ECSEL Joint Undertaking
Electronic Components and Systems for European Leadership

iMOCO4.E

Intelligent Motion Control under Industry 4.E

D2.4 General specification and design of IMOCO4.E reference framework

Due Date: M17 – 2023-01-31

Abstract:

In this deliverable, we explain the general specification and design of the IMOCO4.E reference framework. The IMOCO4.E reference framework is explained at an abstract level. The IMOCO4.E reference framework is extensible, open and modular. The refinements and instantiations of this reference framework for the various industrial cases – Pilots, Demonstrators and Use-cases – are also elaborated. This deliverable also provides a functional breakdown of the IMOCO4.E architecture, from embedded control hardware at the edge to cloud-based services.

The project has received funding from the Electronic Component Systems for European Leadership Joint Undertaking, under Grant Agreement n°101007311. This Joint Undertaking receives support from the European Union's Horizon 2020 research and innovation programme and Netherlands, Czech Republic, Latvia, Spain, Greece, Portugal, Belgium, Italy, France, Ireland

Project Information

Grant Agreement Number	**101007311**
Project Acronym	IMOCO4.E
Project Full Title	Intelligent Motion Control under Industry 4.E
Starting Date	1st September 2021
Duration	36 months
Call Identifier	H2020-ECSEL-2020-2-RIA-two-stage
Topic	ECSEL-2020-2-RIA
Project Website	https://www.imoco4e.eu/
Project Coordinator	Arend-Jan Beltman
Organisation	SIOUX TECHNOLOGIES BV (SIOUX)
Email	Arend-Jan.Beltman@sioux.eu

Document Information

Work Package	WP2 Business requirements and reference system architecture
Lead Beneficiary	ITEC B.V.
Deliverable Title	General specification and design of IMOCO4.E reference framework
Version	2.0 (final, after formal review)
Date of Submission	31/01/2023
Author(s)	Sajid Mohamed (ITEC), sajid.mohamed@itecequipment.com
Contributor(s)	Gijs van der Veen (ITEC), gijs.van.der.veen@itecequipment.com All Pilot owners (SIOUX, ITEC, CRIT, PMS, NORMET) Demonstrator owners (PCL, ECS, STL, MADARA) Use-case owners (WEG (GDM), FAGOR, Tyndall, UWB) Many partners (ADI, EDI, EMD, EXE, GNT, VTT)
Internal Reviewer(s)	Hans Kuppens (SIOUX), hans.kuppens@sioux.com Martin Čech (UWB), mcech@kky.zcu.cz

Document Classification	**Draft**				**Final**		X	
Deliverable Type	**R**	X	**DEM**		**DEC**		**OTHER**	
Dissemination Lever	**PU**	X	**CO**		**CI**			

History					
Version	**Issue Date**	**Status**	**Distribution**	**Author**	**Comments**
0.1	13/12/2022	Draft	PU	ITEC	Table of contents, structure, initial content
1.0	20/01/2023	Draft	PU	ITEC	Initial complete draft
1.1	27/01/2023	Draft	PU	ITEC	Addressing reviewer comments' and updates – added Figure 1, Section 3.6, updated Figure 8, Figure 13, Figure 17, Figure 18, Figure 20
1.2	30/01/2023	Draft	PU	ITEC	Editing, Added Section 4.9 (UWB), Section 4.10 (PCL), 4.12 (AB STILL) and updated Section 4.8
2.0	31/01/2023	Final	PU	ITEC	Final version for dissemination

Type of Contribution	
Partner	**Description of Contribution to Contents**
ITEC	Ideation, reference framework design and specifications, reference framework viewpoints, preparation and editing of the document
Pilot owners, Demonstrator owners, Use-case owners, Many partners	Greenfield reference architectures, framework refinements, reviewing respective content
GNT	Data management viewpoint: Figure 3
SIOUX	Digital twin viewpoint: Figure 6, reviewing

Table of Contents

List of Figures 5
Abbreviations 6
Executive Summary 8
Keywords 8
1. Introduction 9
1.1 Purpose of the document 9
1.2 Structure of the document 10
1.3 Intended readership 10
2. Design of the IMOCO4.E reference framework 11
2.1 Gathering the greenfield architectures from industrial users 12
2.2 Requirements verification, traceability and fulfillment 13
3. The IMOCO4.E reference framework and specifications 14
3.1 Overview 14
3.2 Architecture viewpoint 14
3.2.1 Functional breakdown of the framework and its overall specifications 14
3.3 Data management viewpoint 16
3.3.1 Versioning – source code, data and model management over the life cycle 17
3.4 AI viewpoint 18
3.5 Digital twin viewpoint 20
3.6 IMOCO4.E objectives and the reference framework 22
4. Refinements of the IMOCO4.E reference framework 23
4.1 Pilot 1 Tissector architecture refinement 23
4.1.1 Pilot 1 architecture refinement for simulation using VR and DT 24
4.2 Pilot 2 reference framework refinement for the ADAT machine 24
4.3 Pilot 3 architecture refinement in the absence of a machine prototype 26
4.4 Pilot 4 architecture refinement with CI/CD infrastructure 27
4.5 Pilot 5 architecture 29
4.6 UC1 architecture refinement 30
4.7 UC2 architecture refinement 31
4.8 UC3 architecture for remote teleoperation 32
4.9 UC4 architecture refinement 34
4.10 Demonstrator 1 architecture refinement 35
4.11 Demonstrator 2 architecture refinement 36

4.12 Demonstrator 3 architecture refinement 37
4.13 Demonstrator 4 architecture refinement 39
5. Conclusions 40
References 41

List of Figures

Figure 1 Flow diagram for the suggested way-of-working for the industrial users 12
Figure 2 Architecture viewpoint of the IMOCO4.E reference framework 14
Figure 3 Data management viewpoint for the IMOCO4.E reference framework (from D5.1 [7]). 16
Figure 4. AI viewpoint with BB interactions 18
Figure 5. AI viewpoint during a machine lifecycle (from D2.3) 19
Figure 6 Digital twin outline for an intelligent motion control system (from D6.1) 20
Figure 7. Digital twin viewpoint during a machine lifecycle (from D2.3) 21
Figure 8. Digital twin viewpoint with BB interactions 22
Figure 9 Pilot 1 Tissector architecture refinement from Sioux. The envisioned BBs that will be integrated are annotated. 23
Figure 10 Pilot 1 Tissector simulation infrastructure based on the refined reference framework. 24
Figure 11 Pilot 2 refinement of the reference architecture for the ADAT machine. 25
Figure 12 Pilot 3 architecture and example of refinement in the absence of a machine prototype 26
Figure 13 Pilot 4 architecture refinement with CI/CD infrastructure 28
Figure 14 Pilot 5 architecture refinement 29
Figure 15 Use Case 1 architecture refinement 30
Figure 16 Use Case 2 architecture refinement 31
Figure 17 Local (user) end reference architecture for UC3 32
Figure 18 Remote (robot) end reference architecture for UC3 33
Figure 19 UC4 architecture refinement 34
Figure 20 Demonstrator 1 architecture refinement 35
Figure 21 Demonstrator 2 architecture refinement 36
Figure 22 Demonstrator 3 autonomous robot software architecture 37
Figure 23 Demonstrator 3 global reference architecture 38
Figure 24 Demonstrator 4 architecture refinement 39

Abbreviations

Abbreviation	**Explanation**
3D	3 Dimension
ADAT	Automatic Die Attach
AGV	Autonomous Ground Vehicle
AMC2	Access Modular Controller
AI	Artificial Intelligence
API	Application Programming Interface
BBx (e.g. BB1)	Building Block x (e.g. Building Block 1)
CAD	Computer-Aided Design
CAN-OPEN	Controller Area Network – the Open Communication Solution Dissemination Project
CD	Continuous Development
CI	Continuous Integration
CNC	Computer Numerical Control
COTS	Commercial Off-The-Shelf
DC	Direct Current
DevOps	Development Operations
DoF	Degree-of-Freedom
DMS	Distributed Message Service
DT	Digital Twin
DTA	Digital Twin Aggregation
DTI	Digital Twin Instance
DVC	Data Version Control
Dx.x (e.g. D2.4)	Deliverable x.x (e.g. Deliverable 2.3)
EtherCAT	Ethernet for Control Automation Technology
ERP	Enterprise Resource Planning
FB	Feedback
FPGA	Field Programmable Gate Array
GAN	Generative Adversarial Network
GUI	Graphical User Interface
HMI	Human-Machine Interface
HTTP	HyperText Transfer Protocol
HW	Hardware
I/O	Input/Output
IMOCO4.E	Intelligent Motion Control under Industry 4.E
IPC	Industrial Personal Computer
IRT	Isochronous Real Time
IT	Information Technology
LIDAR	LIght Detection And Ranging
LIMS	Laboratory Information Management Systems
MBSE	Model-Based Systems Engineering
MCP	Motion Control Platform
MES	Manufacturing Execution System
MIMO	Multi-Input Multi-Output
ML	Machine Learning
MLOps	Machine Learning Operations
MQTT	Message Queuing Telemetry Transport

M&E	Motor and Encoder
NFC	Near Field Communication
OPC-UA	Open Platform Communications – Unified Architecture
PC	Personal Computer
PHP	Hypertext Preprocessor
PIL	Processor-in-the-Loop
PL	Performance Level
PLC	Programmable Logic Controller
QA	Quality Assurance
REST	REpresentational State Transfer
RFID	Radio-Frequency Identification
RGB-D	Red Green Blue – Depth
RL	Reinforcement Learning
ROS	Robot Operating System
RTOS	Real-Time Operating System
R&D	Research & Development
SaaS	Software-as-a-Service
SAP	Systems, Applications and Products in data processing
SCADA	Supervisory Control and Data Acquisition
SE	System Exploitation
SI	System Integration
SILx (e.g. SIL3)	Safety Integrity Level x
SLAM	Simultaneous Localisation And Mapping
SO	System Operational
SoC	System-on-Chip
SPI	Serial Peripheral Interface
ST	Scientific and Technological
SW	Software
TCP/IP	Transmission Control Protocol/ Internet Protocol
TOF	Time-of-Flight
TSN	Time-Sensitive Networking
UCx (e.g. UC3)	Use Case x (e.g. Use Case 3)
UI	User Interface
UPS	Uninterrupted Power Supply
USB	Universal Serial Bus
VR	Virtual Reality
WAN	Wide Area Network
WLAN	Wireless Local Area Network
WPx (e.g. WP2)	Work Package x (e.g. Work Package 2)
XIL	X-in-the-Loop
.NET	Network Enabled Technologies

Executive Summary

This deliverable describes the general specification and design of the IMOCO4.E reference framework. In addition, refinements and instantiations of the reference framework from the IMOCO4.E industrial applications – Pilots, Demonstrators and Use-cases – are also provided. The IMOCO4.E reference framework brings together the architecture, data management, AI and digital twin viewpoints from the industrial users of the IMOCO4.E consortium. The IMOCO4.E reference framework envisions a generic architecture platform for designing, developing, and implementing novice and complex motion-controlled industrial systems. The proposed IMOCO4.E reference framework is defined over several discussions after gathering the industrial applications' brownfield and greenfield reference architectures.

The proposed IMOCO4.E reference framework and refinements go beyond the current state-of-the-art and meet the scientific and technological (ST) development objectives, system integration and interoperability (SI) objectives, system operational (SO) objectives, and system exploitation (SE) objectives of the IMOCO4.E project. In addition, the IMOCO4.E reference framework in this deliverable will act as a reference for further technological development and innovation in the IMOCO4.E project.

Keywords

Intelligent Motion Control, Reference Framework, Reference Architecture, Artificial Intelligence, Digital Twin, Data Management

1. Introduction

Deliverable D2.4 is the final deliverable of Work Package 2 (WP2). WP2 focuses on the business requirements and the reference system architecture. Deliverable D2.4 provides the general specification and design of the IMOCO4.E reference framework, and requires inputs from the previous deliverables of WP2, as detailed below. Previous deliverables in this WP are:

- D2.1 State-of-the-art methods in Digital Twinning for motion-driven high-tech applications [1]
- D2.2 Needs for future smart production in Europe from the mechatronics and robotic point of view [2]
- D2.3 Overall requirements on IMOCO4.E reference framework [3].

First, the WP2 establishes the current state-of-the-art of IMOCO4.E relevant technologies and identifies the shortcomings of existing solutions in meeting European manufacturing and production challenges in the future. Second, WP2 assesses the demands on future smart manufacturing in Europe from the mechatronics and robotic point of view and identifies the latest promising emerging trends in smart motion control and smart manufacturing. Third, WP2 links the IMOCO4.E challenges with the technological and business requirements defined by the industrial end-users. Finally, WP2 describes the IMOCO4.E reference architecture at the system level to drive the development actions of WP3, WP4 and WP5 and guarantee successful modular system integration and operation.

1.1 Purpose of the document

Deliverable D2.4 presents the general specification and design of the overall IMOCO4.E reference framework at an abstract level and some initial refinements of the overall framework targeting industrial pilots, demonstrators and use cases. This will be the basic but extensible, open and modular framework to be realised, demonstrated and validated in the IMOCO4.E project [4]. This deliverable considers the ST, SI, SO and SE objectives of the IMOCO4.E project.

This document does not include the building block (BB) level refinement of the IMOCO4.E reference framework. Detailed specification, design and refinement of the reference framework at the BB level can be more extensive, and these are left to the respective deliverables in WP3, WP4 and WP5. In addition, this document does not include the specifications and design of the IMOCO4.E methodology, toolchain and their integration within the reference framework, as these are left to the respective deliverables in WP6. Finally, the refinements of the IMOCO4.E reference framework for the industrial applications – Pilots, Demonstrators and Use-cases – are provided at an abstract level in this document. The detailed description and specifications of the respective applications will be part of the respective deliverables in WP7 for Pilots and Demonstrators and WP6 for the Use-cases.

1.2 Structure of the document

Chapter 1 contains sections that help the reader understand the context of this deliverable with respect to the overall project. Chapter 2 explains the steps taken in the design of the IMOCO4.E reference framework. A strategy on how the requirements for the reference framework can be verified, traced and fulfilled is also provided. Chapter 3 proposes the IMOCO4.E reference framework and its overall specifications. The reference framework's different viewpoints in relation to this project's BBs are also presented. In addition, suggestions to use model versioning for model management are proposed. Using model versioning implies that we could use the existing data management framework/viewpoint for model management. Chapter 4 provides the refinements and instantiations of the IMOCO4.E reference framework for the various industrial applications in the IMOCO4.E project. Finally, Chapter 5 concludes the deliverable.

1.3 Intended readership

Deliverable D2.4 is disseminated as a public document. This means that the general specifications and design of the IMOCO4.E reference framework specified in this deliverable are available to everyone.

2. Design of the IMOCO4.E reference framework

An initial version of the IMOCO4.E reference framework was already introduced in D2.3 [3]. The initial version of the reference framework paved the way for defining IMOCO4.E concepts and formed the basis for the deliverables D3.1 [5], D4.1 [6], D5.1 [7] and D6.1 [8]. The deliverables D3.1, D4.1 and D5.1 detailed the first version of the requirements and specifications based on D2.3 [3], and D6.1 provided the guidelines for the IMOCO4.E methodology and toolchain with a focus on digital twins (DTs) and artificial intelligence (AI). The steps leading to the design of the reference framework are the following:

1. Identifying shortcomings from the state-of-the-art reference architectures. Understanding the current state-of-the-art and identifying the gaps were essential for defining the IMOCO4.E reference framework. The state-of-the-art methods for motion-driven high-tech applications and shortcomings were reported in deliverable D2.1.

2. Identifying the needs for future smart production from the mechatronics and robotic point of view. The IMOCO4.E reference framework should address these needs through relevant requirements. The needs summary gathered per BBs was used to define the requirements for (some of) the relevant topics. The needs were reported in deliverable D2.2.

3. Gathering and characterising the brownfield architectures of the pilots, demonstrators and use cases from the industrial partners in the IMOCO4.E consortium. The brownfield architecture (in the context of the IMOCO4.E project) characterises the existing legacy systems (software, hardware, or platform) that would be part of the corresponding pilots, demonstrators or use cases during the IMOCO4.E framework development and integration. The constraints and interfacing options with the brownfield architecture in the generic IMOCO4.E reference framework definition were reported in deliverable D2.3.

4. The first iteration of the detailed requirements and specification of the different layers in the IMOCO4.E project were reported in the deliverables D3.1, D4.1 and D5.1. The requirements and specifications were also detailed on a building block (BB) level. These requirements and specifications must be adhered to while designing the final IMOCO4.E reference framework.

5. The guideline of the IMOCO4.E methodology and toolchains were reported in deliverable D6.1. The guideline must be adhered to in the design of the IMOCO4.E reference framework.

6. Gathering and characterising the greenfield reference architectures of the pilots, demonstrators and use cases from the industrial partners in the IMOCO4.E consortium. The greenfield architecture (in the context of the IMOCO4.E project) characterises the envisioned systems (software, hardware, or platform) on top of the existing legacy systems that would be part of the corresponding pilots, demonstrators or use cases during the IMOCO4.E framework development and integration.

