Project Title	Artificial Intelligence in Secure PRIvacy-preserving computing coNTinuum
Project Acronym	AI-SPRINT
Project Number	101016577
Type of project	RIA - Research and Innovation action
Topics	ICT-40-2020 - Cloud Computing: towards a smart cloud computing continuum (RIA)
Starting date of Project	01 January 2021
Duration of the project	36 months
Website	www.ai-sprint-project.eu/

D1.2 - Requirements Analysis

Work Package	WP1 | Requirements & Architecture Definition
Task	T1.2 | Use Cases and Framework Requirements Analysis
Lead author	Enrico Abate (BECK)
Contributors	Enrico Abate-Daga, Andrei Popa (BECK), Michal Klosinski (AF), Davide Cirillo, Daniele Lezzi (BSC), Patrick Thiem (C&H), Danilo Ardagna, Matteo Matteucci (POLIMI), André Martin (TUD), German Moltó (UPV), Grzegorz Timoszuk (7BULLS)
Peer reviewers	Matteo Matteucci (POLIMI), Amanda Calatrava (UPV)
Version	V1.0
Due Date	30/06/2021
Submission Date	30/06/2021

Dissemination Level

X	PU: Public
	CO: Confidential, only for members of the consortium (including the Commission)
	EU-RES. Classified Information: RESTREINT UE (Commission Decision 2005/444/EC)
	EU-CON. Classified Information: CONFIDENTIEL UE (Commission Decision 2005/444/EC)
	EU-SEC. Classified Information: SECRET UE (Commission Decision 2005/444/EC)

AI-Sprint - Artificial Intelligence in Secure PRIvacy-preserving computing coNTinuum, has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement no. 101016577.

Versioning History

Revision	Date	Editors	Comments
0.10	24/02/2021	Andrei Popa	Initial draft
0.50	29/04/2021	Andrei Popa	Version 1.0 provided for review
0.51	07/05/2021	Danilo Ardagna, Matteo Matteucci	First Review+ Comments
0.52	14/05/2021	Enrico Abate-Daga, Andrei Popa	Document revision
0.53	20/05/2021	Germán Moltó, Miguel Caballer	Contributions to section 5.3
0.54	14/06/2021	Enrico Abate-Daga, Andrei Popa	Completion of section 4.3
0.55	16/06/2021	Andrei Popa	Adding Category field to Requirements, added section 6.1.1, 1.1, contributed to section 1.4
0.56	21/06/2021	Davide Cirillo, Michal Klosinski	Completion of sections 4.1, 4.2
0.57	23/06/2021	Andrei Popa, Daniele Lezzi, Patrick Thiem, André Martin, Grzegorz Timoszuk	Contributed to sections 4.3.2, 2.1, 5.3.9, 5.4, 5.3.5
0.58	24/06/2021	Enrico Abate-Daga	Completed sections 4.5.3, 4.3.6, 4.3.7
0.59	24/06/2021	Matteo Matteucci	Reviewed: 1.x, 2.x
0.60	25/06/2021	Matteo Matteucci	Reviewed: 3.x
0.61	26/06/2021	Matteo Matteucci	Reviewed 4.x, 5.1, 5.2, 5.3, 5.4, 6.x Written 5.2.3 5.3.6, 5.3.8,
0.62	26/06/2021	Amanda Calatrava	Reviewed: 1.x, 2.x, 3.x, 4.1.x, 4.2.x
0.63	27/06/2021	Amanda Calatrava	Reviewed: 4.3.x, 5.1, 5.2, 5.3.1-5.3.4, 6.1
0.64	27/06/2021	Enrico Abate-Daga	Added Table of Content, List of Tables, List of Images, List of Abbreviations
0.65	28/06/2021	Andrei Popa	Added Section 7: Conclusions
0.66	28/06/2021	Danilo Ardagna	Reviewing the whole document
0.67	29/06/2021	Andrei Popa	Review and accept suggestions. Contributed to section 2.1.5 + Abbreviations
0.68	29/06/2021	Amanda Calatrava	Revision of the whole document
1.00	29/06/2021	Enrico Abate-Daga	Final formatting

Keywords

Cloud Computing; Cloud Infrastructure; Artificial Intelligence; Edge Computing; Computing Continuum; Software Design & Development; Cloud Trust & Security; Privacy;

Disclaimer

This document contains confidential information in the form of the AI-SPRINT project findings, work and products and its use is strictly regulated by the AI-SPRINT Consortium Agreement and by Contract no. 101016577.

Neither the AI-SPRINT Consortium nor any of its officers, employees or agents shall be responsible, liable in negligence, or otherwise however in respect of any inaccuracy or omission herein.

The contents of this document are the sole responsibility of the AI-SPRINT consortium and can in no way be taken to reflect the views of the European Commission and the REA.

Executive Summary

D1.2 summarizes the initial requirements analysis of the AI-SPRINT project. The outcome of this analysis is a list of requirements for all tools which will be evolved or developed within the project. This analysis will guide all the AI-SPRINT technical activities.

The aim of AI-SPRINT (**A**rtificial **I**ntelligence in **S**ecure **PRI**vacy-preserving computing co**NT**inuum) is to define a novel framework for developing and operating AI applications, together with their data, exploiting computing continuum environments, which include resources from the edge up to the cloud. AI-SPRINT will offer novel tools for AI applications development, secure execution, easy deployment, as well as runtime management and optimization. AI-SPRINT tools will allow trading-off application performance (in terms of end-to-end latency or throughput), energy efficiency, and AI models accuracy while providing security and privacy guarantees. AI-SPRINT framework will support AI applications data protection, architecture enhancement, agile delivery, runtime optimization, and continuous adaptation.

A critical success factor in the development of high quality, state of the art, software frameworks is a deep understanding of the requirements of users of the framework (e.g. developers, AI experts, service providers etc.) and of end-users of the application and services created using the framework (e.g. farmers, stroke patience, maintenance engineers etc.). The elicitation of the AI-SPINT requirements started with the description of Personas, i.e. fictional images of typical AI-SPRINT users with their specific interests, experiences and demands. User Stories and Requirements were then formulated by looking at the AI-SPRINT from the perspective of the selected Personas. The providers of the three real-life Use Cases wrote the first draft of the requirements, the tool providers reviewed and enhanced these requirements from a more technical point of view, also keeping in mind current state of the art solutions (see deliverable D1.1).

The requirements analysis helped to outline the importance of some critical AI-SPRINT features. The ability to develop cloud/edge agnostic applications and to define at runtime which component will run on the edge, on the cloud or on a hybrid-cloud setup will be a pivotal strength of the tools. The excellent security features of the framework will allow its usage also in security and privacy critical domains (e.g., Personalized Healthcare). The federated training of AI models will also prove to be extremely important in areas in which data cannot be easily shared among the different parties involved in the project.

The requirements analysis also showed that the proposed reference architecture (see deliverable D1.3) can be used for all three Use Cases, with minor specific modifications. The proposed project assets (SCAR, PyCOMPs, Krake, SCONE etc.) present a very good starting point for the development of the framework; the tools need to be adapted and especially integrated to form a coherent, easy to use framework.

It must be mentioned that the requirements analysis process is intended to be a continuous process, hence this report does not coincide with the end of requirement analysis. A requirement repository was defined and will be continuously updated until month 24.

Table of Contents

1. Introduction 9
1.1 AI-SPRINT Project Scope and Vision 9
1.2 Scope of the document 9
1.3 Target audience 9
1.4 Structure of Document 9
2. Requirements Analysis Approach 10
2.1 Methodology 10
2.1.1 Step 1: Use Case Overview and Application Architecture 11
2.1.2 Step 2: Epics 12
2.1.3 Step 3: User Stories 12
2.1.4 Step 4: Requirements 13
2.1.5 Step 5: Consolidation and requirements for AI-SPRINT Tools 14
3. Personas 15
3.1 Lucía the Application Architect 15
3.2 Paolo the Application Developer 15
3.3 Christine the AI-Expert 16
3.4 Piotr the Application Manager 16
3.5 Loredana the Infrastructure Provider & Sysops 17
3.6 Alexandre the Application and Service Provider 17
3.7 Application End-user (Personalized Healthcare) 18
3.7.1 Henrik the stroke patient 18
3.7.2 Xavier the manager at the stroke foundation 18
3.7.3 Claudia the biomedical engineer of the smartband company 19
3.7.4 Jeanne the advisor neurologist 19
3.8 Application End-user (Maintenance and Inspection) 19
3.8.1 Ana the drone operator 19
3.8.2 Whit the inspection analyst 19
3.9 Application End-user (Farming) 19
3.9.1 Chiara the Factory Owner 19
3.9.2 Zeno the Factory Factotum 20
4. Requirements Definition 21
4.1 Use Case 1: Personalized Healthcare 21
4.1.1 Use Case overview 21
4.1.2 High level Architecture 23
4.1.3 Epics 24
4.1.4 User stories 24
4.1.5 Requirements 27
4.1.6 Summary and Overview of Requirements 34

4.1.7 Initial Roadmap 35
4.2 Use Case 2: Maintenance and Inspection 35
4.2.1 Use Case overview 35
4.2.2 High level Architecture 37
4.2.3 Epics 37
4.2.4 User stories 38
4.2.5 Requirements 40
4.2.6 Summary and Overview of Requirements 44
4.2.7 Initial Roadmap 45
4.3 Use Case 3: Farming 4.0 45
4.3.1 Use Case overview 45
4.3.2 High level Architecture 48
4.3.3 Epics 49
4.3.4 User stories 50
4.3.5 Requirements 63
4.3.6 Summary and Overview of Requirements 68
4.3.7 Initial Roadmap 70
5. AI-SPRINT Assets Technical Requirements 71
5.1 Overview of assets 71
5.2 Design time tools 71
5.2.1 Design and Programming Abstractions 72
5.2.2 Performance Models 73
5.2.3 AI Models Network Architecture Search 74
5.2.4 Application Design Space Exploration 75
5.3 Runtime tools 75
5.3.1 Infrastructure Manager 77
5.3.2 EC3 79
5.3.3 OSCAR 80
5.3.4 SCAR 81
5.3.5 Monitoring Tools 82
5.3.6 Federated Learning 84
5.3.7 Scheduling for Accelerated Devices 86
5.3.8 Privacy Preserving Continuous Training 87
5.3.9 Programming Framework Runtime 88
5.3.10 Application Reconfiguration 89
5.4 Security tools 89
5.4.1 TEE and Code Attestation 90
5.4.2 Secure Boot 90
5.4.3 Secure Networks 92

6. Summary of Technical Requirements 93
6.1 Use Case Requirements summary 93
6.2 Design Time Tools Development plan 94
6.2.1 Development plan 95
6.3 Runtime Tools Development plan 95
6.3.1 Development plan 95
6.4 Security Tools Development plan 96
6.4.1 Development plan 97
7. Conclusions 98

List of Figures

Figure 2.1 - Methodology 10
Figure 4.1 - High Level Architecture Personalized Healthcare 23
Figure 4.2 - Initial Roadmap Personalized Healthcare 35
Figure 4.3 - High Level Architecture Maintenance and Inspection 37
Figure 4.4 - Initial Roadmap Maintenance and Inspection 45
Figure 4.5 - High Level Architecture Farming 4.0 48
Figure 4.6 - Initial Roadmap Farming 4.0 70

List of Tables

Table 2.1 - Template for Requirements Overview 11
Table 2.2 - Template for User Stories 12
Table 2.3 - Template for Requirements 13
Table 4.1 - Requirements Overview Personalized Healthcare 22
Table 4.2 - List of Requirements Personalized Healthcare 34
Table 4.3 - Requirements Overview Maintenance and Inspection 36
Table 4.4 - List of Requirements Maintenance and Inspection 44
Table 4.5 - Requirements Overview Farming 4.0 47
Table 4.6 - Smart Farming Device Hardware 49
Table 4.7 - Smart Farming Device 3D-Camera 49
Table 4.8 - List of Requirements Farming 4.0 69
Table 6.1 - Tools usage for Use Cases 93
Table 6.2 - Tools usage for Personas 94

1. Introduction

1.1 AI-SPRINT Project Scope and Vision

The AI-SPRINT (Artificial intelligence in Secure PRIvacy-preserving computing coNTinuum) project's main objective is to define a novel framework for developing and operating AI applications, together with their data, exploiting computing continuum environments, which include resources from the edge up to the cloud. The AI-SPRINT Framework will offer a set of tools for application development, easy and secure deployment, execution, management, and optimization without the constraints of being locked in commercial solutions and services from major providers.

1.2 Scope of the document

This document aims to identify, define, analyze, and document all requirements for AI-SPRINT Framework and its associated Use Cases as part of the project's Work Package 1 - Requirements and Architecture Definition. All activities for this deliverable are conducted under the WP1 task T1.2 - Use Cases and Framework Requirements analysis. Requirements are consolidated in the Requirement Repository, where they will be managed and updated after month 6 until month 18 of the project.

This document's output will be used by the following task in WP1 (T1.3- Architecture Definition) and the AI-Framework implementation work packages.

1.3 Target audience

The Requirements Analysis Document is intended for internal use, although it is publicly available. The target audience is the AI-SPRINT project team including all partners involved in the delivery of work packages related to the three Use Cases proposed.

As the AI-SPRINT project will provide a framework to build real-life AI-based applications running somewhere in the computing continuum, this document will be used by the AI-SPRINT partners building the framework to elicit and document also the requirements of their tools.

1.4 Structure of Document

This deliverable is composed by five main Chapters described as follows:

1. **Introduction**: This section describes the scope of the document and target audience
2. **Requirements Analysis Approach** defines the method for the identification and documentation of the project's requirements, and the definition of AI-SPRINT Personas that will be considered as the main AI-SPRINT end users. Personas are fictional images of typical AI-SPRINT users with their specific interests, experiences and demands.
3. **Requirements Definition** is the main section of the document that contains the Use Cases description, their high-level architecture design and the requirements extracted from Epics and User-stories with a summary and initial roadmap.
4. **The AI-SPRINT Assets Technical Requirements** section contains requirements from the AI-SPRINT framework tools/assets perspective and are split into the three main categories of Design, Runtime and Security.
5. **Summary of Technical Requirements** section contains a summary from a tools' perspective and a development plan for each main category (Design, Runtime and Security)
6. **Conclusions** section summarizes all elicited requirements and the deliverable achievements.

2. Requirements Analysis Approach

2.1 Methodology

The requirements for AI-SPRINT are formulated by focusing on personas and looking at the framework from different points of view: the point of view of the future users of the framework, but also the point of view of users of applications developed through AI-SPRINT.

The future users of the framework are developers, AI-experts, application managers etc. like the ones who analyzed three real-life applications (Use Cases) and formulated the framework requirements by abstracting the requirements of these three specific applications. Using this methodology, the requirements elicitation process will take place by following five steps:

- Step 1: Present the Use Case and the architecture of the underlying application
- Step 2: Present the Epics of the application
- Step 3: Present the User Stories resulting from the Epics. The User Stories are always presented using the point of view of future users (Personas)
- Step 4: Formulate the Requirements which results from the User Stories
- Step 5: Define which AI-SPRINT asset is affected by which Requirement

The methodology followed is illustrated below

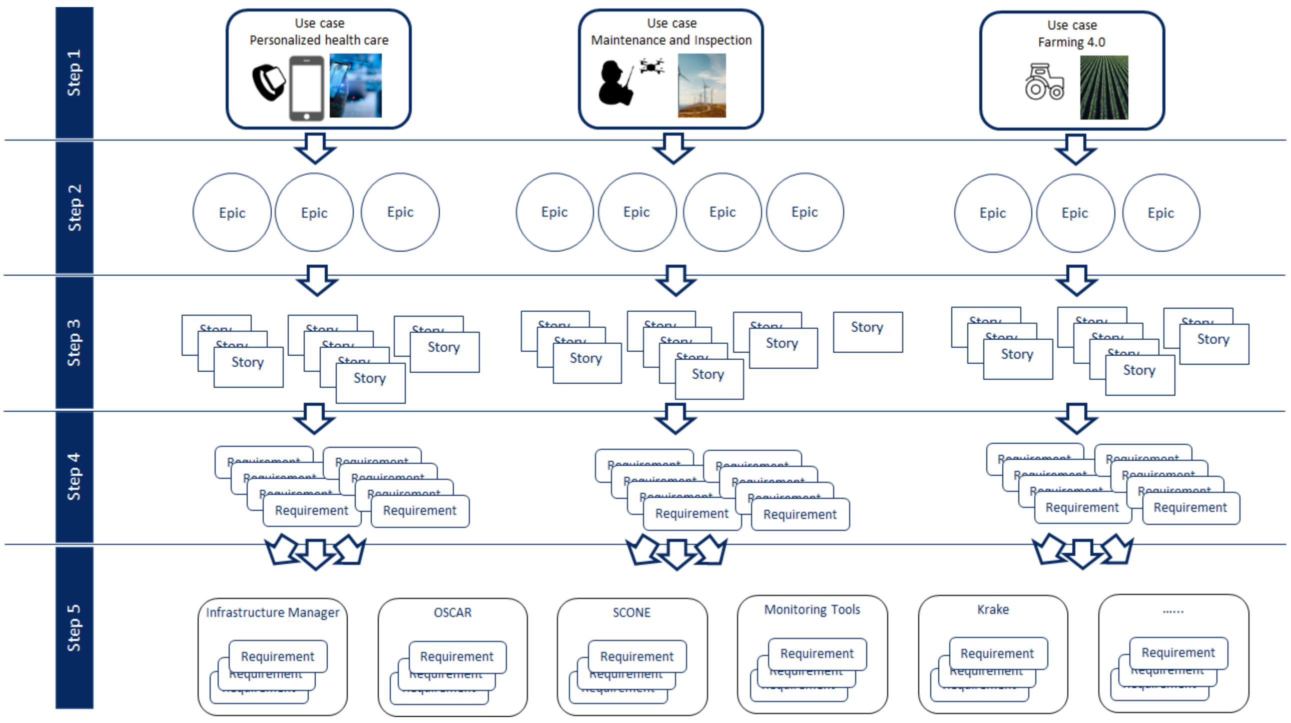

Figure 2.1 - Methodology

The AI-SPRINT Framework Assets list and the initial architecture are defined and described in detail as part of the AI-SPRINT Deliverable *D1.3 Initial Architecture Design*.

In the next subsections, all steps depicted in Figure 2.1 are explained.

2.1.1 Step 1: Use Case Overview and Application Architecture

The first step of the requirements elicitation process is to analyze the Use Case and provide a first high-level draft of the application architecture. The application architecture is derived from the AI-SPRINT reference architecture, Use Case specific elements are added, irrelevant elements are removed. In this step, each Use Case is already assessed against high level requirements as presented in the proposal document with the following table. This will indicate the weight and emphasis on each AI-SPRINT asset component for the specific Use Case.

	Data Type	Advanced AI features	Development Tools	Deployment Tools	Infrastructure Elasticity	Security	Privacy
Use Case Name	Type of data	++	++	++	++	++	++

+ = nice to have; ++ = required

Table 2.1 - Template for Requirements Overview

Explanations of table fields

Data type: The main type of data processed. For example, for the health-care UC, consists of time-dependent sensor data from a wearable device and additional data based on a lifestyle questionnaire.

Advanced AI features: Need for advanced machine learning and AI features. For example, in the healthcare UC, the development of a stroke risk assessment model integrating sensor and lifestyle information requires advanced machine learning approaches. The expected outcome of such a model is a fine stratification of subjects based on their individual characteristics.

Development Tools: The needs of the Use Case development activities are covered by several tools including commonly used programming languages (Python), data analytics and visualization tools (Jupyter notebooks), machine learning frameworks (Scikit-learn, PyTorch).

Deployment Tools: Deployment tools that are adequate for the Use Case. Standard tools include GitLab as well as Docker and Kubernetes. The use of Infrastructure Manager and of FaaS abstractions for edge deployment (through OSCAR) is also foreseen.

Infrastructure Elasticity: Given the target application the Use Case may require resources to be dynamically adapted to workload changes.

Security: The sensitive data in the Use Case mandates high security requirements that can be covered by SCONE tools.

Privacy: The definition of high privacy standards for the Use Case is mandatory due to the sensitive nature of the data processed and modelled.

2.1.2 Step 2: Epics

Epics for the three Use Cases are defined and documented. An Epic is a high-level description of a usage situation. Epic descriptions contain:

- A title (e.g. Applications update: cloud and edge components of the applications are updated)
- A unique id with the format <Use Case No.>.E<Epic No.> (e.g. UC3.E01)
- A short description, usually one or two paragraphs

2.1.3 Step 3: User Stories

Epics are broken down in User Stories. In this context, most users are users of AI-SPRINT (e.g. Application Developer, AI-Expert); nevertheless, some User Stories are written from the point of view of the end-users (e.g. Farmer, Maintenance Engineer). User Stories focus on a unique feature or functionality, clearly mention the persona and are described as a one phrase statement: “As a <persona> when <performing an action> I want to <do something> so that <I can achieve some goal>”.

The naming convention for a user story id is <Epic No.>.S<Story No.> (e.g., UC3.E01.S01).

