Intelligent Motion Control under Industry 4.E

D5.8 Report on digital twins, corresponding supporting technologies and their interaction with the cloud

Due Date: M32 – 2024-04-30

Abstract:

This deliverable is the concluding report of WP5. It describes BB6, BB8 and BB9 components, description of developed interfacing, datasets management on different layers, modelling of complex mechatronics systems and the description of digital twinning technology developed in this WP. The results briefly address the fulfilment of requirements and specifications defined in (D5.1), (D5.2) and in (D2.3). Detailed tests will be the part of WP6 and corresponding reports.

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	WP5 - Digital Twins and their interaction with the cloud
Lead Beneficiary	Brno University of Technology (BUT)
Deliverable Title	Report on digital twins, corresponding supporting technologies and their interaction with the cloud
Version	5.0 (starting from 1.0)
Date of Submission	30/04/2024
Author(s)	Petr Blaha (BUT), petr.blaha@ceitec.vutbr.cz
Contributor(s)	Tariqul Islam (DTT), tariqul.islam@digitaltwin.technology Sajid Mohamed (ITEC), sajid.mohamed@itecequipment.com Michael Geurtsen (ITEC), michael.geurtsen@itecequipment.com Tassos Kanellos (GNT), tasos.kanelos@gnt.gr, Fabien Bouquillon (UNIMORE), fabien.bouquillon@unimore.it Álvaro Arco (SED), alvaro.arco@nav-timing.safrangroup.com Martin Poelstra (SIOUX), martin.poelstra@sioux.eu Ilektra Simitsi (ITML), isimitsi@itml.gr Ronald Rietman (PEN), ronald.rietman@philips.com Javier Arenas (FAG), jarenas@aotek.es

	Calin Husar (SIEMENS), calin.husar@siemens.com Bohumil Klima (BUT), klima@vut.cz Jakub Hnidka (BUT), hnidka@vut.cz Zdenek Havranek (BUT), havranek@vut.cz Martin Dosedel (BUT), martin.dosedel@ceitec.vutbr.cz Diego Gonzales (TEK), diego.gonzalez@tekniker.es Henry Stoutjesdijk (PMS), henry.stoutjesdijk@philips.com Ansgar Bergmann (STILL), Ansgar.Bergmann@kiongroup.com Max van Haren (TUE), m.j.v.haren@tue.nl Manuel Beschi (UNIBS), manuel.beschi@unibs.it Davide Colombo (WEG), dcolombo@weg.net Roberto Pagani (UNIBS), roberto.pagani@unibs.it Martin Goubej (UWB), mgoubej@ntis.zcu.cz Angela Dimitriou (INTRA), angela.dimitriou@netcompany.com Rahul Tomar (NURO), rahul.tomar@nuromedia.com Jakko Nieuwenkamp (REDEN), j.nieuwenkamp@reden.nl Luca Pulina (UNISS), lpulina@uniss.it Kalle Määttä (NORMET), kalle.maatta@normet.com Jan Wytze van der Weit (PEN), jan.van.der.weit@philips.com Diego Navarro Cabrera (UGR), diegonavaca@ugr.es Pekka Kilpeläinen (VTT), pekka.kilpelainen@vtt.fi Amer, Nehal (ADI), Nehal.Amer@analog.com Alan Dunne (EMDALO), alan.dunne@emdalo.com Alfie Keary (UCC), alfie.keary@tyndall.ie Vivian Tai (CYBERTRON), zurich.office@reexen.com Harry Luoma (EXE), Harry.Luoma@exertus.fi							
Internal Reviewer(s)	Jakko Nieuwenkamp (REDEN), j.nieuwenkamp@reden.nl Luca Pulina (UNISS), lpulina@uniss.it Kalle Määttä (NORMET), kalle.maatta@normet.com Manuel Beschi (UNIBS), manuel.beschi@unibs.it Tassos Kanellos (GNT), tasos.kanelos@gnt.gr, Bohumil Klima (BUT), klima@vut.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
1.0	26/09/2023	Draft	WP5 partners	Petr Blaha	Initial version prepared.
2.0	29/01/2024	Draft	WP5 partners	Petr Blaha	Updated template structure.
3.0	17/03/2024	Draft	WP5 partners	Petr Blaha	Integration of sections prepared by task leaders.
4.0	10/04/2024	Pre-final	Reviewers	Petr Blaha	Integrated version for the review.
5.0	29/04/2024	Final	PU	Petr Blaha	Integration of comments gathered by task leaders. Finalization of the document.

Type of Contribution	
Partner	*Description of contribution to contents*
ADI	Inputs to section 6.8 Virtual reality digital twins, enabling human-computer interaction.
BUT	Input to chapter 3.3 and 3.4, leading the preparation of the document.
CRIT	CRIT was involved in requirements preparation phase in Task 5.1. They did not plan to contribute to this deliverable.
CYBERTRON	Input to section 3.7.
DTT	Input to 2.11 Deploying DTT's anomaly detection AI model in Edge and Fog environments via Kafka to achieve predictive maintenance for manufacturing processes. Input to 5.9 Digital twit of a chip manufacture line. Input to 6.9.7 Spline planner for real truck to generate given path from VR-setup for mobile robot. Input to 6.10.2 Technical setup overview and 6.10.3 ROS-Unity bridge interface. Input to Section 7.5 Image Anomaly Detection AI model.
Emdalo	Inputs to section 6.8 Virtual reality digital twins, enabling human-computer interaction.
EXE	Inputs to chapter 6.6 Digital twin toolchains for control system development.
FAGOR	Input to section 5.5 Model for collision detection in hard real-time CNC application.
GNT	Input to section 2.1 'Overview of all realized solutions', Input to section 2.2 'Addressed ST objectives and KPIs', Input to section 2.3 'Distributed Messaging System (DMS)', Input to section 2.4 'Big Data Repository (BDR)', Input to section 2.6 'Cyber-Security Module (CSM)', Input to section 7.6 'ML Algorithms for data analytics and predictive maintenance'.
INTRA	Input to section 7.7.
ITEC	Inputs to the IMOCO4.E reference framework, Section 7.8.
ITML	Input to section 2.7 'Data Visualisation Toolkit (DVT)', Input to section 7.6 'ML Algorithms for data analytics and predictive maintenance'.
NORMET	Inputs to section 6 Augmented and virtual reality through digital twins and chapter 6.6 Digital twin toolchains for control system development.
NURO	Input to sections 6.9, 6.10 and 7.9.

PCL	Inputs to section 6.4 and 6.7.
PEN	Input to 2.9 'Healthcare Robotics Data Management Pipeline (PEN, PMS)', input to 7.10.
PMS	Input to 2.9 'Healthcare Robotics Data Management Pipeline (PEN, PMS)', input to 3.6 and 7.10.
REDEN	Input to section 5.3, 6.8, 7.11, coordination of writing section 5 as Task 5.5 leader.
SED	Input to section 2.8 'Time-Sensitive Networking (TSN)'.
SIEMENS	Input to section 2.10 'UC1 xIL co-simulation through cloud'. Inputs to section 6.5 Building digital twins using reduced order models with measured data.
SIOUX	Inputs to chapter 2.6 'Cyber-Security Module (CSM)', Inputs to chapter 5.8 'Digital twin for transient thermal behaviour of mechatronic systems', Inputs to chapter 6.3 Automated generation of digital twins.
STILL	Input to section 6.9 Comparative User Experience test of driving behaviour in encounters of mobile robots with human traffic participants in a virtual and a real environment. Inputs to section 6.10 VR-based tool for User Experience evaluation of driving behaviour of mobile robots in encounters with human traffic participants.
TEKNIKER	Input to section 4.5 Virtual commissioning of complex PLC/CNC projects.
TU/e	Developed parametric identification techniques of multirate systems for digital twinning, that is reported in Section 4.6.
UCC	Inputs to section 5.8.8 Virtual reality digital twins, enabling human-computer interaction.
UGR	Input to section 5.7 Data-driven dynamic modelling of a robotic arm.
UNIBS, WEG	Contents for Chapter 3.5 and 4.3.
UNIMORE	Input to section 2.5 'Field Gateway Proxy (FGP)', Inputs to 6.5 Platform configuration for high-speed digital twinning.
UNISS	Input to section 6.3, 7.3, 7.4. Activities related with leading task 5.7.
UWB	Reporting on Section 4.3 on Self tuning of fixed structure controllers, Section 5.3.3 on Digital Twin models of Current calibration tables for medical manipulator and Section 4.4.
VTT	Inputs to chapter 6.6 Digital twin toolchains for control system development and inputs to chapter 5.6 Deflection modelling of a high reach mining boom.

Table of Contents

List of Figures

List of Tables

Abbreviations

Abbreviation	Explanation
ADMOS	Anomaly Detection Application For Machine States
AE	Auto Encoder
AGV	Automated Guided Vehicle
AI	Artificial Intelligence
AMDV	Actual Motor Drive Velocity
AMR	Autonomous Mobile Robot
API	Application Programming Interface
APM	Application Performance Monitoring
AR	Augmented Reality
ASIC	Application Specific Integrated Circuit
AUC	Area Under Curve
BB	Building Block / Bounding Box
BDR	Big Data Repository
BLDC	Brushless Direct Current
BLE	Bluetooth Low Energy
BPTT	Backpropagation Through Time
CAD	Computer Aided Design
CatE	Computing at the Edge device
CCT	Current Calibration Tables
CD	Continuous Delivery
CI	Continuous Integration
CNC	Central Network Controller
CNN	Convolutional Neural Network
CoE	CAN over EtherCAT
CPU	Central Processing Unit
CSM	Cyber-Security Module
CSS	Cascading Style Sheets

CTQ	Critical to Quality
CWT	Continuous Wavelet Transform
DAE	Differential-Algebraic system of Equations
DC	Direct Current
DCN	Deformable Convolutional Network
DL	Deep Learning
DMS	Distributed Messaging System
DOF	Degree Of Freedom
DPU	Deep Learning Processing Unit
DSL	Domain Specific Language
DT	Digital Twin
DTC	Development of a Digital Test Cell
DUT	Device Under Test
DVT	Data Visualisation Toolkit
EEV	End Effector Velocity
EKF	Extended Kalman Filter
EST	Enrolment over Secure Transport
ETL	Extract, Transform, and Load
EU	EUropean
FFT	Fast Fourier Transform
FGP	Field Gateway Proxy
FIR	Finite Impulse Response
FMI	Functional Mock-up Interface
FP	Floating Point
FPGA	Field Programmable Gate Array
FPS	Frames Per Second
FRER	Frame Replication and Elimination for Reliability
GA	Grant Agreement
GP	Gaussian Process

GPU	Graphical Processing Unit
GUI	Graphical User Interface
HIL	Hardware In the Loop
HMI	Human Machine Interface
HPC	High-Performance Computing
HRI	Human-Robot Interaction
HW	Hardware
IAM	Identity and Access Management
ID	Identification Data
IDE	Integrated Development Environment
IDS	Intrusion Detection Systems
IGBT	Insulated Gate Bipolar Transistor
IIoT	Industrial Internet of Things
IO	Input Output
IoT	Internet of Things
IP	Internet Protocol
ISO	International Organization for Standardization
IT	Information Technology
KPI	Key Performance Indicator
LAN	Local Area Network
LMS	Learning Management Systems
LPF	Low Pass Filter
LSTM	Long-Short Term Memory
MFCC	Mel-Frequency Cepstral Coefficient
MIL	Model In the Loop
MIMO	Multiple-Input-Multiple-Output
ML	Machine Learning
MLE	Maximum Likelihood Estimation
MPU	Micro Processor Unit

MQTT	Message Queuing Telemetry Transport
MSE	Mean Squared Error
MSO	Mixed Signal Oscilloscope
NN	Neural Network
NVMe	Non-Volatile Memory Express
ODE	Ordinary Differential Equation
P/D/UCs	Pilots/Use cases/Demonstrators
PC	Personal Computer
PD	Proportional-Derivative
PHP	Hypertext Pre-processor
PI	Proportional Integral / Performance Index
PID	Proportional Integral Derivative
PIL	Process In the Loop
PLC	Programmable Logic Controller
QA	Quality Assured
QoS	Quality of Service
QR	Quick Response
RNN	Recurrent Neural Networks
ROM	Reduced Order Model
ROS	Robot Operating System
RPP	Regulated Pure Pursuit
RT	Real-Time
RUL	Remaining Useful Life
SIEM	Security Information and Event Management
SIL	Software In the Loop
SISO	Single-Input-Single-Output
SLURM	Simple Linux Utility for Resource Management
SoC	System on Chip
SQL	Structured Query Language

SNMP	Simple Network Management Protocol
SSD	Solid State Drive / Single Shot Detector
SSO	Single Sign-On
ST	Scientific and Technological
STFT	Short-Time Fourier Transform
SW	Software
TCP	Transmission Control Protocol
TLS	Transport Layer Security
TRL	Technology Readiness Level
TSN	Time-Sensitive Networking
UDP	User Datagram Protocol
UI	User Interface
UKF	Unscented Kalman Filter
USB	Universal Synchronous Bus
USP	Unique Selling Point/Proposition
VC	Virtual Commissioning
VHDL	Very High-Speed Integrated Circuit Hardware Description Language
VM	Virtual Machine
VR	Virtual Reality
VRE	Vector Reconstruction Error
WP	Work Package
xIL	Something In the Loop

Executive Summary

This document contributes to two main pillars of the IMOCO 4.E project. AI principles and Digital Twins. It describes components development for IMOCO 4.E reference architecture defined in WP2 and prepares the ground floor for their modular integration in WP6 and demonstration on Pilots / Use cases / Demonstrators in WP7.

Chapter 2 supports data management needs. It provides a comprehensive solution for collecting, pre-processing, persistently storing, and distributing data sets in the context of industrial environments, all in a trustful and secure manner. Proposed solutions are not developed from scratch, but they build on existing tools and frameworks, which are combined in a way to support data management in the green fields as well as in brownfield applications.

Chapters 3 aims to enable condition monitoring and predictive maintenance at the instrumentation layer. It mainly describes software results for the computation of condition indicators, which enable the estimation of the remaining useful lifetime of the components. Even on the lowest layer, it gives some ideas and examples of how to use AI methods for the predictive maintenance of selected systems.

Chapter 4 comes with algorithms for fast commissioning of the systems. This process is supported by the proper user interfaces developed, especially with the involvement of digital twins. As a result, the commissioning can be performed in a shorter time with less experienced personnel.

Chapter 5 gives insight into the development of complex models of multi-axis systems that can be used within the digital twin. Such models are flexible, fast, general, adaptable, and with a high level of abstraction. They are able to interact with data coming from the real counterpart.

Chapter 6 describes tools that have been developed for building digital twins. Virtual reality and augmented reality concepts are studied and employed in the applications as a bridge between the virtual world of the digital twin and a real user, gaining the advantages of the connection with this virtual world. Several use cases of digital twin development and applications are described in detail.

Chapter 7 deals with the development of robust and AI-based methods for condition monitoring and predictive maintenance. Compared with the solution in Chapter 3, this chapter provides methods for higher layers in IMOCO4.E architecture. Developed ANN and ML approaches are implemented and demonstrated in real applications. Chapter 8 provides the conclusion of the realized work and the presented document.

Keywords

Digital twin, communication interface, secure communication, data processing, data storage, data visualization, secure communication, cyber security, time sensitive networking, condition monitoring, predictive maintenance, smart sensor, wireless sensor, self-commissioning, system identification, system diagnostic, parameter estimation, xIL simulation, models for digital twins, deflection modelling, augmented reality, virtual reality, artificial intelligence, neural network, deep neural network, convolution neural network, anomaly detection, remaining useful life, autoencoders.

1. Introduction

This document describes the achievements in work package WP5 – Digital Twins and their interaction with the cloud in the IMOCO 4.E project. Figure 1 shows areas where WP5 contributes to the IMOCO structure. There are three building blocks addressed in WP5 and in this deliverable. These are:

- BB6 – Algorithms for condition monitoring, predictive maintenance, and self-commissioning of industrial motion control systems (Tasks 5.3, 5.4 are contributing to Layer 1 and 5.7 to remaining three layers)
- BB8 – AI-based components (Tasks 5.3, 5.4 and 5.7)
- BB9 – Cyber-security tools and trustworthy data management (Task 5.2)

Besides these building blocks, components for digital twins are addressed throughout the WP5. They are treated mainly in Task 5.5 and 5.6.