We define the IMOCO4.E reference framework based on the above considerations. Some example refinements of this reference framework from the industrial partners are also provided in this deliverable. Steps 1-5 have been reported in previous (public) deliverables. The greenfield reference architectures were gathered after Step 5 so that the industrial partners could assimilate the information from the previous deliverables and refine their architectures based on their current interests and roadmap.

2.1 Gathering the greenfield architectures from industrial users

The design of the IMOCO4.E reference framework required gathering the greenfield reference architectures from Pilots, Demonstrators, and Use case owners of the IMOCO4.E consortium. An example of refinement of the reference architecture and the way-of-working for refining the architectures was provided by ITEC B.V. as the WP2 leader. From the WP2, a request was sent to the industrial partners to gather the envisioned (greenfield) architecture that the Pilots, Demonstrators and Use-case owners will demonstrate in the IMOCO4.E project.

The suggested way-of-working for the Pilots, Demonstrators and Use-case owners to gather the greenfield architecture (illustrated in Figure 1) was as follows:

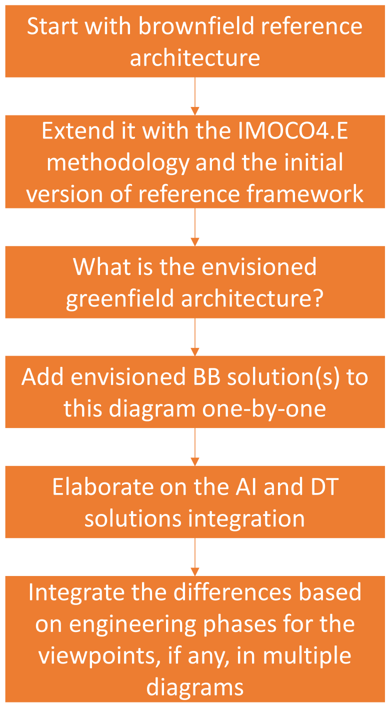
Start with brownfield reference architecture
Extend it with the IMOCO4.E methodology and the initial version of reference framework
What is the envisioned greenfield architecture?
Add envisioned BB solution(s) to this diagram one-by-one
Elaborate on the AI and DT solutions integration
Integrate the differences based on engineering phases for the viewpoints, if any, in multiple diagrams

Figure 1 Flow diagram for the suggested way-of-working for the industrial users

1. Start with the envisioned (greenfield) architecture(s) that will be demonstrated in the IMOCO4.E project. You can modify the brownfield reference architecture you have already provided. If your envisioned architecture is the same as the brownfield, please provide that as the first diagram. Please provide **two** versions of this diagram – one public (for dissemination in D2.4) and one confidential (for dissemination in WP7 or WP6 deliverables).

2. Go through D6.1 for the definitions of digital twins, the IMOCO4.E methodology and the focus on engineering phases.

3. For the final diagram(s), you could use a step-by-step approach:

a. Start by adding the BBs/solutions that you focus on to the diagram
b. Integrate these with your greenfield solution (focusing on AI and DT)
c. Do your architecture/viewpoints look different in each engineering phase? Then, please provide multiple diagrams to distinguish the engineering phases (e.g. design phase, operations phase, etc. or based on the definitions in D6.1). If possible, could you briefly explain the pattern of the architecture?

The greenfield reference architectures were gathered, and the public diagrams were used in this deliverable to showcase the refinements from the IMOCO4.E reference framework. The IMOCO4.E reference framework brings together the architecture, AI and digital twin viewpoints from the industrial users of the IMOCO4.E consortium.

2.2 Requirements verification, traceability and fulfilment

The requirements for the IMOCO4.E reference framework were reported in D2.3. This section discusses how the requirements relevant to the IMOCO4.E reference framework will be verified, traced and fulfilled. The IADT requirement verification method [9] is used for verifying the requirements. The IADT method classifies the verification method as follows:

- I: inspection (observation using basic senses)
- D: demonstration (use the system as it is intended)
- T: test (more precise and controlled demonstration using scientific principles and procedures)
- A: analysis (validation of the system by scientific methods)

The requirements are traceable through the requirement coding scheme (defined in D2.3), where each requirement is assigned a unique requirement ID. The requirements are also prioritised using the MoSCoW method [10]. The fulfilment of the requirements will be detailed by the corresponding tasks/deliverables assigned to the requirement in D2.3.

3. The IMOCO4.E reference framework and specifications

3.1 Overview

This deliverable presents the basic IMOCO4.E reference framework at an abstract level. The IMOCO4.E reference framework is extensible, open, modular, flexible, scalable, future-proof and fully functional. A flexible framework is configurable from the lowest layer (Layer 1) to the human interfaces and supervisory layer (Layer 4). IMOCO4.E will also bring an MBSE (Model-Based System Engineering) approach to all the architecture layers. Furthermore, these layers will demonstrate how AI supports the optimisation of processes using digital twin instances.

3.2 Architecture viewpoint

The architecture viewpoint of the IMOCO4.E reference framework is illustrated in Figure 2. This is a generic version based on the inputs from the industrial partners. Interpretations of this framework will be detailed in the refinements of the IMOCO4.E reference framework in Section 4.

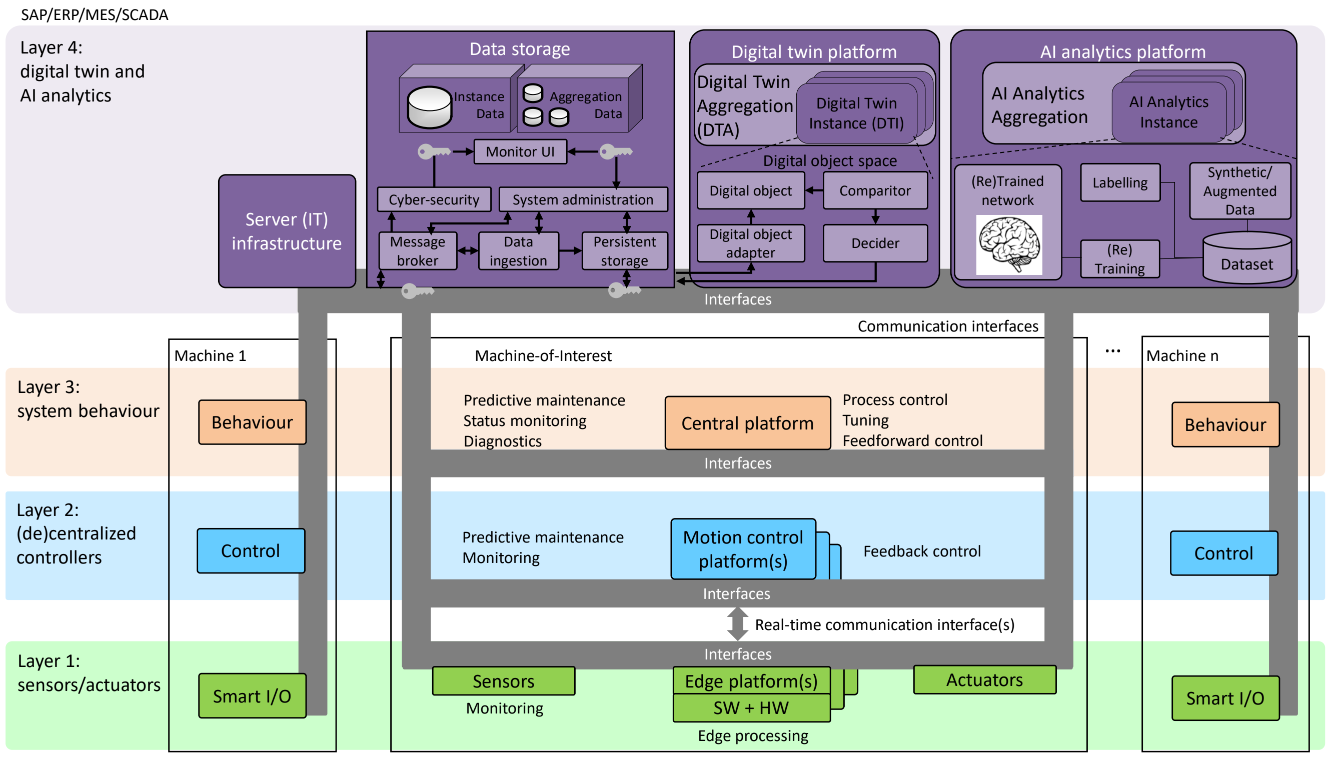

Figure 2 Architecture viewpoint of the IMOCO4.E reference framework

3.2.1 Functional breakdown of the framework and its overall specifications

The components of the IMOCO4.E reference framework and its overall specifications (at an abstract level) are detailed below (adapted from D2.3 [3]). The detailed specifications will be part of future deliverables (e.g. D3.2, D4.2, D5.2).

- *Sensors* are the input devices which provide an output with respect to a specific physical quantity. It is a hardware component for detecting or measuring physical properties or parameters by converting signals from one energy domain to the electrical domain. E.g., temperature sensor, proximity sensor, pressure sensor, position sensor, touch sensor, etc.

- *Actuators* are components that tie a control system to its environment. The actuator is a machine component responsible for moving and controlling a mechanism or system, for example, opening a valve.
- *Smart I/O* refers to smart sensors and smart actuators. Smart sensors may include some local (or edge) processing. Smart actuators may include some local intelligence (and processing).
- *Platforms*: A platform refers to the combination of software (tasks, messages, mapping, scheduling) and hardware (computation, communication, memory). The software performance relies heavily on the predictability and reliability of the deployed hardware. The software can overcome errors to a certain degree when a few hardware functions fail. Still, the overall performance will degrade when input signals, i.e., data, are corrupted in hardware before these are in the 'digital' domain. Therefore, it is essential that the hardware used within the IMOCO4.E project provides reliable signals and data. The platform components considered in the IMOCO4.E reference architecture are the following.
    - Edge platforms (Layer 1): When data needs to be processed at the edge, the IMOCO4.E framework will rely on edge platforms. E.g. for high-speed vision processing, an edge platform is essential since sending image streams over the fieldbus is not ideal due to limited fieldbus bandwidth.
    - Motion control platforms (Layer 2) are mainly required for accurate and predictable feedback control at a high sampling rate (in the kHz range). Such control can be centralised or decentralised. The state of the components controlled by the platforms can also be monitored, and necessary predictive maintenance actions can be taken.
    - A central platform (Layer 3), e.g. a PC, is required for coordinating the machine operation. Process control, feedforward control, parameter setting/tuning, machine status monitoring, predictive maintenance and diagnostics are some of the tasks/applications that run on the central platform
    - *AI platform* (Layer 4) refers to the AI analytics and training infrastructure. The AI platform consists of AI analytics instances and AI analytics aggregation (explained in Section 3.4).
    - The *digital twin (DT) platform* (Layer 4) refers to the software and hardware necessary for the digital twin instances (DTI) and digital twin aggregations (DTA) modelling, operation, monitoring, maintenance, and update. DTI and DTA are explained in Section 3.5.
- *Server (IT) infrastructure* is the backbone of a factory or production line for interactions with human users and factory operations, e.g. integrating the machine operation with SAP, ERP, MES or SCADA solutions.
- *Data storage* for the IMOCO4.E reference framework refers to the data necessary for the AI and digital twin platforms. Instance data refers to the data for a machine instance, and the aggregation data refers to the data for DTA and AI analytics aggregation. The data management infrastructure for data storage is explained in Section 3.3.
- *Interfaces* are the most necessary components in the IMOCO4.E reference framework. Interfaces can be fieldbuses, real-time communication protocols, wireless communication protocols, internet communication and so on. As shown in Figure 2, interfaces can be present between any two architecture layers on the same machine or between architecture layers on two different machines through Layer 4. The interface between Layer 1 and Layer 2 is typically a real-time communication interface (e.g. EtherCAT or TSN).

3.3 Data management viewpoint

The data management viewpoint of the IMOCO4.E reference framework is outlined in Figure 3. The IMOCO4.E reference framework's data management will be realised through BB9, focusing on 'Cyber-security tools and trustworthy data management'. The IMOCO4.E reference framework supports the real-time data exchange of text-based information between multiple endpoints in parallel through a robust and distributed pub/sub messaging system based on Kafka brokers [10]. In addition, the framework aims to support a central aggregation and persistent storage of data based on ElasticSearch [12].

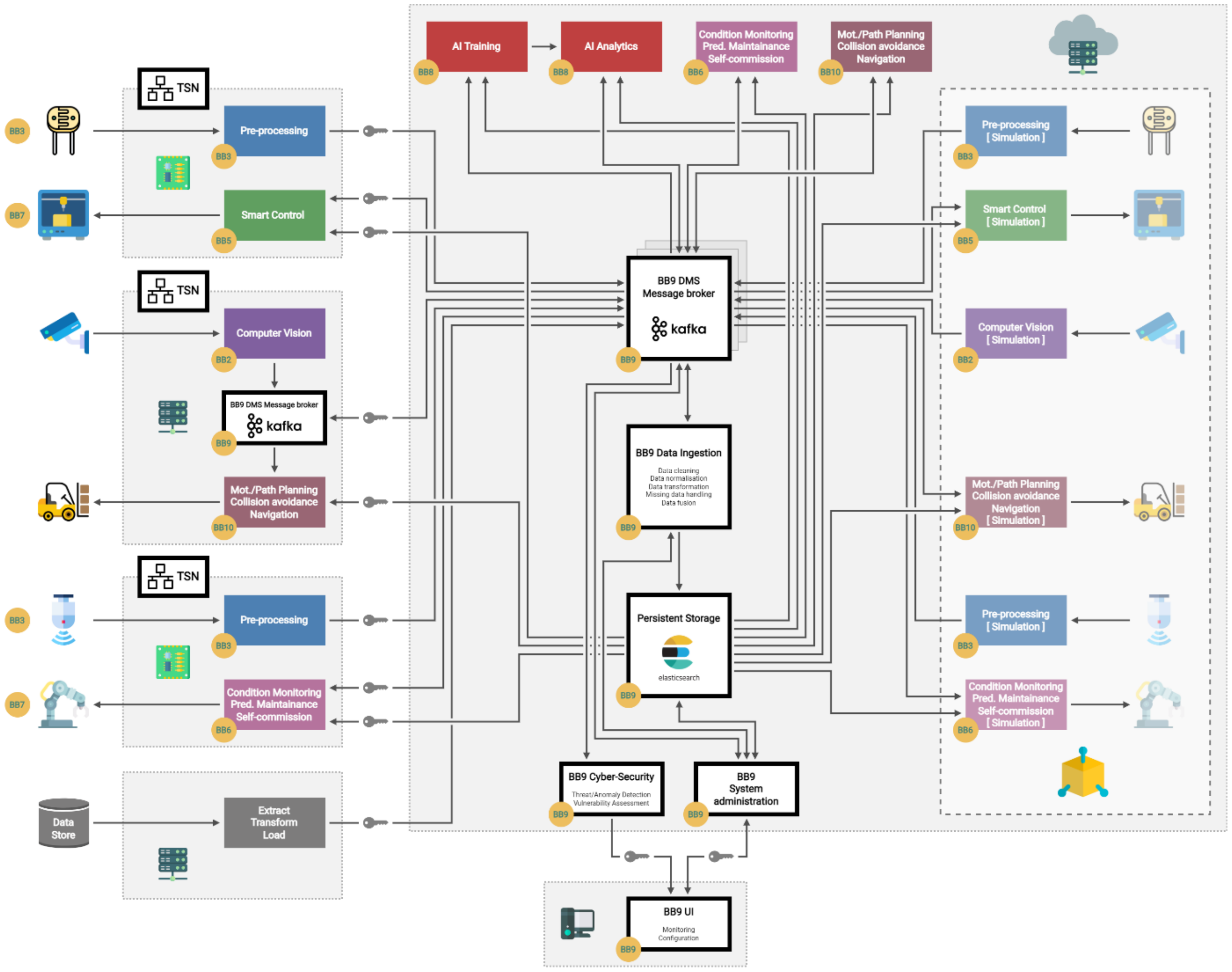

Figure 3 Data management viewpoint for the IMOCO4.E reference framework (from D5.1 [7]).

The functional components in the data management viewpoint are detailed below.

- *DMS Message broker* translates a message from the sender's formal messaging protocol to the receiver's formal messaging protocol. A message broker is an architectural pattern for message validation, transformation and routing. The BB9 solution will have Kafka message brokers.
- *Data ingestion* is the process of moving (and/or replicating) data from one point (e.g. data sources or main database) to another point (e.g. data lake or persistent storage) for some purpose.

- *Persistent storage* or nonvolatile storage is any data storage device that retains data after the power to the device is turned off. The envisioned BB9 solution will have persistent storage of data based on the ElasticSearch stack.
- Cyber-security in the context of the IMOCO4.E framework refers to cyber-secure data transmissions by implementing threat detection and vulnerability assessment.
- *UI* for *system administration*, monitoring and configuration purposes.