User-stories are documented with additional information using the following template:

Field name	Details
ID*	Give a unique ID for this story
Title*	Give a title/short name for this story
Priority*	One of the following: ● **Must have**: The system (AI-SPRINT framework) must implement this requirement to be accepted. ● **Should have**: The system should implement this requirement: some deviation from the requirement as stated may be acceptable. ● **Nice to have**: The system should implement this requirement, but may be accepted without it.
Persona*	One of the Persona described in Chapter 3
Category*	Design, Runtime, Security
Description*	As a <persona> when <performing an action> I want to <to something> so that <I can achieve some goal>
Need/Purpose*	The objective that the persona wants to achieve through the story
Comment	Any additional explanatory comment

*Mandatory field

Table 2.2 - Template for User Stories

2.1.4 Step 4: Requirements

The User Stories are transformed into **Requirements** for the components of the AI-SPRINT framework. Requirements are organized in four different categories:

- **Design Requirements** are directly linked to the Design time tools Work Package (WP2) of the AI-SPRINT project, which will provision the programming environments, services and tools to enable the development of AI-SPRINT applications by providing a framework to design AI Applications.
- **Runtime Requirements** cover the Runtime environment as part of Work Package 3 (WP3) and will describe the needs for providing a functional solution for the Use Cases including Continuous Deployment (CI/CD), Programing Framework and Application reconfiguration, Monitoring for all components (application, infrastructure and edge), Continuous training, etc.
- **Security Requirements** will be used as input for the Security Tools Work Package (WP4) that aims to define, develop, and implement all security policies, tools, and configurations to ensure Information Security at all layers within the AI-SPRINT framework including but not limited to: Network Security, Application Execution Security, Patch management and Secure Boot.
- **Out-of-Scope Requirements** are those requirements identified during the elicitation process that are not included in the scope of the project, but are relevant in the context of Use Cases implementation and operation.

Naming convention for Requirements is <Use Case No.>.Req<Requirement No.> (e.g. UC3.Req01)

Requirements will be documented based on the following template:

Field name	Details
ID*	Give a unique ID for this requirement
Title*	Give a title/short name for this requirement
Priority*	One of the following: ● **Must have**: The system must implement this requirement to be accepted. ● **Should have**: The system should implement this requirement: some deviation from the requirement as stated may be acceptable. ● **Nice to have**: The system should implement this requirement, but may be accepted without it.
Required for (User Story)*	Could be more than one User Story. For asset requirements this field will map to the Use-Case Requirements instead of Stories
Category*	Design, Runtime, Security
Description*	Specify the intention of the requirement/assumption
Rationale*	Give a justification of the requirement/assumption
Tool	Proposed tool to address the requirement (if known)
Acceptance Criteria	Metrics, objectives required to achieve the expected outcome of the requirement
Supporting materials	Give a pointer to documents that illustrate and explain this requirement or assumption (in particular those of domain analysis)
Tentative scheduling	Tentative scheduling of accomplishment.
Comments	

*Mandatory fields

Table 2.3 - Template for Requirements

2.1.5 *Step 5: Consolidation and requirements for AI-SPRINT Tools*

All requirements from the Use Cases are consolidated, mapped to the AI-SPRINT assets (IM, PyCOMPSs, etc.). In this phase existing requirements may be reformulated in a more detailed way; additional requirements may be added. Requirements for the assets will be mapped to relevant Use-Case requirements.

All Scenarios, user-stories and Requirements will be added to the Requirements repository where they will be consolidated, analyzed and approved. The Requirements Repository will remain open and will be continuously updated until M18 of the project.

3. Personas

In this section we will analyze each one of the identified personas involved in the AI-SPRINT project. Each subsection corresponds to a single persona. However, the last three subsections analyze the personas involved in each Use Case as the application end users.

3.1 Lucía the Application Architect

Lucía is a senior software architect, she has a technical degree and has 8 years of experience, first as developer, then as architect; in the last two years she manages a team of five developers. She has worked in several software development projects, but her focus is in software related to quality assurance and quality control in production. Lucía has a very good understanding of commercial cloud solutions and she understands very well the challenges posed by cloud orchestration and by coexistence of cloud and on-premise infrastructure elements.

Lucía is obsessed with architecture quality and user experience. She is convinced that well architected software is simple to understand and thus to maintain. She also thinks software developers must understand how software performs in real-life and how users work with the software.

Lucía does not consider herself an AI expert or a data scientist, but she has already worked in projects in which AI was critical. Lucía sees herself as a very good DevOps engineer: her team is responsible for the application, but also for the deployment on the underlying infrastructure.

Lucía has always worked with agile software development methods. Her team currently works with Scrum, with weekly sprints.

Lucía’s tasks in projects which use AI-SPRINT are:

- Define and review the overall architecture of the application based on users’ stories
- Identify the critical elements of the application and define KPIs to measure their performance, both in terms of accuracy of AI models and of application execution time
- Review performance of the application and propose technical improvements
- Review users’ stories and help younger colleagues to formulate development tasks

Lucía uses:

- Design time tools
- PyCOMPSs
- Python
- SCONE
- OSCAR

3.2 Paolo the Application Developer

Paolo is an application developer. He completed his master in Computer Science two years ago. So far, Paolo has only worked for one large software project: the goal of the project is to predict the duration of batteries lifetime used by e-autos. His task was to write the software which collects the relevant parameters from the e-autos’ batteries and sends them to a central cloud-based data repository. The application Paolo wrote is relatively simple, but deploying new versions of the application and keeping track of which version is deployed on which car is a nightmare. He is learning about AI-SPRINT in the hope that the framework will make his life easier.

Paolo’s tasks in projects which use AI-SPRINT are:

- Enhance the software and deploy new versions
- Build a data processing pipeline which moves data from the edge into the cloud

Paolo uses:

- Design time tools
- IM
- Monitoring Tools
- SPACE4AI-D

3.3 Christine the AI-Expert

Christine is a data scientist and AI/ML expert. She has a PhD in Mathematics and 15 years of work experience. She has worked several years in insurance companies as an underwriting risk analyst. Four years ago, she switched to the Digital Transformation unit and has worked in several projects whose aim was to use AI to optimize insurance processes (e.g., individualized risk management, fraud detection, automated claim management, etc.).

Christine's focus is AI and ML, not IT. She has a good understanding of software development and cloud technology, but the deployment and management of IT solutions has never been the focus of her work. She needs tools that help her to build, test and continuously improve AI models.

Christine works frequently with sensitive and/or with personal data. She is well aware of all GDPR-related requirements and needs a set of tools that ensure all data she works with is secure all the time. Christine occasionally works with other insurance companies or from institutions which collaborate with her (e.g., hospitals); she is very interested in the federated learning capabilities of AI-SPRINT.

Christine's tasks in projects which use AI-SPRINT are:

- Requirements definition for the AI Framework and Use Cases
- ML/AI model design and implementation

Christine uses:

- Design time tools
- SCONE
- Network Architecture Search (NAS)
- Federated learning
- AI-Frameworks (e.g. PyTorch, TensorFlow)
- Scheduling for Accelerated Devices

3.4 Piotr the Application Manager

Piotr works as IT Application Owner for a Logistics company. Piotr has a degree in Business Administration and Information Technology. He has worked four years as a Business Analyst before becoming IT Application Owner. Piotr manages the application used to track in real time the performance of trucks, he acts as a bridge between end-users and developers. The application he manages collects data related to the performance of the trucks and detects malfunctions in the used equipment. Based on the collected information, trucks' maintenance is planned. In some cases the maintenance has to be performed by a dealer or a technician; in other cases, the maintenance can be performed by upgrading a software component.

Piotr's job is to review the requirements and the quality of the application together with the Business Application Owners and the application key users. Piotr prepares reports about the application performance.

He formulates requests for changes, he oversees the development work and he coordinates the deployment of the application.

Piotr’ tasks in projects which use AI-SPRINT are:

- Requirements definition for the AI Framework and Use Cases
- Application management
- Owner of the CD process
- Monitoring

Piotr uses:

- Monitoring Tools
- IM

3.5 Loredana the Infrastructure Provider & Sysops

Loredana works for a provider of Manufacturing Execution System (MES) applications as a service. She has a degree in Computer Science and has worked first as IT administrator and later as Application administrator. The company she works for developed an MES application used mainly by car-industry suppliers. The software is centrally managed using a cloud application, but for most of the customers it runs on on-premise appliances.

Her job is to make sure the underlying infrastructure works as required and that the availability and performance Service Level Agreement (SLA) is met. The application is maintained using continuous deployment, there are usually a handful of changes deployed every day. Loredana also tracks the deployment status and ensures the correct components are delivered to the correct customers.

Loredana’s tasks in projects which use AI-SPRINT are:

- Requirements definition for the AI Framework and Use Cases
- Setup and Management of the infrastructure runtime environment
- Component migration
- Infrastructure orchestration

Loredana uses:

- Krake
- IM
- Monitoring Tools
- EC3
- SPACE4AI-R

3.6 Alexandre the Application and Service Provider

Alexandre works as Head of Post-Sales in the same company as Loredana (provider as MES Manufacturing Execution System as a service). He has a Bachelor in Business Administration and he is a co-funder of the company. His job is among others to measure customers’ satisfaction and to improve the quality of the MES application based on the customers’ feedback.

Alexandre’s goal is to ensure customers continue using the MES his company provides and that they sign-up for additional (chargeable) features. He checks daily reports about availability of the application, performance and usage of individual modules, usage distribution over time, number of runtime errors, infrastructure costs etc.

Alexandre uses an AI based application which predicts users' behavior, e.g., usage peak times, risk of termination, probability of upselling etc.

Alexandre's tasks in projects which use AI-SPRINT are:

- Understand the application behavior
- Provides Operational requirements for Use Cases (Service Design)

Alexandre uses:

- Monitoring Tools
- SCONE

3.7 Application End-user (Personalized Healthcare)

The following personas do not directly work with the AI-SPRINT framework, but they work with the software created using the framework.

3.7.1 Henrik the stroke patient

Henrik, a marketing executive, had a stroke at age 49 due to an increased risk owing to high blood pressure and high cholesterol levels, which may have been a part of his family history. His father struggled for years with uncontrolled high pressure and his grandfather passed away from a stroke. After spending weeks in a stroke rehabilitation center, Henrik was compelled to get regular blood pressure and cholesterol screenings and change his eating and activity habits. To this aim, Henrik is considering buying a fitness band or smartwatch with a heart rate sensor.

3.7.2 Xavier the manager at the stroke foundation

Xavier is the president of one of the largest stroke foundations in his country. Having suffered a stroke himself, Xavier's biggest motivation is to encourage others to know their risk for stroke. His foundation partners with the community to prevent stroke and develop education and resources for stroke survivors and their families and health professionals. Xavier's foundation has collaborated on several research projects that directly involve stroke sufferers and has experience in managing their personal data and confidential information.

3.7.3 Claudia the biomedical engineer of the smartband company

Claudia graduated in Computer Engineering and she has a long-standing experience as a full stack developer and freelance developer in different projects performing maintenance tasks for websites and platform design. She is now working as a back-end developer for a renowned company that produces smartband for arrhythmia detection. Claudia has experience working with JavaScript runtime environments for the creation of Web servers and networking tools, database programs such as MongoDB, and Docker technology.

3.7.4 Jeanne the advisor neurologist

Jeanne, MD, PhD, is a Vascular Neurologist working in the largest hospital of her country. Jeanne is specialized in cerebrovascular diseases with expertise in the diagnosis and management of stroke. She is the Coordinator of a group of specialists in cerebrovascular diseases organized by a prestigious scientific society in Europe. She has participated in several research projects as a medical advisor.

3.8 Application End-user (Maintenance and Inspection)

The following personas do not directly work with the AI-SPRINT framework, but they work with the software created using the framework.

3.8.1 Ana the drone operator

Ana is the drone operator. She is responsible for taking shots of turbine blades in the field. Depending on the location of the turbine farm it might either be on the ground or on the sea. In both cases the amount of equipment she can take with her is limited. So most of the time next to the drone it is a notebook and/or tablet she uses for downloading photos from the drone and further uploading them to the cloud. She can benefit from image analysis in a number of ways. She can get close-real-time feedback from the drone if photos taken are of poor quality. Also, if she is using an autonomous flight feature, the AI model can be a source of information for an autonomous mission control unit. Finally, the initial assessment of the images allows Ana to realize that more detailed photos of some parts of the blade are required.

3.8.2 Whit the inspection analyst

Whit works at an inspection or maintenance company as the analyst. He is responsible for taking over photos taken by Ana and annotating, a posteriori, all the damages found on the blade. The final result of his work is a report on the state of the turbines summarizing all the types and severities of the damages found. He is using a SaaS system that allows him to browse through the images, mark damage bounding boxes, measure them and provide their description. The application he is using is able to suggest the damages to him using computer vision algorithms behind the scene. The information on the damages he enters into the system will be fed back into the machine learning process to improve computer vision models in next iterations.

3.9 Application End-user (Farming)

The next personas, Chiara and Zeno, are end users and do not directly work with the AI-SPRINT framework, but they work with the software created using the framework.

3.9.1 Chiara the Factory Owner

Chiara is the owner of a large lot of land in the Veneto region; she inherited the land from her parents and manages it together with her family. Chiara produces several agricultural goods, including wheat, tomatoes, apples and grapes. Her family also manages an Agriturismo, which in the past years has become an increasingly important part of the family business.

Most of the produced wine is sold directly to restaurants in the area or to tourists who spend their holidays in the Agriturismo. Chiara's customers are very conscious about food quality and are particularly interested in buying “bio” products. Chiara wants to reduce as much as possible usage of treatment in her winery. On the other hand, disease control is extremely important: she knows that she could lose the complete crop.

3.9.2 *Zeno the Factory Factotum*

Although Chiara has a very deep understanding of all aspects of farming, she manages the business and has delegated all practical work to employees or seasonal workers. **Zeno** is employed by Chiara; he has been working for Chiara for more than twenty years. He is in charge of the maintenance of the winery. He knows every corner of the farm, every plant on the field, every detail of the winery ecosystem, all friends and foes the grapevine.

4. Requirements Definition

In this section we analyze the requirements of each AI-SPRINT Use Case, by following the methodology described in Section 2 of this deliverable. Thus, the section is mainly divided in three subsections, each one corresponding with one of the AI-SPRINT Use Cases.

4.1 Use Case 1: Personalized Healthcare

4.1.1 Use Case overview

The **Personalized Healthcare** Use Case concerns the development of AI models for health monitoring via wearable devices connected to mobile phones. It will involve external participants, namely subjects who suffered from a stroke and healthy individuals. To prioritize data security, a federated learning setting will be implemented where versions of the model at the edge (i.e., mobile phones) share local updates on the global model parameters in the cloud rather than personal data. The local updates will be computed for groups of subjects in order to simulate a realistic scenario in which personal data are stored by different hospitals with data sharing constraints.

4.1.1.1 **Impact and objectives**

The overarching objectives of the Use Case are to increase the quality of care and lower expenses through personalized AI-powered solutions. Despite a general openness towards digital health, such as wearable technology, concerns about privacy and security are high[1]. AI-SPRINT federated learning environment ensures that smart and secure automated systems deliver personalized assessment of individual health, in particular stroke risk. Main benefits of this approach include:

- Wellbeing benefits objectives:
  - Continuous and non-invasive monitoring over long periods of time
  - Identification of early indicators of health issues
  - Management of at-risk and vulnerable subjects
  - Personalization of technological solutions matching individual needs
  - Patient empowerment and information about their own health
  - Patient engagement in healthier lifestyle and behavior change
- Economic benefits objectives:
  - Reduced strain on the healthcare system
  - Accelerated digitization of healthcare
  - Personalized care treatment plans
  - Innovation and growth of the wearable tech industry
  - Improve confidence in digital health through privacy and security

The digital data, which will be processed to train AI models in a federated learning setting, will be collected anonymously and encrypted to ensure its security and control according to ethical principles and relevant legislation (GDPR).

4.1.1.2 **Challenges**

The challenges specific to the Use Case encompass distinct aspects related to the complexity in terms of technological infrastructure and data modeling, analysis, and evaluation. The development of an actionable stroke risk assessment model is limited by the available data, which, in this Use Case, consists of sensors and lifestyle information. As supported by existing literature on similar studies, we expect this data to be

[1] Accenture 2020 Consumer Research on Digital Health: https://www.accenture.com/_acnmedia/PDF-130/Accenture-2020-Digital-Health-Consumer-Survey-US.pdf

sufficiently rich and informative, allowing us to obtain insights into this medical condition and thus enable an effective subject stratification. Nevertheless, the number and representativeness of the participants in the pilot study, as well as the integration of different types of sensor data (PPG, ECG, etc.) will pose challenges to address, but also opportunities for research in cardiovascular diseases. Subject recruitment and instructing based on a suitable experimental design can have an impact on the execution and outcomes of the Use Case pilot study. In this regard, the Use Case will implement a federated learning setting with hospital simulations, where a global model in the cloud learns from updates communicated periodically by groups of local models in the edge. The main challenges of federated learning are related to the efficiency of model updates communication, the possible device dropouts due to network connectivity or device energy issues, the statistical heterogeneity due to the non-uniform distribution of the data processed at the edge, and privacy and security concerns about sensitive information potentially embedded in the shared model updates. Finally, the distributed computing between the edge and the cloud, which will be attained by the use of COMPSs runtime, brings challenges which are mainly related to the compatibility of COMPSs in specific mobile operating systems.

The AI-SPRINT assets validated by this Use Case include design and performance tools (resource allocation between edge and cloud), privacy-preserving continuous training and deployment (federated model updates and responsive reaction to re-training and inference), programming runtime (task assignment and scheduling), and secure network (compliance with security policies).

4.1.1.3 Initial analysis high level requirements

The high-level requirements for the Personalized Healthcare use case are summarized in Table 4.1.

AI models for risk stroke assessment will be trained using time-dependent sensor data from a wearable device and information from a lifestyle questionnaire based on known cerebrovascular risk factors. These models, based on machine learning approaches enabling the integration of such heterogeneous data and trained using federated learning, will provide a fine risk stratification of subjects based on their individual characteristics. To this aim, development tools using Python and R programming languages as well as dedicated machine learning frameworks, such as Scikit-learn and PyTorch, will be used. The use of deployment tools, including Kubernetes for container orchestration, Infrastructure Manager (IM) and E3C for infrastructure management and elasticity, and OSCAR for FaaS services, as well as the programming model PyCOMPSs, are foreseen. The high security standards required by the use case will be covered by SCONE. Finally, privacy of personal information is guaranteed through the fully GDPR-compliant design of the use case pilot study.

	Data Type	Advanced AI features	Development Tools	Deployment Tools	Infrastructure Elasticity	Security	Privacy
Personalized Healthcare	time series	++	++	++	++	++	++

+ = nice to have; ++ = required

Table 4.1 - Requirements Overview Personalized Healthcare

4.1.2 High level Architecture

The architecture of the use case (see Figure 4.1) maps into the reference AI-SPRINT architecture, joining together specific Design and Run-Time tools that encompass all the required components and services. The deployable infrastructure consists of Kubernetes clusters and groups of edge devices that we call 'simulated hospitals', specifically patients' personal mobile phones connected via Bluetooth to individual smartbands.

The deployable infrastructure, which is secured with SCONE, is used for three main tasks: 1. training preliminary models prior to the pilot study; 2. updating and refining the models using federated learning during the pilot study; 3. exploiting OSCAR FaaS functionality for inference on-demand.

At Run-Time level, COMPSs orchestrates the distribution of the computation, MinIO provides local storage and cloud synchronization, InfluxDB takes care of monitoring services, IM and EC3 manage container deployment and elasticity, respectively. At design level, SPACE4AI-D selects the resources based on defined performance constraints.

