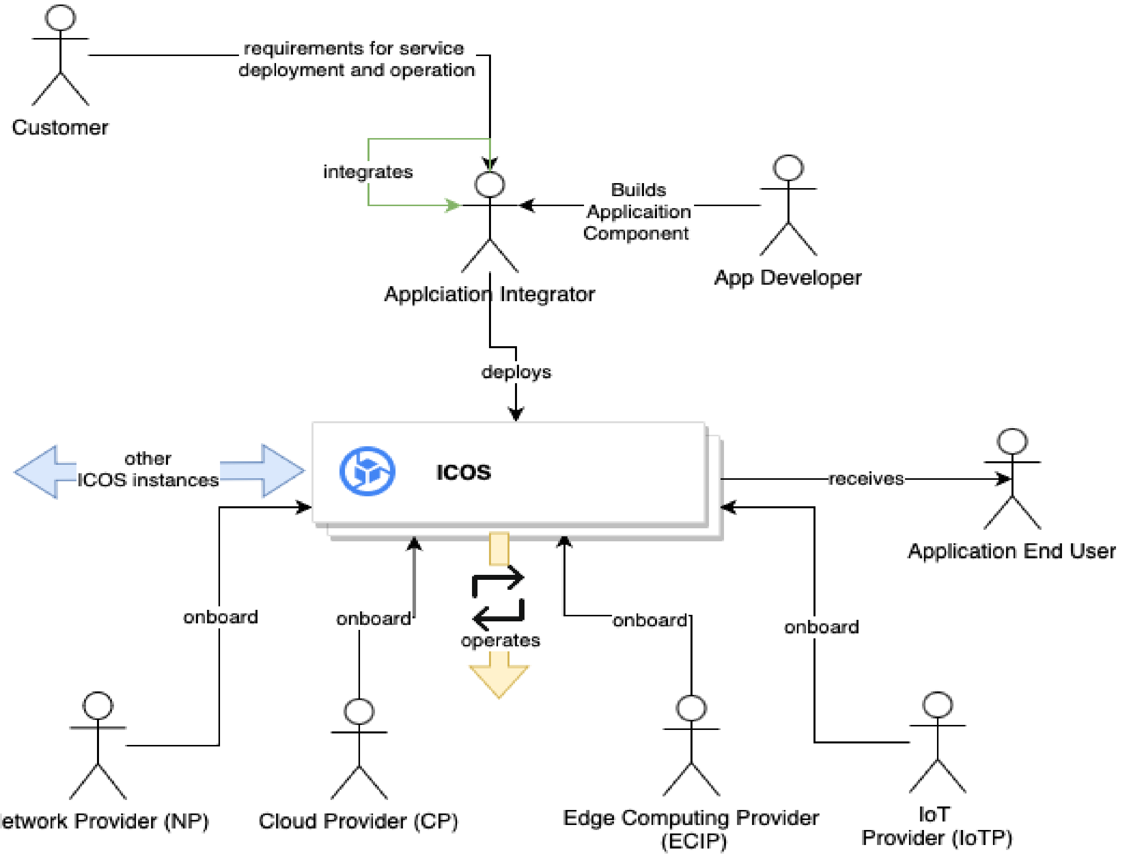

Figure 1: ICOS roles and interactions

3 ICOS User Stories

This section refines several basic ICOS system user stories that will allow better understanding of the ICOS operation as well as elicit functional requirements that will be discussed in the following sections. The User Stories presented in D2.1 [1], provide a high-level overview of the main interactions and workflows anticipated in ICOS. The section is further refined this deliverable, considering development and implementation work that has been made.

3.1 User Story 1 [US.1]: Cloud Continuum Realization

This User Story explains the method by which the ICOS-managed and controlled Cloud Continuum (CC) is realized. According to the definitions of Section 3, whenever an ICOS Node is bootstrapped, that node immediately becomes part of the ICOS CC. The requirement for an infrastructure or infrastructure node to become part of the CC (i.e., ICOS Node), is provided either by exploiting the ability to deploy and execute ICOS (Controller or Agent) or exploit available APIs that provide baseline functionality within the CC. In this context, ICOS provides two operation modes (based on the intent, extent, and capabilities of the infrastructure node), the ICOS Controller and the ICOS Agent mode, which depending on the scenario can be used interchangeably. ICOS, through an ICOS Controller, is able to monitor and maintain the topology of the created CC along with the metadata generated by the ICOS Agents that are on-boarded on the said controller. Multiple ICOS Controllers communicate using east-west interfaces to exchange local views and information.

The process for the realization of the CC can be broken down to the following scenarios:

1. ICOS Node on-boarding

- Onboarding of ICOS Controller
  - The owner of the computing resources installs the ICOS controller distribution.
  - The ICOS Controller is registered to the lighthouse.
  - The ICOS controller advertises the policy of which agents may register.
- Assignment of ICOS Agent to a controller
  - Either by manual configuration (active infrastructure deployment) or through a discovery mechanism (passive infrastructure deployment) the ICOS Agent is assigned to an already running ICOS Controller.
- Onboarding of ICOS Element
  - An ICOS Element (e.g., Edge device) is joined to an existing ICOS Agent.
  - The owner of the ICOS Element must possess credentials for the ICOS Agent and the local Orchestrator and provide a profile of the optional ICOS services to be run on the Element.
  - A container orchestration engine (COE) and an Orchestrator of the COE are assumed to be installed on the ICOS Element. These are onboarded as part of the process.
  - The Element owner uses the ICOS Shell command to launch the process.
  - This process: deploys and commissions the local Orchestrator Agent; joins to the ICOS Agent and publishes the profile.
- The Agent then pushes a job to the controller that will trigger the normal ICOS job deployment procedure that will deploy the optional services to the ICOS Element. Onboarding of ICOS Agent
  - The owner of an ICOS agent contacts the lighthouse to find a suitable controller or contacts a controller owner to arrange permission to join.
  - The Agent onboarding requires that the Agent owner possesses credentials from the ICOS Controller and local Orchestrator.
  - The Agent owner then uses the ICOS Shell to launch the onboarding process.

- This process deploys the ICOS Agent software; uses the gathered credentials to commission itself to the Controller; connect to the relevant services on the Controller (telemetry, Job Manager etc) and connect to the local Orchestrator. ICOS Node removal
- Whenever an ICOS node departs from the CC, the credentials of the node are revoked from the ICOS Identity Access manager. Then the ICOS software can be uninstalled.
- In case of communication failure or any other deliberate failure, the node is only removed after a time threshold.

- ICOS Controller communication
- Each ICOS controller maintains information on capabilities, assets, resources, context concerning the subset of the ICOS nodes associated with it. This information is important for executing parts of a containerized application according to the required KPIs / Service Level objectives.
- ICOS controllers support east-west interfaces to interconnect and expand the CC across infrastructures and domains. Two modes are supported
    - Manual configuration of peers, where known ICOS controllers are configured to talk to each other.
    - Automatic configuration where the ICOS Controllers employ discovery mechanisms in order to discover peers and interconnect with them.
- Similarly, to the ICOS nodes failure/deletion, ICOS controllers may spawn and be removed during operations.
- Peered ICOS controllers comprise the total of the CC by exchanging information and allowing Application deployment over infrastructure spanning a multitude of ICOS controllers and domains.

Related Roles: NP, CP, ECPP, IoTP

3.2 User Story 2 [US.2]: IoT GW/Node or device on-boarding

IoT device onboarding supports the deployment and management of IoT network gateways or devices to be instantiated as ICOS Nodes within the ICOS continuum.

IoT providers should provide at least one IoT network element to deploy the ICOS agent, so that it can be integrated in ICOS. The on-boarding of IoT infrastructure nodes in CC, in general follows the pattern described in US.1. Here we provide some specific provisions for IoT devices. The following scenarios can be observed:

1. IoT Gateway onboarding: The IoT gateway is the entry point to IoT devices from the ICOS perspective. The gateway should always run the ICOS agent, to ensure the management of resources and devices in an integrated and seamless way. The onboarding process should indicate how the gateway needs to be integrated with the other components provided by ICOS: data storage, processing nodes, etc... During the gateway onboarding process, ICOS needs to receive a response from the gateway regarding the IoT configuration options exposed to the system.
2. IoT Node onboarding: The node is an IoT device which will be collecting data and sending such data through an IoT network. In the currently rare occasion, the IoT node is not restricted and ICOS Agent can be deployed, a similar deployment process with the IoT GW will be followed. However, in most of the cases an IoT node may not be able to run the ICOS agent and also not offer a public API in order to interface as a restricted device, and therefore it might not be available to the continuum (but we might consider that IoT nodes are “dumb” data providers) In this case the availability of IoT node is performed through the IoT GW which can deploy an ICOS Agent. The onboarding of a node process should indicate if there are any configuration options available to ICOS, and through which methods are usable.

Related Roles: ECPP, IoTP

3.3 User Story 3 [US.3]: New Edge/Aggr/Core Resource On-boarding

As explained in User Story 1 that covers the CC realization, the infrastructure node on-boarding is also applicable here. However, this user story provides some specific details in particular for cloud (edge/aggr/core) on-boarding.

Infrastructure and resources on-boarding, including Computation, Storage and Network across Cloud, Edge (near and far edges) and IoT initially will have the following procedure:

In order to on-board an ICOS user that does not bring any infrastructure, ICOS must provide the necessary resources to meet the user's requirements. Infrastructure and resources on-boarding, including Computation, Storage and Network across the ICOS Ecosystem must follow specific criteria and be carefully selected depending on whether the related resources are assigned by ICOS for shared usage or exclusively for the specific user. ICOS will be able to discover available resources and attempt matchmaking. If the attempt fails, ICOS has the ability to spawn the resources needed on behalf of the user to ensure the alignment within the SLA. When these resources are shared among multiple users, ICOS will offer a quota in order to limit the usage of mentioned resources.

Matchmaking criteria includes features such as proximity, availability, and applicability of the resources, as well as the latency, bandwidth or network components availability.

ICOS will offer a very specific and dynamic quota to optimize resource usage and operation cost on one hand, and, on the other hand, to ensure that the quota is always respected, being rigorously checked by telemetry services.

Challenges:

- Aggregation of resources and infrastructure is possible at runtime, ICOS should be able to define a topology of resources and be able to offer available / close (GPS proximity) resources to be aggregated. For example, a map with existing edges in Barcelona, when a new edge is added to ICOS, should appear on it as well.
- Usage of shared resources brings the need of accessibility provision and control. Whether the shared resources are owned by ICOS or not they must be provisioned within authority control or be integrated with given authorization and be exploited within all means of privacy and security for all involved users.

Related Roles: NP, CP, ECPP

3.4 User Story 4 [US.4]: Application Deployment

This user story describes the application deployment workflow. Two scenarios are discussed, i) the baseline case - where the ICOS Controller may satisfy the application's resource requirements; and ii) the general case - where the ICOS Controller needs to coordinate in order to satisfy the application requirements.

Scenario 1 - Resource availability on a single ICOS Controller

1. An application request arrives to the ICOS Controller.
   - The application request includes an application descriptor that contains all the deployment requirements for the application components, such as required computation, communication, data sources, etc.
2. The ICOS Controller processes the application descriptor, obtains the available resources, and provides the best matching for the request (one ICOS element per application component).
3. The ICOS Controller notifies the ICOS Agents for the provision of resources.
   - More than one Agent could be involved in the best matching selection.
4. The ICOS Controller deploys the application components on each location, interacting with each Agent. Details of each deployment are specific to the application and technology domain at infrastructure level and are intentionally omitted.

5. By the end of the previous step, the application is considered deployed and active, and all data sources and components are interconnected. The set of ICOS elements involved in the execution of this application becomes a running ICOS instance.
6. During operation, the ICOS Controller keeps track of the health of the execution.
   - Fault tolerance, policies monitoring and migration/relocation loops are active (not detailed here) during the application operation.
7. Telemetry generated for the running application is collected and maintained by the Controller through the Agents.

Scenario 2 - Resources availability via Controller coordination

In this scenario, the application requested resources are not available within the vicinity of a single ICOS Controller. As such, coordination amongst Controllers is activated in order to match the requested resources. It should be noted that the ICOS Controllers should be distributed and dynamically organized. Each controller manages a subset of the total amount of ICOS elements (i.e., infrastructure, according to the organizational structure) but can interact with other Controllers to combine the resources from both groups.

The main difference in this scenario is at the initial steps where the resource request needs to be referred to other Controllers prior to matching decisions.

The complex service deployment user story is the following:

1. An ICOS Controller receives a request to execute an ICOS Application.
2. The Controller analyses the app requirements, obtains the available resources, and realizes only a subset of the application components can be successfully executed in the current context.
   - The reasons for this could be diverse. For instance, some datasets are not available in that context, or there are not enough resources to execute the application according to the agreed SLA.
3. The Controller contacts other Controllers (broader scope, according to the ICOS Control structure) and requests if the remaining application components can be successfully executed.
   - If not, iterate on this step until all components have been mapped.
4. After the application requests have been mapped (across different Controllers) the process of signalling the ICOS Agents is performed via each corresponding ICOS Controller.
   - If after requesting all the Controllers, all application components have not been mapped, this means the application cannot be executed in the current ICOS configuration, so the execution request returns with the corresponding error.
5. The rest of the scenario follows the flow of Scenario 1.

Related Roles: AppInt, AppDev, CCP, Customer

3.5 User Story 5 [US.5]: Dynamic Re-configuration / Optimisation of Application Resources

Dynamic reconfiguration of application resources should be based on appropriate AI/ML models that are trained with network telemetry data in one or more ICOS instances. With respect to the previous version of this user story, it is now updated to support Federated Learning among the involved ICOS instances. The basic steps are summarized below:

- An ICOS user initiates the process for data collection that are related to a particular application.
- Collection Module: The network telemetry module then gathers information from all involved ICOS nodes of the ICOS instance where the application is deployed.
- If another instance of this application is deployed over a separate ICOS instance, then additional training data are collected.

- Data cleaning and pre-processing: Collected monitoring data is pre-processed prior to the ML-training, so as to ensure proper data format and low level of noise in the training dataset.
- Model Training: Using a large part of the collected monitoring data, several local ICOS node-specific models are trained. During the training phase, several model hyperparameter (model deepness, model learning rate) configuration is considered and validated, to finally find the optimal model configuration. To support federation across local models (we assume that each one is deployed over a separate ICOS instance), a periodic aggregation of multiple local parameters is performed to build a global observability model. Maximization of model performance is based on an objective function, directly related to the application optimization. Since the model supports federated learning, targets including latency reduction and/or security and privacy protection are achieved. Further, training might take place across multiple ICOS instances.
- Model Validation: To quantify the efficiency of the trained model(s), a performance metric is calculated over collected samples, not encountered during the model training phase.
- Model Deployment: The validated model is deployed in every decentralized location that comprises the ICOS involved ICOS instances, optionally after model packetizing operations, such as dockerization.
- Model Inference: After model deployment, telemetry data is made available to the model(s), so as to provide corrective actions, reconfiguration activities, predictions, alarms towards achieving the application optimization objective.
- Model Operation: The output of the ML models is exploited to configure Applications and/or resources over all ICOS instances where the application is deployed

Related Roles: NP, CP, ECPP, IoTP, CCP

3.6 User Story 6 [US.6]: Data Access and Processing

Data access and processing refers to the ability of the ICOS system to provide access to the data generated within an ICOS instance and process it. This can be either application data produced within a deployed application, or data automatically collected by the ICOS meta-OS that will be later used for model training in ML-aided ICOS components for network reconfiguration and resource optimization, as described in the previous user story.

For easing the processing of data, ICOS provides two main tools: i) the Intelligence Layer, which supports data processing as an internal module of this and allows training new artificial intelligence models and use them for inference and provide the MetaOS with inputs for decision making, and ii) the Distributed and Parallel Execution, which supports the parallelization and distribution of general-purpose data processing applications across the Continuum.

Scenario 1 - Artificial Intelligence Model training

This first scenario aims to support the training of artificial intelligence models and is expected to be used for training models based on application-level data or data automatically collected by the ICOS system regarding the application's performance or the infrastructure. This user story assumes that all the data is accessible by the Controller either because the data is collected and aggregated by the monitoring infrastructure or because the data is being stored and shared using the Data Management Layer.

In this scenario, the user story is started by an ICOS user interaction with the ICOS system using the ICOS Shell (either through the CLI or the GUI).

1. The user authenticates into the ICOS system using the Shell.
2. The Shell interacts with the Lighthouse to assign an ICOS Controller to interact with.
3. The Shell, on behalf of the user, requests the Controller to train the model, indicating the algorithm and the data to use.