The detailed specifications of the data management aspect will be part of the future deliverable D5.2. The BB9 solution will be part of the future deliverable D5.3. The envisioned BB9 solution can serve the data exchange needs of any IMOCO4.E component that can act as a client to the BB9 message broker and can transmit and receive data over the network. Kafka client implementations are available for most programming languages, including C/C++, Python, Go, Java, .NET, Clojure, Ruby, Node.js, Proxy (HTTP REST, etc.) and Perl. Furthermore, IMOCO4.E components can access the BB9 persistent data storage repository for retrieving historical data by performing ElasticSearch API queries. ElasticSearch clients are available for most programming languages, including Java, JavaScript, Ruby, Go, .NET, PHP, Perl, and Python. This also means that integrating the BB9 solution in brownfield architectures may require additional software and hardware modifications and will be done based on the industrial case requirements.

3.3.1 Versioning – source code, data and model management over the life cycle

Version control [13] has become a necessity in the software development lifecycle. Version control tools, such as git [14] and svn [15], are used to manage and track source codes. However, the typical version control system fails to track changes made to data, including metrics, parameters and hyperparameters (for data used in AI/ML applications) and is limited in the amount of data they are able to host due to storage limits. Data version control (DVC) [16] proposes a git for data and models. DVC can be used to manage big data and (image) data sets.

With the emergence of MBSE, models gained significant importance in the engineering phases and the product life cycle. In MBSE, models are the single source of information, and executable software can be directly generated from the models. There can be different types of models – ML models, 3D CAD models, mathematical and simulation models. Model management for each of these types of models can be different with their own toolchains.

Model versioning [17][18] is the application of version control principles for model management and is necessary to enable efficient team-based collaborative development of the models. The simplest model versioning can be done by DVC using git [16]. Applying model versioning when building applications is important because it allows:

- Referencing previous versions of the model
- Reverting changes quickly, making building models less risky
- Easily reproducing and sharing models among the developers.
- Indirect versioning of the executable software generated from the models.

The IMOCO4.E reference framework proposes state-of-the-art versioning control systems for source code, data and model management. Integration of the versioning control systems is specific to industrial applications. The detailed methodology for the data and model management will be elaborated in the future IMOCO4.E deliverables (e.g. D6.2, D6.7 and D6.9).

3.4 AI viewpoint

The AI viewpoint is adapted from D2.3. In the IMOCO4.E project, “the focus is on the following distinct aspects of AI for motion control: environmental awareness, motion planning, deployment, image pre-processing, signal analysis for drive diagnostics, long-term predictive maintenance of the machine” (from the DoA [4]). The AI viewpoint provides the data flow, interfaces and BB interactions required to achieve the project goal.

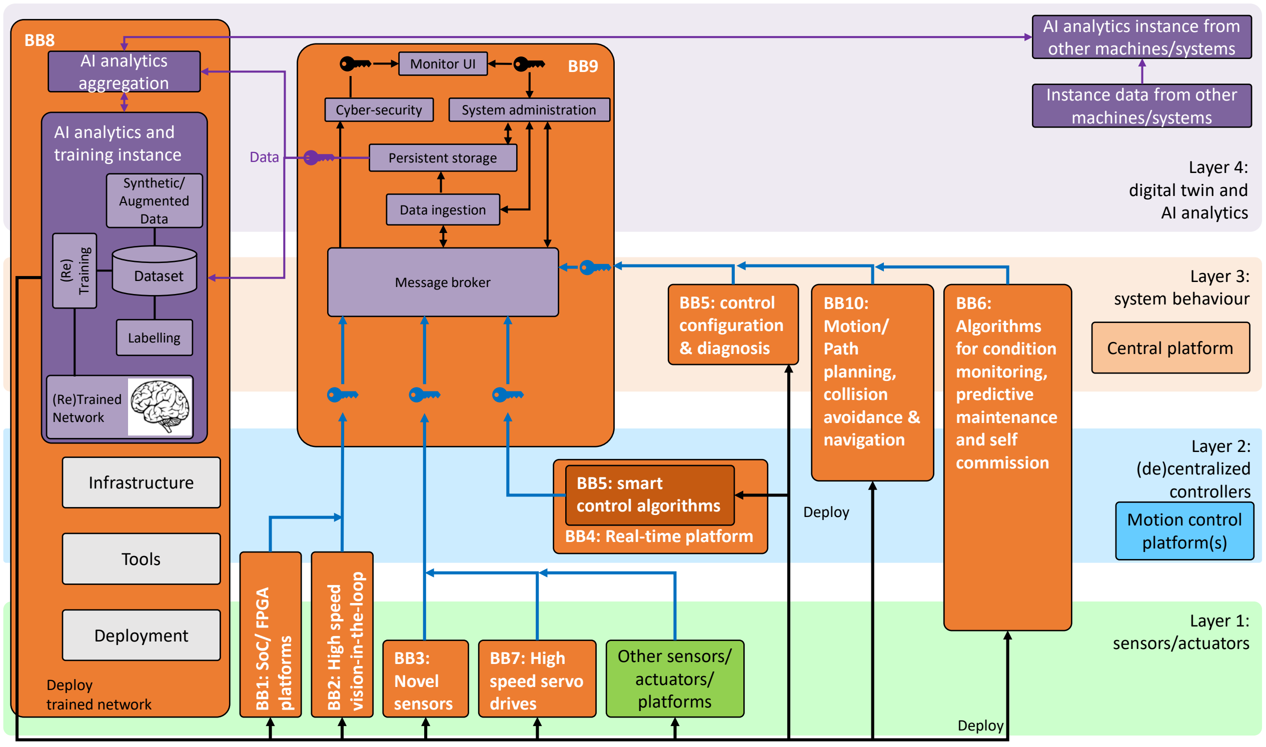
BB8
AI analytics aggregation
AI analytics and training instance
Synthetic/ Augmented Data
(Re) Training
Dataset
Labelling
(Re)Trained Network
Infrastructure
Tools
Deployment
Deploy trained network
Data
BB9
Monitor UI
Cyber-security
System administration
Persistent storage
Data ingestion
Message broker
AI analytics instance from other machines/systems
Instance data from other machines/systems
Layer 4: digital twin and AI analytics
BB5: control configuration & diagnosis
BB10: Motion/ Path planning, collision avoidance & navigation
BB6: Algorithms for condition monitoring, predictive maintenance and self commission
Layer 3: system behaviour
Central platform
BB5: smart control algorithms
BB4: Real-time platform
Deploy
Layer 2: (de)centralized controllers
Motion control platform(s)
BB1: SoC/ FPGA platforms
BB2: High speed vision-in-the-loop
BB3: Novel sensors
BB7: High speed servo drives
Other sensors/ actuators/ platforms
Layer 1: sensors/actuators
Deploy

Figure 4. AI viewpoint with BB interactions

The AI viewpoint with BB interactions of the IMOCO4.E reference framework is illustrated in Figure 4. The general principle is that data is collected from Layer 1 (sensors, edge platforms and actuators) and used by the AI framework for modelling, training, optimisation, analytics and/or services. Additionally, data can be collected from Layer 2 (e.g. from BB5 internal signals). Furthermore, it is convenient to have configuration data (e.g. from Layer 3) available in the data collection, such that the dataset is complete and consistent at all times. The data flows from layers Layer 1, Layer 2 and Layer 3 to the AI framework and back to the corresponding BBs are illustrated in Figure 4. BB8 deals with AI-based components and forms the core BB for integrating the AI framework in the IMOCO4.E methodology. BB8 will specify in detail the AI infrastructure, tools and deployment methods in future deliverables. The data necessary for the AI framework is collected, secured and stored based on the methodology developed as part of BB9 (cybersecurity and trustworthy data management). Some definitions are as follows:

- The *AI instance* refers to the AI framework for a machine instance (or some machine components).
- The *AI aggregation* refers to the aggregation of all AI instances.

The functionality of an AI instance and AI aggregation varies based on the stage in the machine lifecycle and is illustrated in Figure 5. During the machine lifecycle’s design phase, an AI instance’s main functionality is modelling and training. The AI model that is suited for the design objective and satisfies the requirements is identified. Typically, the machine prototype data is used to train the AI model. The AI modelling and training could also start with machine simulation (before the machine prototype is available). Then, the trained AI model is deployed in the machine prototype for testing and validation. The AI instance is optimised for inference performance using the assembled machine data and characteristics during the manufacturing phase. The optimised inference AI model is then deployed in the assembled machine for testing and validation. Finally, during the services phase of the machine lifecycle, the AI instance is used for data analytics and offering other services, e.g. process optimisation. The data monitored by the machine in operation is the input for the AI analytics algorithm, and the AI instance offers optimal services. The AI platform coordinates the AI instance. If required, the AI platform can be independent (with its own hardware and software).

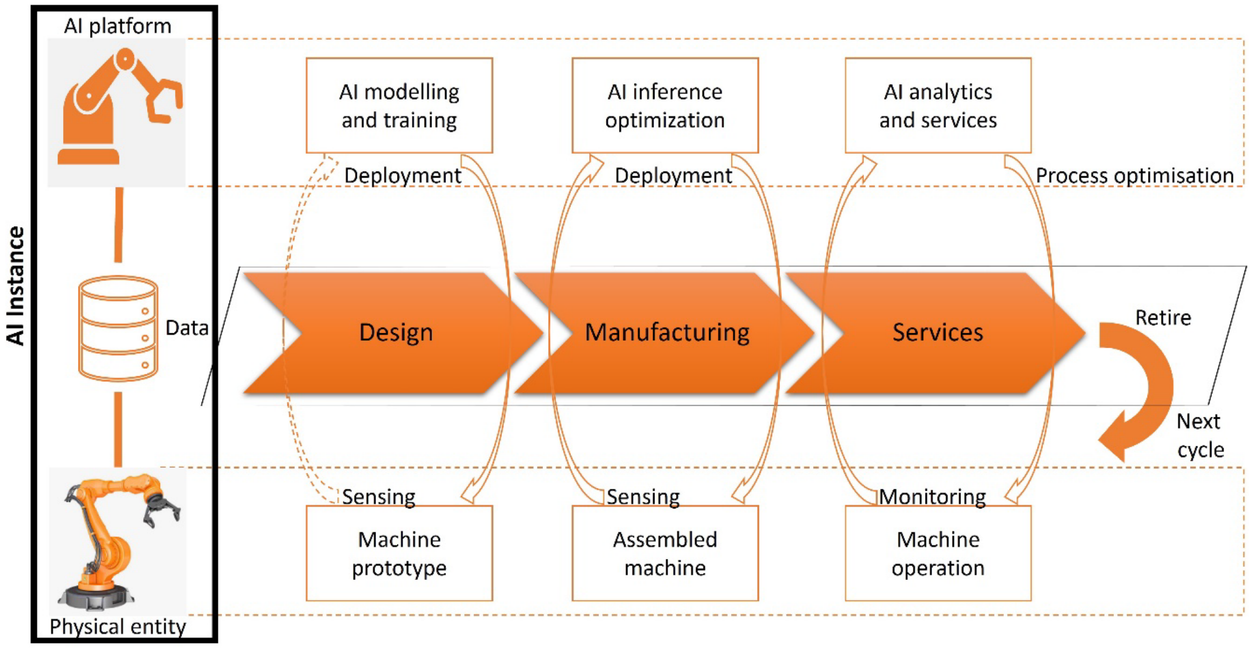

Figure 5. AI viewpoint during a machine lifecycle (from D2.3)

One of the key challenges for AI, however, is to deploy ML products into production rapidly. It is highly challenging to automate and operationalise ML products. "MLOps (Machine Learning Operations) is a paradigm, including aspects like best practices, sets of concepts, as well as a development culture when it comes to the end-to-end conceptualisation, implementation, monitoring, deployment, and scalability of machine learning products. Most of all, it is an engineering practice that leverages three contributing disciplines: machine learning, software engineering (especially DevOps), and data engineering. MLOps is aimed at productionizing machine learning systems by bridging the gap between development (Dev) and operations (Ops). Essentially, MLOps aims to facilitate the creation of machine learning products by leveraging these principles: CI/CD automation, workflow orchestration, reproducibility; versioning of data, model, and code; collaboration; continuous ML training and evaluation; ML metadata tracking and logging; continuous monitoring; and feedback loops" [19]. In the IMOCO4.E project, BB8 solutions will propose the best practices for developing and implementing AI-based solutions.

3.5 Digital twin viewpoint

In the IMOCO4.E framework, a digital twin comprises five dimensions (from D2.3) – the physical entity, virtual model, data, service, and connection or interaction. An outline for the digital twin for an intelligent motion control system is illustrated in Figure 6. The detailed description of the digital twin and the corresponding examples are detailed in the IMOCO4.E deliverable D6.1. The physical entity here is the physical system. The virtual model is the digital object space. The data are the sensor, control, and reference data. Comparitor and decider can provide services during the services phase of the lifecycle. The connections or interactions occur through fieldbuses and/or (wireless) network infrastructure.

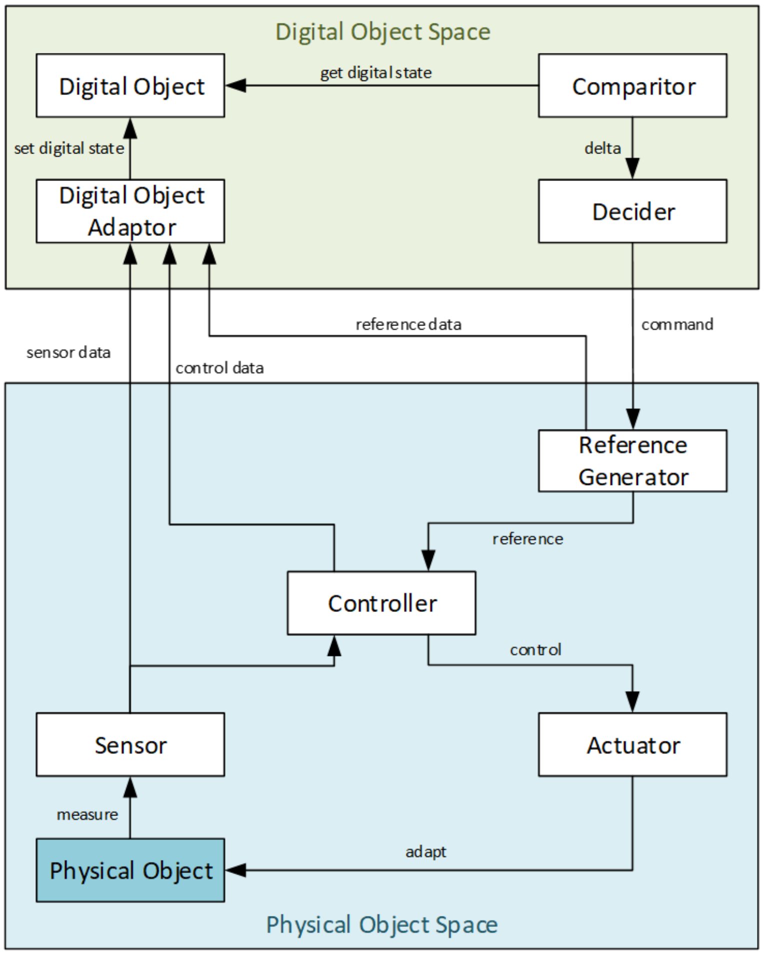

Figure 6 Digital twin outline for an intelligent motion control system (from D6.1)

The digital twin viewpoint during a machine's lifecycle is illustrated in Figure 7 (from D2.3). The IMOCO4.E framework concepts related to digital twins are the following:

- a *digital twin prototype* is a virtual description of a prototype machine containing all the information to create the physical twin prototype. The digital twin prototype can vary from component level to system level.
- a *digital twin instance (DTI)* refers to a specific instance of a physical machine that remains linked to the specific machine throughout its lifecycle.
- a *digital twin aggregation (DTA)* combines all the digital twin instances.

A digital twin is helpful throughout the machine's lifecycle. During the design phase, virtual design models form the basis of the machine prototype development. Machine prototype specifications are required by the virtual models for designing an efficient system through iterative optimisation and virtual verification. A digital twin can be used during the design phase - to design and test a new algorithm, explore use cases, etc., before deploying it to the actual physical system. The physical system may not yet be available at this point. A digital twin also expedites the test time (and hence faster time-to-market) since the physical system has limited test capacity. Testing on the physical system can be expensive if hardware fails due to testing. The digital twin prototype is used for real-time sensing, control, and process optimisation during manufacturing. A digital twin instance during the service phase enables predictive maintenance, fault detection and diagnosis, state monitoring, process optimisation and so on.