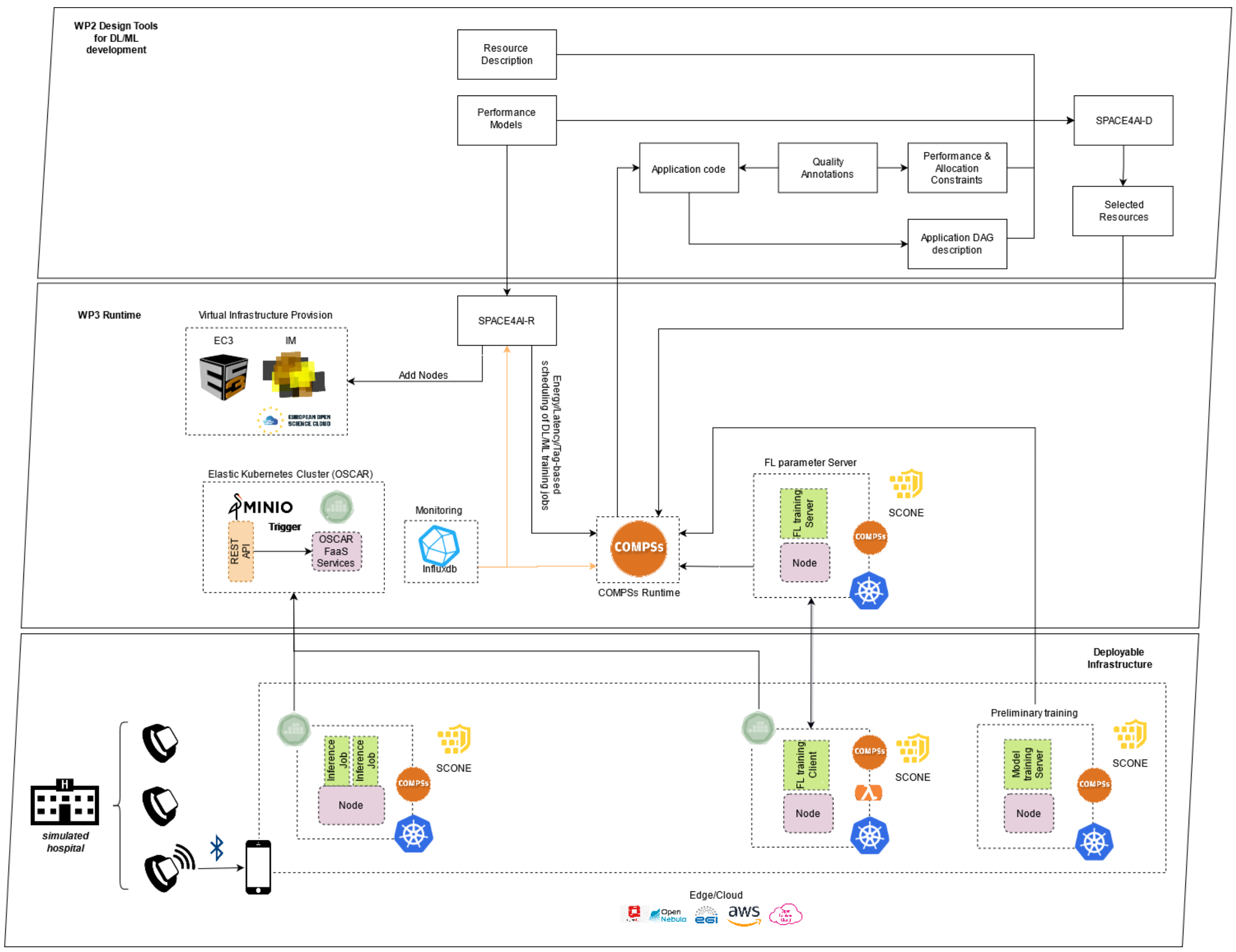

Figure 4.1 - High Level Architecture Personalized Healthcare

4.1.3 *Epics*

4.1.3.1 Epic UC1.E01: Preparation of the pilot study

Before the pilot study kick-off, requirements for the preparation of the pilot study are needed. At this stage, Piotr, Lucía and Paolo (the application manager, architect, and developer, respectively) collaborate with Claudia, the biomedical engineer of the smartband company, to develop a mobile phone app that receives sensor data via Bluetooth and transfers local model parameters to a general model in the cloud. Christine, the AI expert, designs such initial local and general models, which are preliminarily trained using simulated data or historical data, if available. These models will be increasingly rendered stroke-specific over the execution of the pilot study under the guidance of Jeanne, the advisor neurologist.

4.1.3.2 Epic UC1.E02: Execution of the pilot study

Henrik has signed an informed consent to participate in the pilot study. The staff of Xavier’s stroke foundation, who is in contact with Alexandre, the service provider, and Claudia, from the smartband company, instructs him on how to use the devices during the observation time. To ensure privacy, only an anonymous identifier assigned to his smartband is visible to the AI-SPRINT personas. Henrik's smartband is included in a group of others that represents the simulation of a hospital exchanging only model parameters rather than the personal sensor data. In this scenario, all security aspects and the federated learning are set up and ready for use.

4.1.3.3 Epic UC1.E03: Application and model refinements

During the observation time and at every parameter update, Christine, the AI expert, Piotr, the application manager, and Loredana, the infrastructure provider, will evaluate and report on the model and infrastructure performance as well as the strategies adopted to enhance both. The medical relevance of the model outcome is discussed with Jeanne, the advisor neurologist. Possible modifications to improve the infrastructure and model performance are adopted based on monitoring application metrics.

4.1.4 *User stories*

Field name	Details
ID	UC1.E01.US01
Title	Milestone versions
Priority	Should have
Persona	Piotr the Application Manager
Category	Design
Description	As an application manager when a new feature is released I want to monitor version and performance of the application so that the final release can be efficiently attained.
Need/Purpose	The preliminary activities before and after the kick-off of the pilot study imply an incremental addition of features over time that need to be addressed with proper versioning and evaluations.
Comment	The application manager can interact with the application architect to define KPIs based on these preliminary activities.

Field name	Details
ID	UC1.E01.US02
Title	Preliminary data analysis and initial models design
Priority	Must have
Persona	Christine the AI-Expert
Category	Design
Description	As an AI expert when I am developing the models I want to access preliminary sensor data so that an initial processing workflow and ML model can be developed and later refined.
Need/Purpose	In order to realize a federated learning approach, initial local models and a general model need to be implemented. At this stage, these models will not be specific to stroke but they will set the foundation for the subsequent pilot study with real patients. The data used to train the initial models can be simulated and thus basic federated learning components (e.g., sharing of model parameters and re-training procedures) put in place.
Comment	Simulated data can be fabricated from sensor data of volunteers within the staff working with the AI expert or, if possible, from historical data provided by the smartband company.

Field name	Details
ID	UC1.E01.US03
Title	Effective edge-cloud transfer of model parameters
Priority	Must have
Persona	Paolo the Application Developer
Category	Runtime, Security
Description	As an application developer, when facilitating the transfer of the model parameters to the cloud I want to use infrastructure management tools so that the needed functions and operations, such as distributed computing, are tracked and evaluated.
Need/Purpose	Easing the secure and efficient transfer of model parameters to the cloud is crucial to test and use all federated learning components.
Comment	The application developer can interact with the application architect to define the requirements to implement the distributed workflow (PyCOMPSs).

Field name	Details
ID	UC1.E02.US01
Title	Coaching of human subjects about the devices
Priority	Must have
Persona	Henrik the stroke patient
Category	Non-Functional
Description	As a stroke patient when I participate in the research study I want to receive instructions on how to use the smartband and the mobile phone app.
Need/Purpose	Each stroke patient can download a dedicated app, connect his/her smart phone to the smartband using Bluetooth and check the status of all sensors.
Comment	The implementation of this activity is crucial to the success of the UC but it is not related to the AI-SPRINT framework. Importantly, the staff of the stroke foundation need to be instructed first on the devices by the service provider or the biomedical engineer at the smartband company.

Field name	Details
ID	UC1.E02.US02
Title	Hospital simulation
Priority	Should have
Persona	Paolo the Application Developer
Category	Runtime, Security
Description	As an application developer when I design the parameter update for federated learning I want to aggregate local model parameters so that each corresponding group of patients can simulate a hospital.
Need/Purpose	To demonstrate the utility of federated learning in scenarios where privacy constraints limit personal data sharing, we aim to simulate hospitals by grouping different subjects participating in the pilot study.
Comment	The application developer needs to interact with the AI expert in order to evaluate what is the best strategy to aggregate and update the local model parameters of each hospital.

Field name	Details
ID	UC1.E02.US03
Title	Model inference on-demand
Priority	Should have
Persona	Paolo the Application Developer
Category	Runtime
Description	As an application developer when I perform model inference I want to run this computational job on a FaaS platform so that the right execution environment is automatically triggered at convenience.
Need/Purpose	To assess individual stroke risk, the subject or a physician would like to perform a stateless inference using the trained model that is available at the edge.
Comment	This functionality would be desirable to account for efficient responses to immediate model inference requests.

Field name	Details
ID	UC1.E03.US01
Title	Model performance evaluation
Priority	Must have
Persona	Christine the AI-Expert
Category	Runtime
Description	As an AI expert when a model is deployed I want to evaluate its performance so that the stroke risk assessment is accurate and precise.
Need/Purpose	The evaluation of model performance, in terms of accuracy and biomedical plausibility, is crucial to ensure quality and trustworthiness of the offering and identity measures of improvements.
Comment	The biomedical relevance of the model performance should be evaluated by domain-expert, such as the medical advisor of the project.

Field name	Details
ID	UC1.E03.US02
Title	Infrastructure performance evaluation
Priority	Must have
Persona	Loredana the Infrastructure Provider & Sysops
Category	Runtime
Description	As an infrastructure provider when a new infrastructure component is deployed I want to evaluate its performance so that the infrastructure is always responsive and functional.
Need/Purpose	The evaluation of infrastructure performance, in terms of response times and memory usage, is crucial to ensure quality and effectiveness of the offering and identity measures of improvements.
Comment	Beside monitoring indicators, user experience should be taken into consideration.

4.1.5 Requirements

Field name	Details
ID*	UC1.Req001
Title*	Versioning over the three stages of the UC (before, during and after the pilot study)
Priority*	Should have
Required for (User Story)*	UC1.E01.US01/02/03, UC1.E02.US03, UC1.E03.UC01/02
Category	Design, Runtime
Description*	Keeping track of the progress towards the final versions of models and applications is fundamental during the course of the project.
Rationale*	Establishing a versioning scheme allows tracking of new developments. This is important for the UC because preliminary implementations, such as initial models for setting up a federated learning structure, are needed before the pilot study is executed. Moreover, during and after the pilot study, the models and the software can further change and be refined.
Tool	PyCOMPSs, IM, EC3, OSCAR
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC1.Req002
Title*	Exploratory sensor data analysis and modeling
Priority*	Must have
Required for (User Story)*	UC1.E01.US02
Category	Runtime
Description*	Preliminary sensor data are used to create initial ML models that will be further trained with federated learning during the pilot study
Rationale*	To achieve a functional set up of all the components of the federated learning for the pilot study, initial modeling strategies for both the local models and the global model need to be explored and implemented.
Tool	PyCOMPSs, Performance Models
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC1.Req003
Title*	Effective exchange of model parameters between the edge and the cloud
Priority*	Must have
Required for (User Story)*	UC1.E01.US02/03, UC1.E02.US03, US1.E03.US02
Category	Design, Runtime
Description*	The correct execution of the federated learning procedure requires an efficient transfer of model parameters between the edge and the cloud.
Rationale*	The setup and evaluation of the runtime of federated learning features, such as the update of local model parameters at some frequency, enable to effectively reuse and improve such components in the different stages of the pilot study.
Tool	Federated Learning, PyCOMPSs, Monitoring infrastructure
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC1.Req004
Title*	Mobile phone app development
Priority*	Must have
Required for (User Story)*	UC1.E01.US03, UC1.E02.US01/03
Category	Runtime, Security
Description*	A mobile phone app that receives sensor data from the smartband via Bluetooth, updates model parameters by communicating with the cloud, and informs the user about its correct functioning must be developed.
Rationale*	The mobile phone app represents the central piece of the UC as it interfaces with the user and the AI-SPRINT architecture.
Tool	PyCOMPSs, SCONE
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	A feasibility analysis is needed due to possible constraints of Android releases.

Field name	Details
ID*	UC1.Req005
Title*	Hospital model parameters aggregation and update
Priority*	Should have
Required for (User Story)*	UC1.E01.US02, UC1.E02.US02/03, UC1.E03.US01
Category	Design, Runtime
Description*	The model parameters exchange and update through federated learning is arranged to reflect a specific grouping of the participating subjects.
Rationale*	Federated learning is an effective model development strategy to overcome the constraints on personal data sharing of hospitals. As no real hospitals are involved in the UC, we simulate a "hospital" by grouping subjects that participate in the pilot study.
Tool	Federated Learning, PyCOMPSs
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC1.Req006
Title*	Federated learning performance evaluation
Priority*	Must have
Required for (User Story)*	UC1.E01.US02/03, UC1.E02.US02/03, UC1.E03.US01/02,
Category	Design, Runtime, Security
Description*	The performance of federated learning is evaluated in terms of efficiency and biomedical value.
Rationale*	The model evaluation can be performed considering several aspects of the AI-SPRINT technology, in particular the efficient functioning of the architecture for the UC and the biomedical relevance of its outcome for stroke risk assessment. Additional evaluations should include known issues related to federated learning, such as non-identically distributed datasets, poor scalability, low responsiveness, waste of resources (e.g., idle nodes waiting for parameters updates), security aspects, and others.
Tool	Monitoring infrastructure
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC1.Req007
Title*	Ensuring a secure data collection and processing environment
Priority*	Must have
Required for (User Story)*	UC1.E01.US03
Category	Runtime, Security
Description*	Due to the high security requirements of the UC, data at rest, data transfers as well as data processing should all occur in a secure as well as encrypted fashion.
Rationale*	The stroke risk assessment Use Case processes highly sensitive medical data. Therefore, it is important to provide confidentiality and integrity protection for the data while being collected through the edge devices, while being transferred to cloud resources for further processing as well as while being processed and stored there. Traditional isolation techniques such as using virtual machines are not sufficient as memory dumps can be performed by the cloud owners which are not considered as trustworthy. Therefore, all involved components shall utilize trusted environments where available as it is crucial to guarantee a secure transfer and processing of highly sensitive information through all the components of the AI-SPRINT architecture involved in the UC.
Tool	SCONE
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC1.Req008
Title*	Orchestration and resource management
Priority*	Must have
Required for (User Story)*	UC1.E01.US03, UC1.E03.US02
Category	Design, Runtime
Description*	This requirement entails the design, verification and optimization of cloud solutions to distribute and orchestrate the computation between the edge and the cloud during model training. PyCOMPSs, Infrastructure Manager (IM), in combination with EC3, and SPACE4AI-R will take care of applications orchestration, virtual infrastructure management and resources assignment, respectively.
Rationale*	Managing cloud services through orchestration and resource management tools able to select and utilize the most appropriate sites to run jobs in a distributed manner is crucial to achieve efficiency during model training, in particular in scenarios such as frequent aggregations of the local updates from the edge devices into the global model in the cloud.
Tool	PyCOMPSs, Monitoring infrastructure, IM, EC3, SPACE4AI-R
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC1.Req009
Title*	FaaS platform for model inference
Priority*	Should have
Required for (User Story)*	UC1.E02.US01/02/03
Category	Runtime
Description*	Model inference is carried out in FaaS platforms to account for requests to be executed in closed environments and with automated elasticity.
Rationale*	A subject or a physician would like to run the pre-trained model on the edge to check stroke risk.
Tool	OSCAR
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC1.Req010
Title*	Monitoring and log management

Priority*	Should have
Required for (User Story)*	UC1.E01.US02, UC1.E02.US03, UC1.E03.US01/02
Category	Design, Runtime
Description*	Monitoring would be required at two levels: model performance and infrastructure performance. At the model level, training and inference quality metrics would be desirable, such as precision, recall, log loss, and others. Such evaluations will be based on the integration of sensor data and lifestyle information of the healthy and diseased participants of the pilot study, with additional validation of the predicted cerebrovascular risk from the medical advisor. At the infrastructure level, monitoring application execution times, workload statistics, networking quality, memory usage, would be advisable. In this regard, the management of log files should also be properly addressed considering containerization.
Rationale*	Monitoring all aspects of the systems allow carrying out surveillance of the infrastructure and model behavior and device improvement strategies.
Tool	Monitoring infrastructure
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

4.1.6 Summary and Overview of Requirements

The requirements from the Personalized Healthcare Use Case are summarized in Table 4.2.

Requirements	ID	Design	Runtime	Security
Versioning over the three stages of the UC (before, during and after the pilot study)	UC1.Req001	X	X	
Exploratory sensor data analysis and modeling	UC1.Req002		X	
Effective exchange of model parameters between the edge and the cloud	UC1.Req003	X	X	
Mobile phone app development	UC1.Req004		X	X
Hospital model parameters aggregation and update	UC1.Req005	X	X	
Federated learning performance evaluation	UC1.Req006	X	X	X
Ensuring a secure data collection and processing environment	UC1.Req007		X	X
Orchestration and resource management	UC1.Req008	X	X	
FaaS platform for model inference	UC1.Req009		X	
Monitoring and log management	UC1.Req010	X	X	

Table 4.2 - List of Requirements Personalized Healthcare

4.1.6.1 **Design**

Design requirements encompass several aspects that are crucial for a smooth execution of the Use Case, including the evaluation of performance constraints and the optimal selection of resources.

4.1.6.2 **Runtime**

Runtime requirements include the distribution and orchestration of the tasks between the edge and the cloud, the execution of model inference on-demand via FaaS platforms, and the efficient management of monitoring the model and infrastructure performance.

4.1.6.3 **Security**

Security is a priority of the Use Case due to the involvement of human participants and the use of sensitive information. For this reason, all the processes and components involved in the local training of the model and the exchange, aggregation and update of parameters need to be protected through encryption and, when available, Trusted Execution Environments.

4.1.6.4 **Out of Scope**

The main aspect that is out of the scope of the AI-SPRINT technology is related to the management of the human participants in the pilot study. In compliance with GDPR, all personal and sensitive information is anonymized and privacy ensured in all activities of the Use Case.

4.1.7 Initial Roadmap

The first tentative road map for the implementation of the Personalized Healthcare Use Case is illustrated in Figure 4.2:

- Preliminary data collection and model development will be performed between the second half and the end of 2022 (M18-M24). During this period the mobile phone app as well as the federated learning will be implemented, and the monitoring system, both in terms of model and infrastructure performance, will be tested.
- The pilot study will be divided into three phases of three months each. These activities, which are preceded by the subject's recruitment, could be carried out at the beginning of each half year starting 2022. The execution of these three activities requires monitoring and it is followed by model refinement tasks.

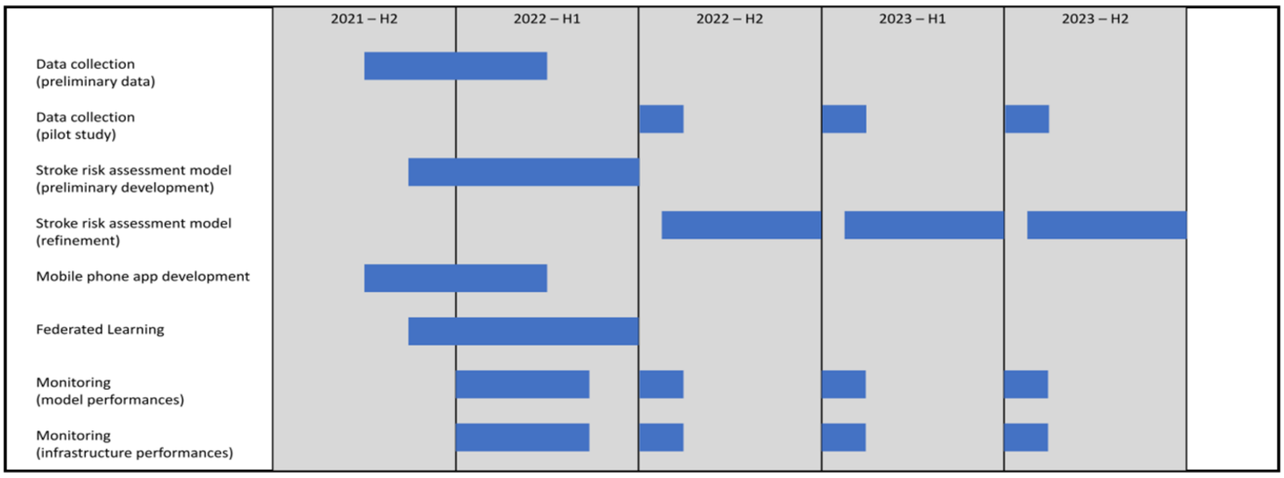

Figure 4.2 - Initial Roadmap Personalized Healthcare

4.2 Use Case 2: Maintenance and Inspection

4.2.1 Use Case overview

This Use Case exploits AI models for identifying windmill blade damage based on vision and thermal images collected by drones (both actual and prospective damages will be considered). Given bandwidth and connection stability constraints coupled with the limited flight time of a drone which is typically below 30 minutes, the selection of relevant images (including objects that need to be further analyzed or general non-repetitive images of clear regions) occurs at the edge (ground station) and only relevant data are transmitted over the edge-cloud channel. Edge processing will also be in charge of calling for a new acquisition (detailed images of certain regions) when required. Inspection time and operator effort can be significantly reduced at IoT level (on the drone, exploiting power/weight-efficient GPU modules, already available as prototype today) by providing image quality feedback to the operator, allowing less conservative flight. Additionally, real-time feedback from AI-powered image analysis can be used as an input for a drone's autonomous mission control unit. AI-SPRINT tools will enable optimal interaction of cloud-based (computationally intensive, longer) analysis and local processing using lighter data pattern recognition routines.

4.2.1.1 Impact and objectives

There a few goals to be achieved by using AI-SPRINT in this Use Case:

- Improve the quality of data collected in the field
- Provide on-line information about surroundings of the drone by ML-based analysis of the video feed
- Reduce software maintenance overhead by providing effective tools for deployment and monitoring of ML models, both on edge and in cloud
- Optimize computation resources required to execute and train ML models

These technical objectives translate to the business effectiveness of inspection operations. Improving image quality and making autonomous operation of the drone possible mean less time spent in the field. It saves both time of people involved and downtime of the turbine. Less maintenance work and good tooling means a more reliable platform, better user experience and generally better product. That supports retention of existing customers and looking for new ones. Finally, optimization of computation resources directly affects infrastructure costs but also saves energy which might be even more important, especially in the green energy sector.

4.2.1.2 Challenges

The biggest challenge of the Use Case is related to the inspection data acquisition, so the edge part of the platform. Wind turbines are very often located in places that are difficult to access (hills, sea), far from telecommunication infrastructure. Drone operators normally cannot take with them an edge server or PC. Notebook or just a tablet is the only option. Drones have limited lift capacity and limited energy sources. All of that means we are operating in a very restrictive environment, at the same time requiring low processing times, high accuracy, good monitoring and reporting. Fulfilling all of these requirements at the same time will require efficient software and careful selection of the hardware elements.

4.2.1.3 Initial analysis high level requirements

The high-level requirements for the Maintenance and Inspection use case are summarized in Table 4.3.