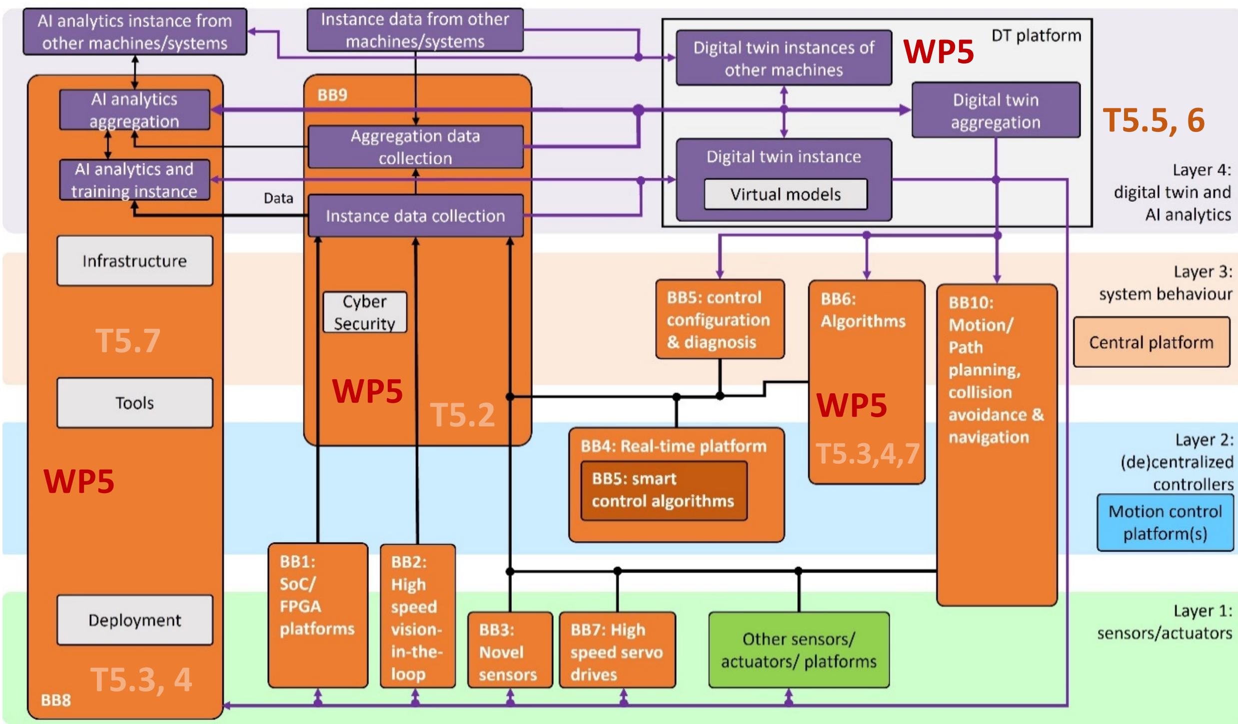


Figure 1: Addressed BBs in WP5 and their location in IMOCO layers structure

BB9 is about data management and secure communication interfaces (Task 5.2). Proper data treatment is demanded due to expanding sensory systems and a huge amount of data generated by digital twins.

The main goal of WP5 is to spread the digital twin concept into a wider range of industrial applications. It deals with the utilization of twins for:

- design,
- assembly,
- training,
- commissioning,
- collision detection,
- maintenance.

The results and achievements from the following tasks are described in this deliverable:

- Task 5.2 Trustworthy and Secure Dataset management, storage, and processing tools
- Task 5.3 AI methods for monitoring and predictive maintenance at instrumentation level
- Task 5.4 Automatic commissioning of motion control systems
- Task 5.5 Modelling and simulation of complex multi-axis systems, complex estimators
- Task 5.6 Augmented and virtual reality through digital twins
- Task 5.7 AI methods for monitoring and predictive maintenance at higher IMOCO4.E layers

1.1 Related scientific and technological development objectives

There are two scientific and technological development objectives which relate to WP5. They were taken from GA and are shown in Figure 2 to make referencing from the following sections in the document easier. ST2 objective relates to WP5 only indirectly through the connection with WP3. The hardware of intelligent and low-power sensors developed in WP3 are also co-developed in WP5 from the software side. This development was realized in Task 5.3, as it covers condition monitoring and predictive maintenance on Layer 1.

code	*Objective description / means of verification (success criteria)*	*WP / CH*
ST1	To develop advanced *model-based* and *knowledge-based methods* for building **digital twins** for design, optimization, customization, virtual commissioning and predictive maintenance of machines and robots, using existing and novel data sets *Means of verification:* digital twins for pilot applications available for MIL, SIL, PIL, HIL stages with functionalities and component KPIs defined in Section 1.3. Toolchains integrated into the *"Digital Twin as a Service"* concept.	4, 5 / 5
ST4	Ensure secure interoperability with State-of-the-Art cloud platform, i.e System Behaviour Layer – Layer 3 and develop specific condition monitoring building blocks providing relevant data for machine digital twins and system behaviour layer, further used either for machine predictive maintenance or re-design, virtual design and optimization; contribute to EU Open Datasets. *Means of verification:* HW and SW blocks, in particular ✓ Robust and AI-based condition monitoring and predictive diagnostics (BB6) ✓ Control system self-commissioning unit (BB6) ✓ Virtual/augmented reality for real-time HMIs ✓ Cyber-security tools for distributed motions control systems (BB9) ✓ AI-based methods for machine predictive maintenance (BB8) (i) reached TRLs, KPIs and functionalities defined in Section 1.3 where also BBs sub-systems are defined. (ii) BBs from Layer 1 and Layer 2 fully connected to System behaviour Layer and digital twins using SotA standards.	5 / 1,5, 7

Figure 2: Related ST development objectives

1.2 Purpose of the Document

The document's purpose is to provide information on the development of digital twins, the development of algorithms for predictive maintenance of mechatronic systems, and the development of self-commissioning algorithms with the use of digital twinning technologies. The critical aspect being discussed is also how to realize data management in industrial environments.

1.3 Structure of the Document

The structure of this deliverable is different from the similar (concluding) deliverables in WP3 and WP4. Only some of the tasks in this work package are directly linked with some building block. Some tasks like T5.5 and T5.6 contribute to the main content of this work package, which is the notion of digital twins. Therefore, it was decided to present the results per individual tasks rather than according to developed building blocks.

Each task has a dedicated chapter in this document. It contains a general description of the achieved results. In case the task is related to one or more building blocks, this fact is treated in a dedicated subsection. Information about the targeted pilots, use cases, and demonstrators are provided for easier orientation, where the results achieved are integrated, tested, and validated. The fulfilment of requirements is treated carefully, as well as the link to development objectives and KPIs. Progress beyond the state of the art is assessed. The conclusion is provided at the end of the document.

1.4 Relation to other activities in the project

This deliverable provides the technologies, which are high-level components for constructing digital twins. They are built to fit in the system-level design proposed in WP2. This deliverable also uses the components developed in WP3 and smart control systems developed in WP4. This deliverable concludes the work in WP5. It started with the definition of requirements and specifications in (D5.1). They were slightly refined in a follow-up deliverable (D5.2). This part was done in Task 5.1. The remaining tasks in this work package started shortly after Task 5.1 in M5. At milestone 4, Tasks 5.2 to 5.7 provided deliverables about the initial design of the building blocks in M21. This deliverable describes the achieved results and developed components in the whole WP5. Due to the wide span of the work and number of results, this description is rather informative, giving a general overview of achievements with an emphasis on how components fit into the proposed IMOCO framework, how they can be combined, and how they address the KPIs of the project.

Developed components are subsequently transferred to WP6, where they are tested and validated. Besides this, they are also integrated in several use cases. Further integration of the components in pilots and demonstrators is realized in WP7. As with the other work packages, WP5 also contributes to WP8, mainly through dissemination activities, the preparation of training material, and finally, it will form the basis for the exploitation plans.

1.5 Intended readership

This deliverable is public. Therefore, it will be available to a broad spectrum of potential readers. During the preparation of this deliverable, we intended to address mainly professionals dealing with industrial control and with the design of production lines, having knowledge about the existing components, and trying to become familiar with new ones, which are specifically helpful for the involvement of digital twins in different phases of product development, production and during its lifetime. The topic of condition monitoring and predictive maintenance on different layers is treated there carefully, and it can be helpful to reduce downtimes and decrease the price and the complexity of service operations.

The deliverable can also be helpful for students. It contains new ideas, and it can serve as a source of inspiration for further development of advanced systems that utilize digital twins and incorporate AI methods.

2. Trustworthy and Secure Dataset management, storage, and processing tools

2.1 Overview of all realized solutions

Section 2 presents the work related to Task T5.2 'Trustworthy and Secure Dataset management, storage, and processing tools' that is also closely related to BB9 'Cyber-security tools and trustworthy data management'. Data management is a critical aspect in the implementation of Industry 4.0 practices. The rise of connected devices, sensors, and other IoT technologies lead to the generation of vast amounts of data across the manufacturing value chain. In the context of Industry 4.0, an effective strategy for the management of the resulting big data is imperative for unlocking the data's full potential and for driving improvements in production efficiency, quality, and safety. To achieve this, a holistic approach needs to be adopted that considers aspects of data collection, storage, distribution, processing and analysis. Furthermore, quality and integrity must be guaranteed, while managing access and security to protect sensitive information from unauthorised access.

To meet the IMOCO4.E data management needs, BB9 provides a comprehensive solution for collecting, pre-processing, persistently storing and distributing data sets in the context of industrial, IoT-enabled environments, ensuring trustful and secure data transmission, storage and accessibility. The internal design and features of BB9 render it highly suitable for supporting AI, i.e., ML and data analytics operations.

BB9 is essentially a group of components that are conceived to act as a central data bus in the IMOCO4.E framework, to which other IMOCO4.E BBs and their components can connect for exchanging data to fulfil their purposes. Therefore, BB9 assumes a pivotal role in supporting components and BBs that need to send data to other endpoints or receive data from other sources. To that end, BB9 offers a range of relevant functionalities, which are enabled by a set of corresponding components that compose BB9. Specifically, the core functionalities provided by BB9 components are the following:

1. BB9 allows the real-time data exchange of heterogeneous, non-binary semi-structured information among multiple IMOCO4.E BBs/components in parallel, through a robust and **Distributed Messaging System (DMS)** that relies on the publish/subscribe (pub/sub) data exchange scheme implemented by the Apache Kafka framework (Section 2.3).
2. BB9 offers aggregation and persistent storage of data originating from multiple sources (IMOCO4.E BBs/components) in a **Big Data Repository (BDR)**, while making the data accessible to other IMOCO4.E BBs/components through an efficient query and retrieval system based on the Elasticsearch technology. The BDR can be configured to automatically ingest the data flowing through the BB9 DMS and optionally perform ETL (extract-transform-load) operations on the data before storing it (Section 2.4).
3. BB9 can facilitate data collection from devices at the edge (BB4) that may implement diverse communication protocols (e.g., TCP, UDP, etc.), and relay the collected data to the BB9 DMS through the dedicated **Field Gateway Proxy (FGP)** component (Section 2.5).
4. BB9 provides a **Cyber-Security Module (CSM)** for comprehensive cyber-security protection by implementing end-to-end mechanisms for authenticating devices, microservices and users that connect to the BB9 distributed messaging system, which is complemented by an AI-based threat detection tool (Section 2.6).
5. BB9 provides a **Data Visualisation Toolkit** for visualising live data flowing through the DMS, stored data in the BDR and for monitoring the health and performance of the DMS (Section 2.7).

6. BB9 includes a HW/SW **Time-Sensitive Networking (TSN)** solution for providing bandwidth and latency guarantees in Ethernet-based network communications (Section 2.8).

Furthermore, in the context of T5.2 and BB9, the following case-specific data management implementations took place for the needs of specific P/D/UCs:

1. A large-scale Healthcare Robotics Data Management Pipeline was implemented for the needs of data collection from medical robotic manipulators that are used for image guided therapy in Pilot 4 (Section 2.9).
2. A secure cloud-based data exchange mechanism was established for the needs of xIL co-simulation in UC1 (Section 2.10).
3. A deployment of an anomaly detection AI model was implemented and coupled with the BB9 DMS in Pilot 3 for predictive maintenance (Section 2.11).

Figure 3 illustrates the internal reference architecture of BB9 that is composed of the 6 main components described above. The diagram displays the interactions between the components as well as the interactions with other software IMOCO4.E components that act as data producers, data consumers or both, all indicated as grey circles. Distinct host devices are represented by grey rectangles at the background. At the left side, devices operating on site at the edge layer are presented. At the right side, a cloud-based host infrastructure is presented.

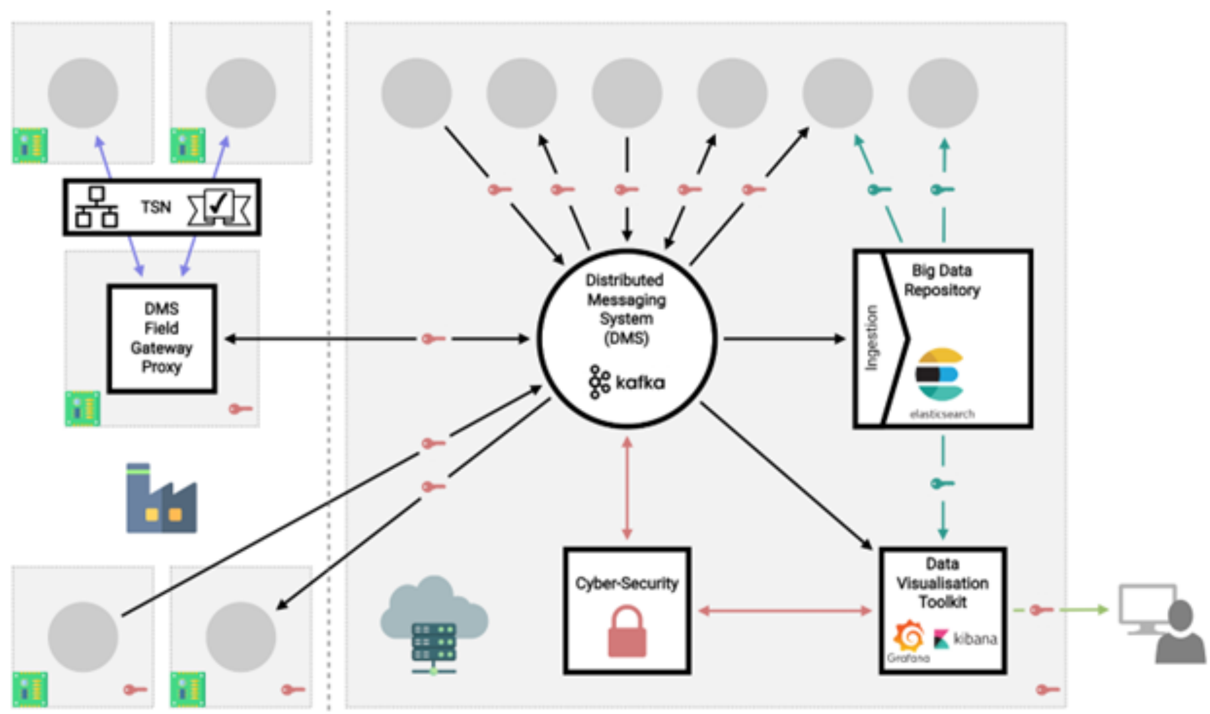


Figure 3: BB9 internal reference architecture

The BB9 internal architecture is indicative and only meant to be used as a reference. It does not preclude other structural variations in potential BB9 applications that include alternative deployment configurations in terms of host infrastructure and connections to third party components, such as the components of other IMOCO4.E BBs. Not all BB9 component need to be implemented in potential deployments and individual component application is possible, although the dependencies between BB9 components need to be considered when designing new implementations to establish the necessary functionalities.

BB9 is delivered as a fully scalable system with increased data reliability, safety and security features based on a microservice architecture with advanced replication, authorisation, and authentication features. BB9 software components are provided as dockerised containers, facilitating their deployment and configuration.

BB9 can be tailored to the exact needs of each P/D/UC where it will be implemented and be adapted to the available infrastructure and data exchange requirements of IMOCO4.E components from other BBs that participate in each P/D/UC. BB9 is highly scalable and can be configured to meet specific performance demands by taking full advantage of the available computational resources that are present in the host infrastructure. Regarding the BB9 TSN support, a TSN HW node needs to be installed at each device that needs to exchange data with other devices that are connected to the same network of an IMOCO4.E deployment.

2.2 Addressed ST objectives and KPIs

2.2.1 Project and WP5 Objective Status

Task T5.2 contributes to the achievement of the **project objective ST4** through the achievement of KPIs associated with BB9, which are elaborated in the following Table 1 and in sub-sections 2.2.1 – 2.2.3.

Table 1. BB9 KPI table

KPI #	KPIs /smart functionality	IMOCO4.E target (TRL 5)
KPI_BB9_1	Intelligent anomaly detection mechanisms capturing cyber-threats real-time and at-the-edge with increased accuracy	Deploy AI-based anomaly detection innovations at the edge
KPI_ BB9_2	Real-time edge-based event management and visualisations	Ensure real-time event management
KPI_BB9_3	Efficient vulnerability assessment in complex Industrial IoT systems and in communication channels.	Thorough vulnerability assessment in IIoT environments

T5.2 is also related to the **WP5 objective** 'Propose a general framework for big data management in industrial applications - bringing digital twins in the loop leads to a requirement to transfer big amounts of data with extremely diverse latency requirements between layers in real-time for their further processing for reasoning and decision making. The situation is even aggravated in case of predictive maintenance because these data must be efficiently stored.'. **T5.2 achieves this objective** through the provision of the BB9 components and implementations elaborated in the following sub-sections 2.2.2 - 2.2.4 and in sub-sections 2.3 – 2.11.

2.2.2 KPI_BB9_1 Status

The BB9 Cyber-Security Module (CSM), described in section 2.6, includes the implementation of a Deep Learning approach for anomaly detection in a cyber-security context (threats), which presents advantages over traditional Intrusion Detection Systems (IDSs) that face limitations in detecting entirely unprecedented threats. The approach is based on a Long-Short Term Memory (LSTM) Auto Encoder (AE) model that can analyse the network communications of a device and detect unanticipated patterns when trained with corresponding time-series datasets that include data points that are labelled as normal or as cyber-security threats. The model was successfully deployed and validated on an edge device (Jetson) in the context of Pilot 3, effectively achieving the KPI target that focuses on the transfer of anomaly detection mechanisms from the cloud to the edge.