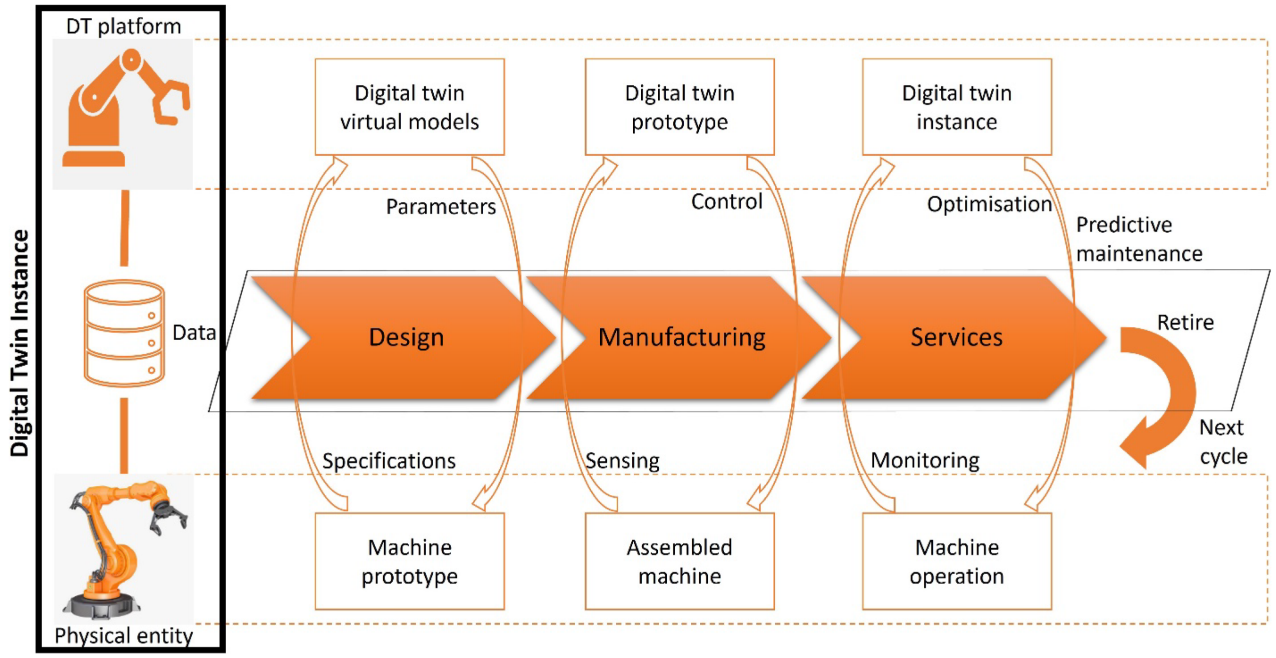

Figure 7. Digital twin viewpoint during a machine lifecycle (from D2.3)

The digital twin viewpoint with BB interactions is illustrated in Figure 8. The general principle here is that the physical entity comprises the machine (the sensors, platforms, actuators and interfaces represented through the various BBs and other components, e.g. COTS). The virtual entity of the digital twin is represented by the digital twin (DT) platform. The digital twin virtual models are part of the DT platform. The services and analytics are performed through the AI framework (BB8). The BB9 handles the data collection, storage and cyber-security. The DT platform uses the data from the physical twin, services, and models. Finally, the DT platform sends the parameter changes for optimal machine performance to the relevant physical components or provides warnings or predictive maintenance schedules to the human users, e.g. operators and service engineers.

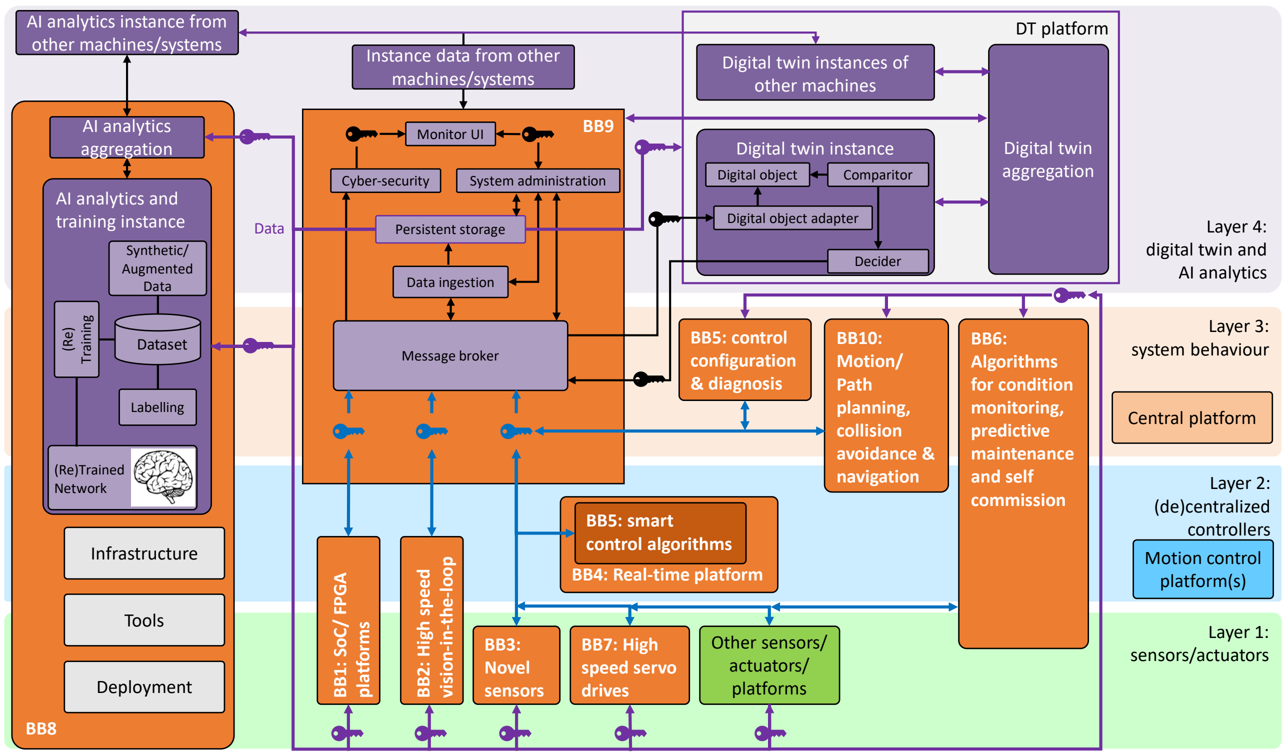
AI analytics instance from other machines/systems
Instance data from other machines/systems
DT platform
Digital twin instances of other machines
AI analytics aggregation
Monitor UI
BB9
Cyber-security
System administration
Digital twin instance
Digital object
Comparitor
Digital object adapter
Decider
Digital twin aggregation
AI analytics and training instance
Synthetic/ Augmented Data
Data
Persistent storage
Layer 4: digital twin and AI analytics
(Re) Training
Dataset
Data ingestion
Labelling
Message broker
BB5: control configuration & diagnosis
BB10: Motion/ Path planning, collision avoidance & navigation
BB6: Algorithms for condition monitoring, predictive maintenance and self commission
Layer 3: system behaviour
Central platform
(Re)Trained Network
Layer 2: (de)centralized controllers
Motion control platform(s)
BB5: smart control algorithms
BB4: Real-time platform
Infrastructure
Tools
Deployment
BB1: SoC/ FPGA platforms
BB2: High speed vision-in-the-loop
BB3: Novel sensors
BB7: High speed servo drives
Other sensors/ actuators/ platforms
Layer 1: sensors/actuators
BB8

Figure 8. Digital twin viewpoint with BB interactions

3.6 IMOCO4.E objectives and the reference framework

The IMOCO4.E objectives are detailed in [4]. The objectives relevant to the reference framework and how they are considered in the general design and specification for the IMOCO4.E reference framework are summarised below.

- The digital twin viewpoint in Section 3.5 details how to build the digital twins considering model-based and knowledge-based methods (ST1). The digital object in the digital twin instance is the model of the physical object, and the AI platform enables knowledge-based solutions integration.
- The reference framework enables the smart gathering and processing of sensor information (ST2), modular unified smart control layer (ST3), and ensures interoperability with the cloud platform (ST4).
- The data management, AI and DT viewpoints outline the integration of the developed BBs into the reference framework (SI 1) and specify the interfaces for interoperability (SI 3).
- The reference framework refinements for pilots (SO1) and demonstrators (SO2) are explained in Section 4.
- The reference framework and refinements for the industrial cases (pilots, demonstrators and use cases) will be part of the public deliverable (SE objectives).

4. Refinements of the IMOCO4.E reference framework

In this section, we will provide refinements of the IMOCO4.E reference framework for (some of) the industrial cases (Pilots, Demonstrators and Use-Cases). The refinements also detail how to use the reference framework during the various engineering phases (explained in D6.1) and by various personnel (developer, operator, service engineer, etc.).

The detailed description of the Pilots, Demonstrators and Use-cases can be found in the DoA and is not repeated in this deliverable. In this deliverable, only the framework refinements, instantiations, and relevant aspects are provided. The detailed specifications and further explanations of these refinements and instantiations will be part of future deliverables.

4.1 Pilot 1 Tissector architecture refinement

Figure 9 outlines the IMOCO4.E reference framework refinement for Pilot 1, Tissector. Layer 1 of Tissector abstracts I/O, (quadrature) encoders, physical hardware, motors, and cameras. The I/O, motors (smart actuators), and encoders (smart sensors) are interfaced with the embedded real-time motion controller (Layer 2) through a fieldbus. The standard EtherCAT interface will be used during the development phase, and a customised hardware interface will be used during the operational phase. The cameras (smart sensors) are interfaced with the application processor (Layer 3).

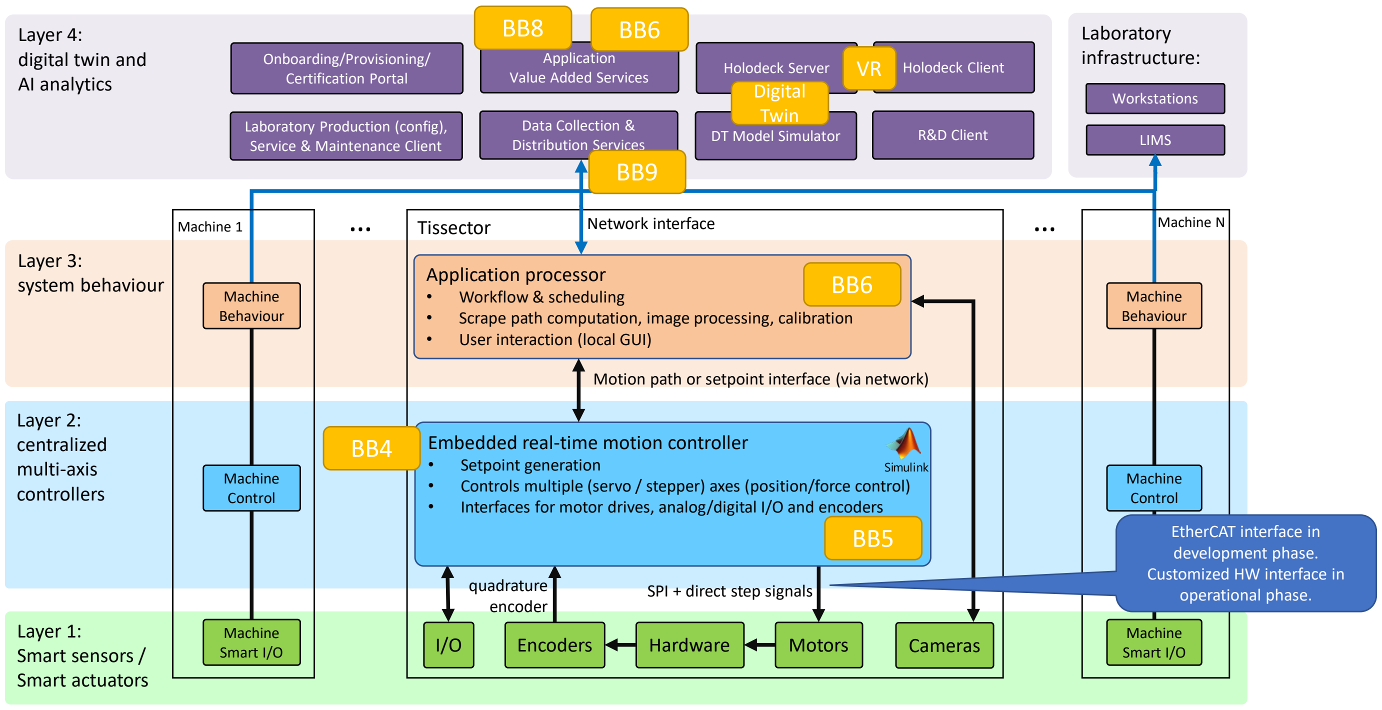
Layer 4: digital twin and AI analytics
Onboarding/Provisioning/ Certification Portal
BB8
BB6
Application Value Added Services
Holodeck Server
VR
Holodeck Client
Digital Twin
Laboratory Production (config), Service & Maintenance Client
Data Collection & Distribution Services
DT Model Simulator
R&D Client
BB9
Laboratory infrastructure:
Workstations
LIMS
Machine 1
...
Tissector
Network interface
...
Machine N
Layer 3: system behaviour
Machine Behaviour
Application processor
Workflow & scheduling
Scrape path computation, image processing, calibration
User interaction (local GUI)
BB6
Motion path or setpoint interface (via network)
Layer 2: centralized multi-axis controllers
Machine Control
BB4
Embedded real-time motion controller
Setpoint generation
Controls multiple (servo / stepper) axes (position/force control)
Interfaces for motor drives, analog/digital I/O and encoders
Simulink
BB5
EtherCAT interface in development phase. Customized HW interface in operational phase.
quadrature encoder
SPI + direct step signals
Layer 1: Smart sensors / Smart actuators
Machine Smart I/O
I/O
Encoders
Hardware
Motors
Cameras

Figure 9 Pilot 1 Tissector architecture refinement from Sioux. The envisioned BBs that will be integrated are annotated.

Layer 2 for the Tissector has the embedded real-time motion controller platform that controls multiple (servo/stepper) axes, provides interfaces for smart sensors and smart actuators in Layer 1 and generates setpoints based on the input from the application processor (Layer 3). The algorithms in Layer 2 are also developed using Simulink software. Layer 2 interfaces with Layer 3 through a motion path or setpoint interface (via network). Layer 3 performs the workflow and scheduling, scrape path computation, image processing, calibration and provides a local GUI for user interaction.

Layer 4 comprises data collection and distribution services, DT model simulator, R&D client, laboratory production (config), service and maintenance client, holodeck server, holodeck client, application value-added services, certification portal and the laboratory infrastructure. The holodeck server and client enable integration of VR for the Tissector. The DT model simulator enables the digital twin. The application value-added services provide the AI services/solutions.

The detailed specifications of Tissector components will be part of future deliverables. The components where the envisioned BB solutions will be integrated are also annotated in Figure 9 for future reference.

4.1.1 Pilot 1 architecture refinement for simulation using VR and DT

Figure 10 instantiates the Pilot 1 reference framework (in Figure 9) for defining the simulation infrastructure during the development phase. The simulated components are highlighted in the figure. The materials (slides, scraper, tubes) can also be simulated using a digital twin and VR. Stubs are also provided for the laboratory infrastructure. In addition, the Layer 1 sensors are simulated, and Layer 1 behaves as a system simulator.

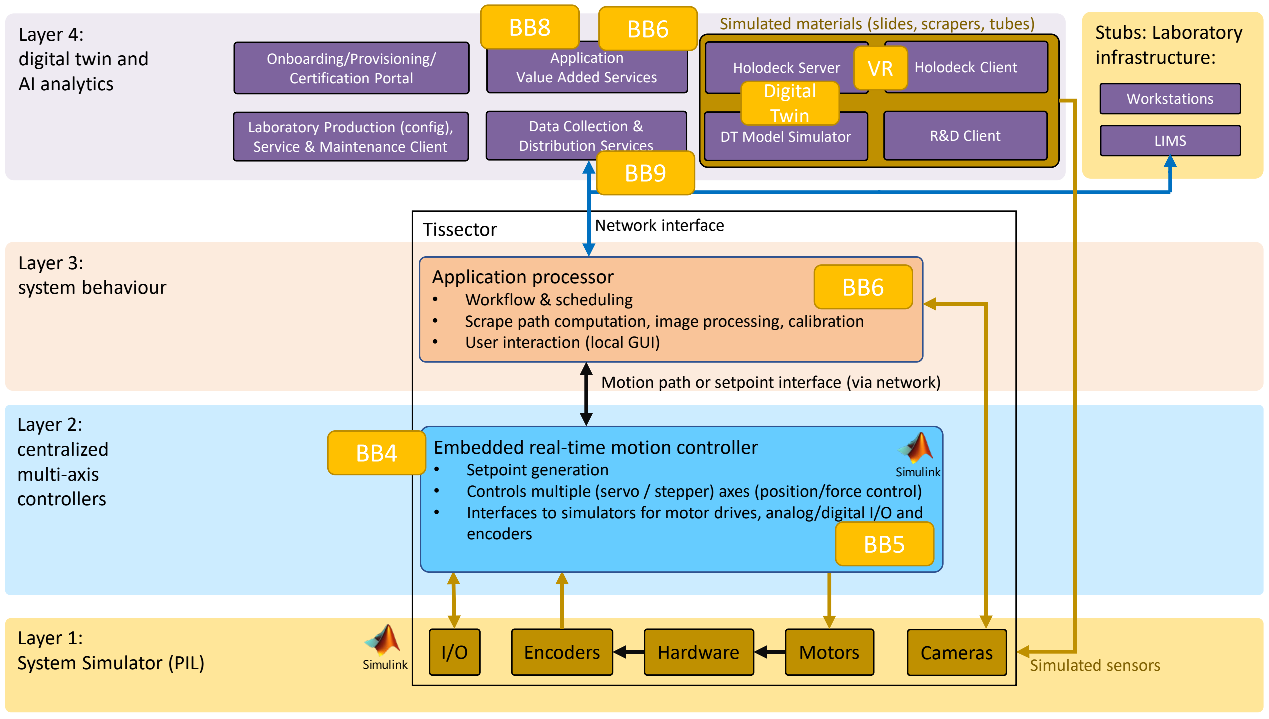

Figure 10 Pilot 1 Tissector simulation infrastructure based on the refined reference framework.