AI models will be trained using images from the drones. Models will allow the drone operator to run autonomous missions. The drone will be able to assess the quality of the images and will ensure all images are taken from the correct side of the blade.

Maintenance and Inspection using drones apply to a fast number of potential customers, not only windmill operators. Powerful development tools are needed to ensure a solid software architecture while flexible runtime tools will support deployment for different customers. Deployment tools will allow migration among different public/private cloud providers thus allowing the support of more customers.

The Use Case does not present any particular challenge in terms of security. Nevertheless, data encryption in transit and at rest will be used. No personal data will be processed by the application, privacy is not critical for this Use Case.

	Data Type	Advanced AI features	Development Tools	Deployment Tools	Infrastructure Elasticity	Security	Privacy
Maintenance and Inspection	image	+	++	++	+	+	

+ = nice to have; ++ = required

Table 4.3 - Requirements Overview Maintenance and Inspection

4.2.2 High level Architecture

The architecture of the use case (see Figure 4.3) maps into the reference AI-SPRINT architecture, including design and Run-Time tools specific to the Maintenance and Inspection Use Case. The deployable infrastructure consists of (1) Training nodes used to train the models (2) edge devices running VM nodes, which will typically run in a laptop and (3) a lightweight edge device running on the drone itself.

All models will be trained on a public or private cloud, the inference will run on either edge device or cloud (also through FaaS), depending on the inference type .

At Run-Time level, SPACE4AI-R and Krake orchestrate the computation; MinIO provides a local storage which can be synchronized with the cloud; IM and EC3 manage container deployment and elasticity, respectively.

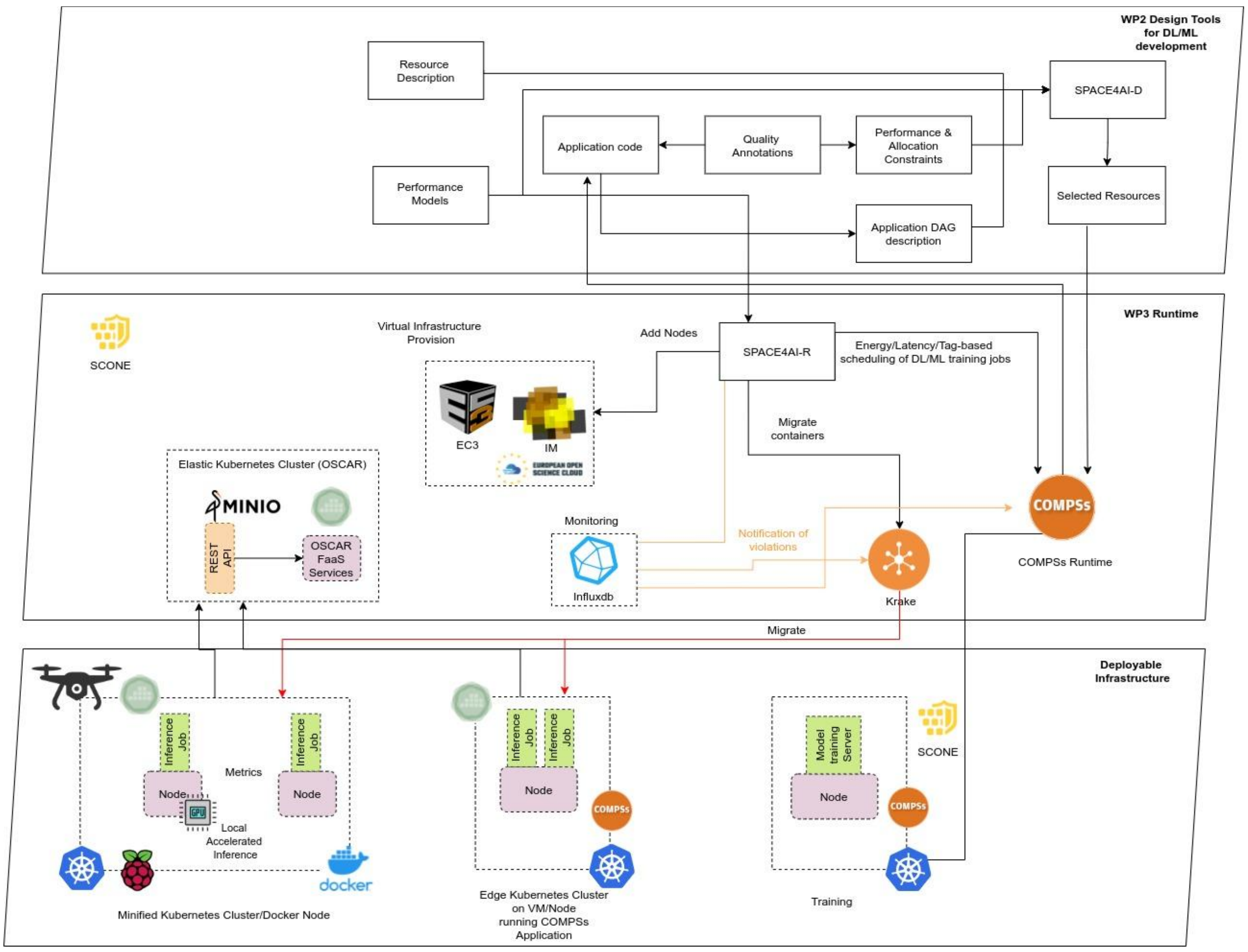
WP2 Design Tools for DL/ML development
Resource Description
SPACE4AI-D
Application code
Quality Annotations
Performance & Allocation Constraints
Performance Models
Selected Resources
Application DAG description
WP3 Runtime
SCONE
Virtual Infrastructure Provision
Add Nodes
SPACE4AI-R
Energy/Latency/Tag-based scheduling of DL/ML training jobs
EC3
IM
EUROPEAN OPEN SCIENCE CLOUD
Migrate containers
Elastic Kubernetes Cluster (OSCAR)
MINIO
REST API
OSCAR FaaS Services
Monitoring
Influxdb
Notification of violations
COMPSs
COMPSs Runtime
Krake
Migrate
Deployable Infrastructure
Inference Job
Inference Job
Metrics
Node
Node
GPU
Local Accelerated Inference
docker
Minified Kubernetes Cluster/Docker Node
Inference Job
Inference Job
Node
COMPSs
Edge Kubernetes Cluster on VM/Node running COMPSs Application
Model training Server
Node
SCONE
COMPSs
Training

Figure 4.3 - High Level Architecture Maintenance and Inspection

4.2.3 Epics

4.2.3.1 Epic UC2.E01: Taking photos in the field

Ana, the drone operator, is using UAV (Unmanned Aerial Vehicle) to take photos of the wind turbine blades. While taking photos, she is interested in image quality or verification of blade face exposure. She also needs to know whether there are some damages that require taking close up photos. Additionally, Ana might want to use autonomous mission mode in her work. In such a case, there are two major ways of managing the flight. One is to pre-program the flight pattern that will be followed by the drone, based on GPS and RTK data. This option does not require additional information from AI modules. The second one is fully autonomous mode where the drone operates based on its sensors. Such mode of operation relies on the real-time data about surroundings. One of the sources can be real-time analysis of the video feed from the camera that can be performed by the on-board ML model.

4.2.3.2 Epic UC2.E02: Inspection report preparation

Whit, the inspection analyst, is working on the report preparation. He needs to find all the damages visible on the photos delivered by Ana. He is working with a SaaS application for image browsing and, by computer vision models to suggest damage types, severity and location he annotates the images for the report that will be used also in the following for model retraining. Whit is the implicit user of the AI Sprint framework. He is using his SaaS application for his everyday work. The application is calling AI models hosted, deployed and controlled by the framework.

4.2.4 User stories

Field name	Details
ID	UC2.E01.US01
Title	Check image quality
Priority	Should have
Persona	Ana the drone operator
Category	Runtime
Description	As a drone operator I need a way to verify that shots I have taken are of good quality without the need of going through all individual photos manually but relying on AI models.
Need/Purpose	Assuring good quality of photos.
Comment	

Field name	Details
ID	UC2.E01.US02
Title	Check blade face exposure
Priority	Should have
Persona	Ana the drone operator
Category	Runtime
Description	As a drone operator I need a way to verify that shots I have taken show blad face at an angle consistent with SOP (Standard Operating Procedure).
Need/Purpose	Assuring good quality of photos.
Comment	

Field name	Details
ID	UC2.E01.US03
Title	Navigation points detection
Priority	Nice to have
Persona	Ana the drone operator
Category	Runtime
Description	As a drone operator I want to be able to set up autonomous mission mode. My navigation unit needs to recognize and locate on camera feed characteristic construction elements of the blade (tip or hub).
Need/Purpose	Navigation aid.
Comment	

Field name	Details
ID	UC2.E01.US04
Title	Clearinghouse functionality
Priority	Nice to have
Persona	Ana the drone operator
Category	Runtime
Description	As a drone operator I want to be able to do initial screening of the images and locate these that need to be further checked for damages.
Need/Purpose	Operations optimization
Comment	

Field name	Details
ID	4.2.4.1 **UC2.E02.US01**
Title	Detect damages to the blade
Priority	Must have
Persona	Whit the inspection analyst
Category	Runtime
Description	As an analyst I need an application to locate damages on the images, suggest its classification and severity.
Need/Purpose	Damage detection
Comment	

4.2.5 *Requirements*

Field name	Details
ID*	UC2.Req001
Title*	Off-line operation
Priority*	Must have
Required for (User Story)*	UC2.E01.US01,UC2.E01.US02,UC2.E01.US03
Category	Runtime
Description*	Modules operating in the field need to be able to work off-line.
Rationale*	Drones often operate in places where there is no internet or cellular range.
Tool	Monitoring Infrastructure, Krake, SPACE4AI-R, IM
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC2.Req002
Title*	Model updates
Priority*	Should have
Required for (User Story)*	UC2.E01.US01,UC2.E01.US02,UC2.E01.US03
Category	Runtime
Description*	Edge devices running ML models should be able to update/change model versions easily.
Rationale*	There are two scenarios where models are changing. In the first scenario, simply a new version of the model. In the second one, running different models/pipelines for various sites (for example different for ground inspections, different for off-shore) is needed.
Tool	SPACE4AI-D, IM, OSCAR
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC2.Req003
Title*	Operation metrics
Priority*	Should have
Required for (User Story)*	UC2.E01.US01,UC2.E01.US02,UC2.E01.US03
Category	Runtime
Description*	Mission metrics (operation times, model results) should be available for analysis.
Rationale*	Good monitoring provides system health verification and mission quality (number of good shots for example).
Tool	Monitoring Infrastructure
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC2.Req004
Title*	Model’s performance: response time
Priority*	Must have
Required for (User Story)*	UC2.E01.US01,UC2.E01.US03
Category	Runtime, Design Time
Description*	Some of the analytical modules used on the UAV require near-real-time operation (<1s response).
Rationale*	Some applications of the models (blade part/location, exposure check) require quick feedback (either for drone operator or for autonomous mission control unit). This can be achieved through a proper initial deployment of the components and selection of the AI intelligent devices at design time, and through components migrations, change of the AI model partitioning, scaling of cloud VMs , etc. at runtime.
Tool	SPACE4AI-D, SPACE4AI-R, Krake, Monitoring Infrastructure
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC2.Req005
Title*	Cloud deployment
Priority*	Must have
Required for (User Story)*	UC2.E02.US01
Category	Runtime
Description*	Flexible deployment mechanism allowing to run ML models in various infrastructure scenarios (serverless, VM, K8S cluster, etc.)
Rationale*	Data processing pipelines are in constant development and their hardware needs are changing (amount of memory, computation power needs, access to GPU etc.). It is essential that both experiments and production models can be easily deployed and run in various configurations.
Tool	IM, OSCAR, EC3
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC2.Req006
Title*	Model's performance: throughput
Priority*	Must have
Required for (User Story)*	UC2.E02.US01
Category	Runtime
Description*	Data analysis infrastructure needs to be able to process a significant amount of data in a relatively short period of time. Single turbine inspection consists of 250-700 images (normally 20MPix). Single turbine should be processed within 1-2 minutes.
Rationale*	End users of the system expect almost real time operation of the system. As soon as data is uploaded into the system it is expected to be available for report preparation.
Tool	SPACE4AI-R, Krake, Monitoring infrastructure
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC2.Req007
Title*	Cloud computing metrics

Priority*	Must have
Required for (User Story)*	UC2.E01.US01,UC2.E01.US02,UC2.E01.US03
Category	Runtime
Description*	Model operational metrics (processing time, resources consumed, any processing failures etc.) should be collected and available for analysis. Any critical issue should be alerted to the system administrator.
Rationale*	Good monitoring is essential to catch and eliminate any issues in the platform. Especially important for a production deployment.
Tool	Monitoring infrastructure
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	**Details**
ID*	UC2.Req008
Title*	Dynamic deployment and scaling
Priority*	Should have
Required for (User Story)*	UC2.E01.US04, UC2.E02.US01
Category	Runtime
Description*	Platform should allow for component migration between edge and the cloud or scale up cloud resources in case of increased need for computation power.
Rationale*	Even though the initial model deployment pattern might be good for most of the cases, sometimes computation needs might require additional computation power (more defects, higher resolution photos etc.). In such a case migration of some components to the cloud or cloud resources scale up might be necessary.
Tool	IM, EC3, Krake, SPACE4AI-R
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC2.Req009
Title*	Log collection and browsing
Priority*	Must have
Required for (User Story)*	All USs
Category	Runtime
Description*	All the components of the platform constantly generate logs of various nature (warnings, errors, additional information etc.). The requirement is to have a consistent way of gathering the logs, sending them to the central location and analyzing them afterwards.
Rationale*	In distributed systems log management and good tools for its analysis is crucial for effective platform management. Logs are being generated in various locations by different components. To have a good overview of platform operations, log entries need to be centralized and proper analysis tools need to be available.
Tool	Logging infrastructure
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

4.2.6 *Summary and Overview of Requirements*

The requirements from the Maintenance and Inspection Use Case are summarized in Table 7.

Requirements	ID	Design	Runtime	Security
Off-line operation	UC2.Req001		X	
Model updates	UC2.Req002		X	
Operation metrics	UC2.Req003		X	
Model's performance: response time	UC2.Req004	X	X	
Cloud deployment	UC2.Req005		X	
Model's performance: throughput	UC2.Req006		X	
Cloud computing metrics	UC2.Req007		X	
Dynamic deployment and scaling	UC2.Req008		X	
Log collection and browsing	UC2.Req009		X	

Table 4.4 - List of Requirements Maintenance and Inspection

4.2.6.1 **Design**

There are no special design time requirements coming from the Use Case. However, the application architecture should be defined properly to allow components migration and resource scaling at runtime to provide near real-time performance.

4.2.6.2 **Runtime**

Runtime is where most of the Use Case needs are focused. For infrastructure inspection it is crucial to have a flexible deployment and computation model that supports various situations drone operators can face in the field. Monitoring and logging infrastructure should be resilient enough to work properly even in case of longer off-line periods.

4.2.6.3 Security

There are no specific security requirements coming from the Use Case.

4.2.6.4 Out of Scope

There are numerous aspects of UAV operations that are important from Use Case perspective (flight control software, autonomous mission control, etc.) that are not in the scope of the AI-SPRINT project.

4.2.7 Initial Roadmap

The first tentative road map for the implementation of the Maintenance and Inspection Use Case is illustrated in Figure 4.4.

- During the first part of the project, the AI platform will be prepared, the AI models will be trained. These two phases are the most challenging phases of the Use Case and will extend into the first half of 2023
- Once the first version of the AI models has been prepared, the models will be tested, improved and integrated into the existing services

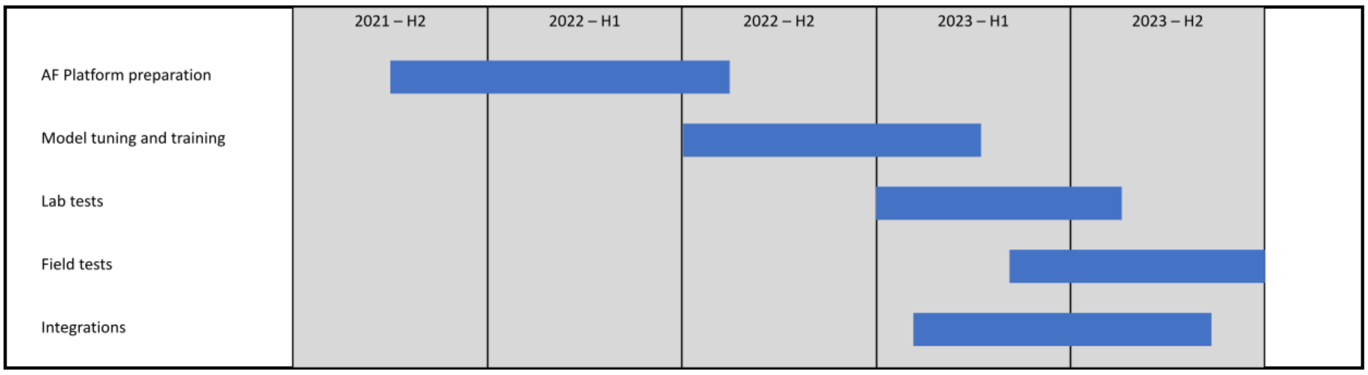

Figure 4.4 - Initial Roadmap Maintenance and Inspection

4.3 Use Case 3: Farming 4.0

4.3.1 Use Case overview

Farming 4.0 Use Case focuses on the optimization of phytosanitary products used in vineyards. AI models will be developed to process at the edge (i.e., field machinery) image sensor data collected on a grape harvester (e.g., multispectral, fluorescent, foliar humectation). These data will be matched on the cloud with historical data sources (e.g., phytosanitary product usage and crop yield) for the monitoring, planning, and optimization of quantity, timing and placement of products.

4.3.1.1 Impact and objectives

The high-level objective is to achieve precision farming in viticulture, where every step of the winegrowing process from pruning to harvesting is constantly monitored and optimized. Overall this Use Case will achieve the following objectives:

- Environmental benefits objectives:
  - Pollution reductions by min. 35%
  - Increases in good insects for the farm ecosystem of around 20%
  - Prevention of new species resistant (-25%) to treatment, fewer chemical products, fewer products harmful to biodiversity

    - Identifying parasites or diseases in vines at a very early stage means being able to cure only the vines that need treatment with the exact number of products
- Economic benefits objectives:
    - Cost reduction by optimizing the quantity of chemical products (-10%) used for spraying
    - Improved overall vineyard efficiency by stabilizing crop quantities and reducing the number of sick plants
    - Increased sales in smart spraying segment

4.3.1.2 **Challenges**

Besides the novelty of the Use Case subject, there are many potential blockers identified for achieving the proposed goals. The isolated areas with poor or no Internet connectivity where this solution will operate, the environmental conditions, limited computing power on Edge, or the fast changing of the elements that need to be analyzed (leaves, grapes) are raising various challenges like:

- Necessity to execute AI models on the edge, with no connectivity to the cloud
- Unstable edge-to-cloud connection and consequent need to cache collected data, which include logs, monitoring data, but also images
- Need to keep device and maintenance costs low against strong application constraints (edge devices are widely distributed and/or difficult to act upon) requiring central device management with frequent updates and monitoring
- Need to develop novel models for phytosanitary product optimization, never exploited so far in the farming field, continuously adapting models for phytosanitary use based on the data gathered on the field

The AI-SPRINT assets validated by this Use Case are related to design tools (the possibility of learning AI models for the unconventional data collected by the grape harvester), programming abstractions (simplify development and deployment of solutions on heterogeneous and difficult to manage IT environment), continuous training (to iteratively improve the AI models initial models rely on limited data and need interactive improvement), support of open source standards and multi-cloud (prevent lock-in, leaving GREG customers free to choose the hosting cloud by themselves).

4.3.1.3 **Initial analysis high level requirements**

The high-level requirements for the Farming 4.0 Use Case are summarized in table 4.5.

AI models for the Use Case will be used to evaluate foliage volume (RGB images and 3D Infrared images), to estimate yield (RGB images) and to detect diseases (RGB and GPS). Velocity and acceleration are needed to compute the exact time at which the computed quantity of phytosanitary treatment needs to be used. Training AI models in this domain is quite challenging, however, advanced features such as Federated Learning will not be needed.

The mid-term goal is to develop a service out of the application, which means that Development and Deployment Tools will be needed to ensure the service can be continuously improved and that improvement can be continuously and easily deployed to farmers subscribing to the service.

Infrastructure Elasticity might be needed in a later phase, e.g. to easily migrate the application between Cloud Providers or possibly deploy on an on-premise infrastructure managed by the farmers.

Security is critical in all modern software applications, however, there are no specific requirements for this Use Case. Data will be encrypted in transit and at rest, each device will authenticate using a certificate.

Privacy is not an issue, no personal data will be stored or processed by the application.

	Data Type	Advanced AI features	Development Tools	Deployment Tools	Infrastructure Elasticity	Security	Privacy
Farming 4.0	Images, GPS, Velocity Acceleration	++	++	++	+	+	

+ = nice to have; ++ = required

Table 4.5 - Requirements Overview Farming 4.0

4.3.2 *High level Architecture*

Overall architecture of the Use Case is illustrated in Figure 4.5.