2.2.3 KPI_BB9_2 (Real-time edge-based event management and visualisations) Status

The achievement of this KPI requires the implementation of an event management and visualization system that (a) has a distributed nature (not centralized), (b) it considers event management at the edge layer and (c) operates in real time. **BB9 implements a Distributed Messaging System (DMS) that is an inherently real-time and distributed event management system by design**. The DMS underlying technology (Apache Kafka) natively supports the deployment of message brokers in cluster mode on host infrastructure nodes that may belong in the same computing environment (e.g., HPC) or may be distributed across multiple geographic regions. The sizing of the cluster depends on the level of demands for data production/consumption throughput, the number of expected publisher/subscriber endpoints and fault tolerance requirements. The cluster can be deployed exclusively at the cloud, on-premises at the factory floor or combining both options, while it can also be dispersed across different geographic regions. In general, the positioning of brokers in a large-scale production environment should consider the proximity to the data sources and sinks, data security isolation and disaster resilience aspects. Besides the distributed nature of the DMS, BB9 also implements **the Field Gateway Proxy (FGP) component that introduces another dimension to the distributed nature of the BB9 event management system towards the edge**. More specifically, the FGP is meant to be deployed at the edge layer, near the data sources and edge-based inference components. Its role is to collect raw data and inference results from edge devices and components in cases where Kafka client implementation is not possible by the data producers (e.g., in case of legacy system in brown-field architectures). The FGP can implement other basic communication protocols (such as the TCP/UDP implemented for Pilot 3) to receive the data, transform them to another format if necessary and relay them to the DMS via a Kafka producer client. **The BB9 event management pipeline is built on top of real-time protocols and is designed to operate in real time**. Data points and events are either published directly from their source to the DMS via Kafka client implementations or through the mediation of the FGP that is also meant to receive data via real-time protocols, such as TCP/UDP. Once published to the DMS Kafka topics, events can be consumed almost immediately by intended AI components for further analysis, which in turn publish their inference results to dedicated DMS Kafka topics. These inference results, but also the original data points and **events can be visualized with minimal latency within the BB9 Data Visualization Toolkit (DVT)**. Experiments in Pilot 1 with a single cloud-based broker showed that the average latency introduced by the BB9 event management system is 160ms when the producer and consumer are also hosted on the same VM as the broker. In the context of the addressed predictive maintenance and Remaining Useful Life estimation a sub-second latency is considered real-time. However, if demanded by prospective use cases, even lower latencies can be attained through careful configuration of the DMS cluster and the procurement of sufficient computational infrastructure resources through the extensive scalability and customization features of Apache Kafka. For example, according to a benchmark test conducted by Confluent (Apache Kafka Performance), a median end-to-end latency of 3ms and a latency of 18ms at the 99th percentile can be achieved (measured at 200K messages/s with 1 KB message size) with suitable tuning of the cluster. Finally, in cases of on-premises deployments, **the BB9 Time Sensitive Networking (TSN) component can be employed for transporting traffic from multiple endpoints while providing differentiated Real-Time Quality of Service (RT-QoS)**. Protected traffic can be defined, and it will have guaranteed packet transport, in terms of bounded low latency, low packet delay variation and low packet loss. The TSN component also features an **advanced tool for monitoring latency** and has been validated in Pilot 1 and Pilot 2.

In conclusion, **KPI_BB9_2 is achieved** as **a real-time** and **distributed event management system** has been successfully applied with provisions for **edge-cloud communications**.

2.2.4 KPI_BB9_3 Status

The BB9 Distributed Messaging System (DMS) can provide logs of key health and performance metrics (e.g., cluster broker status, topic traffic, partition and leader numbers, replication rates, offsets) that need to be monitored by the administrator for conducting a **vulnerability assessment of the system in terms of data integrity and availability**. Through the introduction of the BB9 Data Visualisation Toolkit (DVT) and the generation of a Grafana dashboard that can visualize key health and performance metrics of the DMS, a more thorough vulnerability assessment is achieved. In addition, the BB9 DVT offers a more comprehensive **vulnerability assessment of industrial equipment on the factory shop floor** by visualizing data that refer to multiple machines and/or attributes simultaneously, organized into intuitive dashboards. These data can range from raw sensor readings to inference results that refer to predictive maintenance analysis and Remaining Useful Life (RUL) estimation. Finally, **another facet of enhanced vulnerability assessment provided by BB9 is in the domain of cyber-security**, where the introduction of an AI-based tool for threat detection as part of the BB9 Cyber-Security Module can offer a more complete mapping of possible threats across multiple devices at the edge, fog or cloud, which can be coupled with the BB9 DVT in the form of dashboards that reveal possible risks across the entire Industrial IoT environment.

In conclusion, **KPI_BB9_3 is achieved** as a thorough vulnerability assessment in IIoT environments has been established in terms of (a) data integrity and availability, (b) industrial equipment health and (c) cyber-security threats by collecting relevant key parameters and exposing them in appropriate visualisations.

2.3 Distributed Messaging System (DMS)

2.3.1 Tech Overview

A **Distributed Messaging System (DMS)** acts as the backbone of BB9. It is based on the pub/sub communication model and relies on the Apache Kafka technology. The DMS is registered as SW-040 in the IMOCO4.E software component catalogue.

The industry 4.0 paradigm, exemplified by IMOCO4.E, leverages sensing technologies and IoT to generate large volumes of data for monitoring industrial equipment. Artificial Intelligence needs to acquire this data so that it can be analysed for various purposes like predictive maintenance and process optimization. Effective communication of analysis results back to the equipment or other endpoints is essential, requiring robust data exchange mechanisms. The pub/sub communication model is chosen for its ability to handle a loosely coupled real-time communication between multiple agents simultaneously while ensuring data integrity and scalability. In this model, publishers send messages to a messaging system, which distributes them to interested subscribers based on predefined topics, enabling fault tolerance and scalability without direct connections between publishers and subscribers.

IMOCO4.E adopts the pub/sub messaging model using Apache Kafka technology (Apache Kafka website, 2024), (Dobbelaere & Sheykh, 2017) and (D'silva et al., 2017) for its data management needs. In this context, Producers publish events to a Kafka broker or cluster, while Consumers subscribe to the broker or cluster and receive these events independently. Events are organized and durably stored in topics that act like folders in a filesystem. Multiple producers can publish events to a single topic and multiple consumers can be subscribed to the same topic to receive the published events. Topics are partitioned across multiple brokers (cluster) for scalability, ensuring events with the same key are consistently ordered within partitions. Topics can be replicated for fault tolerance, with Kafka clusters spanning multiple data centres or cloud regions, providing redundancy and fault tolerance.

The DMS includes system administration tools for monitoring the Kafka cluster to ensure smooth operation (e.g., health and performance of brokers, disk space availability, etc.) and for configuration purposes (e.g., managing Kafka topics, consumer groups, partition replication, etc.). Through the DMS, BB9 can serve the data exchange needs of any IMOCO4.E component, which can act as a producer or consumer to the DMS, if it can transmit and receive data over the network. Kafka producers and consumers can be deployed via respective client implementations that are available for most programming languages, including C/C++, Python, Go (AKA Golang), Java, .NET, Clojure, Ruby, Node.js, Proxy (HTTP REST, etc.) and Perl. The deployment of such Kafka clients within IMOCO4.E components is simple and straightforward, while relevant documentation is abundantly available (Apache Kafka documentation, 2024) and (Apache Kafka clients, 2024).

2.3.2 Implementation aspects

The BB9 DMS was implemented in Pilot 1 and in Pilot 3.

An integration between the Pilot 1 data platform and BB9 has been built to provide the BB9 DMS with real-world equipment data from Tissector devices and to demonstrate the versatility of IMOCO4.E and the capacity to integrate with pre-existing industrial systems via BB9. The integration is realised in the form of a data bridge that performs basic message format conversion and sets up a secured connection to the DMS to publish messages on a dedicated Kafka topic. In the context of a predictive maintenance use cases, an AI component (SW-097) subscribes to this DMS topic to receive the corresponding data as input and publishes its inference results back to the DMS on another Kafka topic. The results are in turn consumed by (a) the Pilot 1 data bridge and propagated to the internal Tissector network, (b) by the BB9 BDR for permanent storage and (c) by the BB9 DVT for visualization purposes.

In Pilot 3, the DMS is deployed at a server on the fog layer, as the entire system is meant to operate on-site. The DMS receives sensor data and image data (converted to text) from the BB9 Field Gateway Proxy component (SW-009) at the edge and relays it to BB6 AI components at the fog (SW-044, SW-109 and SW-101) in real time so that they can perform monitoring, predictive maintenance, and quality control analysis tasks. The DMS receives inference results from these three AI components and from another that is positioned at the edge (SW-018). The inference results are consumed by the BB9 Data Visualisation Toolkit (DVT) for visualization purposes. At the same time, all data published at the DMS are also consumed by the BB9 Big Data Repository for permanent storage. Separate Kafka topics have been established for each type of input data (sensor, images) and for the inference results of each AI component.

In both cases, it was possible to monitor critical health metrics of the DMS through a Grafana dashboard that was set up via the BB9 DVT and a Prometheus implementation for storing the metrics.

2.3.3 Results

The BB9 DMS has managed to successfully handle all data management tasks in the tests conducted for both pilots, when deployed in the target production environments. More specifically, in Pilot 1 the DMS successfully implemented 2 Kafka topics with 2 producers (P1 data bridge, SW-097) and 3 consumers (P1 data bridge, SW-097, SW-041). In Pilot 3, the DMS successfully implemented 6 Kafka topics (sensor-blade-data, image-data, lstm-ae-ad-results, rnn-ad-results, cc-ad-results, dnn-results) with 5 producers (SW-009, SW-018, SW-044, SW-109, SW-101) and 4 consumers (SW-044, SW-109, SW-101, SW-041).

In terms of **Data Integrity**, no data loss has been observed in any of the data managed by the DMS during its operation in the context of the conducted tests in Pilot 1 and Pilot 3. More specifically, all data that were published to the DMS Kafka cluster were consistently made available to other IMOCO4.E components without any fault. In terms of **Availability**, the DMS services remained constantly discoverable and available during their operation in the context of the conducted tests in Pilot 1 and Pilot 3. In addition, all DMS interfaces remained **constantly accessible**. More specifically, the DMS was configured to constantly monitor and log any failed Kafka produce and fetch requests and in the context of the tests, there were not any such failures registered in the logs. The results from the implementation of the DMS in Pilot 1and Pilot 3 can be summarized as such: (a) Availability: 100 %, (b) Failed requests: 0, (c) Data loss rate: 0. Finally, the **latency** introduced by the DMS was measured in a single, cloud-based broker configuration, involving a single producer and single consumer hosted on the same VM as the broker. In this setup, the time from message generation at the producer side until message reception at the consumer side was 160 milliseconds on average.

2.3.4 IMOCO4.E requirements

SW-040 meets the following requirements: R054-D5.1-B9, R060-D5.1-B9, R068-D5.1-B9, R066-D5.1-B9, R015-D5.1-L4-sw, R057-D5.1-B9, R058-D5.1-B9, R060-D5.1-B9, R048-D5.1-B9, R050-D5.1-B9, R052-D5.1-B9-sw, R056-D5.1-B9, R066-D5.1-B9.

2.3.5 Capabilities & Limitations (including USP, strengths & weaknesses)

The pub/sub model has been gaining increasing attention in the last decades, following the trend of large-scale distributed systems, in which it is highly applicable. It has currently become the dominant model for data exchange in IoT environments (Palmese et al., 2021), (Kang et al., 2021) and (Çorak et al., 2018), real-time big data analytics (Villalba & Carrera, 2019) and event-driven system architectures (Phuttharak & Loke, 2023). For the needs of IMOCO4.E data management, the pub/sub messaging model is adopted and delivered by the BB9 DMS component, which relies on the Apache Kafka pub/sub messaging technology.

Apache Kafka (Apache Kafka website, 2024), (Dobbelaere & Sheykh, 2017) and (D'silva et al., 2017) is currently considered an industry gold standard for distributed messaging systems based on the pub/sub model, providing a wide range of advantageous features, such as powerful event streaming, fault-tolerance and reliability, high scalability, open-source status, multi-tenancy, real-time processing, and high suitability for big data management. Kafka is also supported by a large community and associated ecosystem. Apache Kafka is used for "building real-time streaming data pipelines that reliably get data between systems or applications" supporting multiple sources that can even be in a distributed system. As a message broker, Kafka offers higher fault tolerance in comparison to traditional message brokers, and it can support significantly higher throughput. It is thus capable of handling great amounts of data and deliver it in real time to the components that require it.

Based on the features of its underlying Kafka technology, the DMS can offer the following advantages to prospective IMOCO4.E framework instantiations:

- **Real-time big data management**: The DMS facilitates real-time management of large volumes of data across multiple endpoints, ensuring minimal latency and immediate consumption of messages upon publication. This capability is essential for IMOCO4.E time-critical applications, where real-time analytics play a pivotal role in enhancing operational efficiencies, reducing downtimes, and optimizing system performance.

- **Interoperability**: The loose coupling between producers and consumers, the abstracted and flexible communication protocol, the versatility of the supported message data structure and the API support for multiple programming languages that are provided by the DMS may facilitate source/sink integrations between existing systems and customised applications that fit specific requirements. For the same reasons, the DMS can contribute to the future-proofness of installations as it can support an easy integration of future data sources and data processors.
- **Scalability**: The DMS Kafka cluster can be extended to span over increasing number of infrastructure nodes when required to accommodate increasing data volumes and growing producer and consumer populations without disrupting system performance. In addition, the loose coupling between producers and consumers reduces the complexity in managing larger populations of these entities. Scalability is in high demand by contemporary multi-site industries that may have hundreds of data sources that need to deliver the produced data to equivalent processing units.
- **Adaptability**: The DMS publishers and subscribers have loose relationships, making changes such as adding or subtracting subscribers/publishers easier without impacting other components in the system. The DMS pub/sub model offers unparalleled flexibility by permitting various forms of message delivery - broadcast or selective - depending on system needs. Subscribers have the option of receiving only specific messages that interest them, thus effectively filtering out irrelevant data.
- **Reliability**: The DMS offers guaranteed message delivery with ordered data persistence regardless of failures or network disruptions, which is considered essential in maintaining data consistency and integrity in industrial systems. Furthermore, the native fault tolerance resulting from the Kafka replication features helps enable smooth recovery from system malfunctions and local data losses. Therefore, the DMS ensures overall data management reliability, as it prevents system or communication failures and partial data losses from affecting the overall data exchange and operations of industrial systems.

In the context of IMOCO4.E, it is particularly critical for proposed solutions to be able to adapt to existing brownfield architectures and legacy systems. The level of customisation offered by the Apache Kafka technology stack, its open-source and self-managed nature render it much more appropriate in comparison to commercial, closed-source and managed alternative solutions (e.g., Azure Event Hubs (Microsoft Azure Event Hubs, 2024), Amazon Kinesis Data Streams (Amazon Kinesis Data Streams, 2024) and Google Cloud Pub/Sub (Google Cloud Pub/Sub, 2024)). Furthermore, in the context of commercialising the IMOCO4.E framework, the use of open-source underlying technologies is more favourable in terms of licensing options and overall versatility, when compared to commercial, managed solutions.

Compared to alternative solutions (e.g., Redpanda (Redpanda, 2024), Apache Pulsar (Apache Pulsar, 2024), (NATS, 2024), MQTT (MQTT, 2024)), Kafka is also associated with a richer ecosystem, it is more integrated with other platforms and offers producer and consumer client implementations for most programming languages, leading to higher levels of compatibility and interoperability. In addition, Apache Kafka has a larger active community that can more readily provide support and ensure its long-term maintenance and sustainability. Furthermore, a more extensive range of supportive monitoring and management tools is available for Kafka compared to other solutions.

Nevertheless, due to its fairly high demands in terms of computational infrastructure, Apache Kafka deployment is better suited to multi-node cloud infrastructure or to local servers with adequate computational resource specifications, when an on-premises deployment is required. In cases where a messaging system needs to be hosted on more lightweight infrastructure with fewer computational resources for handling local communications at the edge, the MQTT and particularly the Mosquitto (Eclipse Mosquitto, 2024) implementation can be considered as a more appropriate option.

2.3.6 Customizations & Adaptations (including possible modifications and extensions)

In the context of IMOCO4.E, the DMS may be deployed at the cloud or in a fog environment at the factory premises, as exemplified in the Pilot 1 and Pilot 3 implementations respectively. In addition, the DMS can be delivered as a single Kafka broker node or in a cluster mode composed of multiple brokers. The sizing of the cluster depends on the level of demands for data production/consumption throughput, the number of expected publisher/subscriber endpoints and fault tolerance requirements. The dispersion of the host infrastructure across multiple locations is another factor to consider in the context of disaster resilience and network traffic optimization. The following indicative minimum infrastructure requirements can be considered for hosting a single-node or small cluster: 4-core CPU, 8 GB RAM, 500GB SSD or NVMe HDD, Linux-based dockerised environment. Depending on the load and use case, certain key parameters of the DMS cluster need to be configured accordingly, such as the active topics, partitioning information, replication factors and retention policies. The cluster needs to be monitored during operation to determine if any re-adjustments are needed to such key parameters. Finally, The DMS can be coupled with other BB9 components to deliver added value, i.e., with the BB9 Big Data Repository for permanent storage of data handled by the DMS, with the Field Gateway Proxy for data exchange with edge legacy components, with the Cyber-Security module for access authentication and communication encryption and with the Data Visualisation Toolkit for monitoring DMS health and performance metrics.

2.3.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

Docker technology and Jenkins pipelines were used for deployment of the DMS. Swagger was used for documentation of the DMS administration REST API.