4.2 Pilot 2 reference framework refinement for the ADAT machine

Figure 11 outlines the IMOCO4.E reference framework refinement for the ADAT machine in Pilot 2. The ADAT is ITEC's die-attach platform for semiconductor manufacturing. It is developed to produce high volumes of semiconductor components and thus at high speed and high accuracy, to be competitive. Its basic purpose is to pick semiconductors from a diced wafer, inspect these for damage, and place them onto a substrate. The substrate can be a copper lead frame, a plastic tape, an RFID antenna, and substrates can be discrete or endless (reels). The placement involves glueing, taping or soldering the device to the substrate. The ADAT machine can achieve outputs of up to 72000 products per hour. The machine is part of a production line, and together, many such production lines form a factory.

Layer 1 shows the machine hardware, sensors and actuators. The servo drives and vision system span across both Layer 1 and Layer 2 for the ADAT machine. A platform/system spans across multiple layers when the algorithms typically intended for these multiple layers are deployed on the same platform. An algorithm's functionality and behaviour determines which layer it typically should be. The real-time motion control platform (Layer 2 and Layer 3) will also be the bus master for the fieldbus interface. The motion control tasks are executed on this platform. In addition, the envisioned architecture for Pilot 2 would have dedicated hardware or RTOS on the motion control platform executing some system behaviour tasks, typically done on Layer 3.

The host PC coordinates the machine and executes the process programs. If needed, data from the lower layers are also collected and aggregated by the host PC and span Layers 3 and 4. The data from the ADAT machine is then sent to the server through an ethernet interface for AI services and DT. An operator has direct access to the machine through the Host PC. Software updates and patches developed during the lifecycle are deployed to the machine through the server/cloud. The developer develops the controllers in the Simulink environment, and custom controllers are deployed on the motion control platform and servo drives.

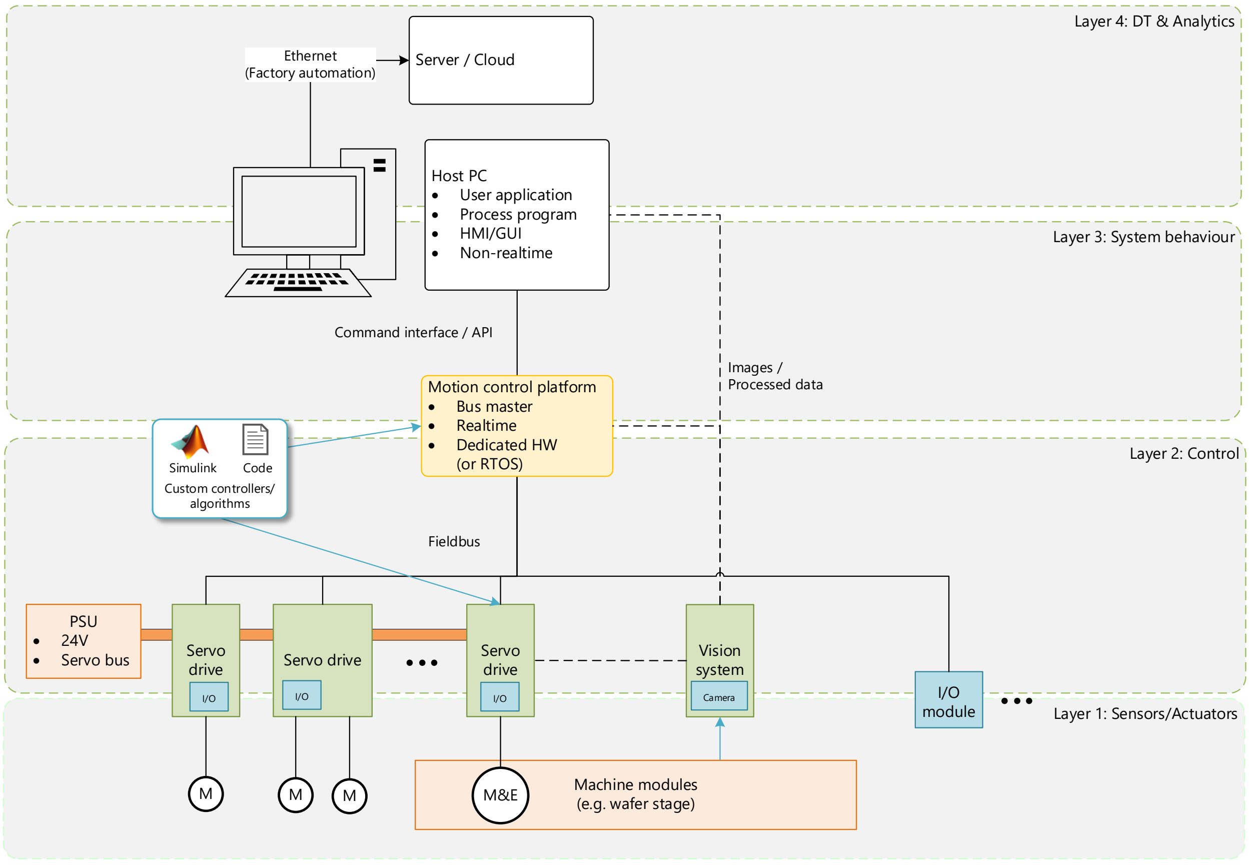

Figure 11 Pilot 2 refinement of the reference architecture for the ADAT machine.

4.3 Pilot 3 architecture refinement in the absence of a machine prototype

Figure 12 outlines the IMOCO4.E reference framework refinement for Pilot 3. This architecture refinement is to face the absence of a machine prototype and real-world sensors for Pilot 3. The main objective of this pilot is to assess the feasibility of improving automatisation for quality checks and alarm detection throughout a whole high-speed packaging process. In this perspective, the AI-based algorithms, in combination with the smart control platform, will help to ensure good quality output and prompt reaction to possible alarms. Within Pilot 3, IMOCO4.E will be implemented and tested in digital twin environments and/or in practice when feasible (e.g., using a hardware-in-the-loop emulator).

In order to cope with the absence of a machine prototype for Pilot 3, the open data set(s) available online will be selected and used: in detail, from one side, a part of these data will be used to generate suitable AI algorithms and, from the other, the remaining part of the data set will be streamed in real-time to the real-time smart-control platform (i.e., using Matlab instances). In detail, this approach will cope with the fact that in Pilot 3, sensors are also not available, and therefore, simulated sensors are enabled for verification and validation of the IMOCO4.e solution. In this perspective, the real-world sensors will be replaced by simulated instances of real-time data generated via Matlab (or equivalent approach) and transmitted to the Real-Time Smart-Control Platform of Layer 3. The real-time smart-control platform in Layer 3 has a heterogeneous and flexible architecture where the sensor data elaboration and AI algorithms can be deployed on an AI (ARM) accelerator, on the Xilinx ZUX platform and/or the Nvidia Xavier platform, depending on the resources required by each software solution to be deployed. The actual adoption of which platform to use for which purpose is still under investigation.

The data fusion bus uses the data management framework explained in Section 3.3 for interfacing Layer 3 with the cloud or fog (Layer 4). AI applications can also be deployed in this environment. When possible, the connection among physical boards will use the TSN protocol. The fog environment for Layer 4 refers to the close (geographical) proximity of the quality of service AI solution to Layer 3.

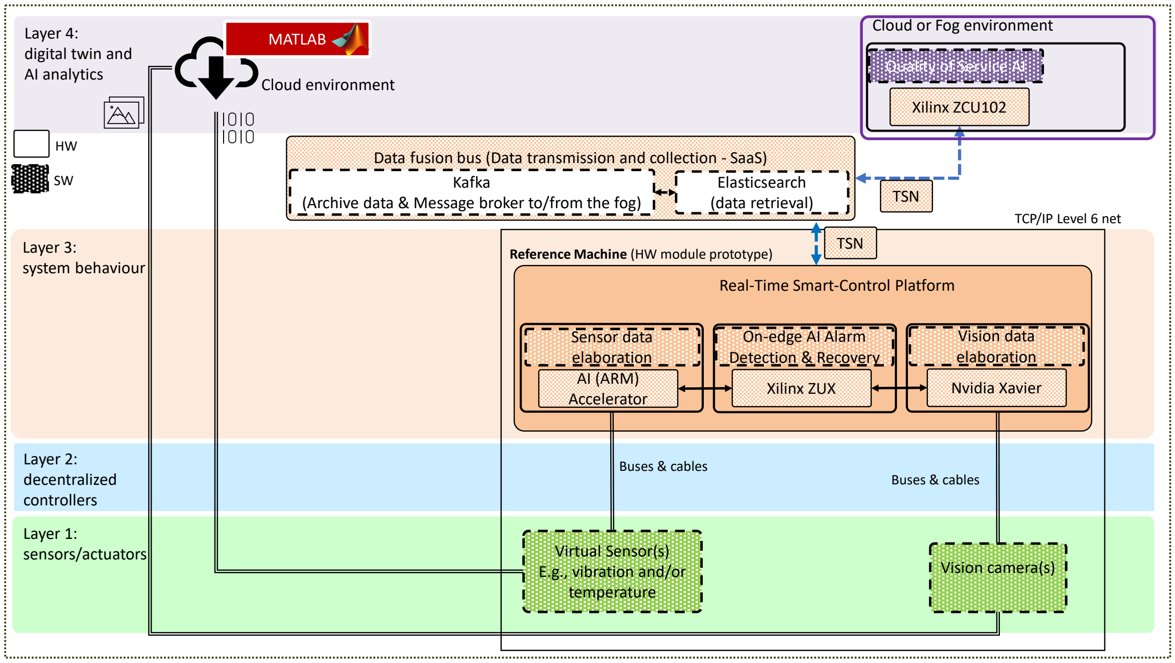

Figure 12 Pilot 3 architecture and example of refinement in the absence of a machine prototype

4.4 Pilot 4 architecture refinement with CI/CD infrastructure

As a healthcare provider, Philips (Pilot 4 owner) focuses on improving treatment for every patient, every time. At Philips Image Guided Therapy Devices, the focus is on creating systems that provide unlimited imaging flexibility for diverse procedures and exceptional positioning freedom for medical teams. With intuitive Axsys side controls, Philips' machines allow the user to make procedures flow naturally and easily, allowing the physician to take action quickly. The goal is to bring meaningful innovations to healthcare providers ready for the procedures now and in the future.

The main objectives of Pilot 4 in IMOCO4.E are to (i) accelerate the development & deployment of meaningful innovations to the market by use of a (digital twin) model-driven approach; (ii) extend system monitoring of the development systems to enable early feedback, fast improvements and test & training data for our (digital twin) model-driven approach; and (iii) time optimised and first-time-right service & maintenance of systems in the field by extending the system monitoring and use of state-of-the-art data model technology.

Figure 13 outlines the Pilot 4 architecture refinement along the CI/CD infrastructure. A clear distinction and interface between the development phase and the production phase components are illustrated. The motors and encoders are the actuators and sensors (Layer 1). For the development phase, additional (smart) obstacle avoidance sensors are envisioned and interface to the application layer (Layer 3) over USB or ethernet. The servo drives and the I/O module span across Layers 1 and 2. For CI/CD, the developers use a motion control platform that spans Layers 2 and 3 for mechatronics development, integration and tests. The mechatronics development uses Simulink software and custom codes for MIMO feedforward and feedback control. A personal development laptop with Simulink software and the required tools is used for development/integration. The laptop can access the CI/CD infrastructure (Layer 4), and the code for deployment to the production machine can be uploaded/generated. Digital Twins, combined with a Virtual (Reality) Test environment, are part of the infrastructure (Layer 4) and enable fast and safe development based on these models.

The motion control platform for the production machine executes the feedforward and motion algorithms in real time. The motion control platform is also the bus master for the EtherCAT fieldbus that interfaces with the servo drives. Local feedback control is executed directly on the drives. The operator of the machine interacts with the host PC, and the code for deployment is updated using the CI/CD infrastructure.

During each phase in the machine lifecycle (development, factory and field), the 'big data' from the machine is fed to the cloud (Layer 4) database. Monitoring data from the machines and training digital twin models enable condition monitoring and predictive maintenance services.

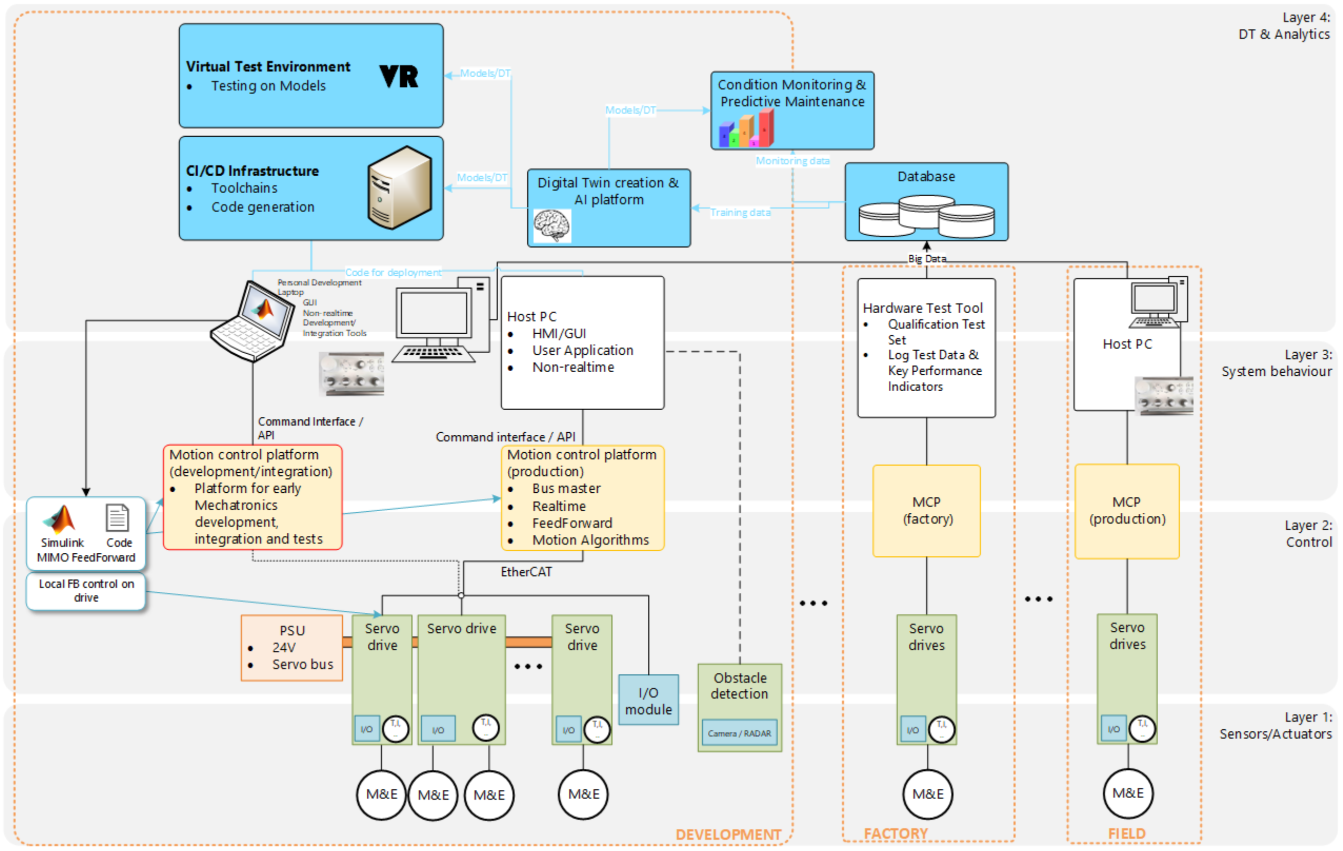

Figure 13 Pilot 4 architecture refinement with CI/CD infrastructure

4.5 Pilot 5 architecture

Figure 14 outlines the IMOCO4.E reference framework refinement for Pilot 5. Robotic boom manipulators are the enabling technology for autonomous underground processes. IMOCO4E technologies are piloted in at least a digital twin of a mining / tunneling robotic boom manipulator on a carrier where feasible. Layer 1 for Pilot 5 comprises motors, analog and digital sensors, safety LIDAR, safety limit switches, and work environment scanning. Servo drives and safety I/O modules span Layers 1 and 2. The manipulator control and the platform control components span Layers 2 and 3. The manipulator control performs AI-assisted tasks, provides a user interface for HMI, uses the point cloud data from work environment scanning for motion planning, and executes motion control algorithms. The manipulator control executes on an RTOS. A safety system (SIL3/PL e standard) is also present in the manipulator control component. The manipulator control component is interfaced with the lower layer components through a fieldbus. The platform control component provides a user interface for HMI and executes process control, platform control and auxiliary equipment control algorithms.