4. The Controller verifies that the user is authorized to run that training and use the data.
5. The Controller forwards the request to the Intelligence layer.
6. The Intelligence Coordination module receives the request and forwards this and the data received to the data preprocessing module, which will prepare the data for model training. The next step will be to send this to the AI Analytics module. However, the Intelligence Layer may offload the task through the data management layer if there is an available node in the topology that can be used to train the model. In that case, this step will be replaced with the following.
    a. The data processing module in the Intelligence Layer will initiate a dataClay client to transfer the training workload.
    b. Data preprocessing will check if data is already present at the dataClay server. Otherwise, this will be transferred from the Controller.
    c. Data will then be prepared (resampled at the right interval for the model, standardised, split, etc.).
7. The AI Analytics module, part of the Intelligence Layer, will start training the model. If this task has been offloaded with data management, the resulting model, model artifacts (e.g., standardisers), and its results in the training set are transferred back to the Controller. Once at the Controller, or if training was not offloaded, models and their artifacts are pushed to the model registry for later inferencing.
8. After pushing a model to the registry for future usage, if the AIOps platform is enabled, it will be linked to the AIOps platform to allow model explainability and experiment tracking.
9. Finally, since the model registry offers version control, but this keeps increasing the footprint of the Intelligence Layer, the model registry will be pruned if the number of versions of a model with the same name exceeds a certain threshold (5 model versions by default).

Scenario 2 - Artificial Intelligence inference

In the second scenario, the trigger of the user story could be either an ICOS user or the ICOS system itself, which requires using a previously computed model to infer some knowledge using specific data. If the request comes from an ICOS user, it uses the Shell to submit the request and undergoes the same authentication process presented in the previous scenario; if the story is triggered by a component from the ICOS architecture, this authentication will happen during the deployment of the component.

1. After the authorization of the inference operation, the corresponding ICOS component -- in the case of being requested by a user, the Shell -- contacts the corresponding Controller indicating which model to use and the input data for the model (e.g., an image or a features vector).
2. The Controller forwards the invocation to the Intelligence layer that queries a model from the model registry matching the model name and version specified using the data received as part of the request.
3. The Intelligence Coordination module receives the requests and forwards the model name and the data to the pre-processing module. This module scales and processes the data if required using the model’s artifacts. The data is then sent to the AI Analytics module together with the request.
4. The AI Analytics module queries the model specified from the model registry with the data received from pre-processing and returns the result of the inference.
5. The requester component makes use of the value or, in the case of a user request, the Shell forwards the result to the user.

Scenario 3 – General-Purpose Data Processing at application level

The last scenario considers the general-purpose processing done at application-level. Applications deployed as described in User Story 4, either as batch jobs or as a service, may process data. All this processing is programmed by the application developer and happens in user space. Therefore, all the data access and all the communication channels among the containers of the application have already been secured. The trigger of this scenario is an external event either a deployment request for a batch job or an invocation to an already deployed application offered as a service.

If the application implementation does not leverage ICOS's Parallel and Distributed Execution (D&PE) component, the components of the application are deployed as described in the manifest and the data processing of each application component will be done in isolation.

Otherwise, if the application uses the D&PE component, the data processing will be distributed across all the containers deployed for that component. In this case, the trigger of this user story is either the deployment of the application in the case of a Batch job, or the arrival of a new request to the service.

1. In either case, as the execution of the code progresses, the runtime system of the D&PE will detect function invocations that can be run asynchronously (also called tasks).
2. Upon the detection of a new task, the runtime analyses the data accesses of that task to find potential data dependencies with previously detected tasks.
3. The runtime system orchestrates the execution of the task guaranteeing the sequential consistency of the whole program and devices to execute it on the local computing devices [move to step 4] or to offload the execution onto another node submitting it as a task [going back to step 2 on a different node]
4. When the D&PE runtime decides to execute it locally, it fetches all the missing input data values necessary to execute the function.
5. Upon the collection of all the input data, the execution of the task handling the execution of the task as a new potential workflow containing nested tasks whose computational workload is to be distributed across the deployed containers for that application component and that will be managed recursively going back to step 1.
6. Over the completion of the execution its nested tasks, the D&PE retrieves all their necessary results to process them and generate the results of the task.

Related Roles: NP, CP, ECPP, IoTP, CCP, ICOS End User, Customer

3.7 User Story 7 [US.7]: Deployment and Controllability from ICOS Shell

In deliverable D2.1 [1], we detailed how the Application Integrator should be able to use the ICOS Shell to interact with the ICOS System to manage the deployment of their applications. The user story was implemented with few modifications with respect to the original expectations.

For the second iteration, this user story is expanded to also cover the steps for an Infrastructure Provider (Cloud Provider or Edge Computing Provider) to deploy and configure new ICOS Agents using the ICOS Shell.

An Infrastructure Provider that wants to create a new ICOS Agent and connect it to an existing ICOS System should follow these steps:

1. The user downloads and installs the ICOS Shell on her machine (or access an already installed version remotely);
2. The ICOS Shell first contacts the Lighthouse to obtain an ICOS Controller to connect to and then contact the Identity and Access Management service to log-in the user;
3. The user requests through the ICOS Shell an "Agent Deploy Token" to the IAM. This token will allow the newly deployed ICOS Agent to bootstrap itself and register to an ICOS Controller;
4. The user specifies all the needed parameters to create a new ICOS Agent. They will include:
   - the Target Infrastructure: a Kubernetes cluster where the ICOS Agent software will be deployed. The endpoint and credentials with appropriate permissions are needed.
   - the capabilities of the new Agent (e.g., local Policy Management, local AI). This information can also be expressed in a simplified manner with multiple pre-defined "profiles" (e.g., full, only-ai, minimal);

- the orchestrator to use for the new ICOS Agent (e.g., OCM, Nuvla). The endpoint and access credentials are required.

5. The user submits a command to start the deployment of the new ICOS Agent specifying all parameters collected in steps 5 and 6. The ICOS Shell should allow to specify these commands in multiple ways (e.g., on command line, in a configuration file, interactively).
6. The ICOS Shell connects to the target infrastructure and deploys the ICOS Agent Operator, which will take care of the installation and configuration of all the required components.
7. The user can check the status of an ICOS Agent querying the ICOS Shell. The ICOS Shell will return the status and the health of the components.

The above steps will allow an Infrastructure Provider to create a new ICOS Agent. In order to on-board new edge nodes to the ICOS Agent, a very similar support should be provided by the ICOS Shell.

Related Roles: CCP, AppInt

3.8 User Story 8 [US.8]: Security and Trust Control and Monitoring

The Security and Trust Control and Monitoring user story has been expanded from D2.1 [1] to describe several user scenarios:

Scenario 1: Identity and Access control and monitoring

In this scenario, ICOS user interacts with the ICOS system using the ICOS Shell (either through the CLI or the GUI). The ICOS Shell then contacts the Identity and Access Management to authenticate the user and authorise the access to the ICOS system according to the role user has in the Identity and Access Management.

Thus, from the point of identity and access control and monitoring, Identity and Access Management of the ICOS system must have APIs supporting Authentication and Authorization.

Additionally, Application end users can use the ICOS Shell to authenticate and access ICOS services targeted for end users within ICOS system. For this, the Identity and Access Management of the ICOS system must support Authentication and Authorization to third parties.

All interactions of the ICOS users and the Application end users with the ICOS Shell and ICOS system should be secured.

Scenario 2: Audit

The ICOS system should be able to monitor and log security related system and network events and report these events to the ICOS user. The ICOS user can use the ICOS Shell to review the audit logs. Moreover, all the authorizations with the API calls pertaining to the ICOS user (API calls to the ICOS Shell) or on behalf of the ICOS user, need to be logged with the ICOS system. In general, all Authentication and Authorization events should also be logged.

Scenario 3: Detection and mitigation of security related issues

From the point of security monitoring and mitigation, the ICOS system should provide vulnerability, threats and anomaly detection of the ICOS nodes and ICOS application as well as mitigation. ICOS Agent performs (periodic) scans of ICOS nodes to detect vulnerabilities and threats. ICOS Controller performs vulnerability and threats scans of the ICOS application. Detected issues are propagated to the ICOS monitoring in the ICOS Controller and the ICOS user is notified about them through the ICOS Shell. The ICOS user can review the issues and enforce mitigation actions over the ICOS nodes and the ICOS application.

Regarding anomaly detection, the ICOS user must be alerted when an anomaly on the cloud/network/edge provider is detected using the ICOS notification system and the ICOS Shell

In this sense, as the data is being aggregated from the ICOS Agents using the ICOS monitoring the stream of data is being applied to the specific pre-trained AI model suitable for the ICOS Application descriptor. The training of the AI model and the inferencing is done in the ICOS Controller.

Next, the ICOS system must be able to categorize anomalies and the ICOS user must be able to review the anomalies. In this sense, anomalies detected by the ICOS Controller are being categorized (compared) and are used by the anomaly detection model. This model can be updated with the new knowledge, considering the knowledge from the ICOS users or other actors with the possibility to categorize it. This process should enable ICOS Shell to report less non-relevant anomalies.

Further, when the ICOS user reviews reported anomalies, he should be able to enforce mitigation actions (e.g., patching process, change to the ICOS Application Descriptor) over the ICOS nodes.

Scenario 4: Compliance detection and enforcement

Regarding compliance detection, ICOS system must be able to monitor and gather the events and map them to controls of specific standards and/or to specific policies and rules. ICOS Agent performs (periodic) scans of ICOS nodes to detect compliance issues. ICOS monitoring aggregates the data the ICOS Controller maps the data to controls of specific standards and/or to specific policies and rules. ICOS Controller has knowledge of standards and controls as well as policies and rules and can be updated with the new knowledge.

Next, the ICOS user is able to review events related to controls of specific standards and/or to specific policies and rules and enforce necessary actions (based on policy recommendations) (e.g., patching process, change to the ICOS Application Template) over the ICOS Node.

Related Roles: CCP, NP, CP, ECPP, IoTP, ICOS user

4 Use Case scenarios addressing ICOS challenges

4.1 UC1: Agriculture Operational Robotic Platform (AORP)

4.1.1 Summary

Concept: Further development of digital and robotic systems based on data exchange ecosystems and services based on their semantic processing to provide knowledge and tools that will increase efficiency, ensure safety, and confirm product quality in the supply chain, while reducing costs and providing valuable and up to date information to farmers.

Challenges: Delays in accessing data, Efficient and optimal utilization of the available edge-to-cloud resources, and connectivity in real conditions.

Expected Benefits: Reduction of decision-making latency, improved AI models, increased system availability, and predictive maintenance.

4.1.2 Use Case infrastructure and devices

The main two devices, considered to function as testbeds, hosting ICOS are described as Robot(s) and User Panel(s). As shown in Figure 2, Robot is equipped with an industrial boxed computer. This computer is connected to the robot's internal control board using a CAN-USB adapter to send control commands to the robot's actuators and read their statues. Additionally, the onboard computer is connected to a WIFI LTE guarantee connectivity to the local network and Internet. Imagery sensors, for instance, RGB and RGB-D cameras are utilised for image processing tasks and other sensors such as IMU and GNSS modules are used for precise localization.

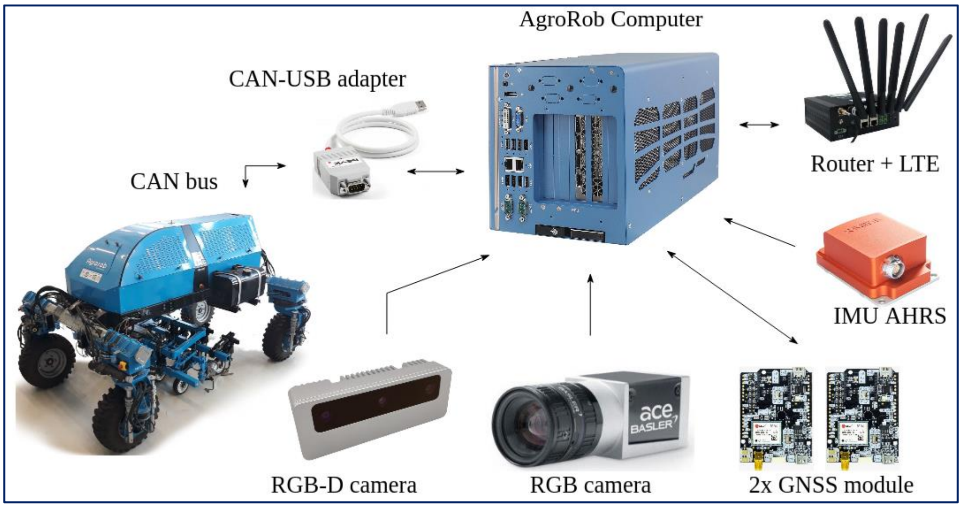

Figure 2 Robot's hardware setup

User Panel, defined as a portable computer equipped with a screen is considered to be used as the hardware for hosting user panel SW, as illustrated in Figure 3.

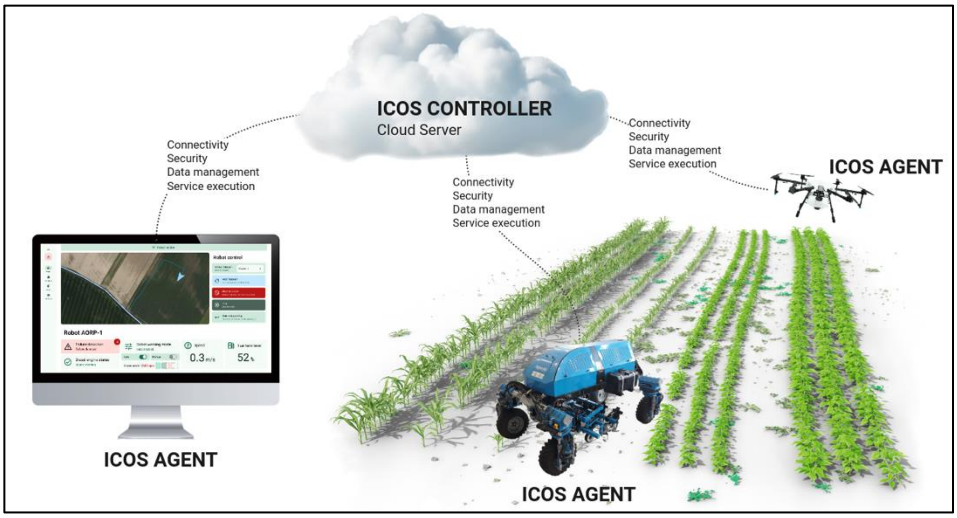

Figure 3 Use case 1 infrastructure setup and devices

4.1.3 Use Case architecture

The use case architecture can be described as a robot operating in the field, executing a given task. Depending on the mission, the robot can take advantage of cloud computing, for instance, for detecting weeds and crops. The data generated as a result of the mission, along with the status and record of the operation in the field, is sent to cloud infrastructure. The cloud processes analyse the farm condition and robot health and generate a farm yield map (as shown in Figure 4). This generated data is then sent to the robot as the next task and mission to be executed on the farm. For instance, the robot might need to spray the part of the farm where weak crop conditions were previously detected.

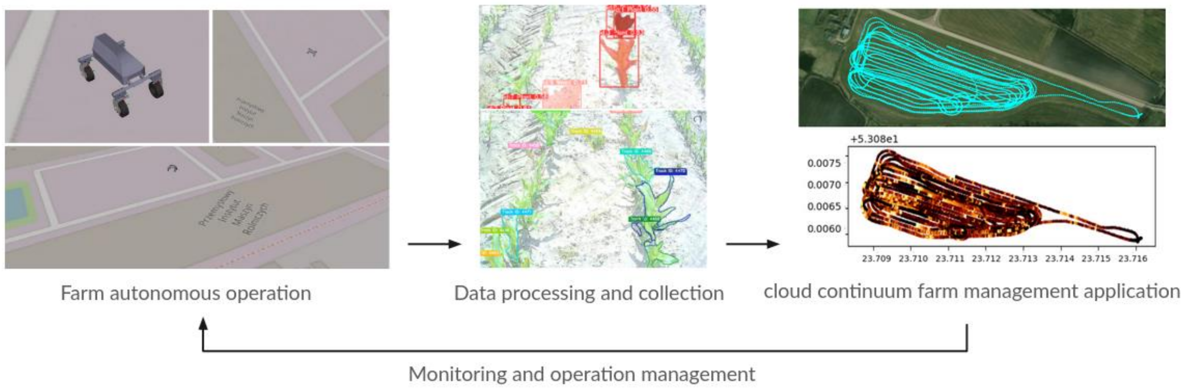
Farm autonomous operation
Data processing and collection
cloud continuum farm management application
Monitoring and operation management
+5.308e1
0.0075
0.0070
0.0065
0.0060
23.709
23.710
23.711
23.712
23.713
23.714
23.715
23.716

Figure 4 Use case system architecture

4.1.4 User stories

UC1 user stories, from a user perspective, are listed in Table below

Table 2: UC1 user stories

Type of user[1]	Goals
User, Robot, Transport platform	Transferring to the field: Robots are prepared, basic maintenance is done. Robot is moved from the farm to field: on wheels (fields in the immediate vicinity) (1) or on the transport platform (larger distance, using roads) (2).
User, Robot	Mission (common): Map upload, path following, task execution, monitoring, video data collection, navigation, status
Robot	Mission – monitoring/ inspection (R): Map creation, path following, task execution, video data collection, navigation, status
Robot	Mission - weeding (R): Map upload, path following, task execution, video data collection, navigation, status
Robot	Mission - spraying (R): Map upload, path following, task execution, video data collection, navigation, status
Robot	Re-filling (R): Established refuelling/refilling point near transport platform, robot calculates low fuel/fluid level (sensor), abort the mission and drive to the refilling point. After refill back to the previous mission.
Robot	Predictive maintenance (R): Before the malfunction (detection), Robot stops
Robot	Communication lost (R): The robot lost communication at the field and connected to the driving platform. Robot and platform lost connection – store all data.
Robot	Robot 2 farm panel data exchange (R): After the mission is completed, the robot starts sending data to the farm panel, after which the robot shuts down the system.
Farm Panel	Data synch at farm (cloud) (P): The Farm panel will communicate with cloud servers and exchange the data,
Common	Cloud calculations (C): NN algorithms calculations, etc.
Admin	System update (A): Process of uploading software/ module updates
Common	ML models improvements (C): AI/ML models improvements on the cloud

[1] User type: (U) – User; (R) – Robot; (P) – Transport platform; (A) – Admin; (C) – Common

4.1.5 Use Case requirements

The following table presents the elicited requirements for the UC1.