2.4 Big Data Repository (BDR)

2.4.1 Tech Overview

Apart from distributing data in real time between IMOCO4.E components, in many cases, it is also necessary to store the vast amount of data that can be produced in an IMOCO4.E deployment for subsequent retrieval and usage. BB9 provides this capability by offering a **Big Data Repository / BDR** (SW-041) that can serve purposes such as:

- Aggregating data from sensors or other IMOCO4.E components to form datasets that can be used for AI training.
- Maintaining log files of IMOCO4.E component activities for debugging them and resolving global issues.
- Maintaining a record of health and performance data of industrial equipment for offline monitoring.
- Performing long-term analytics for predictive maintenance, quality control, efficiency optimisation, etc. By analysing the volume of data that has been accumulated over a time period, trends, patterns and other useful insights can be extracted.
- Using the data for testing and validating new system features.
- Using the data for creating Digital Twin models.

The **Elasticsearch** technology (Elastic website, 2024) has been implemented for the delivery of persistent storage and historical data retrieval capabilities by the BB9 Big Data Repository. Elasticsearch is a distributed RESTful search engine built for the cloud. It relies on a database supporting NoSQL queries that stores data as JSON documents. Furthermore, the BB9 BDR includes a **data ingestion mechanism**. This mechanism is responsible for automatically consuming data that is published to the BB9 DMS and transferring it to the BDR for permanent storage. The same mechanism can be configured to perform data pre-processing before committing the data for permanent storage. The pre-processing may include operations such as data cleaning, normalisation, scaling, composite transformations, imputation, fusion, etc. Furthermore, the data ingestion mechanism can optionally be configured to publish any transformed data resulting from pre-processing back to the DMS in case these data need to be immediately consumed by other IMOCO4.E components.

The BB9 BDR can be deployed on multiple infrastructure nodes in a cluster mode with replicas for increased resilience, reliability, and availability. Administration tools allow the monitoring and management of Elasticsearch indices and shards. The BDR also exposes the Elasticsearch API to other IMOCO4.E components to enable them to efficiently query historical data that are persistently stored. Elasticsearch clients are readily available for most programming languages, including Java, Javascript, Ruby, Go, .NET, PHP, Perl and Python (ElasticSearch Clients, 2024).

2.4.2 Implementation aspects

The BB9 BDR was implemented in Pilot 1 and in Pilot 3. In the first case, it was deployed on a cloud host environment and in the second case on a fog server at the premises of UNIMORE. In both cases, the BDR was connected to the respective DMS instance and implemented ingestion pipelines for transferring and storing the data that were published on DMS topics to corresponding BDR Elasticsearch indices. To that end, there were activities related to the specification and documentation of the data schema of the payload of messages delivered to the DMS. An indicative example can be seen in Figure 4. According to this information, the BDR Elasticsearch index field mappings were configured. The stored data were queried and visualized by both Kibana and Grafana tools of the BB9 Data Visualisation Toolkit.

LSTM AE AD inference result payload schema

Global description

Data schema describing the payload of messages produced and sent by IMOCO4.E SW-044 (LSTM AE anomaly detector for predictive maintenance), developed by GNT and ITML. Each message corresponds to a single inference result

List of properties

- `id`*(string)*: a unique text-based id
- `type`*(string)*: the type of inference result
- `anomalyDetected`*(boolean)*: indicates whether an anomaly has been detected
- `startTime`*(Date)*: The timestamp of the start of the sample sequence that was used for calculating the inference result. Following the ISO 8601 UTC format: YYYY-MM-DDThh:mm:ss.sssZ
- `endTime`*(Date)*: The timestamp of the end of the sample sequence that was used for calculating the inference result. Following the ISO 8601 UTC format: YYYY-MM-DDThh:mm:ss.sssZ
- `timestamp`*(Date)*: The timestamp of the reference sample (mid-sequence) that was used for calculating the inference result. Following the ISO 8601 UTC format: YYYY-MM-DDThh:mm:ss.sssZ
- `dateDetected`*(Date)*: The timestamp of the inference result production by the ML component. Following the ISO 8601 UTC format: YYYY-MM-DDThh:mm:ss.sssZ
- `deviceId`*(string)*: a unique id of the device to which the inference result corresponds
- `data`*(JSON object)*: encapsulates the fields calculated form the original sensor readings for the given time window (optional)
- `MLModelId`*(string)*: a unique id of the ML model used to produce the inference result - versioning information is to be included in the id.
- `inferenceCalcTime`*(long)*: the computation time for calculating the prediction in milliseconds
- `reconstructionError`*(double)*: the reconstruction error produced by the model.
- `threshold`*(double)*: a numerical threshold used to classify a data point as anomaly (optional)
- `anomalyScore`*(double)*: the anomaly score calculated reconstruction error produced by the model.
- `healthLevel`*(double)*: the device health level (0-100) estimated from the anomaly score

Example payload


```
{
  "id": "ADE_2023-06-09T10:07:55.466Z",
  "type": "Anomaly Detection Event",
  "anomalyDetected": "true",
  "startTime": "2023-06-09T10:07:53.614Z",
  "endTime": "2023-06-09T10:07:57.318Z",
  "timestamp": "2023-06-09T10:07:55.466Z",
  "dateDetected": "2023-06-09T10:07:58.462Z",
  "deviceId": "P3_BladeSensor_004",
  "data":
  {
    "pCut::Motor_Torque": "-0.10234699",
    "pCut::CTRL_Position_controller::Lag_error": "-0.02593612",
    "pCut::CTRL_Position_controller::Actual_position": "669263470",
    "pCut::CTRL_Position_controller::Actual_speed": "803.37524414",
    "pSvolFilm::CTRL_Position_controller::Actual_position": "14927670",
    "pSvolFilm::CTRL_Position_controller::Actual_speed": "3077.97021484",
    "pSvolFilm::CTRL_Position_controller::Lag_error": "0.79017323",
    "pSpintor::VAX_speed": "1799.99975585"
  },
  "MLModelId": "LSTM_AE_AD-v0.0.17",
  "inferenceCalcTime": "142",
  "reconstructionError": "4.035",
  "threshold": "0.71",
  "anomalyScore": "106.84",
  "healthLevel": "87.63",
}
```


Figure 4: Documentation of data schema specification for one type of inference results stored by the BB9 BDR.

2.4.3 Results

The BB9 BDR has managed to successfully handle all data management tasks in the tests conducted for both pilots, when deployed in the target production environments. More specifically, the DMS successfully consumed data from 2 DMS topics in Pilot 1 and from 6 DMS topics in Pilot 3, implemented equivalent ingestion pipelines and stored the data in corresponding ElasticSearch indices. The stored data were successfully accessed by both Kibana and Grafana instances of the BB9 DVT in the 2 pilots. No data loss and no data availability limitations (e.g., due to corruption, service failure, etc) were observed during the tests conducted for Pilot 1 and Pilot 3.

2.4.4 IMOCO4.E requirements

SW-041 meets the following requirements: R049-D5.1-B9, R060-D5.1-B9, R066-D5.1-B9, R015-D5.1-L4-sw, R057-D5.1-B9, R058-D5.1-B9, R048-D5.1-B9, R050-D5.1-B9, R052-D5.1-B9-sw, R056-D5.1-B9, R114-D5.1-P3-11

2.4.5 Capabilities & Limitations (including USP, strengths & weaknesses)

Key aspects to consider when designing a big data repository solution for IMOCO4.E include scalability, reliability and interoperability and the Elastic technology stack (Elastic website, 2024) has been found to be most suitable to serve such needs. Its advantages include high scalability and interoperability, fast performance, a versatile and powerful search engine, API-driven and schema-free design, multi-tenancy, open-source status and orientation towards documents. Elasticsearch has matured over recent decades to become an industry standard for a wide range of applications, especially for data analytics purposes. Elasticsearch stores data in documents that are organised into indices, which in turn can be separated into shards and stored in a distributed system with replicas if needed. In contrast to traditional databases, the stored documents can have different formats. In addition, due to its powerful search engine, Elasticsearch can accommodate efficient execution of complex queries for retrieving specific data subsets that are required for data analytics purposes.

Several alternative big data repository solutions are available. For example, **OpenSearch** (OpenSearch, 2024) is an open-source platform, originating from a fork of Elasticsearch but with a smaller community, less comprehensive documentation and fewer integration options. **InfluxDB** (InfluxDB, 2024) focuses in time-series data storage for IoT devices and monitoring systems, while Elasticsearch offers versatility, supporting various data types, full-text searches and horizontal scalability, while it is associated with a larger community and ecosystem. **Solr** (Apache Solr, 2024), provides indexing, replication, querying and significant customisation options, but can be challenging to configure and has a more verbose query language. **Sphinx Search** (Sphinx Search, 2024) offers advanced querying and database integration but is complex to configure and has a limited full-text search features and ecosystem.

As in the case of the DMS, commercial closed-source and managed solutions (e.g., **Algolia** (Algolia, 2024), **Amazon CloudSearch** (Amazon CloudSearch, 2024), **Azure Cognitive Search** (Azure Cognitive Search, 2024)) are not promoted for adoption in IMOCO4.E as they have limited customization and legacy system integration potential and they lead to increased costs, vendor lock-in and obstacles in commercial exploitation. Overall, Elasticsearch offers a powerful and customizable search and analytics platform with distributed, real-time, and full-text search capabilities. Its scalability, flexibility, rich query DSL, ecosystem integrations, and active community make it a preferred choice for building scalable, reliable, and performant search and analytics applications in various domains and industries.

2.4.6 Customizations & Adaptations (including possible modifications and extensions)

The BDR may be deployed at the cloud or in a fog environment at the factory premises, as exemplified in the Pilot 1 and Pilot 3 implementations respectively. In addition, the BDR can scale horizontally according to each use case requirements and available infrastructure, i.e., Elasticsearch can be deployed in cluster mode spanning across multiple infrastructure nodes and the ingestion pipeline service can be replicated. The BDR cluster can be sized according to expected volume of incoming data and query requirements. Furthermore, the BDR ingestion pipelines, Elasticsearch indices and certain key parameters such as index shard size and allocation need to be configured according to use case requirements, expected load and available infrastructure for optimal performance. The BDR can be coupled with other BB9 components to deliver added value, i.e., with the Distributed Messaging System for ingesting and permanently storing exchanged data and with the Data Visualisation Toolkit for providing access to the data that can be visualised.

2.4.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

Docker technology and Jenkins pipelines were used for deployment of the BDR.

2.5 Field Gateway Proxy (FGP)

2.5.1 Tech Overview

The **Field Gateway Proxy (FGP)** component (SW-009) was developed in the context of Pilot 3 but can also be used in other IMOCO4.E deployments. It is a middleware abstraction layer that allows components from edge platforms to communicate with BB9 DMS that resides at a higher infrastructure layer (i.e., fog, cloud). Its necessity comes from the inability of some IMOCO4.E components to adopt the Kafka pub/sub model and to directly communicate with the central Kafka broker of the DMS. Such cases can be encountered where the components use different coding languages, technologies, communication protocols and APIs to interact with BB9 DMS.

Furthermore, the FGP brings advantages to direct communication like the flexibility to change the message transportation mechanism. It also makes the maintenance easier as the different mechanisms do not have to be modified when Kafka is updated. Safety is also increased as the components are not impacted by the performance of Kafka with asynchronous events' transmission. Finally, the deployment is made easier alleviating the components' providers from working with a possibly unknown new framework.

2.5.2 Implementation aspects

The main purpose of the FGP in Pilot 3 is to allow edge components from Layer 2 to exchange data with Layer 4 using the BB9 DMS. It is why the Field Gateway Proxy has been implemented on the SoC-e platform, which is one of the edge platforms of Pilot 3. In the context of Pilot 3, the sensor, camera, and actuator components are communicating with the BB9 DMS in the fog through the Field Gateway proxy. The FGP is executed on the ARM core of the SoC-e platform.

2.5.3 Results

The FGP successfully allows communication between sensor, camera and actuator components and the central Kafka broker in the context of Pilot 3.

2.5.4 IMOCO4.E requirements

The FGP contributes to R005-D5.1-L2-sw, R013-D5.1-L2-sw, R016-D5.1-L2-sw, R047-D5.1-B9, R048-D5.1-B9 and R050-D5.1-B9 by creating a proxy between the Kafka broker and the algorithms of the partners. Due to its ability to make maintenance easier, it also contributes to R066-D5.1-B9.

2.5.5 Capabilities & Limitations (including USP, strengths & weaknesses)

An IoT edge device can be used as gateway to allow connection to a network or IoT Hub (Altimore et al., 2023). There are two types of gateways: protocol translation and identity translation. Using the protocol translation, only the gateway device has an identity in the network or hub, and it translates messages from downstream devices into a supported protocol, and then it sends the messages on behalf of the downstream devices. In the case of the identity translation protocol, the hub and network can identify all the downstream devices behind the gateway. According to (Altimore et al., 2023) such a component is useful when devices cannot have access to the network of IoT Hub. Also, having a device that acts as gateway can have non negligible advantages like device isolation, connection multiplexing, analytics at the edge, offline support, and traffic smoothing. The Field Gateway Proxy is similar to the technologies of the state of the art and has been developed to solve issues when components cannot adopt the Kafka pub/sub model.

2.5.6 Customizations & Adaptations (including possible modifications and extensions)

The Field Gateway Proxy facilitates maintenance and allows the DMS to be modified without having to readapt the partners' algorithms to allow communication with it.

2.5.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

This component has been developed in C using Librdkafka which is a C library implementation of the Apache Kafka protocol (Librdkafka, 2024).

2.6 Cyber-Security Module (CSM)

2.6.1 Tech Overview

BB9 provides a **Cyber-Security Module (CSM)** that relies on a comprehensive **authentication and authorisation mechanism** for safeguarding the BB9 component integrity and managed data**.** This mechanism is dual, securing both the hardware layer (authenticating host devices, implemented by SW-043) and the software layer (authenticating services and users). An overview of how both mechanisms interact in BB9 is given in Figure 5.

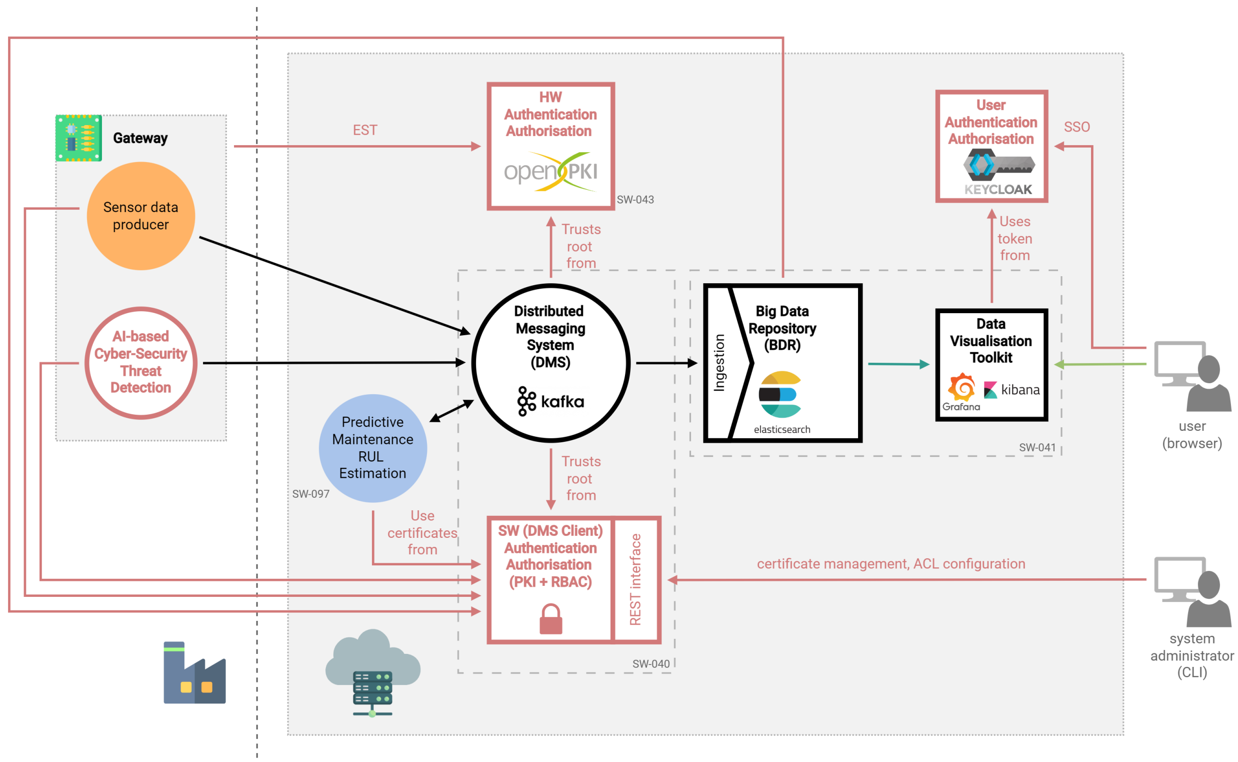


Figure 5: Operation of CSM mechanisms (indicated in red colour), including interaction of SW-043 and microservice certificate service with other BB9 components.

In this picture, the CSM mechanisms, their operation and interaction with other BB9 and other components are indicated in red colour. A field gateway HW device obtains an operational X.509 certificate from SW-043 and can use this certificate for any communication to BB9 DMS (SW-040). Similarly, certificates obtained from the microservice X.509 certificate service secure the TLS-based encrypted communication between IMOCO4.E services implementing Kafka clients and the BB9 DMS. Access to a Grafana dashboard of the BB9 Data Visualisation Toolkit via a browser is only allowed to users authenticated by a Keycloak Single Sign-On (SSO) service that is also part of the CSM (see Figure 6).