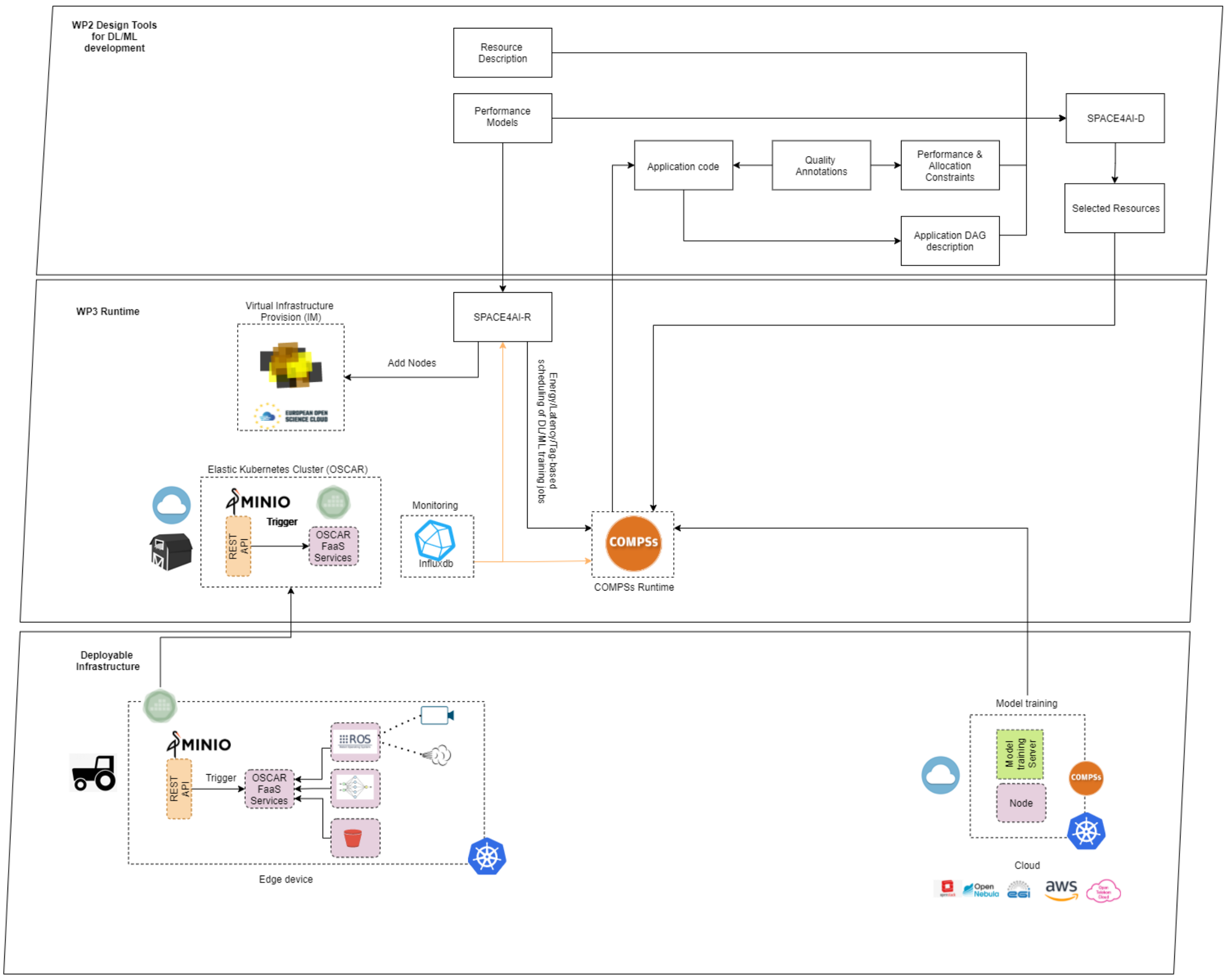

Figure 4.5 - High Level Architecture Farming 4.0

In terms of design and runtime, the architecture of the Use Case is very close to the reference AI-SPRINT architecture, although Federated Learning and encryption of the containers is not relevant for the Use Case, which slightly simplifies the overall picture.

The deployable infrastructure consists of a Kubernetes cluster for training of AI-model and the edge devices. The device will rely on OSCAR for FaaS functionality. At this early stage it is assumed that the edge device will run several containers orchestrated by Kubernetes and deployed using IM.

The edge containers interacting with sensors or with tractor hardware (spraying system) will use containers running ROS. As in other Use Cases, MinIO will be used to provide local storage services and synchronization with the cloud.

4.3.2.1 Smart Farming Device - Hardware

A Smart Farming Device (SFD) will be developed and provided to be installed on the spraying tractor for collecting images in the vineyard and perform the spraying tuning based on foliage density and depth.

The SFD contains a processing unit and a 3D camera with the specifications (an RGB camera and possibly a hyperspectral multispectral camera will be added later in the project) reported in Tables 4.6 and 4.7.

Model	FleetPC-9-B Car-PC (Detailed specifications: https://www.cartft.com/catalog/il/2566)
CPU	Intel Core i7-8700T 6x4.0Ghz
Chipset / FSB	Intel Q370
RAM	32GB
Graphics	Intel UHD Graphics 630
Operating temperature	-40~70°C (with SSD)

Table 4.6 - Smart Farming Device Hardware

Model	O3M271 (Detailed specifications: https://www.ifm.com/de/en/product/O3M271)
Type of light	Infrared Light
Image Resolution [px]	720 x 576
Image resolution 3D [px]	64 x 16
Image repetition frequency 3D [Hz]	25 / 33 / 50

Table 4.7 - Smart Farming Device 3D-Camera

4.3.3 Epics

4.3.3.1 Epic UC3.E01: New SFD Setup

The SFD is configured by Loredana (infrastructure Provider & SysOp) who installs the software elements of the AI-SPRINT framework, a certificate specific to the SFD and the software required for this Use Case. The SFD is installed on the spraying equipment (tractor). It is powered by the tractor and protected against elements (e.g., humidity, vibration, extreme temperatures). It consists of a computing unit that is connected to the cameras which will capture pictures of the plants and yield. The unit will have Internet connectivity and is easily replaceable in case of malfunction.

4.3.3.2 Epic UC3.E02: SFD paring and health check before usage

Zeno (Factory factotum) checks on his telephone if the SFDs (i.e., edge computer and all sensors, from hereafter SFD) on the tractors are working correctly. If so, he starts the treatment of the winery. If the SFD is not working, Zeno may either replace it or spray without using the intelligence provided by the device.

4.3.3.3 Epic UC3.E03: Default spraying conditions

Before beginning treatment, Zeno can check and modify default spraying settings. If during the spraying procedure, some sensor fails, the system will automatically revert to the defined default settings. At any time, Zeno can revert to manual spraying by overriding the system.

4.3.3.4 Epic UC3.E04: Spraying volume tuned based on detected conditions

The quantity of treatment used is computed intelligently based on speed of tractor, humidity, temperature and detected foliage volume. Treatment has to stop if no plants are detected or while the tractor changes row. In this scenario, farmers do not need to perform any action, the SFD manages the spraying nozzle automatically.

4.3.3.5 Epic UC3.E05: Collect vineyard health data and SFD performance data

As the tractor drives through the winery, the SFD collects data about the used doses of treatment and about the health of the plants (trunk, leaves and fruits). Each data point is matched with the position of the plant using GPS. Images are collected to be later uploaded on the cloud for health monitoring status and yield prediction.

4.3.3.6 Epic UC3.E06: Collected data uploaded to the cloud

After the spraying, Zeno drives the tractor back to the factory. In the background all data collected are sent to the cloud. Data transfer happens in the background.

4.3.3.7 Epic UC3.E07: Farmer views report about collected data

When Chiara (farm manager) opens her computer/smartphone, she sees a summary of the spraying process, including additional information about used chemicals, expected yield, found grape parasites, etc.

4.3.3.8 Epic UC3.E08: Update of Configuration, AI-Models or Application Versions

When a new version of the application/AI model is available, Loredana (Infrastructure Provider & Sysops) can push the new version to the SFDs she manages. SFDs are updated as soon as they are online. Loredana can track centrally the status of each SFD.

4.3.3.9 Epic UC3.E09: Overview of used SFDs and their current status

Loredana (Infrastructure and SysOp), Piotr (Application Manager) and Alexandre (Service Provider) need an overview of used SFD, their status and their location. The reports depend on the role of the users.

4.3.3.10 Epic UC3.E10: Track usage and performance of SFDs centrally

Loredana (Infrastructure and SysOp) and possibly Piotr (Application Manager) need to monitor devices and possibly to centrally modify specific monitoring settings such as log level.

4.3.4 User stories

Field name	Details
ID	UC3.E01.US01
Title	SFD installation (software and hardware)
Priority	Must have
Persona	Loredana the Infrastructure Provider & Sysops
Category	Runtime
Description	As Loredana I need to install software and power up the SFD on the tractor
Need/Purpose	The initial installation is performed by Loredana, who installs the AI-SPRINT framework tools on the SFD, it generates an X.509 certificate and copies it to the device and finally uses the AI-SPRINT tool set to install all software components (containers). Once the software has been installed, it has to be wired to sensors on the tractor and an initial test is performed.
Comment	Eventually the SFD will be built into the tractor, this step is necessary in the experimental phase which is the scope of the AI-SPRINT Use Case

Field name	Details
ID	UC3.E01.US02
Title	SFD protection
Priority	Must have (but out-of-scope for AI-SPRINT)
Persona	Loredana the Infrastructure Provider & Sysops
Category	Runtime
Description	As Loredana I need ensure the durability of the SFD
Need/Purpose	The SFD needs to be protected from any potential failures caused by the environmental conditions
Comment	This story is a must have for the Use Case, but actually not in scope of AI-SPRINT.

Field name	Details
ID	UC3.E01.US03
Title	SFD connectivity to cameras
Priority	Must have (but Out-of-scope for AI-SPRINT)
Persona	Loredana the Infrastructure Provider & Sysops
Category	Runtime
Description	As Loredana I need to connect the existing cameras to the SFD
Need/Purpose	The SFD needs to capture images from the cameras provided on the tractor
Comment	This story is a must have for the Use Case, but actually not in scope of AI-SPRINT.

Field name	Details
ID	UC3.E01.US04
Title	Pair SFD to Smartphone over Bluetooth
Priority	Must have (but out-of-scope for AI-SPRINT)
Persona	Zeno the Factory Factotum
Category	Runtime
Description	As Zeno I am able to pair my Smartphone with the SFD to check its status
Need/Purpose	Each farmer can connect his/her smart phone to the SFD using Bluetooth and can check status of all sensors
Comment	This story is a must have for the Use Case, but actually not in scope of AI-SPRINT.

Field name	Details
ID	UC3.E01.US05
Title	SFD Internet configuration and initial settings
Priority	Must have (but out-of-scope for AI-SPRINT)
Persona	Zeno the Factory Factotum
Category	Runtime
Description	As Zeno I am able to attach the SFD to the tractor and/or check proper cabling by myself
Need/Purpose	The SFD needs to be connected to the Internet and installed on the tractor. The SFD has a WIFI interface, which needs to be linked to the factory WiFi router.
Comment	This story is a must have for the Use Case, but actually not in scope of AI-SPRINT.

Field name	Details
ID	UC3.E01.US06

Title	View paired SDF devices
Priority	Should have
Persona	Alexandre the Application and Service Provider
Category	Runtime
Description	As Alexandre I am able to see which SFD are being paired with which Smartphone
Need/Purpose	Application providers need to know which SFD is being used where and how. Collected information can be used to detect usage patterns and improve application. This story will be implemented using the Monitoring tools and custom metrics.
Comment	

Field name	Details
ID	UC3.E02.US01
Title	Check status of SFD
Priority	Should have
Persona	Zeno the Factory Factotum
Category	Runtime
Description	As Zeno I am about to check if the SFD works properly so that I can start the spraying procedure
Need/Purpose	A very simple app shows the status of SFD components (green, red). In case of malfunctions, the SDF will automatically revert to normal spraying based only on tractors' speed. If the SFD is not working Zeno can either disable Smart Spraying and revert to the default spraying level or replace the SFD with a functioning one or call technical support.
Comment	To increase standardization, the possibility to interact directly with ISOBUS will be explored. Integration with ISOBUS standard would allow integrating the SFD with a wide range of tractor models or manufacturers.

Field name	Details
ID	UC3.E02.US02
Title	Disable SFD
Priority	Should have
Persona	Loredana the Infrastructure Provider & Sysops Possibly also Piotr the Application Manager
Category	Runtime
Description	As Loredana I may disable any SDF remotely
Need/Purpose	If an SDF is broken or stolen, it must be possible to disable it so that it cannot interact anymore with the cloud component of the application
Comment	

Field name	Details
ID	UC3.E02.US03
Title	Publish SFD status via Bluetooth
Priority	Must have (but out-of-scope for AI-SPRINT)
Persona	Application Developer
Category	Design
Description	As a developer I am able to collect the status of the sensors connected to the machine via Robot Operating System (ROS) and publish their status using the Bluetooth device on the SFD
Need/Purpose	A monitoring component running on SFD checks availability of sensors and publishes a status notification over the Bluetooth
Comment	This story is a must have for the Use Case, but actually not in scope of AI-SPRINT. Tools which enable the SFD to detect and publish status information of sensors are within scope of AI-SPRINT. Communication with a paired Bluetooth device is relevant for this Use Case, but not related to AI-SPRINT.

Field name	Details
ID	UC3.E03.US01
Title	Set SFD default spraying settings
Priority	Must have (but out-of-scope for AI-SPRINT)
Persona	Zeno the Factory Factotum
Category	Runtime
Description	As Zeno I may modify the default spraying settings and set a default quantity
Need/Purpose	If something breaks during the spraying process, the spraying procedure reverts to the old one, based on speed only, but not on foliage volume etc.
Comment	This story is a must have for the Use Case, but actually not in scope of AI-SPRINT.

Field name	Details
ID	UC3.E03.US02
Title	Set SFD spraying range settings centrally
Priority	Should have
Persona	Piotr the Application Manager
Category	Runtime
Description	As Piotr I can set boundary values to SFD settings (e.g. max spray quantity between x and y ml./sec)
Need/Purpose	Prevent end users from mistakenly setting completely unrealistic default spraying conditions.
Comment	

Field name	Details
ID	UC3.E03.US03
Title	Read/write global configuration settings on SFD
Priority	Must have
Persona	Paolo the Application Developer
Category	Design
Description	As Paolo I may read and write configuration information globally available across the edge devices
Need/Purpose	Some settings must be globally available to all containers running on the edge device. For example, default spraying quantity will be needed by different functions/containers running on SFD. Developers must be able to read/write them easily. Technically, the underlying functionality could be implemented using environmental variables, configuration files, encrypted secrets etc. The management of this configuration information could be managed using the same tools used for managing container deployments.
Comment	Fleet management capabilities are critical because they would enable using the SFD as a service.

Field name	Details
ID	UC3.E03.US04
Title	Read/Write device settings from the Cloud
Priority	Must have
Persona	Loredana the Infrastructure Provider & Sysops Piotr the Application Manager
Category	Runtime
Description	As Loredana I may read/write from the Cloud the local settings of all SFD
Need/Purpose	All SFD local settings (including default spraying settings) must be visible in the cloud. Some settings can also be overwritten from the cloud to the device.
Comment	

Field name	Details
ID	UC3.E03.US05
Title	Report about used SFD settings
Priority	Nice to have
Persona	Alexandre the Application and Service Provider
Category	Runtime
Description	As Alexandre I can view a report showing which settings farmers use and change
Need/Purpose	Application Providers need to understand how the tool is used in real life so that it can be improved for all customers. Knowing which settings some customers frequently change, gives an insight about possible improvement areas.
Comment	Fleet management capabilities are critical because they would enable using the SFD as a service.

Field name	Details
ID	UC3.E04.US01
Title	Foliage volume estimation
Priority	Must have
Persona	Christine the AI-Expert
Category	Design
Description	As Christine I can write an AI model which detects foliage volume
Need/Purpose	This is a core functionality of the Use Case. The model uses images taken by a 3D camera and an RGB camera to detect the foliage volume. Foliage volume is measured in terms of foliage area for 10 centimeters of canopy length. Foliage volume needs to be computed within a margin of 20%. The model also needs to detect complete absence of foliage, e.g., because some plants have been removed.
Comment	

Field name	Details
ID	UC3.E04.US02
Title	Canopy shape identification
Priority	Should have
Persona	Christine the AI-Expert
Category	Design
Description	As Christine I can write an AI model which detects the shape of the canopy
Need/Purpose	Treatment is based not only on foliage volume, but also on shape of canopy, which can be e.g. V-shaped, U-shaped or O-shaped
Comment	

Field name	Details
ID	UC3.E04.US03
Title	Tractor turning point identification
Priority	Must have
Persona	Christine the AI-Expert
Category	Design
Description	As Christine I can write an AI model which detects if the tractor is turning around and moving from one line to the next one
Need/Purpose	When tractor is turning, spraying must be stopped
Comment	

Field name	Details
ID	UC3.E04.US04
Title	Estimation of products needed
Priority	Must have
Persona	Christine the AI-Expert
Category	Design
Description	As Christine I can write a function which computes the required amount of needed product
Need/Purpose	This is a core functionality of the Use Case. The quantity of needed products is estimated using the foliage volume, the tractor speed, the tractor acceleration and some product-specific input parameters. The function also needs to evaluate when the computed quantity of products has to be used, based on the tractor speed and the position of the image captured by the tractor.
Comment	

Field name	Details
ID	UC3.E04.US05
Title	Publish information about spraying quantity
Priority	Must have
Persona	Paolo the Application Developer
Category	Design
Description	As Paolo I am able to send information about spraying volume to the Electronics controlling the nozzles
Need/Purpose	Once the different AI models have computed foliage volume and canopy shape, a function will use this information together with the tractor speed and acceleration to compute the needed quantity of treatment. This information will be published e.g. using ROS. The Sprayer Node will read the published information and tune spraying quantity accordingly.
Comment	The actual spraying device and its controlling elements are out of scope for AI-SPRINT. It is assumed that the spraying device is ROS compatible and can process information sent by the SFD.

Field name	Details
ID	UC3.E05.US01
Title	Report model performance KPI
Priority	Should have
Persona	Paolo the Application Developer
Category	Design
Description	As Paolo I can define and track performance metrics about SFD
Need/Purpose	The Application Developer (with the help of the AI-Expert and the Application Manager) will define KPIs to track the performance of critical aspects of the application. Application Developers can define custom metrics in their code, AI-SPRINT collects the relevant values and sends them to a cloud-based console.
Comment	

Field name	Details
ID	UC3.E05.US02
Title	Store
Priority	Must have
Persona	Paolo the Application Developer
Category	Design
Description	As Paolo I can ensure data collected by sensors is saved on SFD
Need/Purpose	Data collected by sensor is used on the SFD to detect foliage volume. However, the same data can be used, e.g., to create AI models which estimate yield volume.
Comment	Functionality is needed for the Use Case, tools used will not be developed within AI-SPRINT, but AI-SPRINT could provide support, e.g., through a TOSCA template which spins up the required elements.