Table 3 Use Case 1 Requirements

Priority	Application name	Problem	Requirement for ICOS	Expected Validation date
1	User panel software, Real time Monitoring and commanding	Data integrity and synchronisation if there are connectivity problems between Edge-Cloud (Use Case is located in remote, rural area)	ICOS should ensure data synchronisation when connectivity is recovered.	IT2
2	Maintenance software, Edge to cloud data transfer and storage	Data generated as the result of performing tasks in the field must be sent and stored to the cloud, imagery files, database, and edge device (user panel) should be able to access these data.	When connectivity is available, ICOS should make sure that edge devices upload local data to the cloud	IT2
3	Robot control software	Remote deployment	ICOS should provide tools to orchestrate the deployment of control software on the edge devices.	IT2 (optional, out the project scope)

4.1.6 Use Case KPIs

- Validation and mapping of germinated plants (defining statistics on the number of germinated plants and seed quality compared to producer's declaration, etc.) - 90%;
- Effectiveness of weeds and diseases detection - 60%;
- Reduction of the amount of liquid fertilizer used in selected plant cultivation (by predicting the need for preventive treatments for a selected crop, etc.) - from 400 to 170 l/ha (-57,5%)
- Reduction of amount of plant protection herbicide used (task optimization with liquid herbicide and mechanical care with protection zone preservation) - from 300 to 60 l/ha (-80%)

4.2 UC2: Railway Structural Alert Monitoring system (RSAM)

4.2.1 Summary

Concept: The main challenge to be addressed by the use case is related to the continuous monitoring of critical infrastructure on rail tracks to ensure safety and improve maintenance activities.

The railway line along an area selected for the use case is where communications are limited in availability and bandwidth. ICOS Meta OS will make it possible to benefit from processing at the edge while sharing limited amounts of extremely relevant information to the upper layers of other applications.

Challenges: Implementing energy-efficient solutions, improving wireless networking protocols, and efficient and optimal utilization of the available edge-to-cloud resources.

Expected Benefits: Reduced decision-making latency, increased overall system availability, and improved data security.

4.2.2 Use Case infrastructure and devices

In November 2023 (M15), as part of Use Case 2, a multitude of IoT devices (Sensors, Data Loggers and Gateway) were strategically deployed along the FGC rail tracks. The deployment site spans the Lleida-La Pobla line, covering a 4km stretch within the challenging terrain of Gerb. This particular rail line, facilitating 16 train circulations daily, stands as the sole transportation artery for the region. Characterised by its precarious geodesic conditions, including water flow beneath the tracks and non-compacted layers, as well as the presence of small caves and historical instances of ground collapse, the area underscores the critical need for a real-time monitoring solution.

Gateway: 1 CMT Cloud Gateway

The Cloud Gateway, a singular unit equipped with an external antenna, assumes a critical role in linking IoT devices (Sensors and Data Loggers) with the central cloud-based system. Its primary function lies in facilitating the uninterrupted flow of data from the field devices to the central cloud, thereby enabling real-time analysis, decision-making, and feedback.

The deployed gateway on the rail track serves as the conduit for receiving a steady stream of real-time data. Another gateway model with higher capacity will need to be implemented due to specific requirements and parameters for ICOS Agent and installation processes which were not fully defined at the moment of sensor deployment. Consequently, WSE deployed a gateway model tailored to ensure the requisite data flow mandated by FGC for the project.

CMT Cloud application:

The data collection and management layer is an existing solution based on the Connectivity Management Tool (CMT) provided by WSE. The platform supports the processing and storage of sensors deployed on the FGC railway. A dedicated account was opened for FGC to get access to data from sensors in real time.

CMT Cloud allows IoT device onboarding, management and monitoring. Gateways and nodes can be managed and monitored to establish the health status of each of them. Such functionalities are supported by the Network management area of CMT.

CMT Cloud is based on a microservice architecture with several components acting as core services, while some others contribute to the applications of Monitoring, Safety and Maintenance that will be orchestrated by ICOS Meta OS.

4.2.3 Use Case Architecture

For the Railway Structural Alert Monitoring system, ICOS will be managing the Edge and Cloud processing environments. Edge will be supported by the IoT Gateway with limited resources for computing and 4G connectivity through commercial mobile services to the Cloud computing environment. The cloud computing environment used by Worldsensing is provided by Google Cloud Platform. Both the Edge device and the Cloud environment should have the ICOS agent deployed to be able to onboard such elements to the continuum.

The onboarding of both compute services will allow the orchestration of services through ICOS Meta OS according to specific requirements for the Monitoring, Safety and Maintenance applications available in the CMT Cloud solution.

Out of the scope of the ICOS managed environment, data from the IoT sensors and nodes will be aggregated at the IoT gateway through a LoRaWAN radio communication. The data collected from the IoT sensors (tiltmeters) is related to geometry parameters of the rail track, while IoT nodes collect data from geotechnical sensors (extensometers and piezometers) to geological parameters.

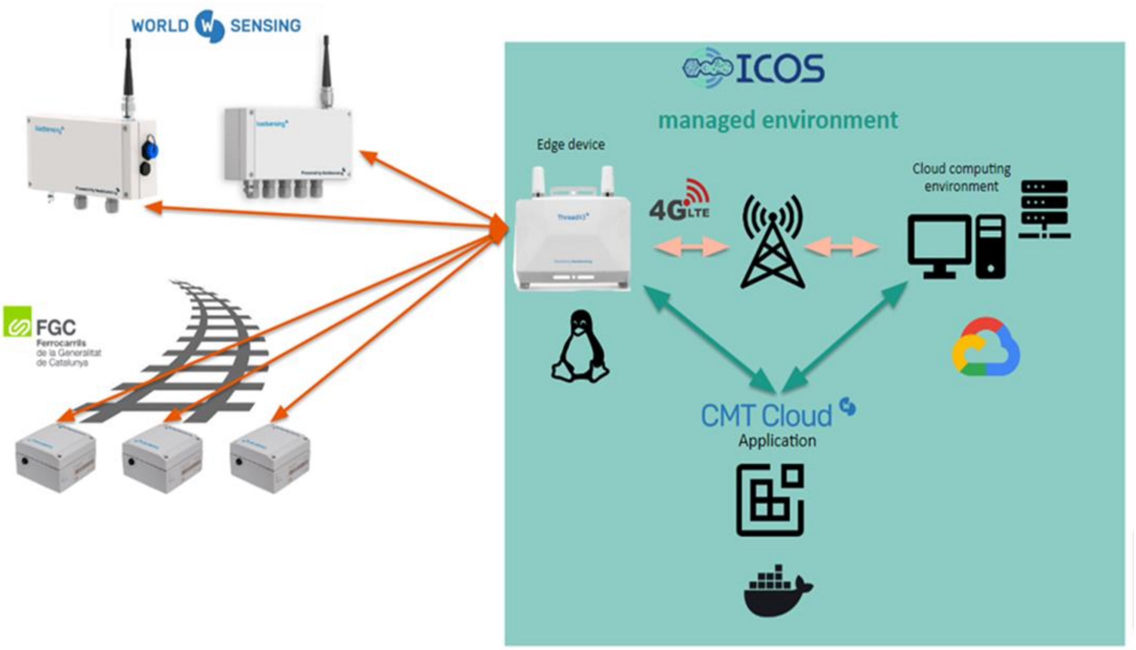

Figure 5 ICOS Use Case 2 architecture

4.2.4 Use Case user stories

The RSAM (UC2), described in detail in the paragraph above, will be based on the monitoring of rail infrastructure for safety and maintenance. The UC2 will aid in the digitization of rail infrastructure operations and resource optimization, resulting in increased rail operational efficiency. UC2 user stories, from a user perspective, are listed in Table 4.

Table 4: UC2 user stories

Type of user	Goals
End User (Control room operator)	I want to receive alerts of possible safety situations from the sensors deployed at the rail infrastructure as soon as they are detected. Then the operator can further explore the operational status and can take actions based on the information provided.
End User (Field geologist)	I want to display the time series of all data parameters collected from sensors at the rail infrastructure, to be integrated in the geological models and visualization tools.
End User (Field Engineer)	I want to review the alert events generated by each of the sensors, so that I can identify possible safety and quality non compliances.
End User (Field Engineer)	I want to visualize and interact with the prediction pattern for each parameter of the sensors, which have a direct effect in the maintenance activities.
End User (Maintenance responsible)	I want to receive warnings on possible quality parameters regarding non-compliance situations, so that I can plan maintenance activities. I also want to be able to visualise data from the evolution of the parameter monitored.

4.2.5 Use case Requirements

The following table presents the elicited requirements for the UC2.

Table 5: UC2 requirements

Priority	Application name	Problem	Requirement for ICOS	Expected Validation date
1	Real time Monitoring	Data integrity and synchronisation if there are connectivity problems between Edge-Cloud (Use Case is located in remote, mountain area)	ICOS should ensure data synchronization when connectivity is recovered.	IT2
2	Critical event detection for safety	Operate regardless of connectivity (taking local decisions)	When connectivity is not available, ICOS should make sure that edge devices process data and execute rules. Critical applications should operate regardless of connectivity availability.	IT2
3	Prediction for maintenance planning	Identify the trend and predict the moment when quality parameters would not be met	ICOS should make decisions when data transfer should be done from Edge to Cloud. Relevant rail applications should be executed according to the available capacity of data transmission and to the processing capabilities of devices.	IT2 (optional, out the project scope)

4.2.6 Use Case KPIs

1. Consolidated and integrated data visibility for critical parameters being monitored (service availability 20% more)
2. Safety application is fully operational offline (normal operation after connectivity is restored in case of temporarily lost connection) (safety factor increases 20%)
3. Reduction in data transferred to the cloud (decrease 20%)

4.3 UC3: In-car Advanced Infotainment and Multimedia Management system (IAIMM)

4.3.1 Summary

Concept: Multi-users and Multi-sites Virtual Sharing Experience to interact in sync with high-definition media contents (3D models, immersive videos, pictures, etc.) with in-car passengers and other users far away. The service provides and enriches multimedia functionalities for planning, enjoying trips and visiting touristic sites. Its deployment architecture includes edge nodes to host rendering and pre-processing and more powerful cloud nodes.

Challenges: Ensuring seamless user experience, providing secure multi-user communication and interaction infrastructure.

Expected Benefits: Optimized multimedia content distribution, enhanced quality of service, and privacy and security of shared data.

4.3.2 Use case Infrastructure and Devices

For the first iteration, in parallel with the complete vehicle integration set-up described in D6.4 [3], a proper workbench has been deployed within our laboratory setting, in order to allow a more accessible hardware validation and testing in a controlled environment.

The workbench includes the components needed to integrate and demonstrate the IAIMM functions:

Onboard Computational node: a CAR - PC (NUC) is used for hosting any customised software components needed onboard. Specifically for our case the Car PC is a CINCOZE DI-1100, which has high computational power to satisfy the requirements for the computing node. The PC forms one node of the ICOS infrastructure and will host the Control Plane that manages the onboard part of the application and the car position component that uses the information provided by the precise positioning module for feeding the application with the vehicle latitude and longitude coordinates needed for the IAIMM App.

Car HMI: the vehicle HMI is provided using a monitor directly connected via display port to the Car Pc.

Connectivity Board: The modem provides the connectivity and is connected to the Car PC through an ethernet cable, which supplies fast and stable internet connectivity to the vehicle.

Precise Positioning: the module that provides vehicle position and vehicle dynamic data is constituted by a GNSS device (GPS/GNSS Module) and the related antenna (GPS/GNSS Antenna) placed close to the lab window for satellite connectivity.

The Car PC and its auxiliaries, which are representative of the vehicle architecture, form one node of the ICOS infrastructure. The near edge cluster, hosted in a server provided by Polytechnic University of Turin, forms another node of the infrastructure located in close proximity to the vehicle. The deployment of clusters at the edge brings many advantages for future iterations of UC3 in terms of scalability, availability, distribution of data and latency.

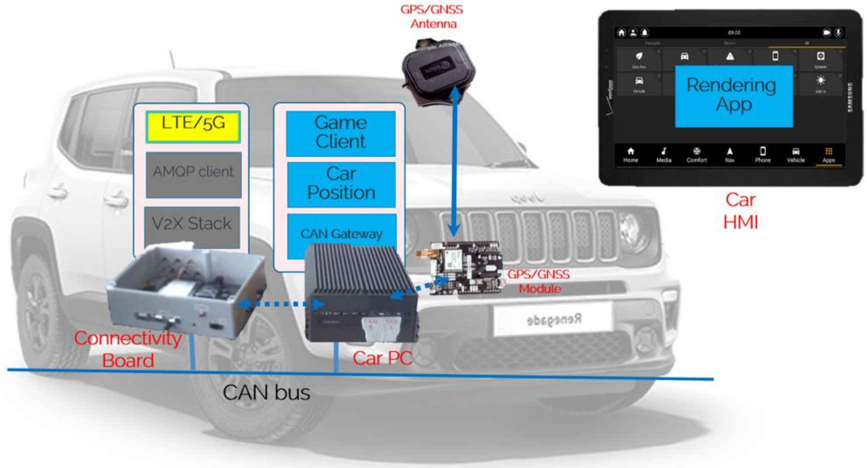

Figure 6 UC3 Vehicle Architecture

4.3.3 Use case architecture

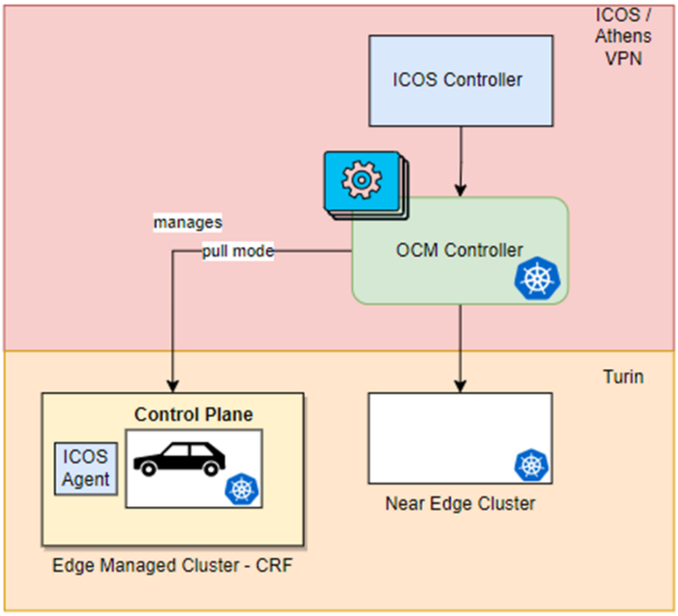

Figure 7 UC3 Architecture

UC3 architecture (see Figure 7) is composed of two nodes hosted in Turin and the ICOS cloud environment located in Athens. The onboarding of both Turin nodes into the ICOS infrastructure will allow the orchestration of the IAIMM application through ICOS Meta OS according to the specific requirements regarding latency, security and availability.

4.3.4 User stories

UC3 user stories, from a user perspective, are listed in the Table 6.

Table 6: UC3 user stories

Type of user	Goals
End User	I want to receive (from the car) the location of where it is, so that media content (3D Models, Videos, etc.) can pop up when the car gets close to a Point of Interest.
End User (Engineer)	I want to display (in the car) multimedia content so that the end user can visualize and/or interact with it.
End User (Engineer)	I want to interact (in the car) with the displayed content so that the user can interact with the service.
End User (Engineer)	I want to visualize and interact with high resolution multimedia content (i.e., 3D models) to maximize the QoE. The car must integrate an on-board computational node to allow the service to be nomadic (edge node) so the visualization and interaction will be with low latency and high bandwidth.
End User (Engineer)	I want to send and receive data from each client towards the server so that every client-interaction is in-sync with the others.
End User	I want to have all users be interacting with in-sync content so that the experience is being performed close to real time.

4.3.5 Use-case Scenario Requirements

Table 6 presents the elicited requirements for the UC3.