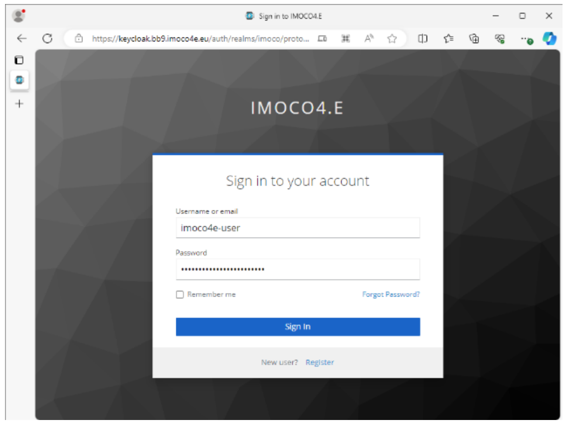


Figure 6: Keycloak SSO deployed and configured for Pilot 1 for logging into the Grafana tool of the BB9 DVT

The aforementioned authentication and authorisation mechanism is complemented by an AI-based cyber-security threat detection tool that can be applied in cases where there are risks for data sources to be compromised, as a fail-safe in cases where the authentication and encryption mechanisms may fail or for unanticipated types of intrusion and cyber-attacks. The tool is based on a Long-Short Term Memory (LSTM) Auto-Encoder Deep Learning model that can monitor key system operations and/or network communications occurring at an infrastructure node and classify them as normal or anomalous. The model can be trained using corresponding time-series data that are labelled as normal or anomalous (e.g., associated with a cyber-security attack).

2.6.2 Implementation aspects

Because SW-043 is built as a stand-alone component, it can also be integrated with existing platforms, as was performed in Pilot 1 (see Figure 7). The (existing) Asset microservice of Pilot 1 implements the Asset verification interface required by SW-043, such that SW-043 can authorize emission of certificates to Pilot 1 equipment.

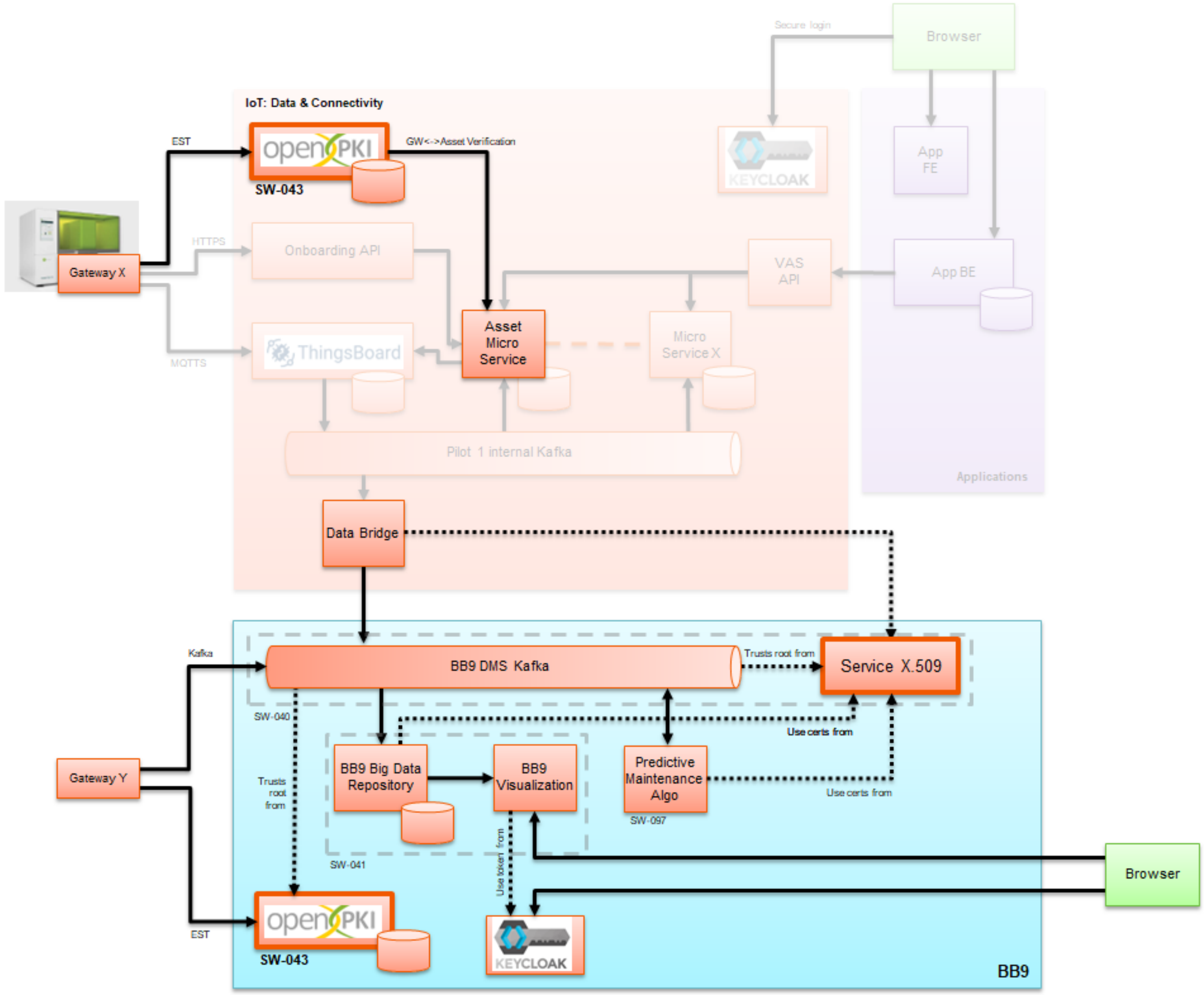


Figure 7: Use of the Cyber-Security Module in Pilot 1

These certificates are then used to secure communication with any other microservice of the Pilot 1 data platform. Additionally, the communication between the BB9 DMS and other components implementing Kafka clients (i.e., the brown-field Pilot 1 Data Bridge, the BB6 Predictive Maintenance AI component (SW-097) and the BB9 BDR) is secured by using certificates from the CSM service certificate issuing service. Users that access the Pilot 1 Grafana dashboard of the BB9 DVT are authenticated by the CMS Keycloak service. In the case of Pilot 3, the entire system is hosted on-premises, involving a series of edge devices and a central fog infrastructure. Under the premise that one of the edge devices collecting and processing data from sensors at the factory shop floor can be susceptible to an attack (e.g., via physical access), the cyber-security threat detection tool is deployed on the edge device (Nvidia Jetson) to monitor its network traffic. Its inference results (detected anomalies) are relayed to the BB9 DMS hosted at the fog node, where they can be visualized and generate alerts via the BB9 DVT, also hosted at the fog node. The DL threat detection model has been trained and validated using the UNSW-NB15 Dataset (UNSW-NB15, 2024).

2.6.3 Results

The setup of Pilot 1 was successfully demonstrated in the IMOCO4.E consortium meeting of November 2023 (see Figure 7). Additionally, specific tests have been performed to prove correct functioning of certificate issuance in both good and bad weather scenarios (such as certificate revocation, invalid GW / Asset ID pairs, etc.). These results are reported in (D6.6).

Regarding Pilot 3, the cyber-security threat detection tool has been tested on the target host environment (Nvidia Jetson Orin Nano 8GB). In order to evaluate its capability to operate at the edge, the performance of the same model has also been tested on a server environment (CPU: AMD Ryzen 7 7700X, RAM: 64 GB). The time to process a single data point by the model in inference mode (classify as normal or anomaly) is **0.622 ms on the server** and **2.407 ms on the Jetson**. In conclusion, the time to evaluate a single data point by the model is 4 times faster on the server than on the Jetson, which is to be expected granted the differences in the computational resources in each case. However, the time of 2.4 ms is perfectly acceptable and is considered to be 18 times faster than real time since the average rate of data point generation in the dataset is 1 every 43.7 ms. The model has also managed to reach a precision of 0.91 in its anomaly detection capacity for the specific configuration parameters that have been applied in the model training.

2.6.4 IMOCO4.E requirements

Requirements R050-D5.1-B9, R052-D5.1-B9-sw, R056-D5.1-B9, R057-D5.1-B9, R059-D5.1-B9, R061-D5.1-B9 (in-flight part), R062-D5.1-B9, R063-D5.1-B9, R064-D5.1-B9 and R066-D5.1-B9, R003-D5.1-LX-sw are applicable to the CSM, and have successfully been implemented, as demonstrated in the November 2023 consortium meeting and test results in (D6.6).

2.6.5 Capabilities & Limitations (including USP, strengths & weaknesses)

Correctly setting up **mutual TLS** infrastructure is complex. Because SW-043 can be deployed as part of BB9 and in a standalone fashion, it can enable both green- and brown-fields projects to benefit from this state-of-the-art device authentication. The standard EST protocol enables efficient implementation of certificate provisioning on any field gateway, by cleverly re-using libraries that such gateways need anyway (e.g., HTTPS). A downside of (mutual) TLS is the need for (offline) root certificates and their corresponding processes to securely generate, store and use their private keys. Additionally, the complexity of (device-specific) end-of-line programming during production for factory-provisioning of certificates may not be worth it for certain (low-cost) equipment, e.g., in case the authenticity of a device can sufficiently be 'proven' during onboarding using a different mechanism (such as scanning a QR code printed on the device itself). Regarding the service (Kafka client) TLS, fine-grained authorization is facilitated by a REST API that is exposed to the system administrator for easily managing Access Control Lists and read/write permissions to individual Kafka topics.

The adoption of Keycloak as an identity and access management (IAM) tool is based on its wide range of features for securing applications and services: It provides robust SSO capabilities, it is highly scalable and customizable, it is highly extensible and has a vibrant open-source community and has an open-source status. Compared to commercial, managed solutions such as **Microsoft Entra ID** (previously Azure Active Directory) (Microsoft Entra ID, 2024), **AWS Identity and Access Management** (AWS IAM, 2024), **Okta** (Okta, 2024), **Auth0** (Auth0, 2024) and **PingIdentity** (PingIdentity, 2024), Keycloak offers higher potential for customization and integration with pre-existing brownfield systems due to its open source status, while avoiding vendor lock-in, increased costs that may prohibit adoption, and licensing obstacles in commercial exploitation. Compared to other open source IAM tools, such as **WSO2 IdentityServer** (WSO2 Identity

Server, 2024), **Gluu** (Gluu, 2024) and **FreeIPA** (FreeIPA, 2024), Keycloak is considered to be a more established solution with a larger community and associated ecosystem behind it, more extensive customization features and integration capabilities.

The AI-based cyber-security threat detection tool complements the CSM by addressing the three main contributors to IoT system vulnerabilities: isolation assumption, increased connectivity and heterogeneity (Yaacoub et al., 2020). In the context of IMOCO4.E, cyber-security attack scenarios include the possibility of attackers gaining control of the internal network either physically (e.g., a USB or LAN port of an edge device) or wirelessly (e.g., exploit Wifi or Bluetooth interfaces) to carry out an attack, such as Denial of Service (DoS), malware injection, False Data Injection, privacy invasion, packet injection or an Adversarial Attack (e.g., Data Poisoning (Zhu et al., 2023), Byzantine Attacks (Shi et al., 2023), Model Extraction Attacks (Lee & Hur, 2017), (Lee et al., 2020), which are particularly relevant considering the extensive use of AI in IMOCO4.E. Traditional **Intrusion Detection Systems (IDSs)** become unreliable in complex systems, such as Industry 4.0 environments, especially when faced with novel attacks that are not deterministically defined. Conventional IDSs use rule-based approaches, like restricting communication protocols or node interactions, requiring frequent updates that are slow, cumbersome, and prone to errors. Signature-based malware detection relies on identifying specific malware signatures in a database, rendering it ineffective against zero-day exploits. Machine Learning (ML) or Deep Learning (DL) algorithms for intrusion detection are not bound by specific rules or malware signatures, allowing them to generalize better by learning complex data representations and modelling nonlinear relationships within the data. In essence, AI-based solutions seek patterns of malicious events rather than hardcoded instances. In complex domains with high-dimensional data, ML/DL algorithms offer scalable solutions, which are impractical for human analysts or conventional IDS systems. This underscores a shift in IDS development from signature-based to anomaly-based systems capable of detecting deviations from normal workflows.

2.6.6 Customizations & Adaptations (including possible modifications and extensions)

For Pilot 1, SW-043 was integrated into the existing Pilot 1 platform, and Pilot 1's Asset Service was extended to include the Asset Verification interface. The Keycloak service has been integrated with the Grafana tool of the BB9 DVT in Pilot 1, but it can also be used to provide SSO support to any other front-end component of IMOCO4.E. The cyber-security threat detection has been deployed on a Jetson edge device in the context of Pilot 3, but it can practically operate at any infrastructure node at the edge, fog or cloud and either monitor the operations and communications of its host infrastructure node exclusively or analyse the resources of multiple infrastructure nodes, especially if these are aggregated to the BB9 DMS.

2.6.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

Integration of the EST protocol on the gateway was relatively straightforward. Deploying OpenXPKI in a kubernetes cluster in such a way that it nicely integrates with other kubernetes aspects such as Ingress controllers proved significantly harder. Also, converting the existing OpenXPKI Docker images into a format that allows for easier configuration through Helm templates (including the ability to distinguish between Factory and Operational instances just by external configuration settings) proved difficult. This highlights the need for a building block that can easily be re-used.

Docker containers were used for deployment of the CSM services. In the case of the cyber-security threat detection tool, Tensorflow was used for ML model development and MLFlow was used for ML automations, model management and experiment tracking.

2.7 Data Visualisation Toolkit (DVT)

2.7.1 Tech Overview

The BB9 Data Visualisation Toolkit provides two main tools, based on the Grafana (Grafana, 2024) and Kibana (Kibana, 2024) technologies, which are used for visualising (a) real-time data flowing through the DMS and (b) historical data stored in the Big Data Repository respectively.

Grafana is an open-source tool that is widely used for visualising data from various sources and particularly live data streams. Grafana can be connected to a diverse range of data sources, including time series databases, relational databases, and cloud-based storage services. Grafana offers an intuitive user interface, enabling users to effortlessly build interactive charts, graphs and dashboards for monitoring and analysing data in real-time.

The DVT Grafana tool is integrated with the BB9 DMS and allows to visualise data that flow through the DMS Kafka cluster in real time. The exact parameters and visualisation types can be configured from the UI, and multiple visualisations can be combined in dashboards. Furthermore, Grafana allows the generation of alerts that can be delivered to interested parties by specifying corresponding alert generation rules and notification delivery channels, which include email and multiple messaging applications (e.g., Slack, MS Teams, Telegram, etc.). The Grafana tool can also be used by BB9 system administrators to quickly create custom dashboards that provide an in-depth view of the health and performance of the DMS clusters to optimise performance and ensure seamless operations of the deployment (see Figure 6). Alerts can also be set up for notifying administrators when specific metrics exceed predetermined thresholds, giving them time to quickly respond to any potential issues that may arise.

Kibana is an open-source data visualisation and exploration tool commonly combined with Elasticsearch to form an advanced data analytics platform. Kibana allows users to explore and analyse data from multiple sources, including both structured and unstructured information stored in Elasticsearch. Kibana users can easily create custom dashboards and visualisations that offer insights into key business metrics. Kibana provides users with various data visualisation options, including bar charts, line graphs and heat maps that can easily be customised to suit specific use cases. Furthermore, Kibana includes powerful search capabilities which enable them to rapidly filter and analyse large volumes of data quickly and efficiently.

The DVT Kibana tool is integrated with the BB9 BDR to provide a GUI tool for accessing, managing and visualising the archived data in the permanent storage. This tool also allows for certain data management tasks to be carried out, including re-organisation of indices, data type mapping specification, etc. Through this tool, the data can be filtered using complex queries that are specified by using the GUI. Individual data entries can be isolated and inspected either visually or in their raw text form for checking data consistency and specific values of interest. Data from specific or multiple indices can be filtered and combined to be fed to visualisations, which can be updated dynamically when new data entries are registered in the system.

2.7.2 Implementation aspects

Grafana and Kibana tool instances were deployed as part of the BB9 DVT in Pilot 1 and Pilot 3 and were associated with the respective deployments of the DMS and BDR. In **Pilot 1**, a Grafana dashboard was set up for monitoring the health and performance of the Kafka broker (see Figure 8) and another dashboard for monitoring key time-series metrics for the health and status of the involved Tissector equipment and the remaining useful life as calculated by the employed AI component (see Figure 9).

Figure 8: Grafana dashboard for monitoring critical health and performance metrics of the DMS Kafka broker

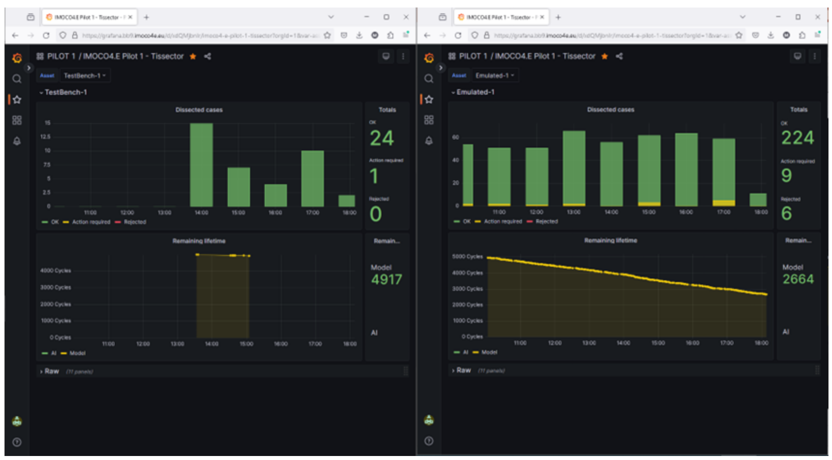


Figure 9: Data from real and emulated Pilot 1 devices visualized in a Grafana dashboard of the BB9 DVT

An alert mechanism was also set up to trigger a notification when certain criteria were met, and action was required (see Figure 10).