Field name	Details
ID	UC3.E05.US03
Title	Tag collected data
Priority	Should have
Persona	Paolo the Application Developer
Category	Design
Description	As Paolo I can write code which tags all collected information using tags I can freely define
Need/Purpose	All sensors (3-D camera, RGB camera etc.) will collect data from the vineyard. For example pictures for which the canopy shape detection model returns low confidence, could be tagged so that they can later uploaded to the cloud and used to improve models. Similarly, pictures on which some illness is detected, could be tagged and shown later to the farmer.
Comment	

Field name	Details
ID	UC3.E06.US01
Title	Selected data to be uploaded to the cloud based on tags
Priority	Should have
Persona	Piotr the Application Manager
Category	Runtime
Description	As Piotr I can decide which information from which SFD can be sent to the Cloud based on tags
Need/Purpose	Sensor data collected by the SFD will be used and deleted. In some cases, the Application Manager will want to keep a subset of the data collected by some SFDs. The application manager must be able to decide at runtime which data will be collected from which specific device
Comment	

Field name	Details
ID	UC3.E06.US02
Title	Centrally modify SFD configuration of specific SFDs
Priority	Must have
Persona	Lucía the Application Architect
Category	Design
Description	As Lucía I can rely on a mechanism which allows me to pull information from and push information to edge devices
Need/Purpose	The SFD must be managed from the cloud. This means that it must be possible to read configuration settings from the SFD (e.g., default spraying settings, set by the farmer), but also to write settings (e.g., overwrite default settings set by the farmer). The mechanism also applies for tagged data (see UC3.E06.US01) or for program code (e.g., Docker image)
Comment	

Field name	Details
ID	UC3.E07.US01
Title	View quantity of product used
Priority	Must have
Persona	Chiara the Factory Owner
Category	Design
Description	As Chiara I am able to see how much quantity of product was used in which lot of the vineyard
Need/Purpose	Farm owners must see how much product was used on which lot of the vineyard. The information send to the sprayer (see UC3.E04.US05) is kept and uploaded to the cloud, together with position of the tractor (detected using GPS).
Comment	

Field name	Details
ID	UC3.E07.US02
Title	View detected illnesses
Priority	Should have
Persona	Christine the AI-Expert
Category	Design
Description	As Christina I can write an AI model which detects grape illness or parasites
Need/Purpose	As the tractor runs, images of grapes and foliage are collected and enriched with GPS coordinates. Once the tractor is back the images are uploaded to the cloud and processed using AI models for disease identification.
Comment	

Field name	Details
ID	UC3.E07.US03
Title	View estimated yield
Priority	Should have
Persona	Christine the AI-Expert
Category	Design
Description	As Christine I can write an AI model which detects fruit size and yield estimation
Need/Purpose	As the tractors runs, collected pictures are processed by a model which tries to identify size of fruits and - using historical data and weather forecast - estimate the yield.
Comment	Accuracy of yield estimation will improve year by year as it depends on data which will be collected as part of the AI-SPRINT project (e.g., number of shape of grapes) but also on data which will be collected and matched in subsequent years (actual yield, weather condition)

Field name	Details
ID	UC3.E07.US04
Title	Publish information about detected illness
Priority	Should have
Persona	Alexandre the Application and Service Provider
Category	Runtime
Description	As Alexandre I can inform my customers that a given illness has been identified in their region
Need/Purpose	If an illness is detected in a vineyard, farmers in the same region should be informed and possibly plan specific protective measures.
Comment	

Field name	Details
ID	UC3.E07.US05
Title	View information published by SFD service provider
Priority	Should have
Persona	Chiara the Factory Owner
Category	Runtime
Description	As Chiara I am able to view illness warnings issued in my region
Need/Purpose	Same as UC3.E07.US04 but from end user point of view.
Comment	

Field name	Details
ID	UC3. E08.US01
Title	Manage passwords and secrets
Priority	Must have
Persona	Paolo the Application Developer
Category	Design
Description	As Paolo I may write code which accesses passwords I might not know
Need/Purpose	Code running in the cloud and on the edge will need access to credentials. In some cases, developers will not have access to the productive environment credentials, which must be securely managed and stored. This functionality is comparable with, e.g., AWS Secrets Manager.
Comment	

Field name	Details
ID	UC3.E08.US02
Title	Continuous deployment mechanism for application
Priority	Must-have
Persona	Paolo the Application Developer Loredana the Infrastructure Provider & Sysops Piotr the Application Manager
Category	Runtime
Description	As Paolo I have a mechanism to continuously deploy new version of the application
Need/Purpose	New versions of the application need to be pushed to the SFD. The functionality is needed by the Application Developer, but also by Infrastructure and Application managers. Infrastructure and Application manager must be able to decide which SFD will receive the new version and must be able to track deployment status.
Comment	

Field name	Details
ID	UC3.E08.US03
Title	Continuous deployment mechanism for application
Priority	Must-have
Persona	Paolo the Application Developer Loredana the Infrastructure Provider & Sysops Piotr the Application Manager
Category	Runtime
Description	As Christine I have a mechanism to continuously re-train and deploy new AI models
Need/Purpose	Same as UC3.E08.US02, but for AI-models
Comment	

Field name	Details
ID	UC3.E08.US04
Title	Tag SFD devices
Priority	Should have
Persona	Piotr the Application Manager
Category	Runtime
Description	As Piotr I have the possibility to tag SFD devices manually
Need/Purpose	The application manager must be able to group devices based on parameters he/she will decide at runtime. Parameters could be, e.g., “VIP Customer” or “Merlot Grape” etc. These tags can be used, e.g., for tuning the deployment process or for reporting purposes.
Comment	

Field name	Details
ID	UC3.E08.US05
Title	Tag SFD devices automatically
Priority	Could have
Persona	Paolo the Application Developer
Category	Design
Description	As Paolo I can set SFD tags programmatically
Need/Purpose	Some tags can be set programmatically, e.g., version of SFD “illness detected”, “Used on tractor moving quickly”, etc. These tags can be used for reporting
Comment	

Field name	Details
ID	UC3.E08.US06
Title	Deployment of AI-model
Priority	Must have
Persona	Piotr the Application Manager
Category	Design-time tools, Runtime tools
Description	As Piotr I have a mechanism which automatically installs new versions of the code and/or the AI models to selected devices
Need/Purpose	The Application Manager takes as input the packaged model and deploys it to SDF (which in this case is an edge device). The deployment tool checks runtime constraints and - if all requirements are met - deploys the model; runtime constraints could be the version of the underlying hardware, the type of sensor etc. The model is deployed on a set of SFDs. The set is selected by the Application Manager either manually or using tags. If the SFD is offline, the model is deployed as soon as the SFD is on-line.
Comment	

Field name	Details
ID	UC3.E09.US01
Title	Reports based on tags or on versions
Priority	Should have
Persona	Loredana the Infrastructure Provider & Sysops Piotr the Application Manager Alexandre the Application and Service Provider
Category	Runtime
Description	As Loredana I have a report which helps me to compare performance of SFD devices with different tags
Need/Purpose	The Service, Application and Infrastructure managers view a list of devices and see status (pending, error, deployed) and the version of the deployed component. Additionally, the dashboard shows some data about application performance, e.g., average inference time for the different models, average confidence value of the AI-models, etc.
Comment	The dashboards for the different users will be different, but the underlying mechanism will be the same.

Field name	Details
ID	UC3.E09.US02
Title	View SFD version
Priority	Should have
Persona	Zeno the Factory Factotum Chiara the Factory Owner
Category	Design
Description	As Zeno I am able to see the version of the SFD software and model
Need/Purpose	Farmers can see the version of the SFD device components. This can be implemented together with UC3.E01.US04
Comment	

Field name	Details
ID	UC3.E10.US01
Title	Collect application logs centrally
Priority	Should have
Persona	Loredana the Infrastructure Provider & Sysops, Piotr the Application Manager
Category	Runtime
Description	As Loredana, I am able to view centrally events logged on each SFD
Need/Purpose	Infrastructure and Application Managers need to see events logged on all SFDs centrally to detect possible runtime errors, performance bottlenecks, unusual usage patterns, etc. Code running on the SFDs logs events to a local text file, which can be synchronized to a cloud management tool.
Comment	

4.3.5 Requirements

Field name	Details
ID*	UC3.Req001
Title*	AI-SPRINT Functions as a Service
Priority*	Must have
Required for (User Story)*	UC3.E01.US04, UC3.E01.US05, UC3.E02.US01, UC3.E02.US03, UC3.E07.US01
Category	Design, Runtime
Description*	Run functions as a service
Rationale*	An AI-SPRINT Edge function is a piece of code which runs on AI-SPRINT edge devices, triggered by some event (similar to AWS Lambda or Azure Functions, but provider agnostic and cloud/edge agnostics).
Tool	OSCAR, PyCOMPSs
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC3.Req002
Title*	AI-SPRINT Cloud Functions Connectors
Priority*	Should have for AWS, Nice to have for Azure
Required for (User Story)*	UC3.E01.US04, UC3.E01.US05, UC3.E02.US01, UC3.E03.US01, UC3.E09.US02
Category	Design, Runtime
Description*	Possibility to deploy AI-SPRINT Cloud functions on AWS Lambda or Azure Functions
Rationale*	AI-SPRINT is provider agnostic and could be used to orchestrate among different providers.
Tool	SCAR
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC3.Req003
Title*	AI-SPRINT Package
Priority*	Must have
Required for (User Story)*	UC3.E01.US01
Category	Runtime, Security
Description*	Software package containing the AI-SPRINT toolset for edge devices
Rationale*	All tools needed on edge (Monitoring, OSCAR, etc.) should be packaged together to simplify initial installation of edge devices. It should also be possible to install certificates, use identity tokens, OpenID connect on Edge devices to secure connection between Edge and Cloud.
Tool	IM
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC3.Req004
Title*	AI-SPRINT Containers as a Service
Priority*	Must have
Required for (User Story)*	UC3.E01.US03, UC3.E01.US04, UC3.E02.US01, UC3.E02.US03, UC3.E07.US01
Category	Design, Runtime
Description*	Run Containers as a Service on the edge and cloud
Rationale*	AI-SPRINT containers run within the AI-SPRINT framework (similar to AWS ECS or Azure Container Service, but runs on Edge and Cloud and is provider agnostic).
Tool	IM, OSCAR
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC3.Req005
Title*	Custom Metrics
Priority*	Must have
Required for (User Story)*	UC3.E01.US06, UC3.E02.US01, UC3.E03.US01, UC3.E03.US02, UC3.E03.US04, UC3.E03.US05, UC3.E04.US05, UC3.E05.US01, UC3.E07.US01, UC3.E07.US04, UC3.E07.US05, UC3.E08.US04, UC3.E08.US05, UC3.E09.US01, UC3.E09.US02, UC3.E10.US01
Category	Design, Runtime
Description*	Create custom metrics to allow monitoring of events and KPIs
Rationale*	Custom metrics are used to monitor all relevant aspects of the application and the infrastructure. Thanks to custom metrics, the monitoring infrastructure can be used to monitor infrastructure, application, models and business usage.
Tool	Monitoring Infrastructure
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC3.Req006
Title*	Edge/Cloud Configuration Settings
Priority*	Must have
Required for (User Story)*	UC3.E02.US02, UC3.E03.US03, UC3.E03.US04, UC3.E07.US04, UC3.E08.US03
Category	Design, Runtime
Description*	Synchronize configuration settings between Edge and Cloud
Rationale*	It must be possible to change application specific configuration settings centrally and update Edge devices accordingly. Similarly, it must be possible to track the value of device-specific settings centrally. Passwords and/or API keys can be shared using this mechanism, i.e. it must be possible to encrypt some configuration settings. Settings can be read/written programmatically.
Tool	IM, Monitoring Infrastructure
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC3.Req007

Title*	Data analysis and modeling
Priority*	Must have
Required for (User Story)*	UC3.E04.US01, UC3.E04.US02, UC3.E04.US03, UC3.E04.US04, UC3.E04.US05, UC3.E07.US02, UC3.E07.US03, UC3.E08.US03
Category	Design
Description*	Analysis of sensor data and creation of AI models
Rationale*	Functionality is needed to create models for detection of foliage volume, of possible grape illness and of size of fruit.
Tool	PyCOMPSs, GPU Scheduler, Performance Models, NAS
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC3.Req008
Title*	Update model and application
Priority*	Must have
Required for (User Story)*	UC3.E04.US01, UC3.E04.US02, UC3.E04.US03, UC3.E04.US04, UC3.E07.US01, UC3.E07.US03, UC3.E08.US02, UC3.E08.US03, UC3.E08.US06
Category	Design, Runtime
Description*	Update code, model version or model parameters on edge
Rationale*	New versions of the application or of the AI model need to be deployed using a continuous deployment mechanism. It must be possible to link the mechanism to a source code management and version control tool like Git. Update of edge devices can be controlled using tags (e.g., only update SFDs used for Merlot grape or used in Piedmont or running a specific version of the hardware).
Tool	IM
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC3.Req009
Title*	Upload files based on rules
Priority*	Must have
Required for (User Story)*	UC3.E05.US02, UC3.E06.US01, UC3.E06.US02
Category	Design, Runtime
Description*	Automatically upload files from edge to cloud based on rules
Rationale*	Inference for illnesses and fruit volume and quantity can be performed in the cloud. It must be possible to upload images or sensor data from edge to cloud. Edge devices will be frequently off-line, the system must detect network availability and start uploading automatically. Rules used could be, e.g., only images collected in a certain region (GPS) or only images whose prediction was in a given confidence interval.
Tool	OSCAR
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	MinIO is used by AI-SPRINT

Field name	Details
ID*	UC3.Req010
Title*	Resources can be tagged
Priority*	Should have
Required for (User Story)*	UC3.E05.US02, UC3.E05.US03, UC3.E06.US01, UC3.E06.US02, UC3.E08.US04, UC3.E08.US05, UC3.E08.US06, UC3.E09.US01, UC3.E09.US02
Category	Design, Runtime
Description*	Possibility to manage meta tags for all resources using a console or an API
Rationale*	Deployment of new versions or models can be controlled by tagging devices and then triggering the deployment only for a subset of the devices.
Tool	Monitoring Infrastructure, IM
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC3.Req011
Title*	Edge/Cloud orchestration mechanism
Priority*	Should have
Required for (User Story)*	UC3.E07.US02, UC3.E07.US03, UC3.E09.US01, UC3.E10.US01
Category	Design
Description*	Allow moving execution of functions or containers between cloud and edge at runtime.
Rationale*	Models for detection of grape illness and grape size will initially run on cloud, the SFD will be used to collect data. In later versions, the model might run on edge and be used to tune the quantity of phytosanitary treatment.
Tool	OSCAR, SCAR
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

Field name	Details
ID*	UC3.Req012
Title*	Off-line/On-line operation of Edge device
Priority*	Must have
Required for (User Story)*	UC3.E01.US04, UC3.E01.US05, UC3.E02.US01, UC3.E03.US01, UC3.E09.US02
Category*	Design, Runtime
Description*	Edge devices must be functional even without any connection to the internet or to a local network.
Rationale*	No connectivity in vineyards is quite normal. The tool must cache sensor data needed in the cloud and - once a connection to cloud services is detected - upload the required information automatically. This mechanism is needed for application management, but also for monitoring and collection of logs.
Tool	IM, Monitoring Infrastructure
Acceptance Criteria	Acceptance criteria will be defined in AI-SPRINT Deliverable D5.1 Evaluation plan at M12.
Supporting materials	AI-SPRINT deliverables D5.3, D5.4 Initial/Consolidated implementation and evaluation; D5.5 Final assessment report, impact analysis, lessons learned and best practices
Tentative scheduling	The requirements implementation schedule will be defined in AI-SPRINT deliverable D5.2 Use Case implementation plan at M12.
Comments	

4.3.6 *Summary and Overview of Requirements*

The requirements from the Farming 4.0 Use Case are summarized in the table below.

Requirements	ID	Design	Runtime	Security
Run functions as a service	UC3.Req001	X	X	
AI-SPRINT Cloud Functions Connectors	UC3.Req002	X	X	
Software package containing the AI-SPRINT toolset	UC3.Req003		X	X
Run containers as a service	UC3.Req004	X	X	
Create custom monitoring metrics	UC3.Req005	X	X	
Synchronize configuration settings between Edge and Cloud	UC3.Req006	X	X	
Data analysis and modeling	UC3.Req007	X		
Update model version or model parameters on Edge	UC3.Req008	X	X	
Automatically upload files from Edge to Cloud based on rules	UC3.Req009	X	X	
Possibility to manage meta tags for all resources using a console or an API	UC3.Req010	X	X	
Allow moving execution of functions or containers between Cloud and Edge at runtime.	UC3.Req011	X		

Table 4.8 - List of Requirements Farming 4.0

4.3.6.1 Design

In terms of Design, requirements are centered around the following key features of AI-SPRINT:

- Possibility to easily create and update AI models
- Possibility to run AI inference on any cloud provider (this is a key differentiation from commercial services such as Azure Custom Vision or AWS Sage Maker)
- Provide an abstraction layer which allows AI inference to run on the cloud, on the edge or on cloud/on-premise hybrid environment
- Provide cloud-like services such as FaaS or Containerization, without having to deal with runtime constraints such as hardware specific or provider specific limitations

4.3.6.2 Runtime

The key runtime features provided by AI-SPRINT and needed in the Farming 4.0 Use Case are:

- Possibility to orchestrate between cloud and edge
- Possibility to manage edge devices centrally, relaying on the same services which also run in cloud environments (e.g., FaaS)
- Intelligent update tools which allow management of edge resources with unstable or intermittent network connectivity
- Flexible monitoring tools which allow monitoring of edge and cloud resources using the same methods

4.3.6.3 Security

The Farming 4.0 Use Case does not present any specific challenge in terms of Privacy or Security: no personal data will be collected or stored and Sensor data will collect data from plants which are almost always on open fields.

Nevertheless, the Farming 4.0 application will make use of the essential AI-SPRINT security features which are required in any modern IT application: data will be encrypted on transfer and at rest, each edge device will have a certificate to secure and authenticate data transfer, management of certificates will be simplified thanks to AI-SPRINT.

4.3.6.4 Out of Scope

As noted above, some user stories are required for the Use Case, but actually not within the scope of the AI-SPRINT framework. AI-SPRINT tools are not about electronics; the quantity of phytosanitary treatment will be tuned using spraying nozzles; the application built with AI-SPRINT will compute the required quantity, but controlling the underlying hardware is not within the scope of it. Hardware management or specific hardware requirements in terms of durability are important, but not in scope. The SFD will communicate with Smartphones, e.g., using a Bluetooth interface. AI-SPRINT does not provide any specific tools for managing the interaction over Bluetooth.

4.3.7 Initial Roadmap

The first tentative road map for the implementation of the Farming 4.0 Use Case is illustrated below in Figure 4.Y:

- Work on data collection and AI-model is strongly constrained by seasonal conditions, it must begin in the first year of the project and must continue through the whole project duration: data will be collected in the period April - September, work on AI models will begin as soon as data is available and will continue until the second half of 2022.
- First experiments with the AI-SPRINT assets will begin as soon as the first beta releases are available. However, the main development work on the SFD itself and on the management and monitoring tools will begin in the second half of 2023, when the tools will be sufficiently mature.

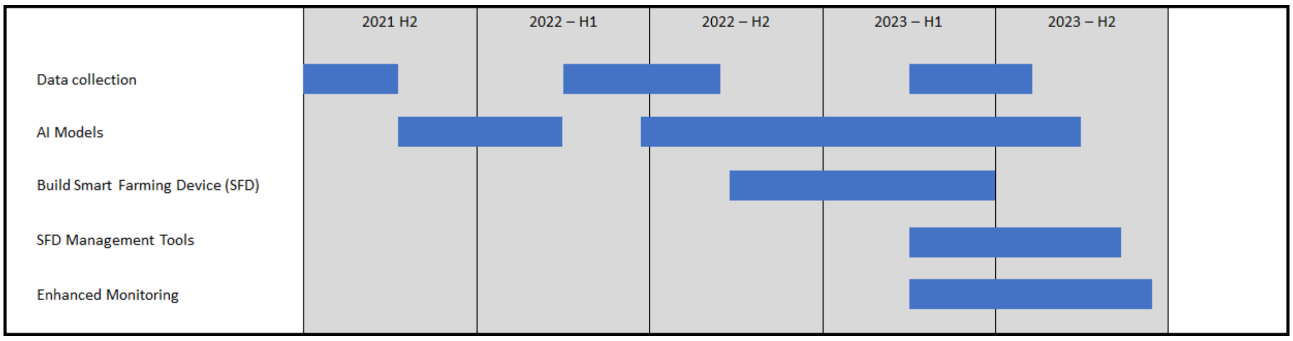

Figure 4.6 - Initial Roadmap Farming 4.0

5. AI-SPRINT Assets Technical Requirements

5.1 Overview of assets

AI-SPRINT vision is to develop an advanced design and runtime framework for developing AI applications running seamlessly in a computing continuum. Existing approaches for developing and deploying software in a computing continuum are not well integrated and leave to the developers the responsibility of: i) partitioning the application and the AI models inference across the full ecosystem, ii) managing explicitly deployment and communication and recovering from failures, and iii) investigating the trade-offs among application performance, security, model accuracy, and data privacy. On the contrary, AI-SPRINT wants to define an integrated approach to simplify application development and runtime management while supporting AI application QoS during the whole lifespan of the application.

In order to address the current limitations, AI-SPRINT aims at delivering tools which cover design and runtime lifecycle and the security and privacy of AI applications. The following sections report the result of the preliminary requirements elicitation and analysis of the main assets that AI-SPRINT will develop. A detailed analysis of the assets interactions and dependencies is reported in the AI-SPRINT deliverable *D1.3 Initial Architecture Design.*

5.2 Design time tools

The objective of the design environment is to provide a layer that abstracts applications from the underlying computing resources, being these edge resources or cloud servers in such a way that the application developer only needs to focus on the actual algorithm and application logic.

In particular, the assets offered to Paolo the Application Developer and to Christine the AI Expert include the Design and Programming Abstractions tool based on the PyCOMPSs parallel programming model and the dislib library; PyCOMPSs will be extended with new constraints annotations to support the definition of performance models. The SPACE4AI-D tool explores and finds an optimal solution for the application component placement problem, according to candidate resources introduced by the AI developer. The AI Models Architecture Search identifies and automates the search of the most accurate deep network which complies with QoS constraints starting from labelled data.

5.2.1 Design and Programming Abstractions

Field name	Details
ID*	DES-REQ001
Title*	Programming model and Design annotation and abstractions
Priority*	Must have
Required for (UC Requirements)*	UC1.Req001, UC1.Req002, UC1.Req003, UC1.Req004, UC1.Req005, UC1.Req008, UC3.Req007
Category*	Design
Description*	Provide a set of programming tools to design AI/ML components and provide quality requirements annotations
Rationale*	The Design Tools will provide a set of predefined AI/ML software components (e.g., traditional ML algorithms such as k-means, linear regression, etc.). The design layer will be based on the PyCOMPSs programming framework but integration with popular frameworks such as PyTorch, TensorFlow, Keras will be also provided. The COMPSs programming model will be extended to define applications following a FaaS model. Quality abstractions will introduce annotations to specify performance parameters for the allocation of tasks to computing continuum resources and security and privacy annotations for data allocation and processing.
Tool	PyCOMPSs; SPACE4AI-D
Acceptance Criteria	Number of AI models implemented using the programming model ≥4 (M24), ≥8 (M30). Percentage of tools integrated into the proposed architecture ≥75% (M18), =100%(M30).
Supporting materials	AI-SPRINT deliverables D2.1, D2.3, D2.5 First/second/final release and evaluation of the AI-SPRINT design tools, D2.2, D2.4 Initial/Final AI-SPRINT design and runtime tools integration.
Tentative scheduling	The first release of the programming model will be released at M12 and will include the definition of quality annotations to specify security policy and performance constraints. The release at M18 will provide an initial integration with the runtime environment and in particular will map quality annotations related to performance constraints to monitoring rules. The release at M24 will provide the final set of abstractions provided by the AI-SPRINT framework, while the final release at M30 will integrate the feedback provided by the industry Use Cases.
Comments	

5.2.2 *Performance Models*

AI-SPRINT will develop novel solutions to model the performance of AI applications running at the full computing continuum stack. The AI-SPRINT performance modelling approach will be mainly based on ML. ML-based performance models will be developed to predict the execution time of AI applications running on heterogeneous multi-devices systems composed of IoT & AI enabled sensors, edge and cloud resources. The tools will provide support and automate AI applications performance profiling. Solutions to identify the ML model providing the highest performance prediction accuracy supporting model selection and hyperparameters tuning will be also developed. The tools will be based on the aMML-library (https://github.com/eubr-atmosphere/a-MLLibrary) which will be extended within the context of AI-SPRINT.