Table 7 UC3 Requirements

Priority	Application name	Problem	Requirement for ICOS	Expected Validation date
1	Seamless experience	Decrease latency / reduce delay maintaining high quality (QoS and QoE) and fluidity in the content provision	ICOS Edge processing capabilities should ensure a reduction of latency ensuring that it will be adequate for the consumption onboard of the multimedia content.	IT1 and IT2
2	Data security and privacy	Ensure high level of data security and data privacy is maintained at all data processing/collection and storage stages including data synchronization and integrity.	ICOS security and data management layer feature implementation should ensure the proper management of data shared by the application according with security and privacy policies	IT1 and IT2

Priority	Application name	Problem	Requirement for ICOS	Expected Validation date
3	Service availability	Ensure the availability of the service optimizing the use of available resource.	ICOS should be able to make the right decision allowing that the application could operate regardless of connectivity and poor resources availability.	IT2

4.3.6 Use Case KPIs

1. Average Time to First Render: Time from the first connection to the first render/image output on the display. Delay(-10%)
2. Accuracy of suggested PoI: How far (m/km) is the vehicle from the PoI when it is suggested and, is it visible from the car at that moment? (distance < X)
3. 3D models fluency and movement delay. Tests will be performed to validate the QoS during mobile operations so the interconnection among several nodes can show the continuum benefits of deploying the service in this architecture. Availability (+20% service available and data synchronization)

4.4 UC4: Energy Management and Decision Support system (EMDS)

4.4.1 Summary

Concept: UC4 focuses on optimising domestic energy use with Electric Vehicles, Heat Pumps, PV systems, and storage playing key roles in achieving net-zero emissions. The ICOS device's algorithm optimises energy management by predicting future demand, forecasting solar PV output, assessing EV charging needs, and anticipating future retail cost signals.

It navigates the complex decisions of using, storing, or selling energy amidst fluctuating demand, supply, and costs. This enables customers to automate intricate cost optimisation decisions while retaining control over their preferred choices

Challenges: Ensuring high level of data protection and security at all stages including data collection, analysis, storage and processing. Provisioning of customised energy solution for optimal energy usage and increase of renewable energy sources. Ensuring viable and sustainable real-time solutions in all settings including areas of poor connectivity.

Expected Benefits: By harnessing Cloud and Edge capabilities for real-time solutions, ICOS will offer reduced data transfers and latency, increased security, and flexibility to tailor to each customer's specific needs, ultimately enhancing customer satisfaction and retention.

4.4.2 Use Case infrastructure and devices

The UC4 implementation consists of three constituent parts, namely, 1) Sensors and associated IoT metering hardware installation; 2) Edge processing and associated Edge component hardware; and 3) Cloud environment and associated ICOS infrastructure.

Sensors and associated IoT metering hardware installation: This includes the installation of Inductive Power Monitoring Clamps in the fuse board of the given test house. The Inductive Power Monitoring clamps are connected with IoT energy metering hardware (ESP32 based) to record timestamped consumption data at 60 second intervals. Importantly, the hardware is connected to an MQTT broker co located within the same local network (hosted by the Edge component of UC4). This provides for transmitting the recorded data in real time via MQTT protocols.

Edge processing and associated Edge component hardware: The Edge component of UC4 is composed of a Kubernetes cluster of Nvidia Jetson Orin Nano devices, where one of the Jetson acts as the master node. These devices were selected for their high specification and capability (CPU/GPU, 8GB shared memory, CUDA cores, Wi-Fi and fast network connectivity). The setup of the master node includes deployment of containerized applications leveraging Docker based virtualization. These applications are open-source and designed specifically for home automation with a focus on privacy and location control.

Cloud environment and associated ICOS infrastructure: A dedicated Cloud environment has been set up for UC4. Currently this is a Microsoft Azure environment within SSE’s Azure tenant. The environment leverages Blob Storage and Data streaming resources. The full data streaming process of the UC4 utilises Azure Event Hub to achieve communication between the IoT, the Edge and the Cloud environment. Within this Cloud environment data is persisted to Blob Storage in parquet format. Additionally, during the Alpha phase of ICOS, UC4 has been leveraging the Test Bed Cloud facilities provided by the ICOS project to ensure seamless integration with the ICOS components.

4.4.3 Use Case Architecture

The UC4 architecture, illustrated in Figure 8, is composed of an Edge Managed Cluster with two nodes and a Cloud component hosted in Athens. The creation of the Edge cluster provides several advantages such as high availability of services, high performances, scalability, and load balancing. K3S was selected for the final UC4 Edge cluster orchestrator due to being a lightweight, cost-effective Kubernetes distribution ideal for resource restricted devices.

The cluster is composed of a master and a slave node with the possibilities to scale out the number of workers if required by the application workload. The cluster nodes are connected through the local (private) network of the households. Due to the presence of the Kubernetes Cluster, UC4 selected OCM as resourcing and clustering manager. The Edge devices of UC4 are networked with an ICOS test bed located in Athens. This is achieved by connecting the cluster’s master node to the test bed via a secured VPN channel backed by the OpenVPN protocol.

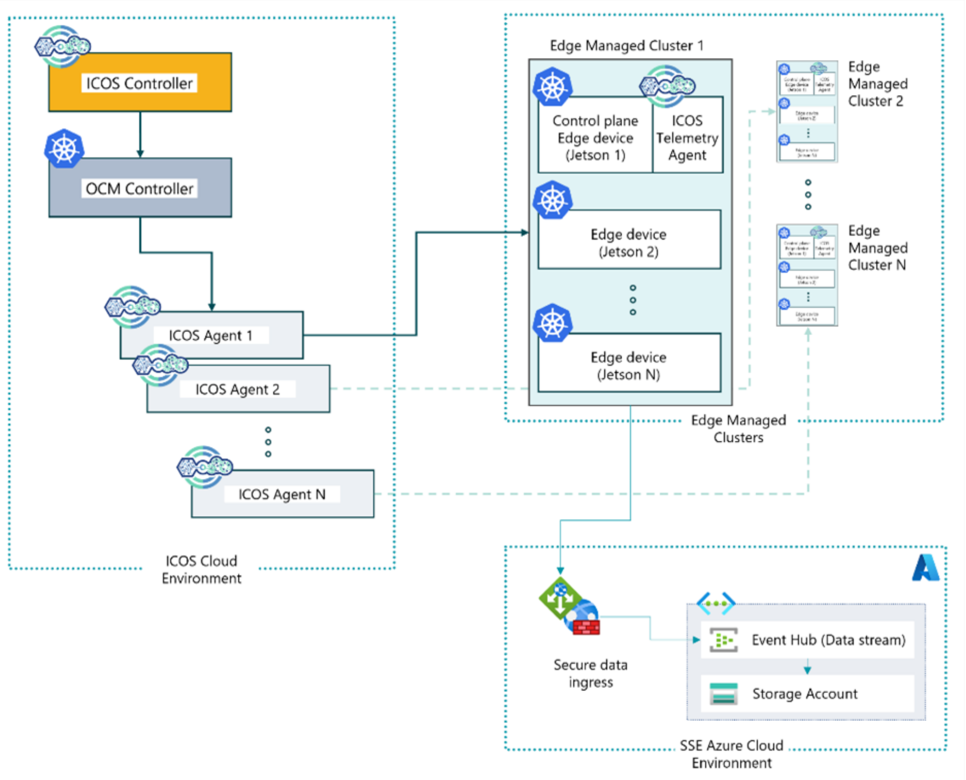

Figure 8 UC4 Architecture

4.4.4 User stories

The EMDS (UC4), will be based on energy trading/usage/storage automated decisions with the aim of improving customer habits for optimal energy consumption, cost savings and CO2 emission reduction. UC4 user stories, from a customer perspective, are listed in Table 8.

Table 8: UC4 user stories

Type of user	Goals
As an owner of PV, home battery and EV	I want to see that automated decisions are being made to get my cost of energy as close to the optimized minimum as possible.
As an owner of Photovoltaic panel	I want to use/store as much of the generated green energy as possible. The goal is to minimize energy transfer from/to the grid and optimize my energy cost.
As a user of edge/IoT devices	I want to be prompted when I have forgotten to set schedules of my EV to ensure I have enough EV range. I want to be prompted when the EV schedule differs from an established usage pattern to ensure I have enough EV range and to maximize my car's sold/used energy.
As a user of edge/IoT devices	I want to be able to have information about my energy savings in graphical or tabular form. I want to be able to track my income from separate sources i.e., P2P, supplier, DSO, TSO.

Type of user	Goals
As a user of edge/IoT devices	I want to be able to track the CO_2 intensity of the energy I pull from the grid. I want to be able to track how I have helped stabilize the grid. I want to be able to see how I have flattened my demand curve. I want to be able to track how I have helped avoid wind farm curtailment.
As a user of edge/IoT devices	I want to be able to track my cost savings if automated decisions were to be switched off. I want to be able to select between decisions I am happy to have made automatically and the ones I can decide upon (accept or reject).

4.4.5 Use-case Scenario Requirements

Table 9 presents the elicited requirements for the UC3.

Table 9 UC4 Requirements

Priority	Application name	Problem	Requirement for ICOS	Expected Validation date
1	Data security and data management	Ensure a high level of data security and data privacy is maintained at all data processing/collection and storage stages including data synchronization and integrity.	ICOS **security** and **data management** layer feature implementation should ensure the detection and mitigation of malicious activity and optimal data management and storage.	IT1 and IT2
2	Latency reduction and real-time prediction	Real-time demand-supply predictions are vital to understand electricity usage and consumption to ensure reliable and interrupted services.	ICOS Edge processing capabilities should ensure a **reduction of latency** and an **increase in security** and operate in all connectivity areas (rural/poor connectivity).	IT2
3	Automated decisions for energy consumption.	Identify the trend and predict optimal usage of energy to flatten the demand curve by removing demand at peak time and boosting energy usage at nighttime	ICOS should provide automated decisions tailored to customer needs by implementing **Trustworthy AI models** including federated learning at the EDGE level with home-to-home model parameters sharing to avail of learnings in other houses.	IT2

4.4.6 Use-case KPIs

Business need: Have a secure and efficient energy management system in place implementing flexible energy solutions tailored to customer needs and sustainability targets for customer trust, **customer retention and satisfaction.**

Client need: Have an energy management system in place where decisions are automated to:

1. Decrease costs.
2. Minimize Grid Consumption.
3. Maximize usage of renewable energy.

Technical KPIs:

1. Ensure a data security and data privacy for all data processing/collection and storage stages with a prospect of 10% fewer cyber security attacks.
2. Reduction of latency for Real-time demand-supply energy predictions (A 30% or more decrease in delay is expected in data transfer and processing)
3. Increase flexibility of services (20% more flexibility of available services with the same use of resources) and data availability

4.5 ICOS 1st Open Call Project Winners

As a result of the ICOS project 1st Open Call, 5 winning use case projects were selected, to complete the 12-month support programme and develop their projects for the ICOS validation purposes.

Open Call information

Project acronym: ICOS

Project grant agreement number: 101070177

Project full name: Towards a functional continuum operating system

Project ICOS, co-funded by the European Union's Horizon Europe research and innovation programme under grant agreement No 101070177, launched its 1st open call for recipients of financial support.

Call publication date: 28th of October 2023 at 09:00 CEST

Call closing date: 17th November 2023 at 17:00 CET

The call was published on project ICOS's website (https://www.icos-project.eu/), on FBA's website (https://icos.fundingbox.com/) and on the Horizon Europe Participants Portal.

4.5.1 Project 1: SHM with Advanced Retrofit analyTics 4 Bridges

Use Case description

Bridges infrastructure faces challenges from increasing service demands, loads caused by natural hazards and extreme weather events driven by global warming, and aging structures. These factors can impact the operation and resilience of the infrastructure. In Europe, most of the bridges were built after World War II to support economic growth and urbanization. However, many of these structures have now reached or exceeded their design life expectancy of 100 years for bridges. Many of these assets have experienced significant changes in loading conditions and may have undergone significant deterioration in their operational life. The aging infrastructure is often worsened by durability issues such as fatigue, corrosion, and creep. This poses safety risks and shortens the functional end-of-life. It is important to note that these civil engineering structures were designed according to codes with lower standards than those of today. Additionally, aging infrastructure faces additional structural challenges due to increasing traffic loads and hazards caused by global warming. For example, over 30% of bridges in EU countries require repairs, with approximately 10% of them being in poor condition.

One of the longest bridges in Greece, with a length of 1,372 m, is the Servia High Bridge (see Figure 9), that has been designed by Riccardo Morandi, and its construction started in 1972 and completed in 1974. The bridge is a part of the GR-3/E65 and is located 15 km southeast of the city of Kozani and 5 km northwest of Servia.

Today a semi-wireless system consisting of numerous sensors and a telemetry system for monitoring the health of the bridge has been deployed. The existing system faces a number of challenges related to the increased number and type of sensors, the inefficient way of data transmission, that can even happen manually in some occasions and the limitations set by the commercial off the shelf monitoring solutions that they do not lend themselves to integration. The main challenge of the Use Case deals with the structural health monitoring of the Servia High Bridge focusing on the monitoring of the behaviour of existing cracks. In particular, engineers pay special attention to the behaviour of the cracks during and after strengthening works.

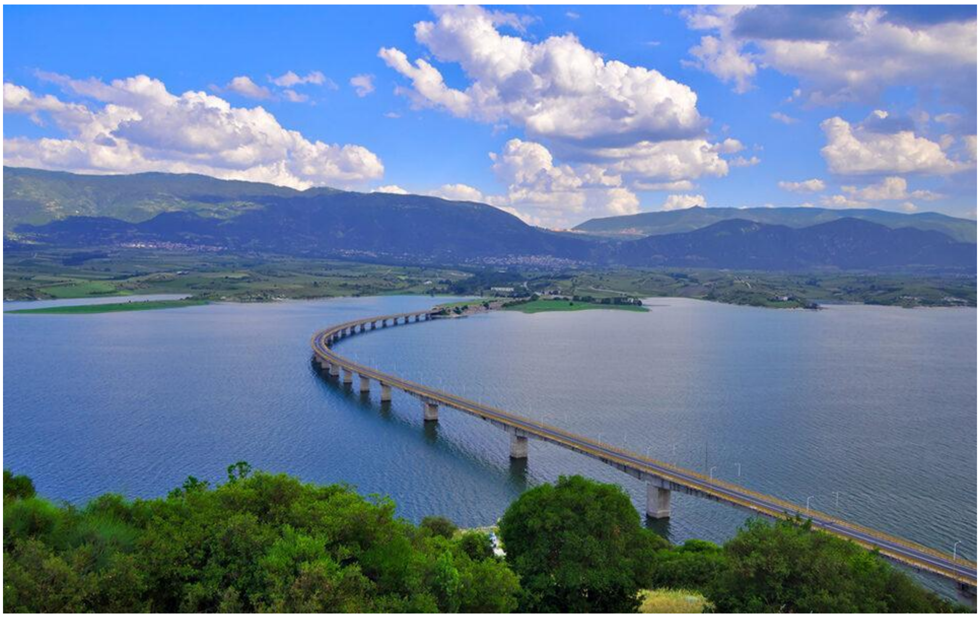

Figure 9 Servia High Bridge

Proposed Solution

SHMart4Bridges solution will have three main layers: i. Edge; ii. Local servers; & iii. Cloud. The edge layer will be responsible for getting the measurements from the sensors that have been already described and feeding cloud with data required for the analysis. The local servers layer will host the repository with the past retrofit interventions. These interventions are the strengthening works of the bridge that need to be correlated with the behaviour of the bridge and the cracks specifically. The layer of the cloud will host the data storage collected from the sensors and will also do a first data ingestions required for the post analysis of the data. The post analysis of the data also happens at the cloud layer and deals with the trend analysis of the bridge measurements, and a retrofit recommendation system.

4.5.2 Project 2: ICOSmart: ICOS for safe and smart mobility in cities

The ICOSmart project will deal with an important social, economic and environmental problem: efficiency and safety at urban road intersections.

In Europe, about half of the road accidents occur at intersections, many of them causing fatalities.

Furthermore, intersections have an important role in urban traffic flow management, and the resulting economic and environmental consequences.

The overall goal of the ICOSmart project is to adopt and integrate ICOS' MetaOS implementation and validate its IoT-edge-cloud continuum concepts under a relevant smart intersection use case within the Jena city's smart city testbed.

Proposed solution

The main output of the project will be an ICOS Meta-OS empowered software platform and associated tools to help cities to enhance traffic flows, to improve citizens safety, and ultimately to enable data-driven decision making to support the broader goal of building smarter and more sustainable cities. The use case architecture will be composed of four main layers :

- Smart intersection testbed using Jena's smart city infrastructure
- Kentyou Data Hub layer where data will be collected, unified and will be prepared for further processing. This will be the layer where most of the ICOS components will be integrated.
- Kentyou AI layer where data processing, correlation, analysis, prediction and simulations take place. Some of the data management related components from the ICOS architecture are expected to be integrated at this layer.
- Kentyou Eye layer which is the user interaction layer with data visualization and decision-making support and impact monitoring features.

The project will test two main case scenarios to test and validate the solution:

1. Monitoring intersections at real time for achieving better situational awareness (e.g., of traffic, soft mobility), forecasting and recommending relevant actions for more efficient traffic management.
2. Increase safety for pedestrians and drivers, by timely detection of potentially dangerous situations.

The project will bring together two complementary enterprises: i) Kentyou, an innovative award-winning startup which will bring its interoperable AI-driven data platform and its tools for visualization and decision making support. ii) Data in Motion, use case partner located in Jena city, a close technological partner of the city, which will bring the testbed infrastructure and domain specific expertise.