Layer 4 of Pilot 5 comprises the process planning software that interfaces with the motion planning algorithm executing on the manipulator control component. The actual data is uploaded to the planning software, and the process planning data is provided to the manipulator control component. The algorithm development and modelling component in Layer 4 use Matlab/Simulink and custom codes. The algorithms are deployed to the manipulator control and platform control components. A XIL development environment with support for VR and HW control modelling is also present in Layer 4. XIL development environment interfaces with the manipulator control with Ethernet bus and the platform control with fieldbus. Diagnostics data from the platform control is also stored in a cloud server for machine diagnostics using WLAN/WAN.

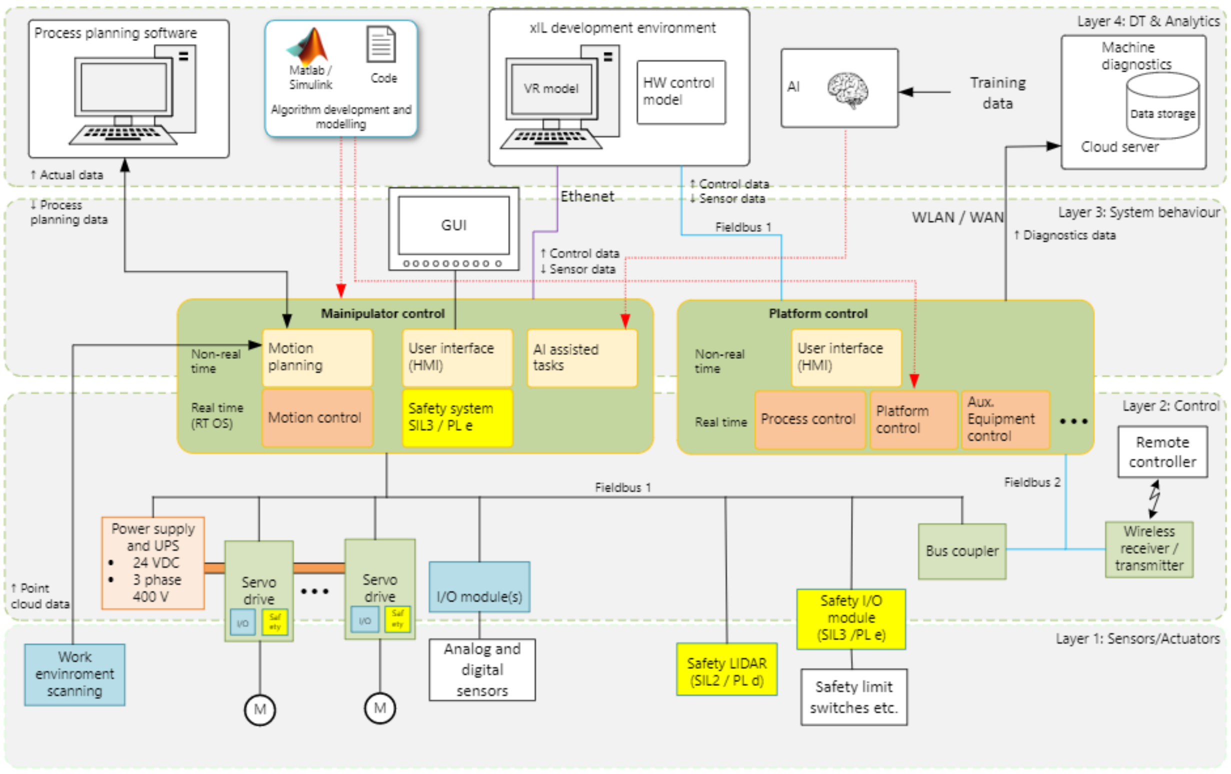

Figure 14 Pilot 5 architecture refinement

4.6 UC1 architecture refinement

Figure 15 outlines the IMOCO4.E reference framework refinement for use-case 1. The focus is on an industrial drive specialised for LIFT application. The drive is a new WEG product named ADL500 and it is inserted in Layer 2. The drive has the capability to manage interfaces like encoders (Layer 1) and has the aim to control the motor and brake dynamics. The drive is connected through a board that can be plugged directly or externally, which makes the control functions and safeties of the LIFT system. In addition, layer 3 is composed of an industrial router that allows the connection of the drive to the Internet services. The internal connection between Layers 2 and 3 is done using MODBUS TCP/IP and CAN-OPEN (the standard used in the lift application). Layer 4 is composed of a cloud service aiming to monitor and process the data provided by each installation.

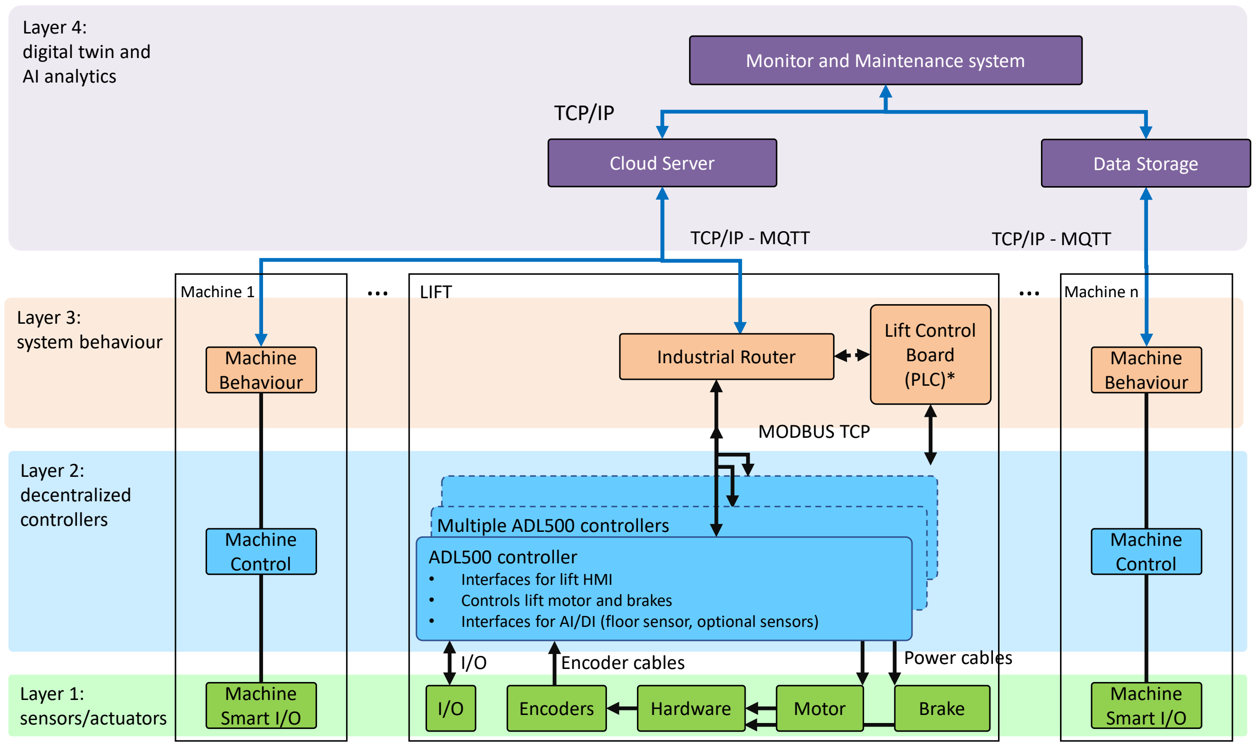

Figure 15 Use Case 1 architecture refinement

4.7 UC2 architecture refinement

Figure 16 outlines the IMOCO4.E reference framework refinement for use-case 2. The focus is on CNC for integrated machine tool and robot control. The motion control platforms in Layer 2 are the FAGOR drives which supports interfaces with encoders (Layer 1), control servo axis and also has 3-phase amplifier. FAGOR drives interface with the CNC platform (Layer 3) using EtherCAT. The CNC does trajectory planning and multi-axes control, PLC, and collision avoidance: machine-robot. A centralised database exists in Layer 4. Layer 3 interfaces with this database using OPC-UA/MQTT protocol. Preventive maintenance, diagnostics and virtual commissioning tasks use this centralised database.

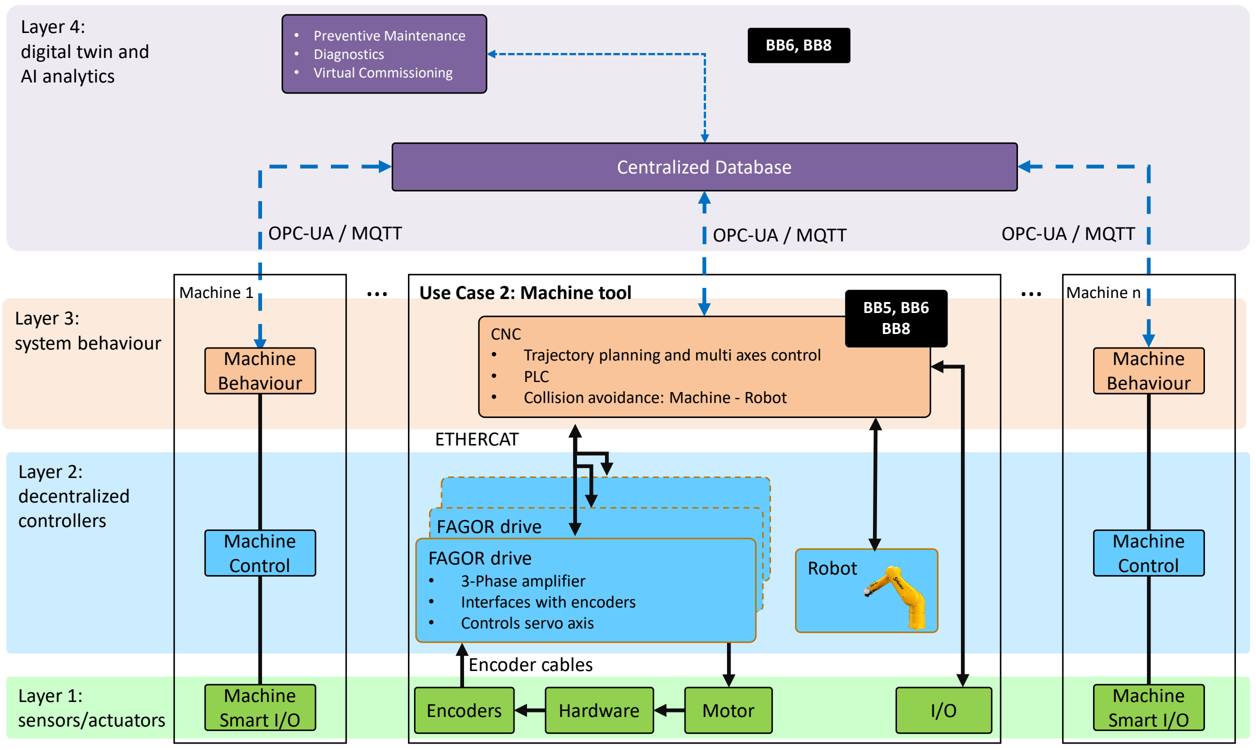

Figure 16 Use Case 2 architecture refinement

4.8 UC3 architecture for remote teleoperation

The Tactile Robot constitutes the next generation of soft collaborative robots equipped with sensing capabilities to process humanlike tactile sensations. Human safety and labour/skill shortages in the industry will be improved dramatically, as potentially dangerous or complex operations involving inspection, repair, or even decommissioning will be performed by a remotely controlled Tactile Robot. Use case 3 will implement safe remote teleoperation via a tactile robot. Human in the loop will be considered through complex HMI coupled with a digital twin representation of the process implemented in virtual reality. The application will be enabled with high-performance AI embedded close to the edge, mitigating motion control errors introduced as a result of sensor and user input limitations.

Figure 17 and Figure 18 outline the IMOCO4.E reference framework refinement for the remote teleoperation application (UC3). The key difference of this refinement from the reference architecture is that the machine-of-interest is distributed. The remote teleoperation requires a local (user) end machine and the remote robot end machine. The local (user) end has three distinct components for HMI processing, TOF processing and HMI & TOF data synchronisation, fusion and error mitigation. The HMI processing component has an HMI Tyndall glove sensor for motion tracking. The human user wears the glove. The TOF processing component has a TOF camera sensor. The HMI and TOF data are extracted at Layer 1. The data is synchronised and fused on the SoC-FPGA-enabled edge device at Layer 2. The prediction/classification of both the robot gripper and the robot arm is executed on the edge device. The AI edge services (self-health, diagnostics, anomalous behaviour detection, and behavioural prediction solutions) are done at Layer 3. Layer 4 supports DT/VR model and updates along with AI analytics. A DT/VR UI enables interfacing with the user.

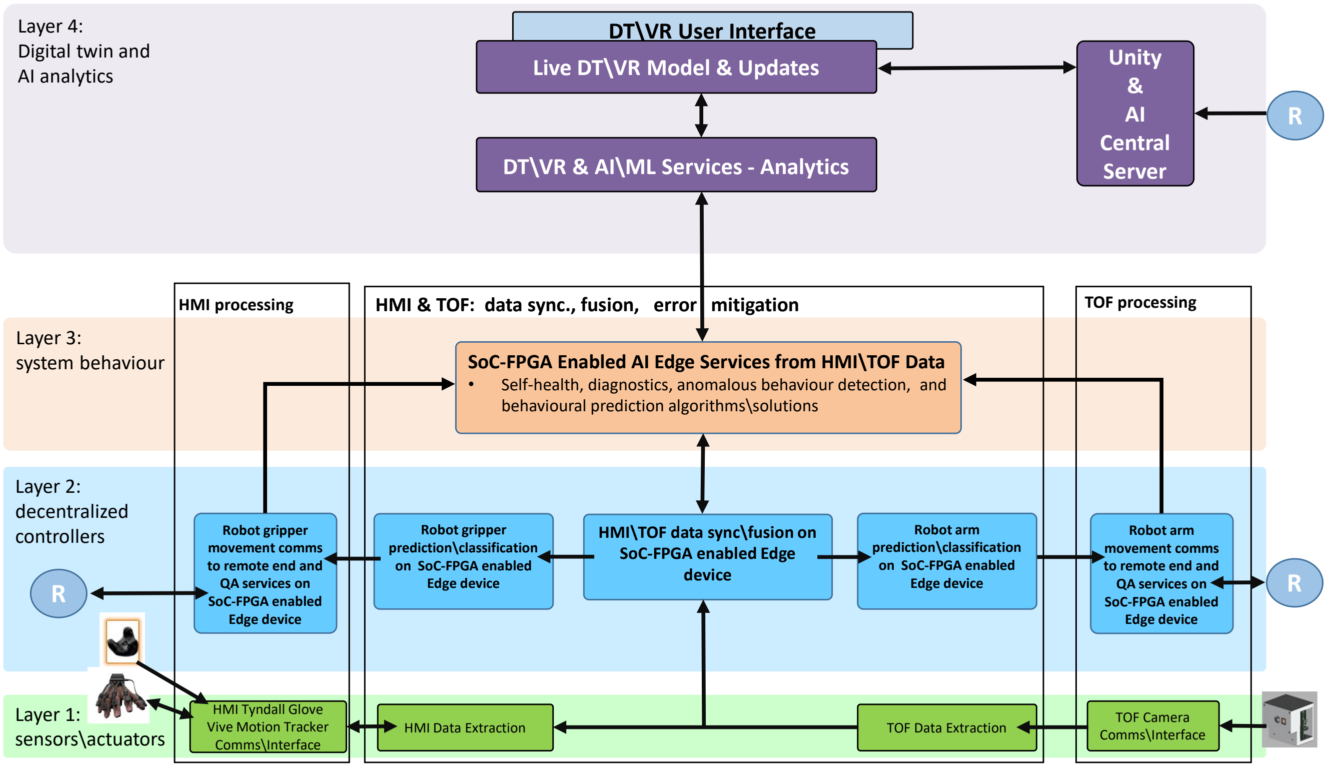
Layer 4:
Digital twin and
AI analytics
DT\VR User Interface
Live DT\VR Model & Updates
DT\VR & AI\ML Services - Analytics
Unity & AI Central Server
R
HMI processing
HMI & TOF: data sync., fusion, error mitigation
TOF processing
Layer 3:
system behaviour
SoC-FPGA Enabled AI Edge Services from HMI\TOF Data
Self-health, diagnostics, anomalous behaviour detection, and behavioural prediction algorithms\solutions
Layer 2:
decentralized
controllers
R
Robot gripper movement comms to remote end and QA services on SoC-FPGA enabled Edge device
Robot gripper prediction\classification on SoC-FPGA enabled Edge device
HMI\TOF data sync\fusion on SoC-FPGA enabled Edge device
Robot arm prediction\classification on SoC-FPGA enabled Edge device
Robot arm movement comms to remote end and QA services on SoC-FPGA enabled Edge device
R
Layer 1:
sensors\actuators
HMI Tyndall Glove Vive Motion Tracker Comms\Interface
HMI Data Extraction
TOF Data Extraction
TOF Camera Comms\Interface