Field name	Details
ID*	DES-REQ002
Title*	Performance modelling
Priority*	Must have
Required for (UC Requirements)*	UC3.Req007
Category*	Design
Description*	The performance model tools will support the Application architect in : 1) profiling the target application to identify its relevant parameters, 2) predict the execution time of the application on unseen configurations (e.g., different number of VMs, different intelligent sensors or deep network model partition).
Rationale*	Once an AI application is designed, performance models can help to evaluate the execution time of the application components before the production deployment or throughout revision cycles. Example of questions that can be answered by performance models include: how many resources (GPUs, memory, CPU threads, etc.) will be required to achieve a given performance target (e.g., training job will end within a due date? the inference time of an AI component will be lower than a given threshold ?).
Tool	Performance models
Acceptance Criteria	Accuracy of performance estimation models for individual components: Average percentage error of the models on estimating the performance of single application components on unseen configuration/scenarios ≤ 30% (M24), ≤ 20% (M30). Accuracy of performance estimation models for distributed applications: Average percentage error of the models for evaluating the end-to-end execution time on unseen configuration/scenarios ≤40% (M24), ≤ 30% (M30).
Supporting materials	AI-SPRINT deliverables D2.1, D2.4 First/Final release and evaluation of the AI-SPRINT design tools, D2.2, D2.5 Initial/Final AI-SPRINT design and runtime tools integration
Tentative scheduling	Initial release of the tools will be provided at M12, final version tentatively developed by M24.
Comments	

5.2.3 AI Models Network Architecture Search

Field name	Details
ID*	DES-REQ003
Title*	Neural Architecture Search Tool
Priority*	Must have
Required for (UC Requirements)*	UC3.Req007
Category*	Design
Description*	The Neural Architecture Tools allow an out-of-the box design of deep neural models directly from data. It allows the design to specify high level requirements in terms of acceptable minimal accuracy (or precision or recall or any similar metrics) and the constraints the obtained model should respect, e.g., minimum inference time, maximum number of flops, maximum memory footprint, maximum power consumption. The tool will perform a Pareto search among the possible models in order to find a suitable tradeoff between machine learning performance requirements and computational ones.
Rationale*	The development of a machine learning model, especially when it is in the form of a deep neural network, requires a long and somehow tedious trial and error process demanded from the experience and sensibility of the designer, not to mention domain knowledge. In the end, the whole process is closer to artcraft than to industrial manufacturing and this hinders somehow the widespread adoption of deep AI neural models for SME which cannot afford hiring AI gurus or are pressed by time to market requirements. Neural architecture search answers this request by providing a tool for the black box development of deep neural models starting from high level requirements and data from the specific application. The tool indeed uses the available data to explore the neural architecture space defined by the user requirements in order to find a suitable tradeoff between such requirements and the model performance. The outcome of this search is a model, or a reduced set of models, which match the required trade off, possibly not being the best possible solution, but providing a feasible solution in a limited time and with limited effort. Such a solution can be further refined by proper fine tuning, and this can be automatized as well.
Tool	NAS
Acceptance Criteria	Number of AI models implemented via Neural Architecture Search ≥4 (M24), ≥8 (M30) possibly also in fields which are not the AI-SPRINT UCs to prove the generality of the approach. Performance gap with respect to manually optimized models <20%.
Supporting materials	AI-SPRINT deliverables D2.3, D2.4 Second/Final release and evaluation of the AI-SPRINT design tools, D2.5 Final AI-SPRINT design and runtime tools integration.
Tentative scheduling	The initial release of the tools will be provided at M24, the final version will be provided by M30.
Comments	

5.2.4 *Application Design Space Exploration*

Field name	**Details**
ID*	DES-REQ004
Title*	Application Design Space exploration
Priority*	Must have
Required for (UC Requirements)*	UC2.Req002,UC2.Req004,
Category*	Design
Description*	The Application Architect submits a description of the system, including application components, with their memory requirements, and edge and cloud resources, with their description in terms of available memory and costs. For each component, the Application Architect specifies the performance demand on the compatible resources. Moreover, she/he also specifies the performance constraints, in the form of response time thresholds. Two types of performance constraints are supported: Local and Global. Local constraints predicate on a single component while global constraints involve a set of components that are running in sequence. Furthermore, the AI developer defines the application components data flow dependencies through a Directed Acyclic Graph and specifies an input exogenous workload. Application components are organized in layers which include a set of candidate resources that can be selected. Data will be transferred across the layers under different network settings.
Rationale*	SPACE4AI-D explores and finds an optimal solution for the application component placement problem (according to candidate resources introduced by the Application Architect) while minimizing the resource cost and satisfying the performance constraints. Moreover, the tool also considers multiple configurations of the components, since a complex deep neural network can be split in different ways. Therefore, SPACE4AI-D selects a configuration for each component and determines on which layer and resource the selected configuration should be placed in order to meet performance guarantees while minimizing the costs. To find the optimal solution, SPACE4AI-D leverages some heuristics such as random greedy, local search and tabu search. The optimal solution includes a description that shows the best configuration for each component and the resource where it should be deployed, and computes the corresponding cost.
Tool	SPACE4AI-D, IM, PyCOMPSs
Acceptance Criteria	Costs reduction of about 20% with respect to first principle solutions (e.g., based on utilization thresholds) and median error between the threshold and the performance of the application at runtime lower than 30%.
Supporting materials	AI-SPRINT deliverables D2.3, D2.4 Second/Final release and evaluation of the AI-SPRINT design tools, D2.5 Final AI-SPRINT design and runtime tools integration
Tentative scheduling	The initial release of the tools will be provided at M24, the final version will be provided by M30.
Comments	

5.3 Runtime tools

The runtime tools consist of a collection of interoperable components that provide the ability to provision computing capabilities from existing cloud-based infrastructures and hardware resources. By building on

DevOps approaches exemplified by Infrastructure as Code tools such as Ansible, this layer provides automated means to deploy Virtual Machines (VMs) that can be dynamically customized at runtime in order to support the requirements of the tools and services employed in AI-SPRINT. By creating automated installation recipes, we facilitate infrastructure uptake and the ability to customize this virtual infrastructure to satisfy the application requirements, and not the other way around. We will adopt TRL-8 components such as the IM and the Elastic Cloud Computing Cluster (EC3) in order to provision these customized virtual infrastructures. Additional Ansible roles and TOSCA templates will be developed to support the requirements coming from the Use Cases. Open-source Functions as a Service (FaaS) tools such as OSCAR and SCAR will be leveraged to support inference of pre-trained Machine Learning models. A monitoring pillar will be established to obtain computational and application-level metrics in order to guide decisions for migration of components, which will be enacted by Krake, and supervised by SPACE4AI-R when performance constraints are introduced, in order to provide better Service Level Objectives.

5.3.1 *Infrastructure Manager*

IM is an open-source tool to deploy customized virtualized application architectures described using the TOSCA (Topology and Orchestration Specification for Cloud Applications) standard on IaaS (Infrastructure as a Service) Cloud platforms.

Field name	Details
ID*	RUN-REQ001
Title*	Application/Service Deployment Requirements
Priority*	Must have
Required for (UC Requirements)*	UC1.Req008, UC2.Req002, UC2.Req005, UC2.Req008, UC3.Req003, UC3.Req004, UC3.Req006, UC3.Req008, UC3.Req010
Category	Runtime
Description*	The application/service to be deployed in a Cloud infrastructure is defined as a TOSCA template. The Application Developer submits the deployment request consisting of the TOSCA template and customizable input parameters, if required, together with the target Cloud infrastructure to the IM and the access credentials to said infrastructure.
Rationale*	The IM receives the TOSCA template and the input parameters and produces the final TOSCA template to perform the automated deployment on the target Cloud infrastructure. The IM interacts with the Cloud infrastructure's API to provision the required Virtual Machines. Then, it is in charge of executing the Ansible Roles to perform the remote configuration and retry in case of any errors, producing a detailed log of the actions performed.
Tool	IM
Acceptance Criteria	Number of back-end resource managers supported, including public clouds (such as Amazon AWS, Microsoft Azure, Google Cloud, Open Telekom Cloud) and on-premise cloud technologies (such as OpenNebula and OpenStack). K3.1≥ 5 (M30).
Supporting materials	AI-SPRINT deliverables D3.1, D3.3, D3.5 First/second/final release and evaluation of the AI-SPRINT runtime environment
Tentative scheduling	The first release and evaluation of the deployment tool will be released at M12 and will include the results of the preliminary tests on the technologies to support the design decisions. At M24, the second release and evaluation will be delivered. The release at M30 will describe the final release integrating the feedback from the use cases.
Comments	

Field name	Details
ID*	RUN-REQ002
Title*	Virtual Infrastructure Query Requirements
Priority*	Must have
Required for (UC Requirements)*	UC2.Req002
Category*	Runtime
Description*	The state of the virtualized infrastructures deployed through the IM is made available to the Application Developer / Application Manager via programmatic and graphical approaches.
Rationale*	The IM server provides a REST API and CLI support to query the state of the virtual infrastructures from a certain programming language and the command-line interface, respectively. It also supports the IM Dashboard, which provides a multi-tenant web-based GUI (Graphical User Interface) to obtain the status of the deployed virtual infrastructures.
Tool	IM
Acceptance Criteria	Number of back-end resource managers supported, including public clouds (such as Amazon AWS, Microsoft Azure, Google Cloud, Open Telekom Cloud) and on-premise cloud technologies (such as OpenNebula and OpenStack). K3.1≥ 5 (M30).
Supporting materials	AI-SPRINT deliverables D3.1, D3.3, D3.5 First/second/final release and evaluation of the AI-SPRINT runtime environment
Tentative scheduling	The first release and evaluation of the deployment tool will be released at M12 and will include the results of the preliminary tests on the technologies to support the design decisions. At M24, the second release and evaluation will be delivered. The release at M30 will describe the final release integrating the feedback from the use cases.
Comments	

5.3.2 EC3

EC3 (Elastic Cloud Compute Cluster) is an open-source tool to deploy self-managed virtual elastic clusters through the IM on several Cloud back-ends. EC3 is a client-side tool for the IM that provides a set of curated TOSCA templates (also supporting RADL, the IM's native language) to deploy several types of clusters such as Kubernetes, SLURM, Apache Mesos, etc. With the help of the CLUES elasticity system, clusters can grow and shrink depending on the workload, typically the number of jobs queued up at the LRMS (Local Resource Management System).

Field name	Details
ID*	RUN-REQ003
Title*	Virtual Elastic Cluster Requirements
Priority*	Must have
Required for (UC Requirements)*	UC1.Req001, UC2.Req005, UC2.Req008
Category	Runtime
Description*	A virtual cluster should be described using a high-level language specification or pre-determined from a set of existing customizable templates so that a user can deploy such clusters on different Cloud back-ends.
Rationale*	Virtual elastic clusters feature the required functionality to support computing capabilities on top of a Cloud infrastructure. These can be used both for training and inference of Deep Learning Models
Tool	EC3, IM
Acceptance Criteria	Number of back-end resources supported by infrastructure deployment. Number of back-end resource managers supported, including public clouds (such as Amazon AWS, Microsoft Azure, Google Cloud, Open Telekom Cloud) and on-premise cloud technologies (such as OpenNebula and OpenStack) ≥ 5 (M30).
Supporting materials	AI-SPRINT deliverables D3.1, D3.3, D3.5 First/second/final release and evaluation of the AI-SPRINT runtime environment; D2.2, D2.5 Initial/final design and runtime tools integration
Tentative scheduling	The first release and evaluation of the deployment tool will be released at M12 and will include the results of the preliminary tests on the technologies to support the design decisions. At M24, the second release and evaluation will be delivered. The release at M30 will describe the final release integrating the feedback from the use cases.
Comments	

5.3.3 OSCAR

OSCAR (Open Source Serverless Computing for Data-Processing Applications) is an open-source tool to support the Functions as Service (FaaS) computing model for file-processing applications packaged as Docker images on an elastic Kubernetes cluster. This cluster can be deployed on multi-Clouds thanks to the IM.

Field name	Details
ID*	RUN-REQ004
Title*	Function Deployment Requirements
Priority*	Must have
Required for (UC Requirements)*	UC1.Req001, UC1.Req009, UC2.Req002, UC2.Req005, UC3.Req001, UC3.Req009
Category	Runtime
Description*	The function to be deployed in a Cloud infrastructure is defined though a high-level format that includes both the performance requirements together with the Docker image and script to be executed upon invocation. The function is deployed to a specific instance of an OSCAR cluster which may be deployed in the edge, on-premises or public Cloud, via the IM
Rationale*	OSCAR supports Docker-based images as the execution runtime for increased support to multiple languages, runtimes, complex application dependencies, legacy software, etc. From the function description, OSCAR creates the corresponding Kubernetes support for that function. Whenever the function is invoked by uploading a file to the object-storage of the cluster, the function will be triggered for processing. OSCAR also supports synchronous invocations via OpenFaaS. OSCAR can also run on low-powered devices such as Raspberry PIs in minified Kubernetes installations to support serverless computing in the edge.
Tool	OSCAR, IM
Acceptance Criteria	Number of back-end resources supported by infrastructure deployment. Number of back-end resource managers supported, including public clouds (such as Amazon AWS, Microsoft Azure, Google Cloud, Open Telekom Cloud) and on-premise cloud technologies (such as OpenNebula and OpenStack) ≥ 5 (M30).
Supporting materials	AI-SPRINT deliverables D3.1, D3.3, D3.5 First/second/final release and evaluation of the AI-SPRINT runtime environment; D2.2, D2.5 Initial/final design and runtime tools integration
Tentative scheduling	The first release and evaluation of OSCAR will be released at M12 and will include the results of the preliminary tests on the technologies to support the design decisions. At M24, the second release and evaluation will be delivered. The release at M30 will describe the final release integrating the feedback from the use cases.
Comments	

5.3.4 SCAR

SCAR (Serverless Container-aware Architectures) is an open-source tool to transparently execute containers out of Docker images in AWS Lambda that is integrated with AWS Batch to execute event-driven highly-scalable file-processing serverless workflows.

Field name	Details
ID*	RUN-REQ005
Title*	Function Deployment on Public Cloud Requirements
Priority*	Must have
Required for (UC Requirements)*	UC3.Req002
Category*	Runtime
Description*	The function to be deployed in a public Cloud infrastructure is defined though a high-level format that includes both the performance requirements together with the Docker image and script to be executed upon invocation. The function is deployed through SCAR on the Amazon Web Services (AWS) infrastructure using AWS Lambda and/or AWS Batch, depending on the application requirements.
Rationale*	SCAR supports Docker-based images as the execution runtime for increased support to multiple languages, runtimes, complex application dependencies, legacy software, etc. From the function description, SCAR creates an AWS Lambda function. Whenever the function is invoked by uploading a file to Amazon S3 (the object-storage of AWS), the function will be triggered for processing. SCAR also supports synchronous invocations via an HTTP endpoint provided by API Gateway. SCAR is also integrated with OSCAR, supporting data-driven serverless workflows in the Cloud continuum.
Tool	SCAR
Acceptance Criteria	Number of back-end resources supported by infrastructure deployment. Number of back-end resource managers supported, including public clouds (such as Amazon AWS, Microsoft Azure, Google Cloud, Open Telekom Cloud) and on-premise cloud technologies (such as OpenNebula and OpenStack) ≥ 5 (M30).
Supporting materials	AI-SPRINT deliverables D3.1, D3.3, D3.5 First/second/final release and evaluation of the AI-SPRINT runtime environment; D2.2, D2.5 Initial/final design and runtime tools integration
Tentative scheduling	The first release and evaluation of SCAR will be released at M12 and will include the results of the preliminary tests on the technologies to support the design decisions. At M24, the second release and evaluation will be delivered. The release at M30 will describe the final release integrating the feedback from the use cases.
Comments	

5.3.5 *Monitoring Tools*

The AI-SPRINT monitoring system will be responsible for monitoring all other AI-SPRINT framework subsystems and the running applications. It will be based on a custom extension of the time-series database InfluxDB (main server) and a set of tools and libraries installed on the client subsystems. These client-side tools will be responsible for gathering data and sending it to InfluxDB. They will be based mainly on Telegraf or Prometheus and - in case of client systems which may work in isolation for a longer time - also on an open source message broker. InfluxDB allows for analyzing a large amount of time-series data, sending alerts and visualizing metrics. All these applications (InfluxDB, Telegraf, Prometheus, RabbitMQ) are open-source software.

Field name	Details
ID*	RUN-REQ006
Title*	Deployment and configuration InfluxDB cluster
Priority*	Must have
Required for (UC Requirements)*	UC1.Req003, UC1.Req006, UC1.Req010, UC2.Req001, UC2.Req003, UC2.Req004, UC2.Req006, UC2.Req007, UC3.Req003, UC3.Req005, UC3.Req006, UC3.Req010, UC3.Req012
Category	Runtime
Description*	InfluxDB must be deployed on the target Kubernetes cluster using the Helm tool. Application and Service Managers manage user accounts using InfluxDB's web interface.
Rationale*	Helm allows to create an InfluxDB cluster on a Kubernetes cluster using predefined, customizable configuration.
Tool	Helm, Kubernetes, InfluxDB, IM
Acceptance Criteria	Influx cluster is able to automatically deploy on Kuberntes cluster (M15). Basic user and authorization structure is also deployed according to configuration =100% (M24).
Supporting materials	AI-SPRINT deliverables D3.2, D3.4 First/final release and evaluation of the monitoring system, D2.2, D2.5 Initial/Final AI-SPRINT design and runtime tools integration.
Tentative scheduling	The initial deployment of the cluster will happen at M12, the final version will be deployed by M24.
Comments	

Field name	Details
ID*	RUN-REQ007
Title*	Configuration of monitoring tools on client subsystems
Priority*	Must have
Required for (UC Requirements)*	UC1.Req003, UC1.Req006, UC1.Req010, UC2.Req001, UC2.Req003, UC2.Req004, UC2.Req006, UC2.Req007, UC3.Req003, UC3.Req005, UC3.Req006, UC3.Req010, UC3.Req012
Category*	Runtime
Description*	Piotr configures and installs necessary monitoring tools in application. In case of applications connected permanently to the Monitoring Subsystem, it will be Telegraf or Prometheus (plus appropriate configuration). In case of applications which can work off-line for a longer time (hours, days) gathering data on the client side must also contain components responsible for buffering monitoring data: RabbitMQ and additional Telegraf instance responsible for sending data to main InfluxDB instance when connection is established. On the other hand, Paolo can use InfluxDB's client software library for a specific programming language, in order to directly send metrics to InfluxDB. Piotr can define custom alerting rules, which check metrics sent by various subsystems, in order to notify the application developed by Paolo. Piotr and Paolo can also view and visualize gathered metrics using the provided web UI.
Rationale*	In order to be monitored, client systems must contain components responsible for sending data to the Monitoring Subsystem.
Tool	Telegraf, Prometheus, InfluxDB client libs
Acceptance Criteria	Monitoring subsystem can be deployed on provided infrastructure and can receive and store at least 99% of sent metrics.
Supporting materials	AI-SPRINT deliverables D3.2, D3.4 First/final release and evaluation of the monitoring system, D2.2, D2.5 Initial/Final AI-SPRINT design and runtime tools integration.
Tentative scheduling	The initial release of the tools will be provided at M12, the final version will be provided by M24.
Comments	

Field name	Details
ID*	RUN-REQ008
Title*	Gathering data
Priority*	Must have
Required for (UC Requirements)*	UC1.Req003, UC1.Req006, UC1.Req010, UC2.Req001, UC2.Req003, UC2.Req004, UC2.Req006, UC2.Req007, UC3.Req003, UC3.Req005, UC3.Req006, UC3.Req010, UC3.Req012
Category*	Runtime
Description*	Piotr registers and manages security credentials needed by client systems to send values of metrics. Metrics are sent to InfluxDB via Telegraf or Prometheus or by application using appropriate InfluxDB's client library.
Rationale*	Client systems must be authorized in order to communicate with InfluxDB.
Tool	InfluxDB, Telegraf, Prometheus, InfluxDB client libs, SCONE
Acceptance Criteria	Average CPU consumption of the monitoring probes for different sampling periods≤ 5% (M24). At least 99% of metrics sent are stored.
Supporting materials	AI-SPRINT deliverables D3.2, D3.4 First/final release and evaluation of the monitoring system, D2.2, D2.5 Initial/Final AI-SPRINT design and runtime tools integration.
Tentative scheduling	The initial release of the tools will be provided at M12, the final version will be provided by M24.
Comments	

Field name	Details
ID*	RUN-REQ009
Title*	Data analysis and alerting
Priority*	Must have
Required for (UC Requirements)*	UC1.Req003, UC1.Req006, UC1.Req010, UC2.Req001, UC2.Req003, UC2.Req004, UC2.Req006, UC2.Req007, UC3.Req003, UC3.Req005, UC3.Req006, UC3.Req010, UC3.Req012
Category*	Runtime
Description*	Gathered metrics can be used to detect failures and anomalies. External systems can be notified when such events occur. Users can view and query metrics using provided web GUI.
Rationale*	Gathered metrics must help analyze system status.
Tool	InfluxDB
Acceptance Criteria	Average CPU consumption of the monitoring probes for different sampling periods≤ 5% (M24). At least 99% of metrics sent are stored.
Supporting materials	AI-SPRINT deliverables D3.2, D3.4 First/final release and evaluation of the monitoring system, D2.2, D2.5 Initial/Final AI-SPRINT design and runtime tools integration.
Tentative scheduling	The initial release of the tools will be provided at M12, the final version will be provided by M24.
Comments	

5.3.6 *Federated Learning*

Federated learning allows the distributed training of models among different facilities, limiting if not avoiding completely the exchange of data, especially when privacy concerns are in place and data cannot be moved/exchanged at all.