4.5.3 Project 3: GridSync

Use case description

The GridSync use case for ICOS addresses critical challenges in low voltage distribution grids, focusing on the monitoring of transformers, which are increasingly strained due to distributed generation and the widespread electrification of structures. This scenario amplifies the need for sophisticated monitoring solutions capable of ensuring operational reliability and longevity of grid infrastructure.

Proposed solution

The proposed solution involves a strategic deployment of IoT sensors in medium-low voltage distribution transformers of Trefor El-net (the project's DSO pilot partner in Denmark), facilitated through the ICOS framework. This solution is designed to significantly enhance grid monitoring and management by providing high-resolution, real-time data analytics capabilities through:

- High-resolution monitoring

- Temperature and environmental monitoring
- Extended monitoring network
- LTE/5G gateway for data relay
- Sub-second time-synchronised data
- Integration with AI Energy
- Value-added analytical outputs

This solution, embedded within the ICOS framework, will enable DSOs to advance their grid management capabilities, enhancing the reliability, efficiency, and adaptability of the grid to meet modern energy demands and integration of renewable energy sources. The deployment of cutting-edge IoT technology and the utilisation of advanced data analytics will transform grid operations, aligning with strategic goals for sustainability and resilience in energy distribution.

4.5.4 Project 4: Distributed Cosmic Ray Observatory Continuum

Use case description

The weather forecasting market was valued at 1.90 billion in 2023 and is expected to grow at a CAGR of 8.2% until 2028 to 2.8 billion. The growth is being driven by the efforts on minimising losses and enhancing production efficiency. Government support and private investment are also boosting the market; other factors are the growth of demand from power companies, the expanding sea and air transportation and the increase of extreme climate conditions.

About Space Weather, Lloyd's quantifies the economic losses in cities in 2015-2025 in Europe due to Solar Storms in US$10.3 billion, compared with US$2.2 billion from Terrorism. It is an estimate for cities and it does not include losses from black-outs, or the losses for sectors such as Telecoms, Aerospace, Space and Satellite companies. The customers can be divided into two main groups: 1) Institutions in charge of Space Weather services (NOAA/NASA, ESA, MET office, European Warning Centres, etc) and 2) Industrial end-users. The industrial customers targeted would be Power, Telecomm, Satellite, surveying and drilling, GNSS service providers, banking and insurance, and weather forecast industry, around the world. Currently the stratosphere is monitored by satellites and balloons. Both methods make discrete measurements whilst they drift. Compared to these technologies, this project will have fixed stations making continuous monitoring of the full column of the atmosphere up to the stratosphere, which again will open the door to data analysis as not possible with the current techniques.

Proposed solution

This project aims to create a network of next generation sensors based on cosmic rays detectors to generate unprecedented data for Climate and Space Weather applications in real time. Each detector will act as a node that performs atmospheric estimations through the use of AI models optimised for each specific location.

The use case includes services in two areas, offering a disruptive proposition in both:

1. Climate research and weather forecasting
2. Space Weather research and forecasting.

The project proposes real time continuous monitoring of the fixed area covered by each detector (node) of the atmosphere 24/7, compared with the current state of single measurements and in randomized areas which are not possible to make in very bad weather conditions, neither balloons nor satellites (which cannot make some readings depending on cloud cover). Having increased spatial sensitivity (up to 400% increase in number of directions) and temporal resolution (up to 300 times faster measurements) enables the use of AI and Big Data techniques with these sensors.

The use of advanced AI techniques to particle detection data creates a lot of opportunities for the exploitation of the sensors, allowing a new era of Space Weather event research and forecasting, opening the door to deeply understanding events such as the earliest proxies of severe Geomagnetic storms.

4.5.5 Project 5: Safe Work Net (SWN)

Use case description

In occupational safety management, the risks to which workers are exposed in their working environment must be identified and, if necessary, appropriate corrective measures implemented. This process of identifying occupational risks has traditionally been done by expert observation or by self-reporting of risk situations by the workers themselves. However, both methods produce significant inaccuracies due, among other factors, to human subjectivity. In this sense, Computer Vision (CV) represents an opportunity for process standardisation and cost reduction. In particular, there are opportunities for measuring ergonomic risks, detecting hazardous situations and monitoring the use of personal protective equipment (PPE). These systems allow a parallel and real-time processing approach of images from different cameras and devices, which can be used to automatically and accurately monitor a working environment. From a computational point of view, the most recently developed Convolutional Neural Networks (CNNs) represent ideal tools for typical CV tasks, such as object detection, person tracking and instantaneous posture identification of a worker, among others.

Proposed solution

Industry 4.0, based on the digitalisation of the working environment, offers new solutions to solve these repeatability problems in ORAs. Risk monitoring in work environments is an area of research of growing interest in Industry 4.0, as its results allow for improved decision-making by experts and thus reduce the economic, social and ethical cost generated by occupational accidents. In recent years, this problem has been addressed mainly in two ways: through the use of inertial gauges (direct measurement), and through CV techniques (indirect measurement).

Today, it is evident that I4.0 technologies present a broad outlook for growth and applicability in key sectors such as industry and construction. However, this is a cumbersome process to adopt, often due to the need for highly trained personnel, time and financial resources. Reducing TMEs is especially costly in changing work environments, typical of the construction sector, particularly with the use of direct measurement ORA methods, as they require inertial gauges that are attached to the worker's body, reducing comfort and limiting the naturalness of the work to be assessed. In contrast, the use of CV methods only requires computer equipment and portable cameras.

To overcome these limitations, the proposed SWN Project aims to investigate and test new alternatives for occupational safety improvement processes based on the use of Artificial Intelligence (AI)-based CV techniques and models, designing and training a Computer Vision System (CVS) to be used in the monitoring of movements that may generate a high ergonomic risk for the worker, to document in an automated way the traceability over time of the use of PPE by the workforce as well as to detect high-risk situations in order to prevent the possible accident.

More specifically, the SafeWorkNet (SWN) project, which we propose as an ICOS use case, aims to deploy an innovative digital occupational safety and health (OSH) surveillance system based on a network of nodes with CV capabilities interconnected through a low latency 4G/LTE/5G network. The system consists of IoT nodes with vision sensors and limited computational capacity (deep edge) and nodes with extended computational capabilities (meta edge). The IoT nodes will collect and analyse real-time video by running lightweight AI models that enable the recognition of safety risks in the workplace. These IoT nodes will be plug&play, can be attached to structures where needed and reused in other installations. The meta edge nodes will run heavier AI models for the detection of more complex situations, such as ergonomic risks. The system is completed by the cloud infrastructure that hosts the central SWN application in charge of both managing and administering the system and its multiple tenants, as well as running deep analytics on the accumulated historical data for the identification of risk patterns and critical areas. Finally, by integrating SWN with the orchestration and computational continuum management capabilities and functionalities offered by ICOS, SWN will ensure a distributed, collaborative, robust and resilient real-time OSH system.

Thus, three specific themes are addressed:

- Monitoring the use of PPE.
- Detection of high-risk situations that may lead to an accident.
- Automated measurement of musculoskeletal risk.

These three themes will be applied throughout the SWN project, individually or in combination, leading to the concrete functionalities and use cases that will be tested in the real test environment.

The end result, in the form of a future commercial product supported by the advances made in SWN, will be a solution characterised by the following advantages for the end-user:

- Ease of deployment as it can be applied to existing camera systems as well as to new cameras installed specifically.
- Flexibility by allowing configuration changes by the end user himself either by relocating cameras or by applying specialised models without the need for experts in programming and/or data science,
- Scalability of a modular solution both in terms of hardware and software including specialised models.
- Adaptability to the detection and analytical needs of each type of risk, offering on the one hand a rapid response in those cases that require it (e.g., pedestrian collisions or entrapment of workers by machinery) and in parallel a more in-depth data analysis for more complex cases (e.g., musculoskeletal risk and use of PPE).
- Continuous monitoring and evaluation of worker safety based on objective criteria.

5 Updates on ICOS Requirements

The ICOS Requirements Elicitation is the result of analysis and a set of activities implemented within the different actors that constitute the value chain of the project. Requirements will rely on the outcomes of sections 5 and 6. The results from mentioned analysis are detailed in the following sections 7.1 and 7.2. This output is produced by following the MoSCoW method to reduce the requirements elicitation extension.

To produce the functional requirements, ICOS is analysed from five different perspectives that are considered the main topics of the project:

- Continuum Creation: Requirements related to the onboarding and setting up of ICOS Ecosystem, including service and resource discovery and configuration.
- Continuum Management: Requirements regarding Governance and Orchestration of the resources that compose the ICOS Ecosystem as well as the software services provided for the end users of this infrastructure.
- Data Resiliency and Transformation: Requirements related to the ICOS internal (meta kernel) data management policies and mechanisms, including data access interfaces, caching policies, and data transformation optimizations.
- Smart Security and Trust: Requirements related to the security and audit when using the ICOS components, detection and mitigation of anomalies as well as detection of compliance issues and their mitigation.
- Operability Serviceability (GUI/CLI): Requirements that affect on one hand the ICOS System usability for end users in terms of graphical or command-line interfaces. And on the other hand, the operability that ICOS system offers to business application operators.

5.1 Functional Requirements

Table 10: Continuum Creation Requirement List

ID	Requirement Name and Description	User Story	Implemented	Validation
CC_FR_01	**Resources Catalogue:** The ICOS MUST support a resource registry to record (publish) available resources to operate application workloads	US.1 Cloud Continuum Realization.	Yes. The Aggregator component contains this information at Controller level and the lighthouse hosts a list of all the registered Controllers	It was already shown in WP3 demo in IT-1.3
CC_FR_02	**Discovery:** ICOS MUST support methods to discover registered infrastructures	US.1 Cloud Continuum Realization.	Currently, IoT and computing capabilities at the Edge are discovered when an ICOS agent is manually onboarded. During the second iteration we expect to enhance the solution by automating the onboarding process.	It was already shown in WP3 demo in IT-1.3

ID	Requirement Name and Description	User Story	Implemented	Validation
CC_FR_03	**Topology Awareness:** ICOS should be able to monitor and maintain the topology of the created Cloud Continuum.	US.1 Cloud Continuum Realization.	Yes. The monitoring infrastructure continuously monitors metrics of the devices and pushes them to the Aggregator of the Controller who has a full view of the system. For the second iteration we aim at reducing the amount of information being transferred.	It was already shown in WP3 demo in IT-1.3
CC_FR_04	**Controller Communication:** ICOS should allow the communication of multiple ICOS controllers to exchange local views, policies and information.	US.1 Cloud Continuum Realization.	No	To be demonstrated as part of the final demo

Table 11: Continuum Management Requirement List

ID	Requirement Name and Description	User Story	Implemented	Validation
CM_FR_01	**Smart resources first allocation and migration:** ICOS MUST be able to find a near-optimal match (considering different metrics, such as response time, energy footprint, monetary cost) in terms of nodes to run one business application taking into account nodes performance, reliability and availability	US. 1 Cloud Continuum. US. 3 New Resource On-boarding.	Yes. The Match-maker component is able to find a first allocation for the application components. During the second iteration we expect to improve the policies of the Match-maker even considering the migration or scaling up and/or down of some components	First allocation was already shown in WP3 demo in IT-1. Smarter policies and migration will be part of the final demo

ID	Requirement Name and Description	User Story	Implemented	Validation
CM_FR_02	**Workload Offloading**: ICOS MUST be able to distribute the workload of the application components offloading part of their computation onto other nodes of the infrastructure, and coordinate the offloaded components	US. 1 Cloud Continuum. US. 5 Dynamic reconfiguration US. 6 Data Access and Processing.	No. The Parallel and Distributed Execution (D&PE) component will be in charge of offloading workload. Its complete integration will be done during the second iteration	To be demonstrated as part of the final demo
CM_FR_03	**Function execution request**: ICOS COULD provide a mechanism to request the execution of a function on the continuum being totally transparent of the device that will host the execution	US. 4 New Service Deployment. US. 7 Deployment and Controllability from ICOS Shell.	Yes. The D&PE component implements this functionality.	Tested in isolated environment
CM_FR_04	**Distributed Control**: ICOS must provide decentralized control nodes along the entire continuum, these control nodes must be distributed according to physical Edge and IoT location in order to provide closer control over specific locality or region	US. 2 IoT GW/Node or Device On-boarding. US. 3 New Resource On-Boarding.	No. Currently, management is centralized in a single Controller. For IT-2, we expect to distribute control using multiple cooperative Controllers.	To be demonstrated as part of the final demo
CM_FR_05	**SLOs Definition**: ICOS must provide the capability to set an application workload Service Level Objective (SLO) by Quality-of-Service levels to be achieved. This agreement must also define the appropriate preventive to ensure SLO compliance.	US. 1 Cloud Continuum realization. US. 3 New Resource On-Boarding.	Yes. The current Application Descriptor model contains a policies section where to indicate SLO for each component.	It was already shown in WP3 demo in IT-1.3
CM_FR_06	**Monitoring**: ICOS MUST collect monitoring infrastructure-level and application-level metrics from various sources as well as provide	US. 1 Cloud Continuum realization. US. 4 New Service Deployment. US. 3 New	Yes. The monitoring infrastructure is already collecting metrics from all the devices and forwarding them to the aggregator where they are stored for processing. The	It was already shown in WP3 demo in IT-1.3

ID	Requirement Name and Description	User Story	Implemented	Validation
	appropriate solutions to preserve and access historical performance data. Collecting metrics from already existing monitoring tools on the infrastructures should be supported.	resources on boarding.	automatic onboarding of an agent will simplify the configuration of the component.	
CM_FR_07	**SLO monitor to raise a corrective plan**: ICOS MUST monitor application workload SLO to raise remedial plan as well as corrective actions when QoS levels are violated.	US. 5 Dynamic Reconfiguration.	Yes. The Policy Manager is already able to detect a violation of the SLO and raises an alert with a corrective action, but it is not yet applied by the JobManager. For IT2, the intelligence driving the selection process for this correction action will be improved with smarter options and the Job Manager will apply remediation actions.	SLO violations were already demonstrated in WP3 demo for IT-1.3. The application of corrective actions will be part of IT2 demo.
CM_FR_08	**Predictive Monitoring/SLOs violations:** ICOS SHOULD predict potential future SLOs violations analysing monitoring metrics with ML techniques, and prepare to avoid/absorb such risk	US. 5 Dynamic Reconfiguration.	No. For IT2, the Intelligence Layer will use monitoring data to produce new predictive metrics that can be used by the Policy Manager.	These predictive metrics will be part of IT2 demo.
CM_FR_09	**Green Policies Monitoring:** ICOS must be able to determine when reserved resources are not used or required for proper application operation at some point in time and provide an alert system and offer an according SLO modifications	US. 4 New Service Deployment. US. 5 Dynamic reconfiguration	No. Currently, ICOS already collects metrics related to the energy consumption of the Edge nodes. However, we have limitations for IoT devices. In IT2, these issues should be solved and allow the definition of Green Policies.	Currently collected energy metrics were part of WP3 demos for IT-1.3. Green Policies will be part of IT2 demo.
CM_FR_10	**Data Management:** ICOS must be able to maintain the data sources topology as well as data source types (metadata)	US. 6 Data Access and Processing.	Not yet.	To be validated through the applications and

ID	Requirement Name and Description	User Story	Implemented	Validation
	for proper application data assignment. This includes data source selection, data source-application binding, and data access			components regular execution.
CM_FR_11	**Data Access:** ICOS must be able to provide different data access methods, including selective access as well as streaming access, according to flexible high level data access application program interfaces	US. 6 Data Access and Processing. US. 2 Device on-boarding	Partially. The high level data access application program interfaces are being used on several components; the upcoming iteration will further increase this number of components.	Components accessing data serve as a validation of this requirement.
CM_FR_12	**Infrastructure-agnostic programming:** ICOS must provide a programming environment that abstracts service developers away from the details of the management of the devices composing the underlying infrastructure and its network topology	US. 7 Deployment and Controllability from ICOS Shell.	Yes. ICOS embraces containerization technologies allowing developers to code their application components totally unaware of the underlying hardware. Using COMPSs, components can internally parallelize and distribute their execution with no infrastructure awareness. Also, the Application Descriptor model sets the requirements for each component without knowledge of the actual devices that will host their execution.	It was already shown in WP3 demo in IT-1.3
CM_FR_13	**Parallelism exploitation:** ICOS MUST provide a mechanism that allows service components to decompose application components into sub-components (or tasks) to enable massive/distributed parallel execution (and achieve lower response times and a better	US. 5 Dynamic Reconfiguration. US. 6 Data Access and Processing.	Yes. The D&PE component automatically parallelizes the execution of a sequential Java/Python code following the COMPSs programming mode.	It was already shown in WP3 demo in IT-1.3

ID	Requirement Name and Description	User Story	Implemented	Validation
	resource exploitation)			
CM_FR_14	**Model optimization techniques**: ICOS SHALL provide different methodologies for the pruning calculation of different deep neural network backbones that provides efficient sparse connectivity for modelling high-performance models on edge nodes. Distillation techniques will be used to shorten training times using the high capability of large models, the ICOS system will offer various techniques for distilling various deep neural network backbones.	US. 5 Dynamic Reconfiguration.	No. These techniques to improve the footprint of AI models already built will be developed in IT-2.	A/B testing is to be performed in isolated infrastructure. This is to be shown as part of an Intelligence end-to-end flow in the final project demo.
CM_FR_15	**ML Scalability and maintenance**: ICOS SHOULD perform model training detached from ICOS nodes. By training the models on a separate platform with more computing power, it is possible to produce models with higher accuracy and to reduce the time required to train them. Once the models are trained, they can be deployed on ICOS nodes to make predictions and carry out specific tasks.	US. 5 Dynamic Reconfiguration.	Yes. This is achieved through data management (dataClay). Intelligence and data management integrate since IT-1 for this purpose.	Testing was performed in infra at Dublin, Barcelona, and Athens using a dataClay backend in the same node as the Intelligence API (controller suite) and at different nodes for AI offloading through the network. This has been demonstrated already in the IT-1 demo.