Figure 17 Local (user) end reference architecture for UC3

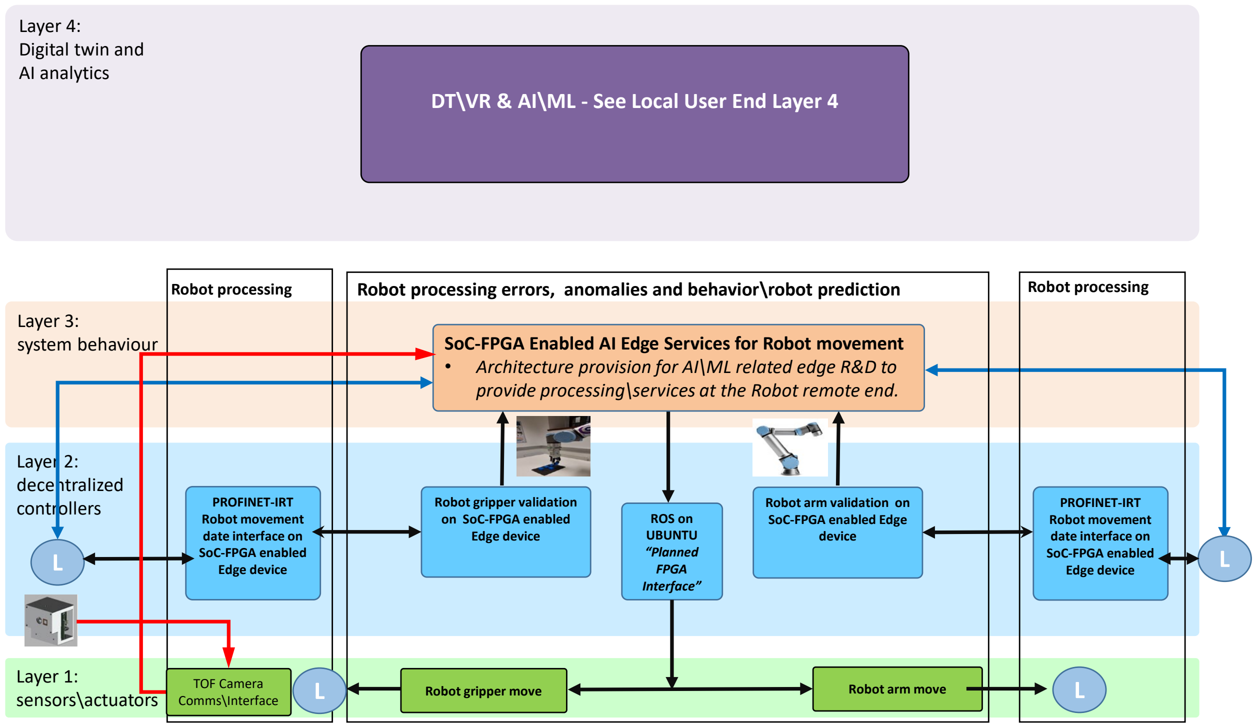
Layer 4:
Digital twin and
AI analytics
DT\VR & AI\ML - See Local User End Layer 4
Robot processing
Robot processing errors, anomalies and behavior\robot prediction
Robot processing
Layer 3:
system behaviour
SoC-FPGA Enabled AI Edge Services for Robot movement
Architecture provision for AI\ML related edge R&D to provide processing\services at the Robot remote end.
Layer 2:
decentralized
controllers
L
PROFINET-IRT Robot movement date interface on SoC-FPGA enabled Edge device
Robot gripper validation on SoC-FPGA enabled Edge device
ROS on UBUNTU "Planned FPGA Interface"
Robot arm validation on SoC-FPGA enabled Edge device
PROFINET-IRT Robot movement date interface on SoC-FPGA enabled Edge device
L
Layer 1:
sensors\actuators
TOF Camera Comms\Interface
L
Robot gripper move
Robot arm move
L

Figure 18 Remote (robot) end reference architecture for UC3

The remote end (illustrated in Figure 18) has three robot processing components - one for the robot arm, one for the robot gripper, and the last one for error processing, anomalies and behaviour/robot prediction. The actuators are the robot gripper and the robot arm (Layer 1). Layer 2 performs robot gripper validation and robot arm validation on SoC-FPGA-enabled edge device. Layer 3 has an architecture provision for AI/ML-related edge R&D to provide processing/services at the robot's remote end.

4.9 UC4 architecture refinement

The collaborative robotic platform provides a 7DoF robotic arm that can be employed in applications requiring fast and flexible adoption of complex motion trajectories in hybrid environments requiring close cooperation of robots and human workers. The goal is to allow a simple definition of motion tasks even by an unskilled operator without the necessity of tedious programming via conventional interfaces, i.e. by hand-coding or teach-pendant jogging. This is achieved by using special haptic interfaces enabling hand-guidance motion teaching regimes. The redundant kinematics of the robot extends its dexterity by enlarging its operational space. This can be beneficial for operation in confined spaces where a redundancy resolution can provide additional degrees of freedom for robot body shaping. Potential application domains include nondestructive inspection and testing or material handling.

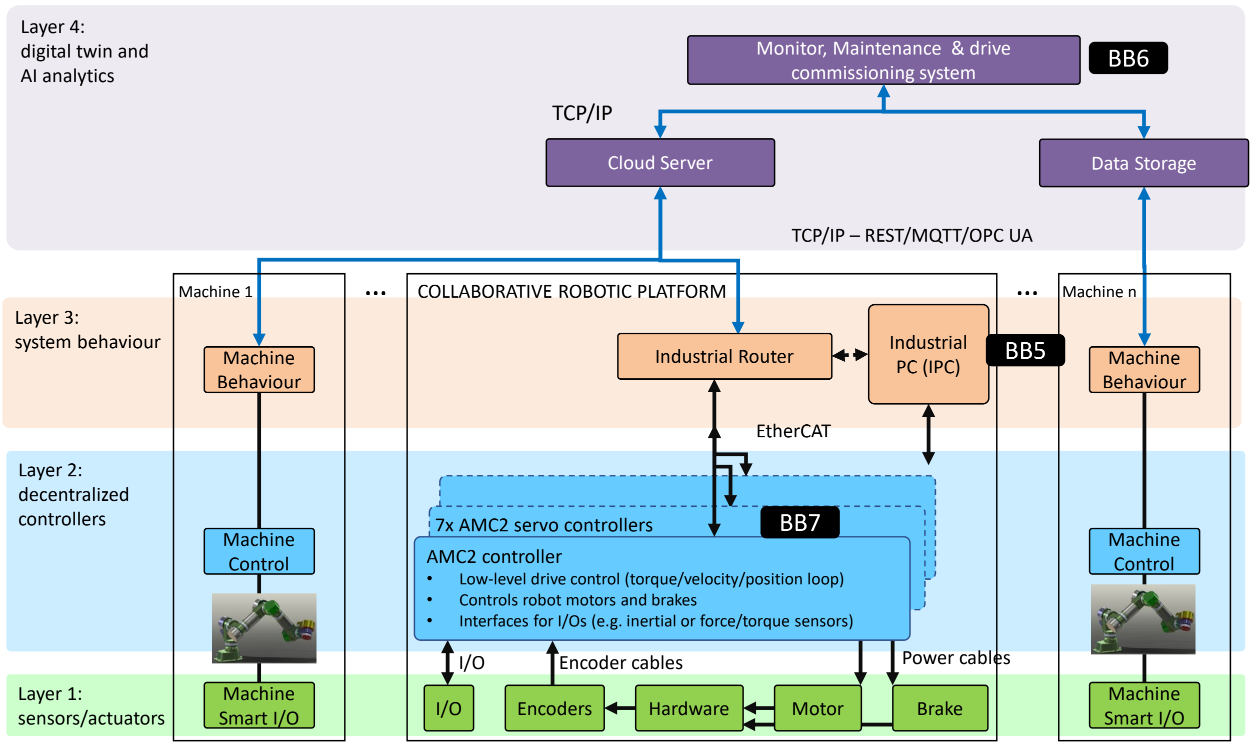

Figure 19 UC4 architecture refinement

Figure 19 provides a decomposition into individual layers of the control architecture. Layer 1 contains sensors and actuators embedded in the integrated joints of the robotic arm. Layer 2 deals with decentralised control at the servo-drives level. Building block 7 developed by UWB will be integrated here, providing high-fidelity torque, velocity and position control. Layer 2 is connected to the sensors/actuators layer by corresponding cabling. EtherCAT communication link is established to Layer 3, with an industrial PC (IPC) serving as a centralised motion controller responsible for coordinated motion planning and synchronisation. Advanced motion control algorithms developed in terms of BB5 will be integrated here. The highest layer, number 4, establishes a monitoring, maintenance and drive commissioning system capable of retrieving and processing relevant machine data. This is a place for employment of developed BB6 functionalities.

4.10 Demonstrator 1 architecture refinement

Figure 20 illustrates the brownfield reference architecture for Demonstrator 1, with the connected Building Blocks. Demonstrator 1 focuses on high-precision cold forming of complex 3D metal parts. Demonstrator 1 targets to measure the selected process using sensors, translating these data to control the machines in real-time. A digital twin (and/or AI analytics) can be added to optimise the process settings or make an autonomous process control. If needed, new sensors could be needed (BB3) for this. The IT infrastructure supports data flow, analytics and process control.

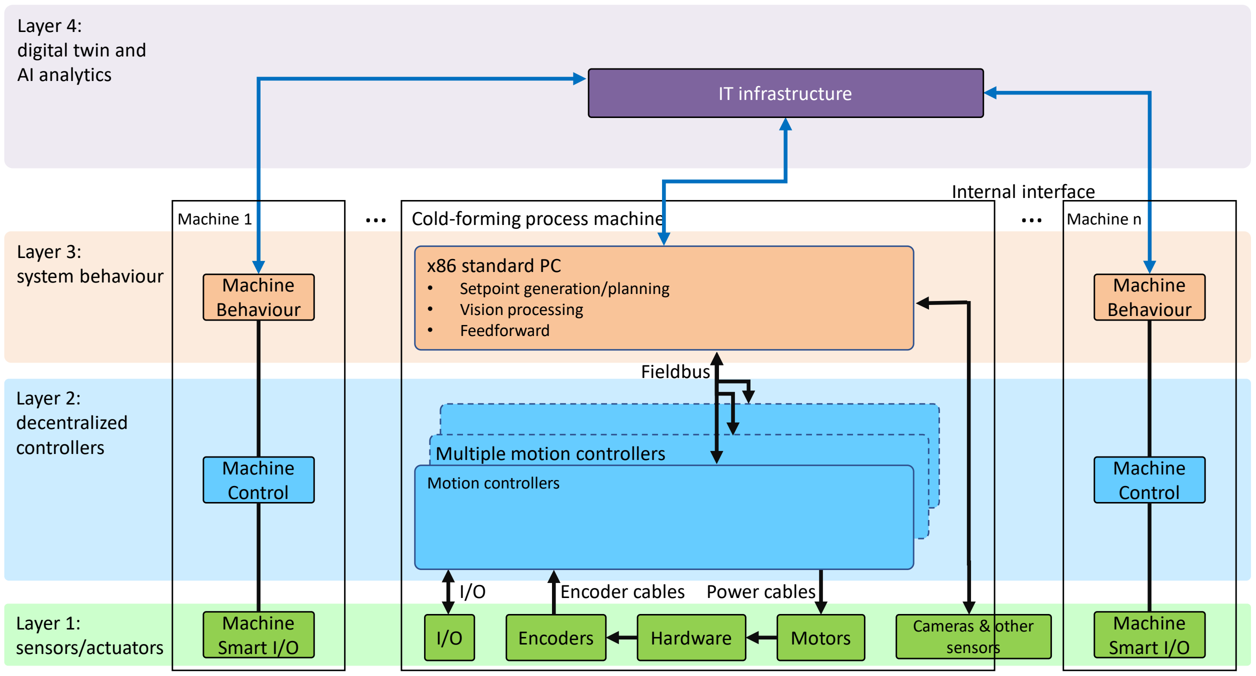

Figure 20 Demonstrator 1 architecture refinement

4.11 Demonstrator 2 architecture refinement

Figure 21 outlines the IMOCO4.E reference framework refinement for Demonstrator 2. Nowadays, equipping tools with artificial intelligence to allow continuous monitoring is a key component in industrial production. Demonstrator 2 targets to overmold wireless sensors (temperature, pressure) and RFID tags on plastic parts. A controller device could read the data transmitted by the sensors in real time. The demonstrator intends to transpose the logic of industry 4.0 into the final product to introduce new functionalities in tools for plastic injection and create an innovative product with more add-value and high incorporation of R&D.

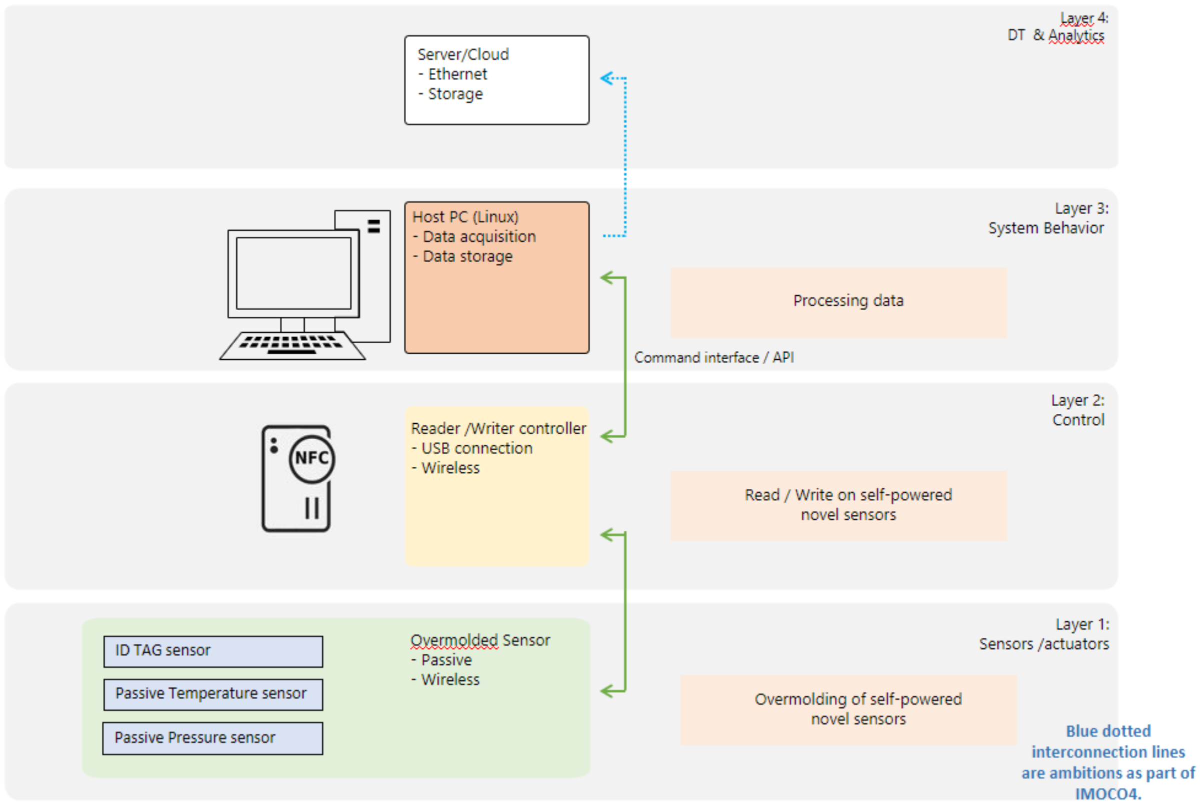

Figure 21 Demonstrator 2 architecture refinement

4.12 Demonstrator 3 architecture refinement

Demonstrator 3 includes all the elements required for the transformation from a classic automated system to a modern autonomous system for internal transport tasks. This means that known standardised brownfield solutions from automation must be extended with new (greenfield) capabilities, so-called senses, for the necessary perception of the environment. This leads to an extension of the sensor technology, as well as to the addition of novel intelligent solution strategies. Figure 22 shows a design sketch for a robust autonomous SW architecture spanning Layers 1-3 of the IMOCO4.E framework of Demonstrator 3, which is shown in Figure 23. The interaction and mapping of the corresponding modules of the framework are explained in the following sections.