Field name	Details
ID*	RUN-REQ010
Title*	Federated Learning
Priority*	Must have
Required for (UC Requirements)*	UC1.Req003, UC1.Req005, UC1.Req006
Category*	Runtime
Description*	This tool will provide a means to specify criteria for safe distributed model updates by techniques such as gradient compression and anonymization. It will also use network partition strategies in order to limit the amount of information exchanged and the need for raw data and full model exchange.
Rationale*	AI-SPRINT will leverage on its the design and runtime tools in order to ease the implementation and deployment of federated learning algorithms on computing continua. Indeed, federated learning assets will leverage on the AI-SPRINT assets to enhance current techniques for distributed privacy preserving learning. Criteria for evaluation of data privacy preservation will be defined and tools for their guarantee will be developed, e.g., epsilon privacy stochastic gradient, when training in the computing continuum.
Tool	Federated Learning
Acceptance Criteria	Federated learning approach which will allow privacy preservation validated on the personalized healthcare Use Case.
Supporting materials	AI-SPRINT deliverables D3.3, D3.5 Second/Final release and evaluation of the runtime environment, D2.5 Final AI-SPRINT design and runtime tools integration.
Tentative scheduling	The initial release of the tools will be provided at M24, the final version will be provided by M30.
Comments	

5.3.7 *Scheduling for Accelerated Devices*

The goal of the GPU scheduling tool is to determine the optimal configuration to run DL training jobs on a given system, characterized by GPU-accelerated nodes or Virtual Machines. The scheduler receives job submissions, profiles them to determine the expected execution times with the available resources and returns the schedule that minimizes execution costs under deadline constraints.

Field name	Details
ID*	RUN-REQ011
Title*	Training Experiment requirements
Priority*	Must have
Required for (UC Requirements)*	UC3.Req007
Category*	Runtime
Description*	The model to be trained is already defined in a Docker container. The AI expert submits the experiment (training job) in a private cluster or public cloud, specifying the deadline/due date for the training to end.
Rationale*	The GPU scheduler manages multiple experiments minimizing the training costs (in terms of energy for the private data center or public cloud operating cost) including the tardiness cost if not enough resources are available. The GPU scheduler decides, possibly, the VM configuration, how many GPUs to assign to the experiment execution and to co-locate multiple experiments on the same physical node/VM. The GPU scheduler predicts training jobs execution time under different configurations through performance models. It can also decide to preempt some experiments if there are not enough resources and re-schedule them when other experiments will complete. The GPU scheduler periodically re-optimizes running experiments to cope with deviation with the predicted time from the performance models and the actual execution time. Remote GPUs access can be also supported through rCUDA.
Tool	GPU scheduler, Performance models, rCUDA.
Acceptance Criteria	Costs savings at least 20% with respect to first principle methods (e.g., earliest deadline first, priority scheduling, first in first out) and job deadline violations below 30%.
Supporting materials	AI-SPRINT deliverables D3.3, D3.5 Second/Final release and evaluation of the runtime environment, D2.5 Final AI-SPRINT design and runtime tools integration.
Tentative scheduling	The initial release of the tools will be provided at M24, the final version will be provided by M30.
Comments	

5.3.8 *Privacy Preserving Continuous Training*

Field name	Details
ID*	RUN-REQ012
Title*	Continuous training
Priority*	Must have
Required for (UC Requirements)*	UC1.Req002, UC1.Req003, UC1.Req005, UC1.Req006, UC1.Req010
Category*	Runtime
Description*	A general-purpose framework towards continuous integration of AI model updates and code changes into the deployment environment. Two possible kinds of updates are foreseen: i) the AI model structure is unchanged (only weight matrices and internal state initialization, in the case of models with memory, are updated); ii) the AI model structure changes because of retriggering of the. Federated learning will be used as well as model partitioning, indeed, via model partitioning only high-level features can be exchanged respecting the privacy of individual users.
Rationale*	AI applications frequently operate in dynamic environments that change over time and models must be continually updated to capture the most recent data trends. This requires a periodic exchange of data and models from the edge to the cloud and back. This has to be allowed in a seamless way while preserving privacy of users and companies involved in the use of AI-SPRINT applications.
Tool	Network Partitioning (module of SPACE4AI-D), Federated Learning, IM
Acceptance Criteria	Acceptance criterion will be defined at M12 when the tool development will start.
Supporting materials	AI-SPRINT deliverables D3.3, D3.5 Second/Final release and evaluation of the runtime environment, D2.5 Final AI-SPRINT design and runtime tools integration.
Tentative scheduling	The initial release of the tools will be provided at M24, the final version will be provided by M30.
Comments	

5.3.9 *Programming Framework Runtime*

Field name	Details
ID*	RUN-REQ013
Title*	Parallel execution of application components
Priority*	Must Have
Required for (UC Requirements)*	UC1.Req003, UC1.Req006, UC1.Req008, UC1.Req009, UC2.Req002, UC3.Req001
Category*	Runtime
Description*	User needs to execute an application developed through the programming interfaces on the edge-to-cloud continuum.
Rationale*	PyCOMPSs runtime provides an interface to execute applications with automatic parallelization. The PyCOMPSs is able to distribute the tasks on the available resources considering the constraints provided by the application developer. A FaaS execution model is also foreseen, through the integration with OSCAR.
Tool	PyCOMPSs, OSCAR
Acceptance Criteria	Acceptance criterion will be defined at M12 when the tool development will start
Supporting materials	AI-SPRINT deliverables D3.1, D3.3, D3.5 Second/Final release and evaluation of the runtime environment, D2.5 Final AI-SPRINT design and runtime tools integration.
Tentative scheduling	The initial release of the tools will be provided at M24, the final version will be provided by M30.
Comments	

5.3.10 *Application Reconfiguration*

Field name	**Details**
ID*	RUN-REQ014
Title*	Application Reconfiguration
Priority*	Must have
Required for (UC Requirements)*	UC1.Req008, UC2.Req001, UC2.Req004, UC2.Req006, UC2.Req008
Category*	Runtime
Description*	The optimal solution identified by the design time tool will be periodically reevaluated in order to account for load variations. These can, indeed, lead to resources saturation or underutilization, having a possibly strong impact on the components response times and the costs predicted at design time.
Rationale*	SPACE4AI-R will evaluate the current status of the system, in terms of incoming loads and response times of the different components. According to this, it will provide an updated solution in terms of components configuration and their deployment on the available resources, optimizing components response times and execution costs. In particular, the type of cloud Virtual Machines selected at design time remains fixed. However, SPACE4AI-R will possibly prescribe to scale resources, adding or removing Virtual Machines if needed. Moreover, components can be migrated from edge devices to cloud or from cloud to FaaS configurations, as well as be deployed with a different configuration, in terms of neural network partition.
Tool	SPACE4AI-R, Krake, IM
Acceptance Criteria	Adaptation mechanisms: Adaptation mechanisms implemented ≥2 (M24), ≥3 (M30). Reconfiguration delay (application unavailability due to reconfiguration): Time an application will not be responsive during the reconfiguration of the infrastructure due to the deployment of new services or the scaling-up/down of the resources between 0 and 5 seconds (M30). Runtime deadline violations: Median deadline violation of runtime management solutions and schedulers (e.g., training time larger than the deadline)≤35% (M30).
Supporting materials	AI-SPRINT deliverables D3.1, D3.3, D3.5 First/Second/Final release and evaluation of the runtime environment, D2.2, D2.5 Initial/Final AI-SPRINT design and runtime tools integration.
Tentative scheduling	The initial release of the tools will be provided at M12 providing mechanisms for the runtime migration of application components. The release at M24, will provide advanced policies for triggering the migration. The final version will be provided by M30, supporting also the change in the partition of deep neural network models.
Comments	

5.4 Security tools

The security tools used within AI-SPRINT target to provide a secure environment for all applications and processes involved in AI operations such that at no point of time confidential information can be accessed by unauthorized personnel. To achieve this goal, the project will use the SCONE framework as a foundation and extend it such that it provides Trusted Execution Environments also for CPU vendors other than Intel (SGX) as well as for edge devices. Furthermore, the SCONE platform will be complemented with measures to provide a secure patch management through trusted compilation, a rollback protection mechanism as well as secure process attestation to mitigate currently existing attacks such as version downgrade and relay attacks.

5.4.1 TEE and Code Attestation

Field name	Details
ID*	SEC-REQ001
Title*	Trusted Execution of applications and processes
Priority*	Must have
Required for (UC Requirements)*	UC1.Req004, UC1.Req007
Category*	Security
Description*	AI-SPRINT ensures that applications run in so called enclaves in case the hardware/the devices the application runs on provides these capabilities.
Rationale*	In AI-SPRINT we have a heterogeneous infrastructure comprising cloud resources as well as edge devices. Our approach is security by design which means we target that applications are always protected depending on the provided hardware capabilities of the underlying nodes. Although some edge devices will not provide so called trusted execution environments that provide confidentiality as well as integrity protection for application data as well as code, we can still leverage mechanisms such as filesystem as well as network encryption and integrity mechanisms. This provides some basic protection which is often sufficient for edge devices as they provide other means of isolation that ensure integrity as well as confidentiality protection for the processes itself.
Tool	SCONE
Acceptance Criteria	Memory snapshots for processes utilizing TEEs will fail. Network communication as well as files cannot be observed by third party processes. Runtime overhead for components with good locality (i.e., paging is within 250 page faults/s)≤30% (M30).
Supporting materials	AI-SPRINT deliverables D2.2, D2.4 Initial/Final AI-SPRINT design and runtime tools integration, D4.1, D4.2, D4.3 Initial/Second/Final release and evaluation of the security tools.
Tentative scheduling	The first release of the TEE and code attestation setup tailored to the applications and runtime components of AI-SPRINT will be available at M12 and will include the use of the file and network protection shield as well as remote attestation. The release at M24 will provide a first prototype for TEEs other than Intel SGX such as confidential GPUs, and patch management, while the final release at M30 will support in a stable manner other TEE vendor as well as full evaluation of its capabilities and performance.
Comments	

5.4.2 Secure Boot

Field name	Details
ID*	SEC-REQ002
Title*	Secure Boot and TPM
Priority*	Must have
Required for (UC Requirements)*	UC1.Req004, UC1.Req007

Category*	Security
Description*	AI-SPRINT ensures that VMs run on securely booted as well as TPM-enabled instances, automated update procedures are implemented, edge cloud nodes as well as servers are always automatically securely booted, and known secure boot policies are followed to address known security vulnerabilities.
Rationale*	Secure Boot and TPM are security measures that have known and unknown vulnerabilities and must be well configured to provide adequate protection. Our approach will be to ensure that every single configuration and update aspect, as well as Secure Boot itself, is enabled and running at boot times to ensure a high level of security. Small gaps can lead to malicious behavior of the booted software. Edge devices in particular, where Secure Boot can be enabled, should be included in an automated test procedure to boot securely.
Tool	AI-SPRINT Secure Boot module
Acceptance Criteria	Attempts to boot a system. The boot process is interrupted and the OS is not booted. The operating system might fail to boot due to a malicious infection. Look for malware or incorrect configurations.
Supporting materials	AI-SPRINT deliverables D2.2, D2.4 Initial/Final AI-SPRINT design and runtime tools integration, D4.2, D4.3 Second/Final release and evaluation of the security tools.
Tentative scheduling	The first release of the Secure Boot implementation will be available at M24 (D4.2). The second and final release will be available at M30 (D4.3)
Comments	

5.4.3 *Secure Networks*

Field name	Details
ID*	SEC-REQ003
Title*	Microsegmentation in the 5G Local Breakout
Priority*	Must Have if using 5G Local Breakout
Required for (UC Requirements)*	UC2.Req001-UC3.Req009 in case of MEC deployments
Category*	Security
Description*	The module implements a permit-list based traffic filtering between the 5G egress interface and the service entry point in the compute cluster. These filters prevent lateral movement between the slices in the 5G network and between the services implemented in the cluster. The tool derives policies from AI-SPRINT secure policy documents.
Rationale*	In any context where 5G Mobile Edge Computing (MEC) is used, either when the MEC is private or it is public, when traffic exits the 5G network the identity of the device is lost and the traffic stream must be re-authenticated at the ingress in the computing cluster. This makes it possible for an attacker to move across slices and across services.
Tool	AI-SPRINT Network security module
Acceptance Criteria	Functional requirement: the tool prevents lateral movement, i.e. traffic coming from a 5G user device is not allowed to connect to the entry-point for a service which is not authorized. Performance requirement: the additional checks add a small performance penalty to user plane traffic (<10% of total end-to-end delay). When new traffic flows are added or removed, there may also be a small performance penalty (M24: <25% of total end-to-end delay; M30 <10% of total end-to-end delay).
Supporting materials	AI-SPRINT deliverables D2.2, D2.4 Initial/Final AI-SPRINT design and runtime tools integration, D4.1, D4.2, D4.3 Initial/Second/Final release and evaluation of the security tools.
Tentative scheduling	The release at M12 will provide the implementation of user plane functions; the release at M24 will provide the implementation of control plane functions; the release at M30 will provide monitoring functions using In-Band Network Telemetry.
Comments	

6. Summary of Technical Requirements

6.1 Use Case Requirements summary

Tables 6.1 and 6.2 provide a summary of the use of the AI-SPRINT tools according to the Use Case and to the Personas identified in this document. As Table 6.2 shows, the 100% of AI-SPRINT tool will be validated by at least one Use Case.

Tool	UC1 - Personalized Healthcare	UC2 - Maintenance and Inspection	UC3 - Farming 4.0
PyCOMPSs	X		X
SPACE4AI-D	X	X	
Performance Models	X	X	X
Network Architecture Search	X		X
IM	X	X	X
SPACE4AI-R	X	X	
EC3	X	X	
OSCAR	X	X	X
SCAR	X	X	X
Monitoring Infrastructure	X	X	X
Federated learning	X		
Scheduling for Accelerated Devices			X
rCUDA	X	X	X
Krake		X	X
SCONE & Security Tools	X	X	X

Table 6.1 - Tools usage for Use Cases

Tool	Lucia-App. Architect	Paolo-App. Developer	Christine-AI-Expert	Piotr-App. Manager	Loredana-Infra & Sysops	Alexandre-App&Service Provider
PyCOMPSs	X					
SPACE4AI-D	X	X				
Performance Models	X		X			
Network Architecture Search			X	X		
IM		X		X	X	
SPACE4AI-R					X	
EC3		X			X	
OSCAR	X					
SCAR	X					
Monitoring Infrastructure		X		X	X	X
Federated Learning			X			
Scheduling for Accelerated Devices	X		X			
rCUDA			X			
Krake					X	
SCONE & Security Tools	X		X	X		X

Table 6.2 - Tools usage for Personas

6.2 Design Time Tools Development plan

This section summarizes the development plan of AI-SPRINT design time assets that will be developed under work package 2 - Design time tools.

6.2.1 *Development plan*

Below is the work allocation for the first year (up to M12) within WP2 plus prospects beyond it. For the first year we are planning to introduce support for different AI frameworks and tools using an iterative approach. Details are given in the month and Milestone-based list below.

M1: Start of the task T1.1 Architecture Definition

M4: Start of the task T2.1 Programming Model Interface, T2.2 AI Models Architecture Design, Optimization and Redesign, T2.3 Cloud and Edge Resource Computing Requirements Evaluation and Performance Models

M6: Creation of a code repository for new assets developed within AI-SPRINT (https://gitlab.polimi.it/ai-sprint)

MS1 (M6):

Definition of initial version of requirements, architecture, and integration approach (see AI-SPRINT deliverable D7.1)

M7: Start of the task T2.4 Application Design Space Exploration and Resource Selection, Design Time Tools and Runtime Environment Integration

MS2 (M12):

- Development of the first release of the programming models including the definition of quality annotations to specify security policy and performance constraints.
- Initial release of the Performance modelling tools (profiling of inference on edge devices and training on GPU clusters). Initial evaluation of the accuracy of performance models on benchmarking applications.

MS3 (M18):

- Initial integration of design abstractions and quality annotations with the runtime environment.
- Mapping of the quality annotations related to performance constraints to monitoring rules.

MS4(M24):

- Final set of design abstractions provided by the AI-SPRINT framework implemented.
- Final release of the Performance modelling tools and initial evaluation of their accuracy on industry Use Cases.
- Initial release of the design time exploration tools and NAS.

MS5(M30):

Final versions of WP2 tools are implemented and integrate the changes according to the feedback provided by the industry Use Cases (developed in WP5) at M24.

6.3 Runtime Tools Development plan

This section summarizes the development plan of AI-SPRINT runtime tools that will be developed/evolved under work package 3 - Runtime environment

6.3.1 *Development plan*

This section describes the work plan for the first year (up to M12) within WP3 and the work planned to be carried out from that point on. The first year will be devoted to achieving the initial development cycle completed towards the first release of the main AI-SPRINT components as independent tools. Details are given in the month and Milestone-based list below.

M4: Start of T3.1 Continuous Deployment, T3.2 Programming Framework Runtime and Application Reconfiguration, T3.3 Application, Infrastructure and Edge Monitoring

M7: Start of the task T3.5 Scheduling for Accelerator Device

M10: Start of the task T3.4 Privacy preserving continuous training

MS1 (M12):

- First release and evaluation of the runtime environment, which will be a prototype addressing the Continuous Deployment and the Programming Framework Runtime as well as the results of preliminary tests on the technologies to support the design decisions.
- First release and evaluation of the monitoring system, which will include the monitoring service, the database of monitoring rules and the basic probes for the low-level monitoring, together with its evaluation.

MS2 (M24):

- Second release and evaluation of the runtime environment, a prototype including the first version of the Scheduling for accelerator devices and of the privacy preserving continuous training.
- Final release and evaluation of the monitoring system, which will provide all the envisioned features, including the high-level application metrics, as well as the probes for the edge devices.

MS3 (M30)

Final versions of WP3 runtime tools are implemented and integrate the changes related to the feedback provided by the industry Use Cases (developed in WP5) at M24.

6.4 Security Tools Development plan

This section summarizes the development plan of AI-SPRINT security tool assets that will be developed under work package 4 - Security tools.

6.4.1 Development plan

In the following, we detail the work planned for the first year (up to M12) within WP4 and the work we plan to perform beyond this initial phase. The first year's plan is to setup a common security policy format as well as implementing the required mechanisms needed to run applications and processes in Trusted Execution Environments as well as provide protection for file system accesses as well as network communication. Details are given in the month and Milestone-based list below.

M4: Start of the task T4.1 Data Security, T4.2 Application Execution Security, T4.3 Network Security

M7: Start of the task T4.4 Patch Management and Secure Boot

MS2 (M12):

- Development of the first release of the security tools which includes a first release of the code and network security components with basic TEE support for security based on security policies that have been setup beforehand.

MS4(M24):

- Second release of the security tools with an initial version for patch management and secure boot. This milestone will also include an initial assessment of the tools behavior as well as runtime performance.

MS5(M30):

Final versions of WP4's security tools are implemented and integrated in the AI-SPRINT stack and fully evaluated with respect to their capabilities and performance. The changes related to the feedback provided by the industry Use Cases (developed in WP5) at M24 are also implemented.

7. Conclusions

This Document has described the AI-SPRINT Use Cases and platform requirements with initial architecture description, personas and tools requirements.

The three Use Cases from various sectors (healthcare, green energy and farming) will focus on demonstrating the AI-SPRINT goal to define and provide a novel framework for developing and operating AI applications, together with their data, exploiting computing continuum environments, which include resources from the edge up to the cloud.

The requirements elicitation process has involved all technical work packages representatives for identifying and documenting the expectations and initial asset requirements of the AI-SPRINT platform and its three Use Cases.

The project identified six main actors for the AI-SPRINT framework and eight actors specific for the Use Cases with a total of 15 epics and 51 user-stories. This analysis led to identifying 52 requirements related to Design-Time, Runtime and Security that will be used as starting points for the development of the AI-SPRINT components as planned in the technical work packages.