ID	Requirement Name and Description	User Story	Implemented	Validation
CM_FR_16	**Federated Learning benchmarking**: ICOS MUST be able to train models over open-source federated learning frameworks. Models trained on ICOS nodes (the edge or on the cloud) will be able to share knowledge when applicable to learn collaboratively.	US. 5 Dynamic Reconfiguration.	No. Federated learning is under implementation in IT-2.	Testing is to be performed in an ICOS testbed. This capability will be described as part of the IT-2 demo.
CM_FR_17	**AI for resource management**: ICOS MUST classify infrastructure that facilitates the resource management on the network, and implements various resource management algorithms for the continuum, as well as clustering the various resources in the ecosystem with the goal of optimizing those available.	US. 5 Dynamic Reconfiguration.	Partially. ▸ In IT-1, intelligence offered models for load forecasting, load anomalies, and CPU utilization. ▸ In IT-2 Intelligence will look at predicting metrics to enhance tasks such as matchmaking and rate different nodes of the available topology exploiting the concept of locality.	A/B testing of models in isolated infrastructure. CPU metrics forecasting was demonstrated in the IT-1 demo. Testing will be done in the testbed for IT-2. To be part of the intelligence flow in the final project demo.
CM_FR_18	**MLOps frameworks**: ICOS MUST allow the storage, retrieval and modification AI models available to be used by clients and AI-as-a-Service (AIaaS) providers from ICOS	US. 5 Dynamic Reconfiguration.	Yes. An API to use AI as a service with a model registry for storage, retrieval, and updates of AI models was provided as part of IT-1.	This functionality was tested in isolated infra and also deployed in the ICOS testbed. This was already shown in the IT-1 demo.
CM_FR_19	**AI Marketplace**: ICOS SHOULD use data science and MLOps frameworks such as MLFlow and BentoML	US. 5 Dynamic Reconfiguration.	Partially. MLOps and BentoML, described in the requirement description, have already been	AI Marketplace to be shown as part of the final project

ID	Requirement Name and Description	User Story	Implemented	Validation
	for version control and similar libraries to ensure reproducibility and proper documentation of each model built or experiment run.		implemented since IT-1 as part of the AI Coordination module. The marketplace covered in the requirement name is planned from M30 of the project.	presentation and/or demo.
CM_FR_20	**Continuous learning**: ICOS SHOULD support algorithms for continuous learning and model retraining, adapting to changes in the data and avoiding catastrophic forgetting.	US. 5 Dynamic Reconfiguration.	Yes. Basic continuous learning functionality implemented. The intelligence layer allows continuous learning by reusing previous models from the model registry and performing incremental model updates. More models to support continuous learning as part of IT-2.	A/B testing was performed in isolated infrastructure.
CM_FR_21	**Multicluster (secure) connectivity**: ICOS SHOULD support direct (secure) networking between Pods and Services in different clusters, either on-premises or in the cloud.	US. 1 Cloud Continuum.	No. Secure connectivity among clusters will be achieved by using clusterlink whose integration is planned for IT-2	Multi-cluster connectivity will be part of IT2 demo.

Table 12: Data Resilience and Transformation Requirement List

ID	Requirement Name and Description	User Story	Implemented	Validation
DRT_FR_01	**Data distribution across the continuum**: ICOS SHOULD take advantage of all available devices to place data. Flexibility to adapt to different configurations (cloud-only, edge-only, cloud-edge)	US. 6 Data Access and Processing.	Not yet	Demonstrated through specific UC in IT2.
DRT_FR_02	**Transparent data access**: ICOS must be able to provide location and format transparent data access methods through flexible high level data access application program interfaces (API)	US. 6 Data Access and Processing.	High-level interface provided for Intelligence Layer. Bus abstraction available. More data access methods will be available for the upcoming iteration.	Components are using the interfaces to access ICOS data.

ID	Requirement Name and Description	User Story	Implemented	Validation
DRT_FR_03	**Minimization of data transfers**: ICOS MUST avoid unnecessary data movements to increase performance, reduce network congestion, and favour privacy by exploiting near-data processing	US. 6 Data Access and Processing.	Data Management supports task offloading which avoids data transfers	Demonstrated through the Intelligence Layer.
DRT_FR_04	**Support for distributed/parallel execution**: ICOS SHOULD provide the integration of data management with the execution runtime to support efficient scheduling and execution of the required tasks.	US. 6 Data Access and Processing.	Yes (COMPSs and dataClay integration support distributed and parallel execution).	Isolated environment (not within ICOS yet)
DRT_FR_05	**Failure recovery mechanism/management**: ICOS MUST provide the capability to restart the failing transfers of the data in case of the failures (e.g., losing the connectivity)	US. 6 Data Access and Processing.	Not yet	Part of IT2 demo.
DRT_FR_06	**Secure data exchange**: ICOS SHOULD support data preserving communication techniques between modules (interfaces) for secure data exchange.	US. 6 Data Access and Processing. US. 8 Security and Trust Control and Monitoring.	Current versions of Data Management Components include TLS support for encrypted communication	Isolated environment (not within ICOS yet)
DRT_FR_07	**Data Recopilation:** Control Nodes must be able to recollect data in a scheduled or periodic way from the nodes it orchestrates	US. 6 Data Access and Processing.	Not yet	Part of IT2 demo.

Table 13: Smart Security and Trust Requirement List

ID	Requirement Name and Description	User Story	Implemented	Validation
SST_FR_01	**Trust Nodes**: ICOS SHOULD be able to provide mechanisms for establishing trust procedures for the onboarding of ICOS nodes and the deployment of applications.	US. 8 Security and Trust Control and Monitoring.	ICOS Controller: Cilium CNI with mTLS ICOS Controller<->Agent: VPN with Wireguard	Deployed ICOS Controller at Staging env and ICOS agent at all UC
SST_FR_02	**Secure comms**: ICOS SHOULD be able to exploit available encryption and isolation mechanisms to provide sufficient levels of security at the cloud continuum level	US. 8 Security and Trust Control and Monitoring.	ICOS Controller: Cilium CNI with mTLS ICOS Controller<->Agent: VPN with Wireguard	Deployed ICOS Controller at Staging env and ICOS agent at all UC
SST_FR_03	**Tenant Isolation**: ICOS SHOULD provide mechanisms that allow for isolation of resources across multiple tenants	US. 8 Security and Trust Control and Monitoring.	To be implemented in IT-3 using multiple Keycloak instances or multiple realms	To be demonstrated as part of the final demo
SST_FR_04	**Secure & robust model training**: ICOS SHOULD provide the ability to train models in a federated learning fashion to ensure privacy and trust. Privacy can be guaranteed by training data on the edge. Robustness is assured by assessing the data against anomaly detection.	US. 8 Security and Trust Control and Monitoring.	Federated learning will be implemented by Intelligence in IT-3. Privacy: To train data at the edge we can offload workloads through Dataclay. Privacy will be allowed there when Dataclay integrates with the ACLs to ensure that only relevant users have access to the data. Robustness: We are working to put a drift detection endpoint to the Intelligence API to provide a functionality that monitors model performance to allow re-training.	To be demonstrated as part of the final demo
SST_FR_05	**Providing trustable models**: ICOS SHALL use models ethically unbiased. By using these models, ICOS strives to preserve trust in the outputs of the ICOS system and provide a fair and equitable experience	US. 8 Security and Trust Control and Monitoring.	Implemented: This is TrustworthyAI. It's a module in the Intelligence layer. For the Beta we are aiming to add model explainability to the	ICOS v1.3 DEMO.

ID	Requirement Name and Description	User Story	Implemented	Validation
	for all.		AIOps platform deployed as part of the Intelligence controller suite.	
SST_FR_06	**Secure infrastructure and code:** ICOS SHOULD assess security of infrastructure (e.g., running Critical Security Controls - CIS benchmarks) and system and application code (e.g., running vulnerability scanning of docker images)	US. 8 Security and Trust Control and Monitoring.	Security of infrastructure - Security Scan (Wazuh). Code security – integrated tools in ICOS CI/CD (e.g., Trivy). Planned for IT-2: integrate Trivy and IaC Scan Runner for scanning of use case application code.	Security Scan – UC, ICOS v1.3 DEMO.
SST_FR_07	**SecureAPI:** ICOS MUST provide API supporting AuthT/AuthZ and Audit capabilities	US. 8 Security and Trust Control and Monitoring.	ICOS Services integration with Keycloak (services AuthT and AuthZ); Each service checks if the user has permissions in its token to access it. Tetragon for Audit – planned for IT-2.	UC, ICOS v1.3 DEMO
SST_FR_08	**SecureLIB:** ICOS SHOULD provide libraries supporting Authentication /Authorization to 3rd parties	US. 8 Security and Trust Control and Monitoring.	Planned for IT-3: Keycloak libraries to be used by use cases for AuthT/AuthZ (integration with Keycloak).	To be demonstrated as part of the final demo
SST_FR_09	**Anomaly detection:** ICOS MUST provide a mechanism for the detection of anomalies (e.g., any kind of abnormal situations, including potential security threats, that is recorded in application, system, or network logs) in the applications/services on the cloud/network/edge provider	US. 8 Security and Trust Control and Monitoring.	Anomaly detection with LOMOS	ICOS v1.3 DEMO
SST_FR_10	**Anomaly mitigation and recovery:** ICOS MUST be able categorize anomalies and recommend specific mitigation actions or recovery process	US. 8 Security and Trust Control and Monitoring.	Anomaly detection (LOMOS) to categorize anomalies, Telemetry->Dynamic Policy Manager->Meta kernel for mitigation actions – planned for IT-3	To be demonstrated as part of the final demo

ID	Requirement Name and Description	User Story	Implemented	Validation
SST_FR_11	**Compliance detection**: ICOS MUST provide a mechanism for the detection of compliance problems regarding controls of specific standards and/or specific policies and rules.	US. 8 Security and Trust Control and Monitoring.	Security Scan (Wazuh) – existing metrics (CIS benchmark) and regulatory compliance metrics-GDPR (not yet enabled). Cilium in ICOS Controller	UC, ICOS v1.3 DEMO
SST_FR_12	**Compliance enforcement**: ICOS MUST provide a system to trigger infrastructure changes to ensure standard and/or policy compliance	US. 8 Security and Trust Control and Monitoring.	Dynamic Policy Manager->Meta kernel for compliance enforcement – planned for IT-3	To be demonstrated as part of the final demo

Table 14: Operability Requirement List

ID	Requirement Name and Description	User Story	Implemented	Validation
OP_FR_01	**Resources publication and discovery**: ICOS MUST support the discovery and publication of resources information (e.g., available cameras etc.) via APIs or Browser so they can associate them to applications	US. 1 Cloud Continuum. US. 3 New Resource On-Boarding. US. 4 New Service Deployment.	As explained for CC_FR_1 and CC_FR_2, ICOS already supports onboarding and discovering devices, but the functionality is not offered through an API or Browser yet. Planned for IT2	This will be part of IT2 demo.
OP_FR_02	**Topology Provision**: ICOS MUST provide the capability to graphically describe the topology and related resources needed to run application workloads including constraints, communication and security policies to apply on each node level	US. 3 New Resource On-Boarding. US. 4 New Service Deployment.	As explained for CC_FR_3, ICOS already supports topology provisioning and is currently described in text mode. Providing this information graphically is planned as part of IT2.	Text mode was already shown in WP3 demo in IT-1.3. Graphical interface will be part of It2 demo.
OP_FR_03	**Resources classification**: ICOS MUST be able to catalogue the reliability, trustworthiness, confidence, of system resources, in order to take into consideration during the resources allocation process	US. 8 Security and Trust Control and Monitoring.	The security layer already provides metrics related to the vulnerabilities of the nodes and this is expected to evolve in IT2.	Security metric already shown in IT1.3 demo.

ID	Requirement Name and Description	User Story	Implemented	Validation
OP_FR_04	**Resource Descriptor Data-model:** ICOS MUST define a data model to be used for describing each device/node resource capability in order to be accessible by the metaOS	US. 6 Data Access and Processing.	The characteristics of each device/node are collected by the Telemetry infrastructure and later used by Policy Manager and Match-Making components of the distributed meta-kernel.	Shown as part of IT1.3 demo
OP_FR_05	**Monitoring system performance:** ICOS MUST display resources and application workloads performances in real time as well as historical performance data within a graphical view. This view should allow service operator to modify the data to display	US. 7 Deployment and Controllability from ICOS Shell	Not yet implemented	Planned for future iteration
OP_FR_06	**Data Processing and ML modelling:** ICOS MUST support dashboard/interface for data processing and ML modelling purposes. A concrete demonstration of the model training and inference performance will be provided via a dedicated GUI. Data processing, ML and FL platforms will be based on open-source tools to offer scalability and transparency of the deployed models.	US. 5 Dynamic Reconfiguration	Swagger interface implemented in IT-1 for basic model management. This is pending for FL, planned for the second phase of the project. JupyterHub and AIOps (MLFlow) were added as part of IT-1 to allow use cases to interact with Intelligence. Dashboarding is supported through telemetry (Grafana). Intelligence pushes model results to telemetry.	Already shown in the IT-1 demo. Unit testing written for D5.2. Testing in ICOS testbed for IT-1.
OP_FR_07	**Intuitive GUI:** ICOS SHOULD provide a GUI that is structured logically and can be understood with minimum requirement of documentation by a reasonably educated (in the scope of ICOS, e.g., network administrator) human.	US. 7 Deployment and Controllability from ICOS Shell	Not yet implemented	Planned for IT-2

ID	Requirement Name and Description	User Story	Implemented	Validation
OP_FR_08	**Interoperable CLI:** ICOS MUST provide a CLI that implements all CUD functionalities of the GUI. Read functionalities might differ for technical reasons, e.g., the lack of notification channels on CLIs or lack of efficient ways to present graphs. This does not only allow for automated testing during development but also for integration with future, at the time of implementation unknown, components. The output of this CLI thus must be machine readable, either by default or after providing a parameter/setting an environment variable.	US. 7 Deployment and Controllability from ICOS Shell	Yes, but will continue to evolve	Shown in ICOS v1.3 DEMO
OP_FR_09	**Granularity of the access**: ICOS SHOULD provide the possibility to define different roles in accessing ICOS	US. 7 Deployment and Controllability from ICOS Shell	Partially; shell transmits authentication token with all requests, this is currently not checked by all other components.	Partially in v1.3 Demo, partially future DEMO

5.2 Non-Functional Requirements

The scope of this section is to define the requirements regarding the quality of the software to be delivered. The quality assurance model is based on the ISO 25010 and represented in several perspectives as follows:

- Functional Suitability: refers to the correctness of the system in terms of properties and functionalities that the system must have and comply with, also includes Functional Completeness and Appropriateness.
- Efficiency: refers to the performance of the system which compares the resources used with the amount of results or capabilities produced by the system. Usually expressed in terms of time, capacity and resource usage.
- Compatibility: refers to the capacity of the system to be integrated or interrelated with other systems or components, as well as the degree of communication and shared resources between them.
- Usability: refers to the effort needed to acquire the capacity to use the system, more specifically, learnability, operability and understandability for an individual user.
- Reliability: refers to the capacity of the system to provide a certain number of services in a specific condition during a given amount of time. Also refers to availability and fault tolerance.

- Maintainability: refers to the effort required to apply modifications to ensure software longevity. Usually expressed in terms of Analysability, modifiability, stability and reusability.
- Portability: refers to the ability of the software to be relocated to a different environment, measured by adaptability, installability and co-existence with other software or systems.
- Security: refers to the level of protection and privacy that the software can offer, including data protection and user protection. Also refers to the ability of software to ensure the authenticity of data or user.

These desired aspects of the software must be structured into Non-functional requirements and by design be covered by the ICOS Ecosystem as well as functional requirements that describe the ICOS Ecosystem itself.