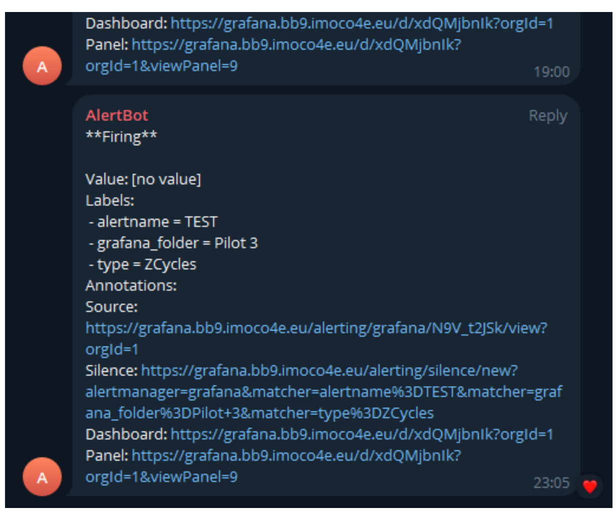


Figure 10: Alert generated by Grafana and sent to Telegram Messenger application

Kibana was used for more detailed data inspections of the permanently stored data, which was particularly useful during configuration activities of the Pilot 1 pipeline (see Figure 11).

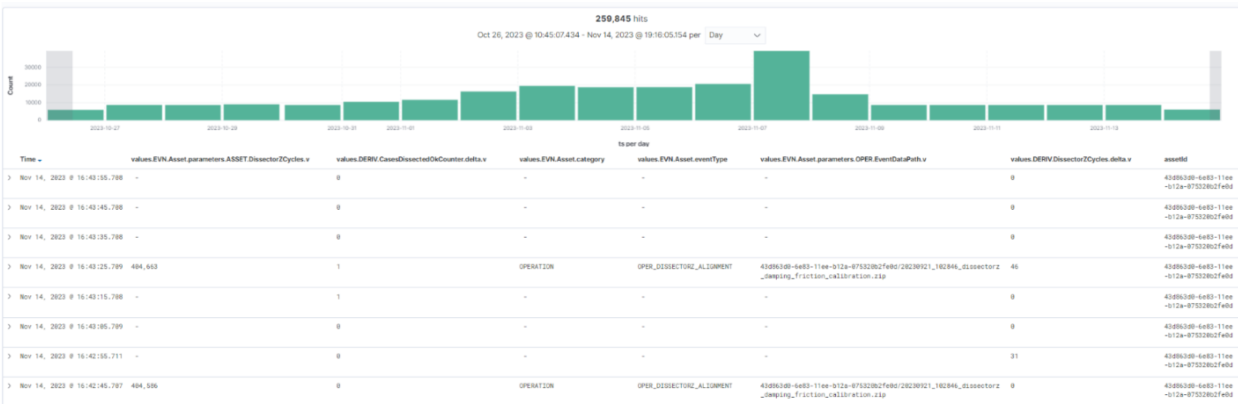


Figure 11: Inspection of Pilot 1 individual events filtered by time in tabular view using Kibana

In **Pilot 3**, a Grafana dashboard for monitoring time series data was set up for the raw sensor data and the inference results from 4 AI components employed in Pilot 3 that were published to the DMS (see Figure 12).

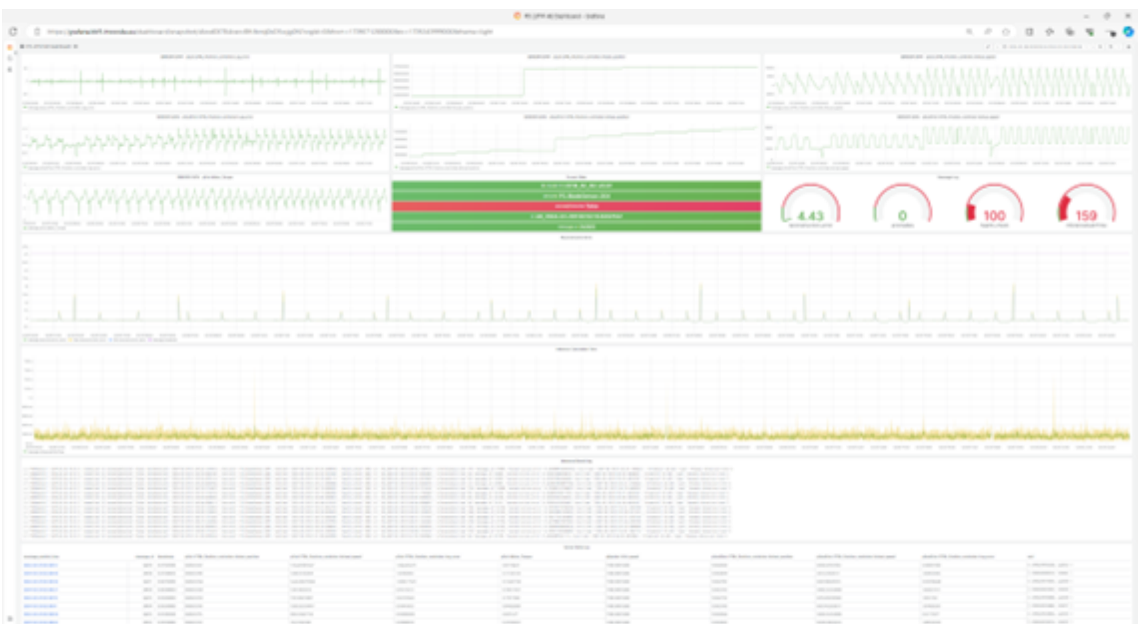


Figure 12: Grafana dashboard for monitoring raw data and anomaly detection results in Pilot 3

Also in this case, Kibana was used to inspect the data that was published to the DMS and permanently stored in the BDR. This was particularly useful to identify data inconsistencies and deviations from the data models that had been specified for usage in Pilot 3. Additionally, it was used to visualize and review key metrics from specific time periods of interest (see Figure 13).

Figure 13: Kibana dashboard for reviewing anomaly detection measurements in Pilot 3 and associated latencies

2.7.3 Results

The DVT successfully fulfilled the set objectives both in the cases of Pilot 1 and Pilot 3 by managing to deliver data visualisations necessary for the operation of the respective use cases. All Grafana and Kibana tool instantiations managed to successfully read data from the intended sources (DMS and BDR), they were appropriately configured by setting up the necessary graphs that were suitable for representing each data type and the graphs were organized into comprehensive and intuitive dashboards.

2.7.4 IMOCO4.E requirements

The BB9 DVT meets the following requirements: R052-D5.1-B9-sw, R056-D5.1-B9, R057-D5.1-B9, R068-D5.1-B9.

2.7.5 Capabilities & Limitations (including USP, strengths & weaknesses)

Data visualisation is an integral part of big data analytics approaches and can effectively assist in intuitively communicating data and conveying their implications. In the Industry 4.0 paradigm of IMOCO4.E, the vast amounts of data produced by connected devices and machines can be better understood using data visualisation techniques. By employing tools like charts, graphs and dashboards for data visualisation purposes, industries can identify trends, patterns, or anomalies in the data, therefore supporting informed decision-making and achievement of improved outcomes in terms of quality, productivity, efficiency safety, etc.

Grafana's flexibility and extensibility have led to its widespread usage by data analysts, developers, and IT professionals. The visualisations supported by Grafana can be particularly useful to IMOCO4.E use cases, as operators have access to a live view of ongoing processes and can quickly gain an understanding of the health and performance of industrial equipment on the factory shop floor, monitor identified patterns, trends and anomalies, potential problems in the production lines, etc. Automated alerts is of high significance in various IMOCO4.E use cases, such as predictive maintenance and quality control, as involved parties can be notified whenever an event occurs, and actions need to be taken.

Similarly, the use of **Kibana** for data analytics can provide industries that adopt IMOCO4.E with a powerful tool for uncovering insights in their data, making data-driven decisions and increasing overall performance and efficiency in their day-to-day operations. By granting access to a visual representation of historical data that have been collected from various sources and over a long period of time, useful insights can be revealed, such as temporal correlations of specific events from distinct data sources that would have not been possible to identify if the respective data had not been aggregated in a common location and fed into a common visualisation.

Regarding the visualisation of the data that are permanently stored in the BB9 BDR, which uses Elasticsearch as its underlying technology, the choice of Kibana was obvious, granted that it integrates seamlessly with Elasticsearch as they essentially belong to the same stack. Regarding the visualisation of real-time data that are published to the BB9 DMS, there are several robust platforms that are used in the field of data visualization, monitoring, and analytics, but they serve different purposes and have distinct features. Some of the most well-known platforms are briefly described in the following paragraphs and are compared to Grafana which is the selected tool for the DVT.

The majority of platforms operate on a paid subscription model**. Datadog** (Datadog, 2024) is a fully managed cloud-based monitoring platform with comprehensive visualization and alerting capabilities and provides a wide range of features, including infrastructure monitoring, application performance monitoring (APM), log management and distributed tracing. Datadog follows a subscription-based pricing model, which could be quite expensive especially for small and medium-sized companies, while Grafana on the other hand is an open-source and free to use tool. Additionally, the setup process of Datadog is rather complex. **Splunk** (Splunk, 2024) is a comprehensive platform for searching, monitoring, and analysing machine-generated big data, including logs, events, and metrics. It is designed as an all-in-one security and monitoring solution for enterprises but is not a low-cost option and is not very easy to set up compared to other platforms. Compared to Grafana, Splunk is more focused on log management and operational intelligence, while Grafana excels in visualizing time-series data and creating dashboards for monitoring systems and applications, which are more relevant to the context of IMOCO4.E. **Instana** (Instana, 2024) is another observability and monitoring software tool offered by IBM. It is an enterprise-ready platform that is relatively easy to set up and it also offers full stack monitoring. However, in terms of pricing it has a relatively high cost compared to other proprietary platforms. **New Relic** (Relic, 2024) is a full-stack observability platform that provides monitoring and analytics for applications, infrastructure, and customer experiences. However, it is relatively expensive and can require a steep learning curve, due to its technical nature and lack of help resources.

With regards to open-source visualization platforms, **Signoz** (Signoz, 2024) is an open-source distributed tracing and observability platform designed for organizations with microservice-based architectures that need deep insights into application performance, latency and errors. It has a main drawback that it does not provide cloud SIEM (security information and event management system) compared to other platforms. **Zabbix** (Zabbix, 2024) is also an open-source IT infrastructure monitoring tool with enterprise-ready functionalities such as full-stack monitoring, data visualization, dashboard customization, alerting, and team management. Even though Zabbix is free to install and use, technical support is upon payment and also it is not very easy to install and set up.

Overall, Grafana's open-source nature, versatility, rich visualization options, scalability, integration capabilities, and extensive plugin ecosystem contributed to its choice to serve the needs of real-time data visualization in IMOCO4.E.

2.7.6 Customizations & Adaptations (including possible modifications and extensions)

The DVT deployment and configuration can be customized to serve specific visualization needs of multiple heterogeneous types of data that may be involved in an IMOCO4.E use case and managed by the BB9 DMS and BDR. The Grafana and Kibana tools offer a wide range of visualization types and configuration options that can be employed to produce dashboards and notification policies that are tailored to use case requirements. Furthermore, even though the DVT has been tightly coupled with the DMS and BDR in the context of the current BB9 implementation, they could also potentially support additional, alternative data sources.

2.7.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

Docker technology was used for the deployment of the Grafana and Kibana instances. Grafana was coupled with Telegram for the needs of alerting functionality demonstration.

2.8 Time-Sensitive Networking (TSN)

2.8.1 Tech Overview

Time Sensitive Networking (TSN) is an extension of the Ethernet protocol that enables real-time synchronization and deterministic communication. In this IMOCO4.E project, ongoing development and improvements have been made to the necessary features and functionalities for creating a device capable of forming complete TSN networks that can be controlled and monitored remotely. The device is called TSN Z16, a platform built on the Xilinx Zynq 7000 family SoC, incorporating all TSN developments.

Task T5.2 and BB9 have focused on the parts most related to data management and security of the entire TSN system. The hardware component essential for TSN's core functionality has been addressed in the context of BB1 and T3.3, while redundancy has been developed here, enhancing security and minimizing the risk of failures. Part of the BB9 also includes the software component for monitoring the TSN system, allowing data control and error prevention. The monitoring aspect in BB9 includes the basic high-level portion of the Central Network Controller (CNC) and specific development for latency measurements, while the platform and the part closest to the cores fall under T3.4, with details provided in (D3.7).

Redundancy. FRER. (HW-015)

Within the HW-015 component of the TSN system, developments related to redundancy fall under BB9. The purpose of redundancy is to enhance communication security preventing information loss, by duplicating packets and sending them through different paths within the network. This is particularly beneficial for critical traffic, ensuring that data continues to arrive without the need for recovery times if any of the links becomes saturated or fails. This effectively increases the reliability and robustness of the network by guaranteeing the delivery of critical flows using redundant data forwarding routes.

In TSN, it is defined that the best way to implement seamless redundancy is by following the 802.1CB standard for Frame Replication and Elimination for Reliability (FRER). The logic to comply with the standard specifications has been developed and implemented at a low level in the FPGA IP cores, and on our platform, links where redundancy will be active can be enabled and configured.

To meet the 802.1CB standard, the frames to be redundantly transmitted must first be identified and replicated, sending them through different paths. Once the frames reach the destination, duplicates must be identified, and unnecessary ones eliminated to prevent duplicate information at the destination. This logic has been programmed into the FPGA cores and is closely related to another core, the traffic identification core, explained in D3.8 of WP3.

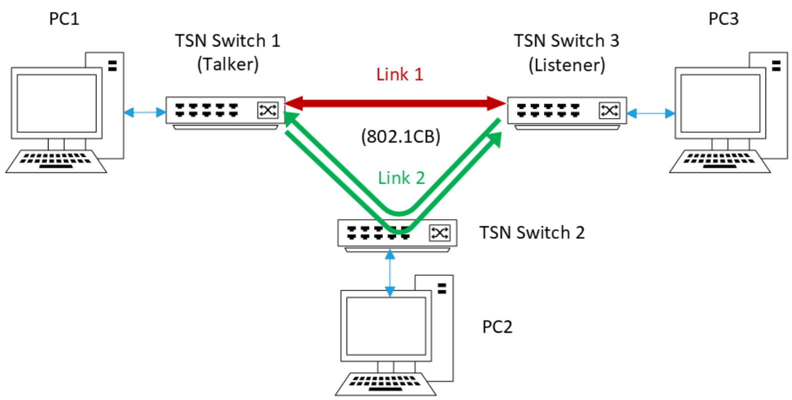


Figure 14: Example of a redundancy scenario with TSN

In Figure 14 an example is presented to illustrate how redundancy could work. In this scenario, there is a critical flow between "TSN Switch 1" and "TSN Switch 3", and redundancy is desired. If "Switch 1" initiates the critical flow, it would be responsible for duplicating packets and sending them through different paths: "Link 1" directly connected to "Switch 3", and "Link 2" passing through "Switch 2". "Switch 3" receives the packets and must identify those already received, discarding duplicates. This way, if "Link 1" were to fail, the messages would continue to arrive.

To activate redundancy on this platform, certain fields need to be configured. To initiate redundancy, one must go to the equipment, the interface and the flow intended for redundancy, specifying that redundancy will take place and the other interface to do the redundancy. In the case of the previous figure, on "Switch 1" one would need to access the interface connecting to "Switch 3" and the critical data flow to indicate redundancy, and the interface connecting to "Switch 2" would be marked as the redundant link. Finally, on "Switch 3" you would need to access the interface connecting to "Switch 2" to enable the packet filtering option, indicating that if the packet has already arrived through another interface, it should be discarded (dropped).

Monitoring tool interface. Latency measurement. (SW-075)

The TSN monitoring tool (SW-075) is divided into two parts in the project: the logic at the FPGA core level and communication up to the CNC, which is in WP3, and the control, visualization, and interface part, which is in T5.2 and BB9. This second part has been incorporated into BB9 as it aligns with the data management aspect. It is from the CNC that everything happening in the network can be securely and remotely controlled, optimal configurations can be sought, the status can be visualized, and potential failures that could compromise system security can be prevented.