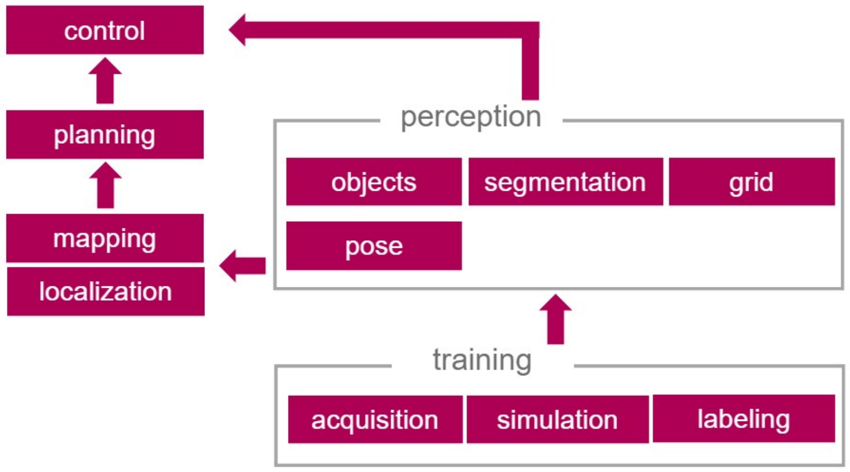

Figure 22 Demonstrator 3 autonomous robot software architecture

In Layer 1, in addition to the existing vehicle control sensors, visual sensors for intelligent detection of semantics like pallets, vehicles, persons and labels were added in the form of cameras and radar sensors. The core task for new functionalities in Layer 1 is in the perception of

- Objects (persons, vehicles or load) - pose estimation of the load carrier and the collisions avoidance
- Segmentation and grid (Floor) – short-term navigation and collision avoidance
- Pose (e.g. pallet) – load handling

These elements include new interactions related to a new link between Layer 1 (optical detection by extended sensors) and Layer 2 (image processing and onboard evaluation) with special detector SW modules optimised for the requested functionality.

By using industry-proven training methods (acquisition, simulation and labelling), the functional output of the greenfield vision modules of Layer 2 is validated, and functional interaction with the behaviour-relevant brownfield modules of Layer 3 can be guaranteed. The basic functions of the modules from Layer 3 are localisation, mapping and the resulting planning for autonomous load handling. In IMOCO4.E in this section, industry-relevant modules such as global and local planning rely on various industry standards such as ROS 2. In Demonstrator 3, the behaviour of the vehicle (control) is composed of the pre-processing by Layer 2 and the final implementation of the vehicle instructions in Layer 3.

Layer 4 is a higher-level layer for Demonstrator 3, which is not included in Figure 22 because it is not a direct part of the internal vehicle control. Clear instructions to the subordinate vehicles, based on the information flow from the vehicles combined with external conditions, come from this Layer 4. Several vehicle systems are combined there to form a fleet, which is also controlled accordingly in this layer. Layer 4 is, therefore, necessary for an overall functional system. The possibilities of Layer 4 are also used to provide relevant data for the training of neural networks and to emulate a possible system behaviour in the simulation section of the cloud. A core advantage is the higher computing capacity available there than on the individual vehicles.

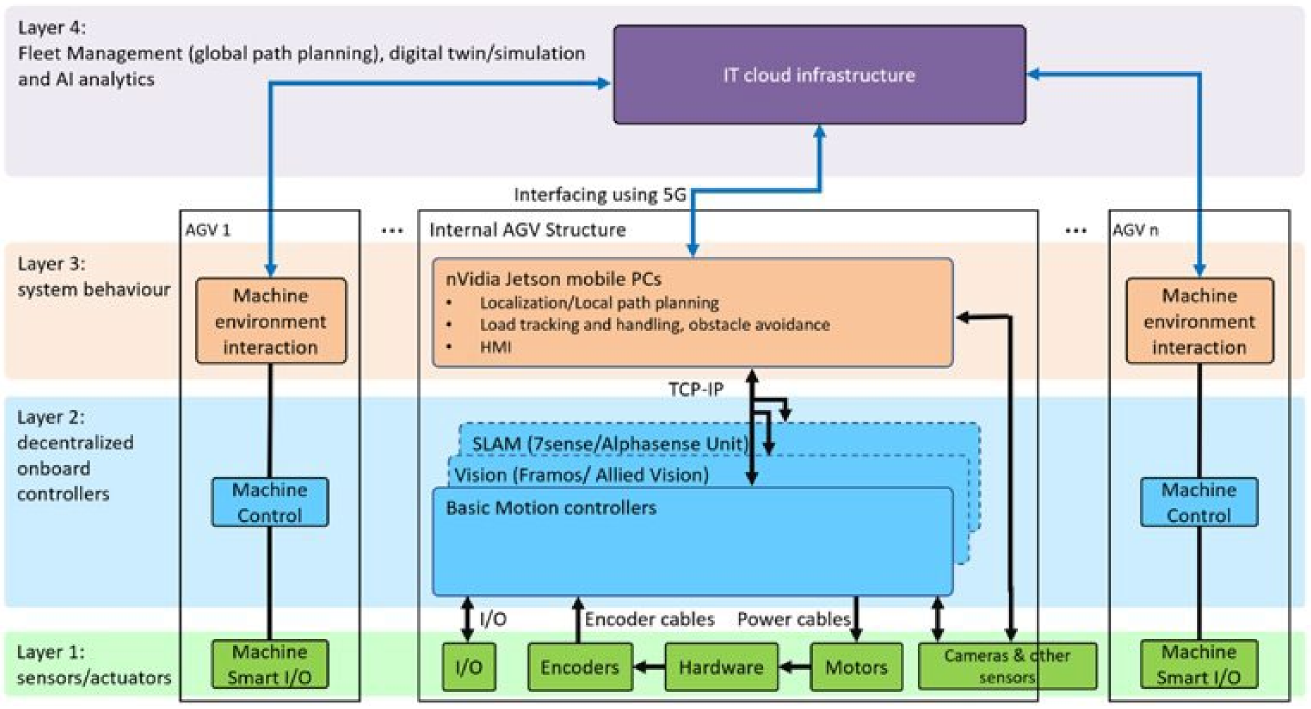

Figure 23 Demonstrator 3 global reference architecture

4.13 Demonstrator 4 architecture refinement

Figure 24 outlines the IMOCO4.E framework refinement for Demonstrator 4. The aim is to develop a computer vision-aided robot that could work in changing environments by applying reinforcement learning for faster deployment of the robot for different tasks. Ultimately the robot could replace manual human work with the vision-based (AI) pick & place robotic solution to be used on multiple industrial production lines to pick and place randomly arranged and differently shaped bottles in placeholders of various shapes and sizes.

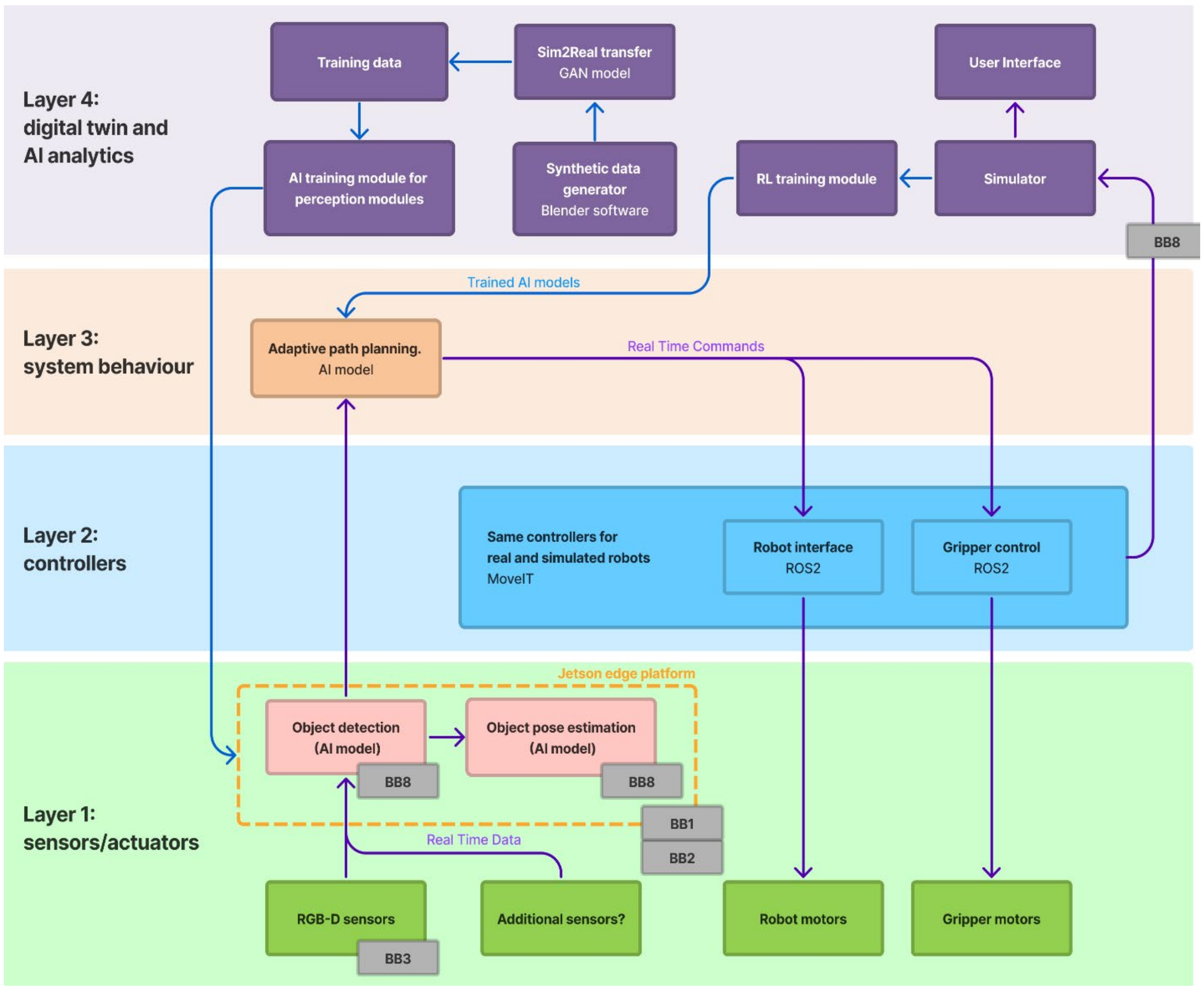
Layer 4: digital twin and AI analytics
Training data
Sim2Real transfer GAN model
User Interface
AI training module for perception modules
Synthetic data generator Blender software
RL training module
Simulator
BB8
Trained AI models
Layer 3: system behaviour
Adaptive path planning. AI model
Real Time Commands
Layer 2: controllers
Same controllers for real and simulated robots MoveIT
Robot interface ROS2
Gripper control ROS2
Layer 1: sensors/actuators
Jetson edge platform
Object detection (AI model)
BB8
Object pose estimation (AI model)
BB8
BB1
BB2
Real Time Data
RGB-D sensors
BB3
Additional sensors?
Robot motors
Gripper motors

Figure 24 Demonstrator 4 architecture refinement

5. Conclusions

The IMOCO4.E reference framework design and specifications are detailed in this deliverable. The reference framework is designed by following a step-by-step approach starting with gathering the brownfield reference architectures of the industrial cases (pilots, demonstrators and use cases) in the IMOCO4.E consortium. Based on the greenfield architectures of the industrial cases and their envisioned building blocks (BBs) integration, the architecture, data management, artificial intelligence (AI), and digital twin (DT) viewpoints are specified. The refinements of the reference framework for the pilots, demonstrators and use cases validate the applicability of the IMOCO4.E reference framework for various industrial domains throughout the engineering phases and under different constraints imposed on the industrial cases.

The proposed IMOCO4.E reference framework and refinements go beyond the current state-of-the-art and consider the scientific and technological (ST) development objectives, system integration and interoperability (SI) objectives, system operational (SO) objectives, and system exploitation (SE) objectives of the IMOCO4.E project. A strategy on how the requirements relevant to the reference framework can be verified, traced and fulfilled is also provided. In addition, suggestions to use model versioning for model management are proposed. Using model versioning implies that we could use the existing data management framework/viewpoint for model management.

The IMOCO4.E reference framework specified in this deliverable will form the basis for the further development of BB solutions and their integration with the pilots, demonstrators and use cases. In addition, the architecture, data management, AI and DT viewpoints will be refined and instantiated with the corresponding toolchains and methodology in the future deliverables of IMOCO4.E.

References

[1] Hans Kuppens, "D2.1 State-of-the-art methods in Digital Twinning for motion-driven high-tech applications." *Zenodo*, 2022.
Available online at: https://doi.org/10.5281/zenodo.7575974 (accessed 31 January, 2023).

[2] Matias Vierimaa, "D2.2 Needs for future smart production in Europe from the mechatronics and robotic point of view." *Zenodo*, 2022.
Available online at: https://doi.org/10.5281/zenodo.7588726 (accessed 31 January, 2023).

[3] Sajid Mohamed. "D2.3 Overall requirements on IMOCO4.E reference framework (3.0)." *Zenodo*, 2022. Available online at: https://doi.org/10.5281/zenodo.7529265 (accessed 27 January, 2023).

[4] IMOCO4.E Description-of-Action (Annex I Part B).
Public link (with limited details): https://www.imoco4e.eu/overview;
SharePoint link for IMOCO4.E consortium (confidential).

[5] Mart Coenen, "D3.1 Perception and instrumentation Layer requirements and specifications (first iteration)." 2022. Available online at:
imoco4e.eu/app/download/13271569949/D3.1.pdf?t=1673862894 (accessed 31 January, 2023)

[6] Diego Navarro Cabrera, Niceto Rafael Luque Sola, and Eduardo Ros Vidal. "D4.1 Requirements for advanced motion control (first iteration)." *Zenodo*, 2022.
Available online at: https://doi.org/10.5281/zenodo.7575243 (accessed 27 January, 2023).

[7] Rahul Tomar, "D5.1 Integral (system level) requirements for valuable twinning methods (first iteration)." *Zenodo*, 2022.
Available online at: https://doi.org/10.5281/zenodo.7588517 (accessed 31 January 2023).

[8] Carlos Rodriguez de Yurre, and Hans Kuppens. "D6.1 Guideline of IMOCO4.E methodology and toolchains." *Zenodo*, 2022.
Available online at: https://doi.org/10.5281/zenodo.7573258 (accessed 27 January, 2023).

[9] Charles S Wasson. "System engineering analysis, design, and development: Concepts, principles, and practices." *John Wiley & Sons*, 2015.

[10] Jennifer Stapleton. "DSDM, dynamic systems development method: the method in practice." *Cambridge University Press*, 1997.

[11] Jay Kreps, Neha Narkhede, and Jun Rao. "Kafka: A distributed messaging system for log processing." *Proceedings of the NetDB.* Vol. 11. 2011.

[12] Gormley Clinton, and Zachary Tong. “Elasticsearch: the definitive guide: a distributed real-time search and analytics engine.” *O’Reilly Media, Inc.*, 2015.

[13] Zolkifli, Nazatul Nurlisa, Amir Ngah, and Aziz Deraman. “Version control system: A review.” *Procedia Computer Science* 135 (2018): 408-415.

[14] Loeliger, Jon, and Matthew McCullough. “Version Control with Git: Powerful tools and techniques for collaborative software development.” *O’Reilly Media, Inc.*, 2012.

[15] Pilato, C. Michael, Ben Collins-Sussman, and Brian W. Fitzpatrick. “Version control with subversion: next generation open source version control.” *O’Reilly Media, Inc.*, 2008.

[16] Ruslan Kuprieiev, et al. “DVC: Data Version Control – Git for Data & Models” *Zenodo*, 2021. Available online at: https://zenodo.org/record/5654595 (accessed 27 January, 2023)

[17] Petra Brosch, Gerti Kappel, Philip Langer, Martina Seidl, Konrad Wieland, and Manuel Wimmer. “An introduction to model versioning.” In *International school on formal methods for the design of computer, communication and software systems*, pp. 336-398. Springer, Berlin, Heidelberg, 2012.

[18] Kerstin Altmanninger, Gerti Kappel, Angelika Kusel, Werner Retschitzegger, Martina Seidl, Wieland Schwinger, and Manuel Wimmer. “AMOR–towards adaptable model versioning.” In *1st International Workshop on Model Co-Evolution and Consistency Management, in conjunction with MODELS*, vol. 8, pp. 4-50. 2008.

[19] Dominik Kreuzberger, Niklas Kühl, and Sebastian Hirschl. “Machine learning operations (MLOps): Overview, definition, and architecture.” *arXiv preprint arXiv:2205.02302*, 2022.