Table 15 General Non-Functional Requirements

ID	Requirement Name	Main ICOS component	Thresholds	Validation
NFR_1	Number of supported Agents from a single controller (for a specific baseline resource configuration)	ICOS Agent ICOS Controller	<100	Will be evaluated during the validation of the UCs
NFR_2	ICOS Node on-boarding time	ICOS Controller ICOS Agent	Less than 5 min	On-boarding of a node, UCs
NFR_3	ICOS Lighthouse maximum supported controllers	ICOS Lighthouse	<10	Simulation test and evaluation from the UC implementation
NFR_4	Minimum observability metrics update time	ICOS Telemetry	1 minute for topology metrics 15 seconds for performance metrics	Use Cases, benchmarking in staging environment
NFR_5	Minimum Application deployment time (for a single ICOS controller and max 2 agents)	ICOS Controller ICOS Agent	Less than 1 min	Use Cases 1,2,3,4
NFR_6	Minimum Remediation time	ICOS Controller	Less than a minute	Use Cases 1,2,3,4
NFR_7	Minimum Device Resource Capability for ICOS Orchestration	ICOS Agent	NA	Evaluated during the UC demonstration
NFR_8	Number of Orchestration technologies supported	ICOS Controller ICOS Agent	Two	Use Cases / Projects
NFR_9	Number of CPU architectures supported	ICOS Agent	Two	Use Cases / Projects
NFR_10	Secure ICOS software deployment	All Docker Images that compose the	Trivy scanner reports 0 critical and at most 3 high	Scanner Reports

ID	Requirement Name	Main ICOS component	Thresholds	Validation
		ICOS MetaOS	vulnerabilities	
NFR 11	Source code quality in ICOS Final release	All software part of the ICOS MetOS developed in the ICOS project	Sonarqube analysis for each components has rating "A" for Reliability, Security and Maintainability and 100% of Security Hotspots reviewed	Code quality report
NFR_12	ICOS authentication mechanism is compliant with requirements from the Service Authentication section of the OWASP Application Security Verification Standard (v4.0.3-2.10)	Identity and Access Management	50% of L2 requirements	Secure code review
NFR_13	ICOS authorisation mechanism is compliant with requirements from the General Access Control Desing section of the OWASP Application Security Verification Standard (v4.0.3-4.1.)	Identity and Access Management	50% of L2 requirements	Penetration testing
NFR_14	ICOS monitoring mechanism is compliant with requirements from the Log content section of the OWASP Application Security Verification Standard (v4.0.3-7.1.)	Audit, Telemetry	50% of L2 requirements	Secure code review
NFR_15	ICOS secure (encrypted) communication is compliant with requirements from the Communication chapter of the OWASP Application Security Verification Standard (v4.0.3-9.)	N/A	50% of L2 requirements	Penetration testing, Secure code review

Additional Non-Functional requirements mapped to functional requirements are included in D2.1 [1].

6 Short review of MetaOS architectures

The Table 16 presents a summary of each MetaOS project approach for each of the following topics i.e., orchestration, federation, networking, descriptors, IoT, AIOps, data management, data processing, security, and trust. The data were collected from public information that has been made available by the projects and from consortium members that participate in both projects. As all the projects are now in the second phase and are updating their architecture, design and approaches, the provided information in the table may not be fully aligned with their evolution.

Table 16 Survey of the MetaOS Ecosystem

Topic	ORCHESTRATION
NEPHELE	▸ **Features/Components:** Synergetic MetaOrchestrator, Multi-Cluster Resource Manager, Network Resource Manager ▸ **Technologies:** Kubernetes, helm, Karmada, OpenHorizon, OSM, custom intent-based orchestration
NebulOuS	▸ **Features/Components**: Local & Global layers of orchestration, Resource Management, Digital Twin Orchestration, Placement Optimisation ▸ **Technologies:** K8s for local orchestration
NEMO	▸ **Features/Components:** Meta-orchestrator, Intent-based Migration Controller, Secure Execution Engine ▸ **Technologies:** K8s/K3s/Unikernels, Open Cluster Management (OCM), ArgoCD, Velero, Confidential Containers (CoCo), Hermit unikernels
FluidOS	▸ **Features/Components:** Intent-driven Node Orchestrator and Available Resources database, coordinating service requests and managing resource information ▸ **Technologies:** K8s/k3s/KubeEdge, Microk8s, karmada

aerOS	▸ **Features/Components:** data fabric provisions. Continuum status, encompassing IEs capabilities and availability and hosted services deployment, rooting on NGSI-LD cross registrations (see 5.4.3.3), customized version of TOSCA to express the services to be deployed, novel decomposition in service components following a defined format, aerOS HLO/LLOs corresponding to different container management frameworks, Self-orchestrator module within aerOS self-* toolsuite ▸ **Technologies:** Operators, TOSCA, custom, Kubernetes, Redpanda
ICOS	▸ **Features/Components:** Distributed Meta-Kernel Layer, Resource & Clustering manager ▸ **Technologies:** Nuvla.io, NuvlaEdge, Kubernetes/OKD , Open Cluster Management (OCM), InterPlanetary File System (IPFS)
Topic	**FEDERATION**
NEPHELE	▸ **Features/Components:** Multi-Cluster Resource Manager
NebulOuS	▸ **Features/Components:** Smart Contracts, Brokering
NEMO	▸ **Features/Components:** Federated meta Network Cluster Controller (mNCC), Cybersecure Federated Deep Reinforcement Learning, Federated Resilience Enhanced with Differential Privacy (FREDY), Federated Monitoring & Alerting systems, Monetization and Consensus-based Accountability (smart contracts), Federated Identity and Access Control ▸ **Technologies:** thanos, prometheus, pixie, quorum, flower
FluidOS	▸ **Features/Components:** Synergetic MetaOrchestrator, Multi-Cluster Resource Manager, Network Resource Manager ▸ **Technologies:** Kubernetes, helm, Karmada, OpenHorizon, OSM, custom intent-based orchestration
aerOS	▸ **Features/Components:** FIWARE Orion-LD Context Brokers Own federated version, ▸ Technologies: NGSI-LD
ICOS	▸ **Features/Components:** Multi-cluster management, No federation is offered however ICOS Lighthouse maintains the Controller information and topology to be used by the Users. East - West interfaces for Controllers allows communication for specific operations ▸ Technologies: OCM

Topic	Networking
NEPHELE	▸ **Features/Components:** Network Resource Manager ▸ **Technologies:** NFVCL, Submariner, OSM, VPN
NebulOuS	▸ **Features/Components:** Overlay Network Management ▸ **Technologies:** CNIs, Mesh and eBPF
NEMO	▸ **Features/Components:** Federated meta Network Cluster Controller (mNCC), Link-Layer Secure connectivity for Microservice platforms (L2S-M), ▸ **Technologies:** Application-Layer Traffic Optimization, IETF Network-Slice Controller (NSC), 5G/TSN
FluidOS	▸ **Features/Components:** Liqo network fabric for extending the Kubernetes network model across multiple independent clusters, MicroController Units with integrated sensors are interconnected to a bus (e.g., Modbus, CAN-bus) or via wireless protocols (e.g., Bluetooth, LoRaWAN, WIFI) ▸ Technologies: Virtual kubelet, VPN tunnel setup
aerOS	▸ **Technologies:**5G/TSN, MetalLB, CNI plugins, Cilium, VPN - Wireguard - own, customized version, NFV
ICOS	▸ **Features/Components:** Distributed Meta-Kernel Layer, Continuum Manager ▸ **Technologies:** Submariner, Wireguard, Clusterlink, Eclipse Zenoh, Cillium

Topic	Descriptors
NEPHELE	▸ **Features/Components:** OCI-artifacts, Hyper-distributed Application Graphs -> Custom, (composite)Virtual Objects -> helm-based, WoT, OMA, TOSCA -> extension for intent, Network Slice Blueprints -> Custom ▸ **Technologies:**TOSCA, Helm Charts, OMA
NebulOuS	▸ Features/Components: Semantic Modelling

NEMO	▸ **Features/Components:** ntent-based API/SDK, mNCC, 3GPP Information model definition for Intent – extension for computing workloads ▸ **Technologies:** OpenAPI/YAML, HELM Charts, K8s Manifests, OCM ManifestWorks, OCI
FluidOS	▸ **Technologies:** High-level intent-description languages (e.g., TOSCA)
aerOS	▸ **Features/Components:** K8s Operators and Custom Resources, Helm Charts. TOSCA ▸ **Technologies:** TOSCA, Helm CHarts, Protobuf, Redpanda topics, YAML, NGSI-LD
ICOS	▸ **Features/Components:** ICOS Controller manifest extensions, ICOS application descriptor ▸ **Technologies:**K8s Manifest, OCM, Helm Charts, Docker Compose
Topic	**IoT**
NEPHELE	▸ **Features/Components:** IoT software Stack - VOStack (W3C WoT, OMA L2M2M) ▸ **Testbeds:** NTUA+CNIT testbed ▸ Use Cases: - Emergency/Disaster recovery - AI-assisted Logistics Operations - Energy management - Remote healthcare services
NebulOuS	▸ **Features/Components:** IoT Devices, Multi-cluster, Machine Sensors & Actuators, Drones, Smart Lampposts ▸ **Testbeds:** Pilot Sites ▸ Use Cases: - Windmill Maintenance - City Maintenance - Precision Agriculture - Fresh Food Supply - Crisis

NEMO	▸ **Features/Components:** Large set of IoT resources including mobile and wireless IoT devices equipped with various sensors such as ambient light, temperature, atmospheric pressure and temperature sensor, tri-axis accelerometer, tri-axis magnetometer, tri-axis gyrometer. Federated NITOS (Network Implementation Testbed using Open Source platform) Lab facility, a SDR (Software Defined Radio) testbed consisting of wireless nodes attached with USRP devices and a SDN testbed equipped with multiple OpenFlow enabled switches. Federated with FIT (Future Internet Testing facilities) ▸ **Testbeds:** OneLAB Testbed (aka NEMO testbed) ▸ Use Cases: - Precision agriculture (bio-spraying) - Smart Grid Flexibility and smart mobility (dispatchable charging of EVs) - Industry 4.0 (Fully automated indoor logistics/supply chain & Human-centred indoor factory environment safety) - Smart City & Smart Media (Round of Athens Race, VR Experience about ancient Workshop, Enhanced AV experience in VR Theatre)
FluidOS	▸ **Features/Components:** IoT environments feature diverse devices like Micro Controller Units (MCUs), Micro Processor Units (MPUs), integrated accelerators, GPUs, Tensor Processing Units (TPUs), ML cores, cryptographic cores, and sensors. ▸ Testbeds: ▸ Use Cases: - Smart Viticulture/agriculture: Use Terraview's climate SaaS platform to exploit data from multiple sources with proprietary AI/ML pipelines to help create intelligence for the practitioners on the ground. - Robotic logistics: Optimize movement processes to improve battery life of factory robots and achieve energy saving and connectivity - Energy grid resilience: Monitoring of the grid state on both transmission and distribution grid, addressing issues related to increasing grid complexity, power production flexibility and new services and operators introduction.
aerOS	▸ **Features/Components:** KubeEdge, Leonardo IoT ▸ Testbeds: Pilots ▸ Use Cases: - AGV robots in manufacturing - Tractors in agriculture - RTGs and STS in maritime ports (logistics), including computer vision - Smart Building - Energy efficiency in edge computing close to Renewable Energy Sources

ICOS	▸ **Features/Components:** IoT devices, Sensors and actuators, Robotics, embedded systems ▸ **Testbeds:** Pilots, Athens Testbed hosting development and integration activities ▸ Use Cases: - 4G/WiFi (ROS based) Agriculture Operational Robotic Platforms - LoRaWAN/4G IoT Nodes/Gateways - 5G Car On board units - WiFi smart energy measurement boards/meters
Topic	**AIOps**
NEPHELE	▸ **Features/Components:** Multi-Cluster Resource Manager, RL-driven autoscaling, Optimal Deployment Mechanisms, AI-assisted orchestration mechanism ▸ Technologies: - VOStack
NebulOuS	▸ **Features/Components:** Proactive reconfiguration, Anomaly Detection, Self-adaptive configuration ▸ Technologies:- ML Algorithms
NEMO	▸ **Features/Components:** Cybersecure Federated Deep Reinforcement Learning, Federated Resilience Enhanced with Differential Privacy (FREDY), Generative Adversarial Network attacks (GAN), Intrusion Detection System (IDS) ▸ **Technologies:** Flower extensions, FREDY, EVEREST, Pytorch, Tensorflow, Keras
FluidOS	▸ **Features/Components:** MEC Federated learning , AI/ML models for privacy preserving AI-assisted orchestration, energy demand prediction of future edge computing tasks, and node energy efficiency optimization. ▸ Technologies: -
aerOS	▸ Features/Components: - ▸ Technologies: Flower, AsyncAPI, Flask, OpenFaaS

ICOS	▸ Features/Components: Intelligence Layer, Intelligence Layer Coordination ▸ Technologies: FastAPI, Flask, - AI Analytics (Tensorflow, Pytorch, Scikit-learn, MXNet), AI Models Repository (MLFlow, HuggingFace Hub)
Topic	**Data Management**
NEPHELE	▸ **Features/Components:** Monitoring and Observability ▸ **Technologies:**Thanos, opentelemetry, flutentd, logstash, Software stack (VOstack) that support distributed data management
NebulOuS	▸ **Features/Components:** DevOps Support, Application and Data Life-Cycle Management ▸ Technologies: - Blockchain (smart contract encapsulator)
NEMO	▸ **Features/Components:** Monetization and Consensus-based Accountability (MOCA), PRESS & Policy Enforcement Framework, Intercommunication Module ▸ **Technologies:** IPFS, DApps, RabbitMQ, SmartContracts, quorum, CEPH, thanos, prometheus, pixie, Fiware Context Broker
FluidOS	▸ **Features/Components:** Storage Fabrics support ▸ Technologies: Liqo
aerOS	▸ **Features/Components:** Orion-LD, Data Fabric of aerOS ▸ **Technologies:** Protegé, Morph-kgc, RDF, OpenLDAP
ICOS	▸ **Features/Components:** Data Management Layer ▸ **Technologies:** DataClay , Eclipse Zenoh
Topic	**Data Processing**
NEPHELE	▸ **Features/Components:** Monitoring and Observability ▸ **Technologies: T**hanos, open telemetry, flutentd, logstash
NebulOuS	▸ **Features/Components:** Intelligent applications data streams ▸ Technologies:- Apache kafka

NEMO	▸ **Features/Components:** Cybersecure Federated Deep Reinforcement Learning, Intercommunication Module ▸ Technologies:- Federated and Deep Reinforcement learning Algorithm
FluidOS	▸ **Features/Components:** Data Gravity approach ▸ Technologies: - Liqo storage fabrics subsytem
aerOS	▸ **Features/Components:** Kafka and Redpanda (messaging), OpenFaaS for serverless analytics (including own templates by aerOS) ▸ Technologies:- Apache kafka, OpenFaas
ICOS	▸ **Features/Components:** Intelligence Layer, Matchmaking component processing workload mapping to appropriate node ▸ **Technologies:** - Dataclay, Distributed & Parallel Execution (COMPSS)
Topic	**Security**
NEPHELE	▸ **Features/Components:** Identity and Access Management ▸ **Technologies:** Keycloak, SIOPv2, DIDComm channel, OIDC4VP,PEP, PDP, PAP/PMP, and PIP
NebulOuS	▸ **Features/Components:** Authentication, Authorization, ▸ **Technologies:**ABAC, SDN/NFV, VPN
NEMO	▸ **Features/Components:** Identity Management, Access Control Management and Intercommunication Security, Cloud-Native Application Protection Platform (CNAPPs), Cybersecure Microservices' Digital Twins, Secure Execution Engine ▸ **Technologies:** Keycloak, RabbitMQ, Falco, Checkov, Grype, Syft, Kong, NGINX, Rust, Unikernels, Nerves, Elixir
FluidOS	▸ **Features/Components:** - Identity Management, Anomaly detection and security orchestration ▸ Technologies:- AI/ML Algorithms, SDN/NFV
aerOS	▸ **Features/Components:** IdM, Auth and Access Management ▸ **Technologies:** Keycloak, KrakenD, OpenLDAP, eBPF(Cilium), DevPrivSecOps (SAST, DAST)

ICOS	▶ Features/Components: Security Layer, ▶ Technologies: - Identity and Access Management (Keycloak) - Anomaly detection (LOMOS) - Security Scan (Wazuh, Trivy, IaC Scan Runner) - eBPF (Cilium)
Topic	**Trust**
NEPHELE	▶ **Features/Components:** Multi-Cluster Resource Manager, Cybersecurity & Trust Management Mechanisms in proposed synergetic orchestration mechanism ▶ **Technologies:** SIOPv2, DIDComm channel, OIDC4VP,PEP, PDP, PAP/PMP, and PIP
NebulOuS	▶ **Features/Components:** - Security and privacy of communications, Data protection and privacy, Detection and mitigation of threats ▶ Technologies:- AI-driven Anomaly detection Engine, Secure overlay network
NEMO	▶ Features/Components: - (in AIOps) • Cybersecure Federated Deep Reinforcement Learning [attack detection and mitigation in FL/RL, GAN, privacy preserving FL(Private Aggregation of Teacher Ensembles-PATE] - (in Kernel) • PRESS, Safety & Policy Enforcement framework • Plugins & Application Lifecycle Management • ZeroTrust/Observability ▶ **Technologies:** Prometheus, Thanos, pixie, falco, OpenPolicy, Scaphandre, Kepler, Checkov, Grype and Syft
FluidOS	▶ **Features/Components:** Isolation and trusted computing for secure environments to run workloads ▶ Technologies: - TEE

aerOS	▸ **Features/Components:** Trust Score algorithm Calculation, IOTA (key messages exchange) ▸ Technologies: - Distributed Ledger Technology (DLT)
ICOS	▸ Features/Components: Intelligence Layer ▸ Technologies: - Trustworthy AI (Flower, Explainer Dashboard , SHAP) - Grafana observability - eBPF observability (Tetragon)

In summary the following results may be drawn for each topic:

Orchestration:

Most projects utilize Kubernetes (K8s) or related technologies (K3s, Microk8s) for orchestration. NEPHELE, NEMO, FluidOS, aerOS, and ICOS all employ Kubernetes-based solutions to manage resources and orchestrate services across clusters.