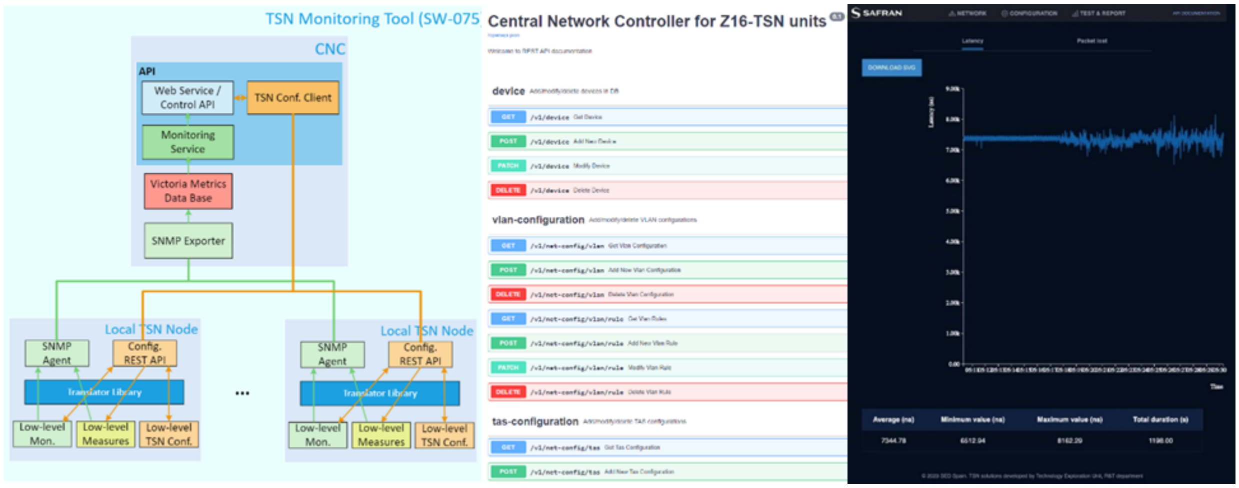


Figure 15: a) TSN monitoring tool architecture. b) CNC configuration services. c) Latency results example

The CNC is composed of multiple services, as depicted in a) of Figure 15. The first service through which most of the data enters is the SNMP Exporter, responsible for querying the state of nodes and transferring the results to the Victoria Metrics database. The database stores them, making them available for retrieval when requested by the API. The API encompasses the web and monitoring service, allowing access to various tabs for numerical or graphical visualization and monitoring of different parameters of the equipment and the network. Utilizing the TSN configuration service, as shown in b) of Figure 15, requests for the state and configuration of TSN-specific options can also be made.

In this project, in addition to continuously improving the services enabling external control of the platform, a completely new one has been developed. This new development allows the initiation of tests and the measurement of specific parameters, such as latency between devices or packet loss in a flow, over a specified period. These parameters are crucial and typically more challenging to obtain since they do not depend on a single device. By providing parameters into the interface to do the measurement, a request is sent to the node's API, which communicates with the FPGA core performing the measurement. Once completed, the results are returned, allowing visualization of the latency graph and data in nanoseconds, along with the percentage of packet loss (c) in Figure 15).

2.8.2 Implementation aspects

The implementation of T5.2/BB9 in Pilot 2 has been based on integrating the TSN system with the Pilot at ITEC offices. Before initiating this integration, several meetings were held to define the integration's objective, the point and layers where it would take place, and the timeline to follow. It was decided to integrate TSN as an alternative deterministic communication between Layer 3 (where the main PC is, receiving information from the pilot) and Layer 4 (where the server is, storing machine data).

Before the in-person integration, network traffic captures from the pilot were requested to study its structure, traffic types, and individual needs. This is aimed at establishing a complete and optimized network configuration for the scenario. Once ready, a visit to the facilities was made with the equipment to set up the scenario, allowing measurements for the final results of the project.

To conduct tests at the ITEC offices, two PCs were set up: one acting as Layer 3 with the simulator and digital twin model of a BIM line, and the other acting as a server, replicating the real case that would occur in the pilot. The configuration previously created was attempted, but some issues and minor bugs were encountered, and they had to be addressed on the spot. In Figure 16, on the left, the two PCs and their screens are visible, with the simulator running on the upper screen and the server monitoring proper functionality on the lower screen. On the right, the equipment forming the TSN network is shown.

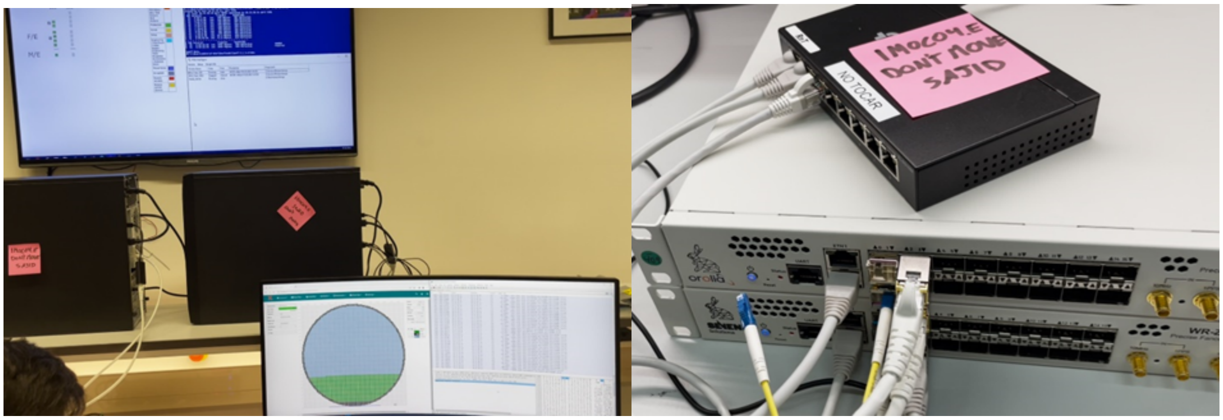


Figure 16: a) Setup at ITEC offices b) TSN devices

Regarding the specific components of the BB9, the proper functioning of the redundancy system was verified, allowing two links to run in parallel and disconnecting either of them without causing disturbances or packet loss in the network. This increases the system's overall security by enhancing its reliability in case of link failures. As for the monitoring software, the system's status could be always visualized, and latency measurements between the equipment were taken using the tool included in the CNC, as shown in the next Figure 17.

The integration was successfully completed after about two weeks of work, meeting all the previous objectives, getting the entire system up and running, and taking measurements. This integration was custom-built for this pilot in the project because in TSN, the topology used, the services to be implemented, and the prioritization of traffic are critical aspects of the configuration.

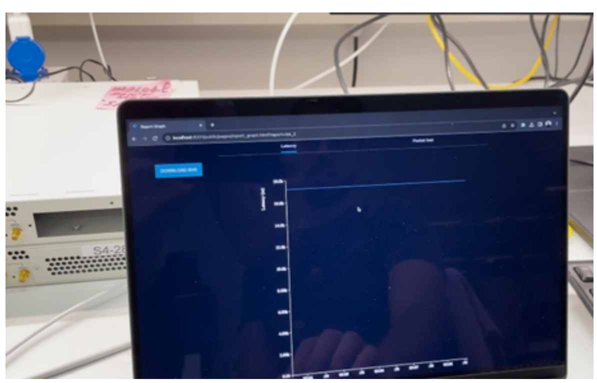


Figure 17: Latency monitoring tool

2.8.3 Results

During the development of the components, tests were conducted to verify their functionalities. Additionally, the first operational tests on the HW-015 and SW-075 were included in WP6, in (D6.4). The remaining tests and results will be included in the final deliverables of WP6, with specific tests on redundancies on the platforms. As for the monitoring tool, its correct operation will be verified, but no concrete tests with data to measure its performance will be conducted.

The most significant sign of achieving the results is the successful integration of the platform and the tool in both P1 and P2. This was demonstrated by verifying with the pilot traffic that redundancy was implemented correctly, with no packet loss when disconnecting one of the links, and by monitoring the entire network from the tool.

2.8.4 IMOCO4.E requirements

The components discussed in this section must meet the requirements defined in BB9. There are additional requirements related to certain aspects of development, such as R053-D5.1-B9 and R060-D5.1-B9 for TSN and redundancy, as they enable the handling of time-sensitive data streams in real-time while adhering to bandwidth and latency requirements and allow for high availability and fault-tolerant operation.

Regarding monitoring, R050-D5.1-B9 and R067-D5.1-B9 are fulfilled as information is collected and can be gathered in real-time with an API, and a GUI is provided. Additionally, R057-D5.1-B9 and R066-D5.1-B9 are met, as integrations have been carried out in pilots, and the components are easily adaptable.

2.8.5 Capabilities & Limitations (including USP, strengths & weaknesses)

The main feature of these components is the ability to create a complete TSN system, both by establishing the network and being able to control and monitor it easily. The network component focused on redundancy offers the advantage of being able to configure each parameter, allowing for full customization of the traffic flow to be redundant or the alternative path through which it should be sent. The monitoring part, in addition to the node status parameters, allows for obtaining data on the network and flows, which is less common. This advantage can be seen with latency measurement, where extra development has been implemented to obtain these flow data.

The redundancy system has some limitations, as it has been designed to duplicate and filter certain critical traffic within the network. This means that if redundancy is to be applied to a large amount of traffic, there may be some limitations. However, this implementation of redundancy is common, as scalability of redundancy is usually not required and would significantly reduce bandwidth. As for the monitoring part, it presents some limitations as it is an initial version of the development, but new features may be added to enhance other monitoring and configuration options.

2.8.6 Customizations & Adaptations (including possible modifications and extensions)

These components have been adapted for integration into the project pilots, specifically in P1 and P2. The TSN network and redundancy part has undergone most of the changes by applying configurations for the specific network. In each case, there was a different network topology with different traffic, which implies that the way of configuring the network and the paths through which redundant messages will be sent is different. The fact that the network is configurable makes these components adaptable to different use cases. Apart from that, the monitoring component is somewhat more fixed, without the need to change it in each implementation since it operates on the network components. This also makes it easier to start using once installed.

2.8.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

The toolchain used in this component encompasses a wide range of tools. For implementing code within the TSN nodes, Vivado tool has been utilized for low-level FPGA programming using VHDL language.

Additionally, C code has been employed for testing and programming device software, while high-level Python code has been used for automation and function dispatch. For the user interface part, web languages such as HTTP, CSS, and JavaScript have been used, and databases like VictoriaMetrics have been programmed. Moreover, network tools like Iperf for traffic injection and Pyshark for monitoring have been employed.

In the development of these components, we had to use this diverse set of tools, learn how to integrate them, and ensure that the ensemble becomes intuitive and useful for the users. We have learned that configuration is a crucial part of TSN, and that monitoring is a vital aspect for network systems.

2.9 Healthcare Robotics Data Management Pipeline (PEN, PMS)

2.9.1 Tech Overview

In Pilot 4, Healthcare Robotics, a Data Management Pipeline is built. It comprises three components:

- **Data lake**
- **Extract-Transform-Load pipeline**
- Vertica **database**

2.9.2 Implementation aspects

Pilot 4 is related to a medical robotic manipulator that is used for image guided therapy. The robotic system has a C-arc with an X-ray tube on one side and an X-ray detector on the other side. This medical device is used to position the X-ray beam in different orientations with regard to the patient on the table. The objectives of this pilot, as described in deliverable D7.1 are the following:

1. Accelerate development & deployment of meaningful innovations to the market by use of a (digital twin) model driven approach.
2. Extend system monitoring of our development systems to enable early feedback, fast improvements, and test & training data for our (digital twin) model driven approach.
3. Time optimized and first-time right service & maintenance of our systems in the field by extending the system monitoring and use of state-of-the art data model technology to automate processes and commissioning.

To achieve these objectives, data is collected from the device that includes both low-level sensor measurements, as well as machine log data. These data sources can provide valuable information for continuously monitoring the device and enable early detections of situations where the device works outside of the desirable conditions.

2.9.3 Results

Data lake. The first element is a data lake that is the first landing spot for device logs. The data lake serves as a storage place for large amounts of raw device logs and needs to scale to petabytes since it continuously collects data from several thousand devices that operate in the field on a daily basis. The storage solution employed for the data lake is AWS S3 that supports the handling of a large number of small files that are typically generated by devices relevant to Pilot 4.

Extract-Transform-Load pipeline. The second element is a parallel ETL process associated with data preprocessing that extracts the information from the device logs that reside in the data lake and stores them in a structured way in a data warehouse. The parallelization of the ETL process is done using SLURM workload management. SLURM is a free and open-source job scheduler that supports Linux and is very popular for scheduling jobs in supercomputers and computer clusters. Each file generated by the medical device is associated with a specific ETL module. This means that for the device logs associated with the robotic manipulator, there is a dedicated ETL process that extracts the relevant information and stores it in a structured way in the data warehouse. Most of the ETL modules are implemented in Java.

Vertica database. The third element is the data warehouse, also associated with data preprocessing, where the information extracted from the device logs is stored in a structured way. The solution used for this is Vertica, which is a column store that is compatible with SQL. Vertica is designed for data to be written once (small number of updates and deletes) and read multiple times. The column-oriented nature of Vertica means support for denormalised data models that allow fast data access with small number of tables joins.

2.9.4 IMOCO4.E requirements

The Healthcare Robotics Data Management Pipeline addresses the following T5.2/BB9 requirements: R048-D5.1-B9, R049-D5.1-B9, R050-D5.1-B9, R051-D5.1-B9, R058-D5.1-B9, R059-D5.1-B9, R060-D5.1-B9, R061-D5.1-B9, R062-D5.1-B9.

2.9.5 Capabilities & Limitations (including USP, strengths & weaknesses)

The data related to Pilot 4 are sensitive medical device data and thus are primarily stored in a database with limited access rights. For the needs of Pilot 4, a case-specific, tailored **Data Management Pipeline** is implemented. Due to the sensitive nature of the medical device data, this processing is implemented with limited access rights. The Data Management Pipeline comprises three main elements.

Software component SW-037 of the IMOCO4.E framework contains a dedicated ETL module for each device log file that is generated by the device. There are two files that are relevant to the robotic manipulator of Pilot 4. One file contains all the operations that the robotic manipulator is performing as well as error and warning messages that are generated in the process. These logged events contain an ID, a text description, a category (like Informational, Warning, Error) as well as a timestamp. Typically, a robotic manipulator similar to what is considered in Pilot 4, generates ~10^3 machine log events during a typical day of usage.

The second device log file that is related to the robotic manipulator of Pilot 4 contains motion trace data. These are essentially sensor measurements extracted from and around the motor drive that controls the movement of the device during its use. The sensor measurements include the motor drive current that is measured at 500 Hz during the movement of the C-arc. It should be noted that the motion trace data is not included in a single file but rather a collection of files that are generated by a C-arc movement from the moment it is initiated, until it stops. With a sampling rate of 500 Hz, a 20 second movement of the C-arc will generate 10k lines of sensor measurements in the device log file. A normal working day of such a system can generate 10^6 sensor measurements.

For both these device log files, the appropriate ETL modules have been implemented that extract the relevant information and store it in the Vertica database.

2.9.6 Customizations & Adaptations (including possible modifications and extensions)

Pilot 4 uses state-of-the-art industry grade data lake (Amazon S3) and database solutions (Vertica). As the ETL process is data specific, it was developed in-house.

2.9.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

1. An Amazon Web Services Simple Storage Service (**Amazon S3**) data lake, to which the logs and motion trace data from the healthcare devices are continually uploaded.
2. An in-house developed Extract-Transform-Load (ETL) pipeline, extracting data from the data lake, processing it, and transforming it into proper storage format to load it onto the database. The pipeline is implemented on **a Linux cluster running the SLURM workload manager.**
3. A **Vertica** database. Vertica's features include column-oriented storage orientation, massively parallel processing, and a **standard SQL interface**.

2.10 UC1 xIL co-simulation through cloud

2.10.1 Tech Overview

The component is an advanced control system designed for elevator systems in buildings. It utilizes cloud co-simulation, advanced control algorithms, and secure communication through an SSH tunnel and TCP/IP protocols.

Function: The function of this component is to control the performance of the lift system with particular emphasis on speed/flight time, accuracy, vibration compensation and comfort. It aims to maintain or improve lift performance as the building rises and car speeds increase.

Internal Operation: The control system uses advanced algorithms to handle control challenges like nonlinearity, uncertainty, and time-varying systems. These algorithms are tested through cloud co-simulation, using two models: a Simcenter Amesim model for simulating lift components and a MATLAB Simulink model for control. Both models exchange and react to each other's output through a secure SSH tunnel for data transmission.

Features: Cloud co-simulation is employed by the control system to validate control algorithms. This allows for a dynamic and interactive simulation environment, where models can exchange and respond to each other's outputs. **Simcenter Amesim Model:** The control system incorporates a Simcenter Amesim model that accurately simulates the mechanical and electrical aspects of the elevator. **MATLAB Simulink Model:** The control system includes a MATLAB Simulink model for the control part of the system. **SSH Tunnel:** The communication between the models is achieved through an SSH tunnel, which provides a secure and encrypted channel for data transmission. **TCP/IP Communication:** The transferred data between the models is of type double and represents the parameter values of the models. The data exchange takes place in real-time at one millisecond intervals within the co-simulation. TCP/IP communication is used for reliable end-to-end communication between the models, ensuring accurate and consistent data transfer.

2.10.2 Implementation aspects

In Use Case 1, the integration of T5.2/BB9 includes various significant advancements and tailored modifications to cater to the specific needs of the elevator system. These advancements and modifications encompass:

Cloud-based virtual machine and scalability: Using a cloud-based virtual machine, an EC2 AWS instance, to run the Simcenter Amesim model offers several advantages. First, it allows for easy scalability, as additional computing resources can be allocated as needed to handle larger simulations or higher volumes of lift data. Second, it reduces the dependency on local hardware and infrastructure, allowing greater flexibility in accessing and running the simulation from anywhere with an Internet connection. This scalability and flexibility contribute to a more efficient and adaptable lift control system.

Integration of MATLAB Simulink: The integration of MATLAB Simulink into the co-simulation architecture enables the implementation of advanced control algorithms specific to the elevator system. These algorithms can address the nonlinearity, uncertainty, and time-varying nature of the elevator's mechanical and electrical components, improving overall performance. By leveraging MATLAB Simulink's extensive library of control blocks and tools, it was possible to develop customized control strategies that optimize speed, accuracy, vibration compensation, and comfort.