Federation:

Projects like NEPHELE, FluidOS, ICOS and NEMO incorporate multi-cluster resource management, although their specific implementations and additional features vary. They all focus on managing resources across distributed environments and ensuring communication and coordination between clusters.

Networking:

Several projects, including NEPHELE, FluidOS, aerOS, and ICOS, use common networking technologies such as Submariner and Cilium. These technologies facilitate network connectivity and management across multiple clusters and environments.

Descriptors:

Projects such as NEPHELE, NEMO, aerOS, and ICOS use descriptors like TOSCA, Helm Charts, and Kubernetes Manifests to define and manage applications and services. These descriptor technologies provide standardized methods for describing, deploying, and managing applications across cloud and edge environments.

IoT:

All projects focus on integrating IoT devices and supporting diverse IoT environments. They use IoT software stacks and testbeds to support applications in various areas showcasing similar approaches in handling IoT integration and management.

AIOps:

In AIOps, projects such as NEPHELE, NebulOuS, NEMO, and ICOS leverage AI/ML models and frameworks for orchestration, anomaly detection, and optimization. These projects emphasize using AI to enhance the management and efficiency of their systems.

Data Management:

Several projects, including NEPHELE, NEMO, and ICOS, focus on monitoring and observability for data management. They employ technologies like Thanos and OpenTelemetry to collect, process, and manage data across distributed environments.

Data Processing:

Projects like NEPHELE, NEMO, and ICOS share approaches in data processing, leveraging monitoring and observability technologies to handle and process data streams. They incorporate AI and ML models to enhance data processing capabilities.

Security:

Security mechanisms are similar across NEPHELE, NEMO, aerOS, and ICOS, which all use identity and access management tools like Keycloak. They also incorporate additional security technologies and frameworks to ensure the protection and integrity of their systems.

Trust:

Projects like NEMO and ICOS focus on establishing trust through AI models, policy enforcement, and observability tools. They use technologies to monitor and manage trust levels within their systems, ensuring reliability and security.

In summary, while each project has its unique features and specific implementations, they share similar approaches in these core topics, utilizing common technologies and methodologies to address orchestration, federation, networking, descriptors, IoT, AIOps, data management, data processing, security, and trust.

7 Technology selection and justification

This section summarises the technology selection for the implementation of ICOS components and Cloud Continuum environment. For each thematic area the decisions along with the justification for the selection are presented.

7.1 Application Description frameworks, descriptors and definitions

According to Statista, Kubernetes and Docker Compose are the most popular containerization technologies respectively holding 75% and 5% of the market share. To facilitate the adoption of ICOS to those developers and service providers already using containerization technologies, the project has opted for leveraging the application descriptor models typically used for application distribution for these containerization technologies. ICOS proposes an application descriptor model that allows the combination of Kubernetes and docker-compose manifests. Another widely used model to describe Kubernetes deployments requiring parametrized values is Helm; ICOS may also integrate Helm Charts as part of its model describe components of its applications.

7.2 Cloud/Edge/IoT Orchestration and Management

Regarding Container Orchestration and infrastructure management technologies, ICOS has selected two frameworks for the following reasons:

1. Nuvla
   - offers an integrated platform for comprehensively managing edge, IoT, and cloud infrastructures seamlessly;
   - supports scalable and adaptable deployments, minimizing human intervention and allowing dynamic management of resources;
   - offers extensive API access, enabling seamless integration with other systems and custom workflows, and supports the deployment on Docker and Kubernetes;
   - includes robust security features with federated identity management, authorization, ensuring data integrity and trustworthiness across the infrastructure;
   - it is a European-developed Open Source software ensuring transparency, adaptability, and community-driven enhancements and promoting interoperability and innovation
2. Open Cluster Management (OCM)
   - follows agent-based (hub-spoke) model, allowing to administrate heterogeneous agents abstracting underlying physical or virtual technology details;
   - Despite the abstraction of this heterogeneity, it provides features to manage the resources under its control and reports enough information about their status to allow ICOS to manage them;
   - enables ICOS to deploy Kubernetes Engine and Helm based applications along the entire continuum (cloud-edge-IoT);
   - is CNCF project

Following technologies have been selected as the core of the ICOS Logging and Telemetry:

- Data Collection and Protocols: OpenTelemetry Collector
  - covers not only metrics but also logging and tracing providing a full observability framework;
  - great interoperability with other existing monitoring and logging tools;
  - powerful and flexible processing capabilities that matches the ICOS needs;
  - CNCF project;

- Monitoring Data Storage: Thanos
  - matches very well the ICOS architecture supporting distributed storages support high availability and provides mechanisms to
  - includes advanced features out-of-the-box needed by ICOS such as automatic compaction, rule engine and metrics backups;
  - extremely flexible deployment and configuration;
  - CNCF project;
- Logs Storage: OpenSearch:
  - existing integration with other tools selected for ICOS like OpenTelemetry and LOMOS;
  - flexible and scalable deployment options;
  - Open Source and plugin-based to allow easy adaptation and extensibility to match the ICOS needs;

7.3 Inter-cluster Network Connection Services

The technology that ICOS uses to connect applications across the multi-cloud is ClusterLink (https://clusterlink.net/). The following features distinguish ClusterLink from other similar technologies.

- Use native kubernetes constructs (e.g., Service) to manage traffic; seamlessly works with any kubernetes distribution and any CNI.
- Share at the kubernetes service abstraction; less information to share between clusters.
- Explicit sharing; each cluster maintains control over its sharing (which services it exposes and to whom).
- Every connection is subject to policy check.
- Explicit NetAdmin and DevOps personas.
- Flow based (as opposed to packet based) handling of traffic.
- 1:1 mapping between LAN and WAN sockets - no multiplexing and tunnelling.
- Application centric management model.

7.4 Data Processing and Management and Intelligence

The following technologies have been selected for providing Data Management in ICOS across the continuum:

1. Distributed Object Store: dataClay (https://dataclay.bsc.es/)
   a. Justification:
      i. Distributed design, suitable for cloud-to-edge environments with distributed access.
      ii. Novel data locality mechanisms which can be leveraged from ICOS Components (e.g., from the Intelligence Layer) allowing task offloading and more efficient resource utilization.
      iii. Flexible data model, supporting complex data structures with multiple concurrent readers/writers
      iv. Proved solution in continuum (used in previous continuum EU projects: CLASS, ELASTIC, mF2C)
2. Eclipse Zenoh (https://github.com/eclipse-zenoh/zenoh)
   a. Justification:
      i. Communication protocol that unifies different communication patterns such as pub/sub, storage/query and supports computations.

ii. It supports geo-distributed storage based on data-named networking (i.e., resource/key) enabling location transparency.
iii. Flexible network topology, it supports peer-to-peer, brokered and routed communications that can span from the cloud to the microcontroller
iv. It has several connection plugins, enabling connection with other communication protocols, such as: MQTT, DDS, REST.
v. It has a ROS/ROS2 bridge, enabling robotics applications to seamlessly connect to a cloud-server or with other robots (swarm mode).

For the Intelligence Layer, models, model pipelines, and part of the solution have been developed during the project. As a ground base for them, we have used the following libraries and frameworks.

1. Intelligence coordination:

- BentoML (https://www.bentoml.com/)
  - a. Justification:
    - i. Inbuilt SwaggerAPI
    - ii. Asynchronous model inference calls.
    - iii. Model registry inbuilt that saves models together with any artifacts (e.g., standardisers and metadata).
    - iv. Enabled integration with Prometheus to output Intelligence logs.
    - v. Enabled integration with MLFlow for AIOps.
    - vi. Inbuilt mechanism for containerisation of intelligence environments.
- MLFlow (https://mlflow.org/)
  - a. Justification:
    - i. It integrates well with BentoML.
    - ii. It enables experiment tracking and logging model results and parameters.
    - iii. New versions provide model explainability for experiments run.
- Jupyter Hub (https://jupyter.org/hub)
  - a. Justification:
    - i. Web interface for users to use the intelligence API environment and models in the registry, or to upload, train, push or pull models.
    - ii. Secure user management that can interact with the IAM security module. Authentication is pluggable, supporting a number of authentication protocols (such as OAuth and GitHub).

2. AI Analytics:

- Tensorflow (https://www.tensorflow.org/) and PyTorch (https://pytorch.org/):
  - a. Justification:
    - i. state-of-the-art deep learning and machine learning frameworks.
    - ii. The use of one or another depends on user preference and for this reason both are provided.
- XGBoost library (https://xgboost.readthedocs.io/en/stable/):
  - a. Justification:
    - i. state-of-the-art ensemble method winning AI competitions.
    - ii. Scalable and easy to use.
- Scikit-learn (https://scikit-learn.org/):
  - a. Justification:
    - i. state-of-the-art Python library for classical machine learning algorithms.
- Statsmodels (https://www.statsmodels.org/):
  - a. Justification:
    - i. timeseries models used to fit univariate numerical data streams on the go.

3. AI Trustworthiness:

- Flower (https://github.com/adap/flower):
    a. Justification:
        i. Active and well documented framework for Privacy-Aware Machine Learning and Federated Learning.
        ii. It provides local model registries to connect with global registries at the ICOS controllers.
- SHAP (https://shap.readthedocs.io):
    a. Justification:
        i. Active and well documented project for model explainability.
        ii. It integrates with new versions of MLFlow.
- NannyML (https://www.nannyml.com/):
    a. Justification:
        i. Active and well documented project monitoring machine learning pipelines to have trustable models and robust pipelines.
        ii. It provides alarms when the model performance degrades over time.

4. Data Preprocessing:

- Pandas (https://pandas.pydata.org/):
    a. Justification:
        i. Commonly used Python library for simple data preprocessing.
- Scikit-learn (also in AI Analytics):
    a. Justification:
        i. Standard Python library for data scaling and data splits.
- dataClay client to connect to the ICOS data management layer.
    a. Justification:
        i. Integration of these tools is required by ICOS so Intelligence communicates to data management.

For supporting parallel and distributed execution of general-purpose data processing operations, other than Artificial Intelligence, we have selected COMPSs (https://compss-doc.readthedocs.io/) for the following reasons:

a. Justification:
    i. It parallelizes and distributes the workload of either batch jobs or service request
    ii. It seamlessly integrates the use of traditional task-based workflows with stream-based dataflows
    iii. It is prepared for a dynamically changing infrastructures
    iv. It is a European-developed Open-Source project
    v. It is successfully used in several H2020 and Horizon Europe projects (NextGenIO, MUG, Expertise, mF2C, CLASS, TANGO, AI-Sprint, CAELESTIS, DT-GEO)

7.5 Security and Trust

Technologies selected for implementation of Security and Trust:

1. Security Coordination API: FastAPI (https://github.com/tiangolo/fastapi)
   a. Justification:
      i. Fast: Very high performance, on par with NodeJS and Go. One of the fastest Python frameworks available.
      ii. Intuitive and easy to learn
      iii. Standards-based: Based on (and fully compatible with) the open standards for APIs: OpenAPI (previously known as Swagger) and JSON Schema.
      iv.
2. Anomaly detection: Log Monitoring System (LOMOS).
   a. Justification:
      i. robust AI technology (NLP) and methodology for anomaly detection on logs, tailored to adapt to new data sensitivity concerns.
      ii. designed to detect security-related events and incidents within the deployed application environment and is deployable automatically, providing users with timely notifications about security episodes.
      iii. in-depth knowledge of technology as it is successfully used in several H2020 and Horizon Europe projects (FISHY, SUNRISE, CYLCOMED, TANGO)
3. Identity and Access Management: Keycloak (https://www.keycloak.org/):
   a. Justification:
      i. Uses modern, state-of-the-art protocols (e.g., OpenID Connect) with lightweight and modern web technologies like REST, JWT and JSON,
      ii. supports multiple authentication methods and authorization flows that can adapt to different use cases and security requirements
      iii. management of authentication and authorization is centralized, and services never directly access the user's credentials (improving the overall security of the system)
4. Security Scan: Wazuh (https://wazuh.com/), Trivy (https://trivy.dev/) and IaC Scan Runner (https://github.com/xlab-si/iac-scan-runner)
   a. Justification for Wazuh:
      i. Modern and unified SIEM and XDR with multiple functionalities for endpoints and cloud environment (detection, incident response, compliance checks etc.)
      ii. Lightweight and easy to use
      iii. In-depth knowledge of technology as it was used in several H2020 projects (FISHY, MEDINA, PIACERE)
   b. Justification for Trivy:
      i. Powerful container and code scanner (vulnerabilities in containers, IaC misconfigurations)
      ii. Reliable, fast and easy to use
   c. Justification for IaC Scan Runner:
      i. Integrates a wide variety of scanners for most popular and used IaC (Ansible, Terraform, Yaml, Python...)
      ii. in-depth knowledge of technology as it is successfully the H2020 project PIACERE

5. Audit: Tetragon (https://tetragon.io/):
    a. Justification:
        i. a flexible Kubernetes-aware security observability and runtime enforcement tool that applies policy and filtering directly with eBPF, allowing for reduced observation overhead, tracking of any process, and real-time enforcement of policies
        ii. Use of modern eBPF (extended Berkeley Packet Filter) technology for observation and tracing
6. Trust and privacy: Cilium (https://cilium.io/) and Wireguard (https://www.wireguard.com/)
    a. Justification for Cilium:
        i. an open source, cloud native solution for providing, securing, and observing network connectivity between workloads, fuelled by the eBPF (extended Berkeley Packet Filter)
        ii. Can also be used for the enforcement of the security policies in Kubernetes
        iii. Uses mTLS for encrypting network traffic
    b. Justification for Wireguard:
        i. simple yet fast and modern VPN that utilizes state-of-the-art cryptography (Noise protocol framework, Curve25519, ChaCha20, Poly1305, BLAKE2, SipHash24, HKDF)
        ii. Simple and easy to use
        iii. High performance (a combination of extremely high-speed cryptographic primitives and the fact that WireGuard lives inside the Linux kernel means that secure networking can be very high-speed)
        iv. Minimal attack surface

8 Conclusions

The deliverable D2.3 "ICOS Ecosystem: Technologies, requirements and state of the art" is the final technical report delivered by the ICOS consortium. It presents the results of the work carried out by the activities T2.1 - Ecosystem Identification: Baseline Technologies; T2.2 - Compute Continuum Requirements Definition and T2.3 - AI, Data Management and Trust/Security Requirements during the first six months of the project.

The document updates the concepts of ICOS as those have changed during the first iteration of ICOS implementation. Some minor updates may be expected by D2.4 that finalises the ICOS architecture.

New descriptions for the User Stories are provided with minor changes that occurred while maturing of the ICOS concepts prevailed during the ICOS MetaOS implementation as well as due to design evolution.

Furthermore, D2.3 has provided a final high level description of the four Use Cases will assess and validate ICOS technologies:

- UC1: Agriculture Operational Robotic Platform;
- UC2: Railway Structural Alert Monitoring system;
- UC3: In-car Advanced Infotainment and Multimedia Management system;
- UC4: Energy Management and Decision Support system)

In addition, the first five projects awarded from the Open Call, are being summarised.

ICOS Requirements (Functional and Non-Functional) Elicitation has been as the result of analysis and a series of activities carried out within the different actors and stakeholders of the project's value chain.

This analysis has been carried out from five different points of view, which are considered to be the main themes of the project, in order to produce the functional requirements:

- Continuum Creation:
- Continuum Management:
- Data Resiliency and Transformation:
- Smart Security and Trust:
- Operability Serviceability (GUI/CLI)

In order to provide a set of requirements, the User Stories and the validation Use Cases were studied and the results from mentioned analysis are detailed in a set of tables. This output is produced by following the MoSCoW method in order to reduce the requirements elicitation extension.

The deliverable provides a survey of the MetaOS ecosystem, through the analysis of the 6 sister projects that are contemporary being developed. The survey attempts to highlight commonalities and differences in the approach towards the development of MetaOS for the Cloud Continuum.

Finally, this deliverable presents the technology selection and design decisions that were made during the IT-1 of the ICOS development cycle. The selection provides a summary of the features and a justification for its selection.

9 References

[1] ICOS. D2.1 – "ICOS ecosystem: Technologies, requirements, and state of the art", D'Andria, Francesco. 2023 (https://www.icos-project.eu/deliverables)

[2] ICOS. D2.2 – "ICOS Architecture Design (IT-1)", Giammatteo, Gabriele. 2023 (https://www.icos-project.eu/deliverables)

[3] ICOS. D6.4 – "Use Cases settings and demonstration strategy", Zrazinska, Izabela. 2023. (https://www.icos-project.eu/deliverables)