Integration of Simcenter Amesim: The use of Simcenter Amesim model in the co-simulation architecture offers a comprehensive solution for simulating the mechanical and electrical components of the elevator system. By utilizing an EC2 AWS Cloud virtual machine, the model can be accessed and executed remotely, providing flexibility and scalability.

A collaborative and iterative simulation process: The bi-directional exchange of data and events between the Simcenter Amesim and MATLAB Simulink models promotes a collaborative and iterative simulation process. This collaboration allows continuous refinement and improvement of the control system. The output of one model can be analysed and used to adjust the parameters or inputs of the other model in real time, enabling rapid iteration and fine-tuning of the control algorithms. This iterative process ensures that the control system continuously adapts and optimises its performance based on the dynamic behaviour of the lift system.

Secure and encrypted communication: The use of SSH tunnels for communication between the local machine running MATLAB Simulink and the cloud-based virtual machine running Simcenter Amesim ensures secure data transmission. SSH tunnels provide authentication and encryption, protecting the parameter values exchanged between the models from unauthorised access or manipulation. This security measure is critical to maintaining the integrity of the simulation and the confidentiality of sensitive lift system information.

Real-time data exchange for accurate control: Real-time data exchange between Simcenter Amesim and MATLAB Simulink models at millisecond intervals enables precise and timely control of the lift system. This high-frequency data exchange ensures that the control algorithms can respond quickly to changes in elevator speed, position and external factors. By continuously exchanging data in real time, the control system can make precise adjustments to optimise lift performance.

TCP/IP Communication: The data transferred between the models is in the form of double type, representing the parameter values of the models. This exchange of data occurs in real-time, with intervals as short as one millisecond, within the co-simulation. To ensure accurate and consistent transfer of data, TCP/IP communication is utilized, establishing a reliable end-to-end communication mechanism between the models.

These developed ideas highlight the benefits of the T5.2/BB9 implementation in Use Case 1, emphasising cloud-based scalability, integration of advanced control algorithms, collaborative simulation process, secure communication, and real-time data exchange. By harnessing these innovations and adaptations, the lift system achieves superior performance, reliability, and adaptability, meeting the critical requirements of speed, accuracy, vibration compensation and comfort.

2.10.3 Results

The control system was able to improve or maintain elevator performance as the building increased in height and elevator speed by using an advanced control system based on sophisticated algorithms and cloud simulation. The control algorithm adapts in real time to changes in speed, position, and external factors, ensuring accurate and timely control of the lift system. The integration of Simcenter Amesim and MATLAB Simulink models in a collaborative co-simulation architecture allows continuous refinement and optimisation of the control system. Secure communication via SSH tunnelling and TCP/IP protocol ensures accurate and consistent data transfer between models, protecting the integrity and confidentiality of sensitive lift system information. Overall, the implementation of these advanced technological capabilities has resulted in a high performance, reliable and adaptable lift system that meets critical requirements for speed, accuracy, vibration compensation and comfort.

2.10.4 IMOCO4.E requirements

The following requirements are met through this implementation: R047-D5.1-B9, R052-D5.1-B9-sw, R053-D5.1-B9, R054-D5.1-B9, R056-D5.1-B9, R057-D5.1-B9, R058-D5.1-B9, R059-D5.1-B9, R060-D5.1-B9, R062-D5.1-B9, R065-D5.1-B9, R066-D5.1-B9, R068-D5.1-B9, R003-D5.1-LX-sw.

2.10.5 Capabilities & Limitations (including USP, strengths & weaknesses)

The state-of-the-art in the field of this elevator control system includes the use of control algorithms to manage lift performance. Innovations that surpass the state-of-the-art include the dynamic simulation environment enabled by cloud co-simulation, secure data transmission through an SSH tunnel, and real-time TCP/IP communication for accurate and reliable data exchange.

These advancements not only enhance the overall performance of the elevator control system but also improve the safety and security measures. The integration of cloud co-simulation allows for efficient testing and analysis of various control algorithms, leading to optimized lift performance. Additionally, the use of secure communication protocols ensures the protection of sensitive data and prevents unauthorized access to the system, ensuring a reliable and secure experience.

2.10.6 Customizations & Adaptations (including possible modifications and extensions)

Customization may include changes to existing control algorithms, adjusting control parameters or adding additional functionality. The system can also be extended by integrating with other building automation and management systems or by adapting to the special requirements of certain industries or environments.

2.10.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

The cloud co-simulation involves the interaction of two models through an SSH tunnel that allows TCP/IP communication. The two models used are a MATLAB Simulink model and a Simcenter Amesim model. One of these models is stored in an AWS resource, specifically a virtual machine, which requires configuration of the environment. To ensure secure communication, an SSH tunnel was established using the Putty application. This aspect of tool integration is crucial in facilitating the secure and seamless exchange of data between the models.

2.11 Deploying DTT's anomaly detection AI model in Edge and Fog environments via Kafka to achieve predictive maintenance for manufacturing processes.

2.11.1 Tech Overview

The anomaly detection AI model developed by DTT within the context of the IMOCO4.E project relies on Kafka for data streaming and transmission. By leveraging Kafka's capabilities, data integrity within communications between the Edge and Fog environments are ensured. Additionally, the BB9 cybersecurity module enhances safety by recording, filtering, and curating traffic streams, providing comprehensive protection against potential threats. The deployment of BB9 not only addresses the security aspect of the anomaly detection model but also demonstrates its applicability and effectiveness in ensuring the safety and security of data transmission within manufacturing environments.

2.11.2 Implementation aspects

This component was developed in the context of Pilot 3. However, it can be used in other IMOCO4.E deployments. For the anomaly detection, the bottle packaging process was considered as a use case where predictive maintenance is planned to be achieved through deploying an Image Anomaly detection ML model. At the beginning of the process when bottle image data are generated via camera sensors which are then fed into the ML model, it identifies any irregularities as *Anomalies* within the images; the model is deployed on a Huawei Edge device. The final output of the identification of irregularities is displayed through a user interface (UI). From there, these images are then converted to JSON format and streamed using Kafka. The data flow among different components of this AI model is illustrated in the block diagram depicted in Figure 18.

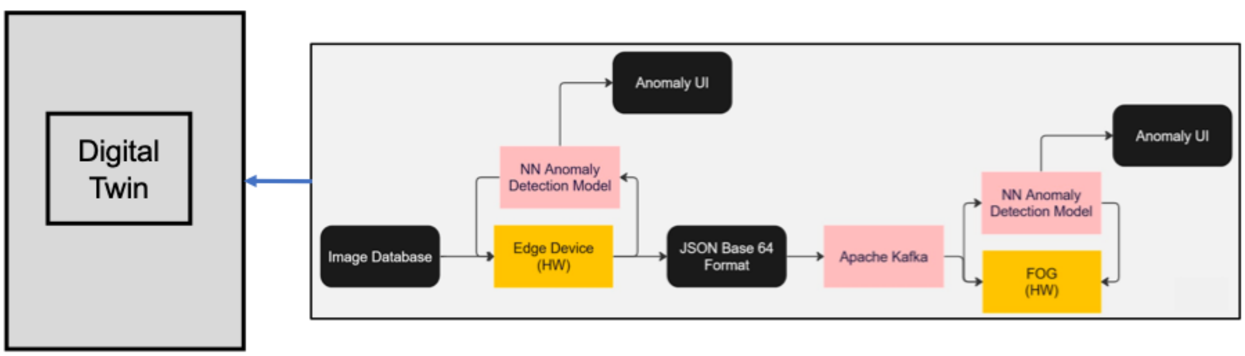


Figure 18: Block diagram of the ML/NN model showcasing the data flow among different components of the solution

Further processing is executed in the *Fog* environment where the image anomaly detection ML algorithm operates.

2.11.3 Results

DTT's model is trained on the "bottle" dataset provided by Pilot 3. Here, we achieved an overall accuracy of 88 %. The precision and recall values of this model are 75 % and 85 % respectively.

The usage of Kafka within the implementation of this AI model helps to achieve timely delivery of image frames from the source to the Fog environment so that the AI model can process it and generate the prediction of containing anomalous content on the streamed images. This also makes sure that the image data are streamed without losing quality.

2.11.4 IMOCO4.E requirements

The most relevant requirements which are addressed by the scope of DTT's work are R047-D5.1-B9, R048-D5.1-B9, R049-D5.1-B9, R050-D5.1-B9, R053-D5.1-B9, R055-D5.1-B9, R057-D5.1-B9, R058-D5.1-B9, R060-D5.1-B9, R061-D5.1-B9, R062-D5.1-B9, R063-D5.1-B9, R064-D5.1-B9, R066-D5.1-B9, R067-D5.1-B9, R005-D5.1-LX-sw.

2.11.5 Capabilities & Limitations (including USP, strengths & weaknesses)

Leveraging Kafka's features, the AI model ensures data integrity of the streamed data. Moreover, supported by Kafka for data streaming, the AI model exhibits scalability and versatility, allowing for seamless integration into various manufacturing environments. Kafka's ability to handle large volumes of data ensures the AI model's capability to process and analyze diverse datasets efficiently.

The AI model's unique selling point (USP) lies in its integration with Kafka for maintaining data integrity features in manufacturing environments. This integration enhances the model's reliability and trustworthiness, making it a robust solution for predictive maintenance and anomaly detection in industrial settings.

The AI model's limitations primarily revolve around potential complexity of implementation and resource overhead. Integrating Kafka may introduce complexity during development, integration, and maintenance phases, requiring specialized expertise in data streaming technologies. Furthermore, Kafka's features may impose additional resource overhead, including computational resources and network bandwidth, which need to be carefully managed to ensure optimal performance of the AI model while maintaining data streaming capabilities. These challenges may impact the model's deployment and scalability in industrial environments.

2.11.6 Customizations & Adaptations (including possible modifications and extensions)

Customizations of the AI model supported by Kafka involve fine-tuning parameters within Kafka to align with specific data streaming requirements and compliance standards. Additionally, customizations may include user interface customization. These customizations enable the AI model to adapt to the unique needs of manufacturing environments, ensuring compliance with industry standards.

Adaptations, possible modifications, and extensions of the AI model supported by Kafka encompass various enhancements to improve functionality and address evolving challenges. These include integrating the model with Security Information and Event Management (SIEM) systems for comprehensive security monitoring, implementing dynamic security policies based on real-time threat intelligence, and enhancing scalability to support deployment in large-scale manufacturing environments. Possible modifications involve incorporating advanced optimizations within the anomaly detection algorithm and integrating with threat intelligence platforms for automated threat detection and response. Extensions may include blockchain integration for immutable data storage and enhanced data integrity, providing an additional layer of security for sensitive manufacturing data. These adaptations, modifications, and extensions bolster the AI model's ability to effectively mitigate cybersecurity risks and ensure robust security measures in manufacturing environments.

2.11.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

These are already explained in the above sections.

3. AI methods for monitoring and predictive maintenance at instrumentation level

3.1 Overview of realized solutions

The purpose of this section is to present results of contributing partners dealing with the Task 5.3 'AI methods for monitoring and predictive maintenance at instrumentation level'. In general, the goal is to provide SW tools enabling condition monitoring and predictive maintenance at instrumentation layer (layer 1) of IMOCO4.E common architecture. The joint effort is focused on BB6 'Algorithms for condition monitoring, predictive maintenance and self-commissioning of industrial motion control systems. BB6 further includes components developed in Task 5.4 'Automatic commissioning of motion control systems' as well. The outcomes of Task 5.3 are, for demonstration purposes, mainly integrated in UC1 and Pilot 4.

The condition monitoring can provide crucial information about the state of the observed system, enabling proper action to either prolong the operation of the system or to commission a precisely aimed maintenance action, reducing both the cost and down time. Task 5.3 provides a comprehensive insight into actions required to apply AI methods for predictive maintenance of selected systems.

One of the addressed applications concerns the monitoring and RUL estimation of power inverter components (BUT). A critical portion of the development of AI methods is the availability of datasets of sufficient quality. In this endeavour, accelerated aging tests were utilized. However, the existing test bed needed to be altered to accommodate additional measurements of thermal transients. Additionally, some data-based condition indicators were tested, with their capabilities of RUL prognosis evaluated and their inherent limitations described. In order to perform these tests, a framework enabling data analysis and design of condition indicators was developed in MATLAB environment. This further opens a possibility to generate a C code, which could be potentially deployed on a low-level controller unit, monitoring the health of IGBT(s) under supervision.

Another topic deals with the condition monitoring based on the measurement and analysis of vibrations (BUT). The chosen application focuses on lifts, both in terms of the health estimation and ride quality evaluation. The proposed diagnostics package contains a Smart Wireless Sensor, developed in WP3 (BB3) and a wireless gateway, enabling convenient data transfer. The system for obtaining the datasets, which could be then used for, e.g., training the NN is proposed. Additionally, a tool allowing the evaluation of the gather data following the related ISO standards was developed in LabVIEW. This led to the derivation of condition indicators based on the parameters normally monitored in order to evaluate human perceived comfort of the lift ride.

A different approach to condition monitoring is model based, possibly utilizing a digital twin of the monitored system (WEG, UNIBS). This possibility was also explored, again in relation to the health estimation of a lift. A model, reflecting the nonlinear dynamics of such system was developed and identified. The digital twin was implemented in Siemens Amesim environment. This enabled simulation of conditions reflecting standard operation, but also an insertion of faults.

As Pilot 4 includes a robotic manipulator, its component failure was of concern as well (PEN, PMS). Two main failure modes were explored: specifically friction wheel slip and motor drive failure. There are multiple methods which can be leveraged to determine which failure is about to occur. Velocity measurements were used for detection of first failure, while current measurements help with the detection of the other. Real-life experiments in a controlled environment were conducted, allowing evaluation of selected condition indicators for failure prediction. The calculated condition indicators were then used as inputs for machine learning algorithms further explained in Section 7.

A final contribution in this section dealt with the detection of audio anomalies and the possibility of the utilization of captured audio signal for condition inspection and fault diagnostics (REEXEN). Several approaches to anomaly detection were examined. Finally, a system based on the extraction of mel-frequency spectral coefficients and subsequent application of depthwise separable CNN was developed and tested. The achieved results proved that the proposed system worked well on the selected dataset, containing acoustic signature of selected machines in normal and faulty states.

3.2 Addressed ST objectives and KPIs

This task relates to objective ST4 (see Figure 2). To be more precise, it provides robust and AI-based condition monitoring and predictive diagnostics which contribute to building block BB6. The KPIs of BB6 are listed in Table 2. Some BB6 indicators were also addressed in task 5.7, i.e., in section 7 of this document.

Table 2. KPIs of BB6

KPI #	KPI / smart functionality	IMOCO4.E target (TRL 4) MS6/M30
KPI_BB6_1	Wireless smart diagnostic IIOT sensors.	Smart IIoT diagnostic sensors with low power design and fully interoperable interfaces, preferably wireless. Parametrization of the sensor properties. Software support with the algorithms for use in the sensor and also outside of the sensor. Cheap, easy to implement sensor will enable widespread utilization of diagnostics.
KPI_BB6_2	Diagnostic models of common parts of mechatronic systems and methods for their parameter identification based on measured quantities. The data acquisition parameters definition for individual fault models.	Reusable fault models for common parts/faults of mechatronic systems. Specification of data acquisition performance for the model functionality. Increased reliability and shorter downtimes. Modelling and analysis can lead to new requirements on sensors for new inverter/motor designs with condition monitoring and predictive maintenance functionality.
KPI_BB6_3	A software library of data/model/digital twin - based condition indicator calculation for mechatronic systems, algorithms for fault identification, classification, and prediction.	A dedicated set of functions and methods which extend Predictive Maintenance Toolbox functionality. The precision of maintenance planning will be improved by the use of twinning technology and the AI techniques.
KPI_BB6_4	Self-commissioning of industrial control loops	Utilization of AI methods to reduce or even suppress the need for human expert intervention into the process of

		self- commissioning. Utilization of digital twins for controller parameters fine-tuning and the possibility to select user-defined criteria for parameter optimization like energy-efficient operation. Virtual commissioning enabled by digital twin-in-the-loop will bring safe, fast and robust tuning and early control quality assessment.

This task addresses Layer 1 (instrumentation level) of IMOCO4.E four layers structure.

3.2.1 KPI_BB6_1 status

The hardware of wireless smart diagnostic IIOT sensor was designed and manufactured in WP3. The software fulfilling KPI_BB6_1 is described in section 3.4.

3.2.2 KPI_BB6_2 status

Thermal model of IGBT power modules and related parameter identification in chapter 3.3, digital twin of the lift for the realisation of the controller and for virtual failure generation in chapter 3.5, machine learning models from chapter 3.6 and generative models utilizing autoencoder approaches in chapter 3.7 fulfil KPI_BB6_2.

3.2.3 KPI_BB6_3 status

Chapter 3.3 is about selecting and evaluating the condition indicator of IGBT power modules serving for remaining useful life estimation. Chapter 3.5 deals with condition monitoring in lift applications. The chapter 3.6 deals with analysis of motor drives failures in the medical manipulator. Chapter 3.7 deals with audio anomaly detection. These chapters contribute to fulfilment of KPI_BB6_3.

3.2.4 KPI_BB6_4 status

This KPI is related with task 5.4 (section 4 – Automatic commissioning of motion control systems)

3.3 Condition monitoring of inverter power components (BUT)

In this section, the diagnostics system for inverter power components is presented. The system was developed thanks to the data obtain during the accelerated aging tests of selected IGBT modules under well-controlled conditions. This allowed simulation of specific type of thermal load of the module, ideally inducing a desired failure mode. Generally, the tests span throughout several days and the gathered data were processed in developed MATLAB library. Thus, the tools for data-based diagnostics were established and validated. Furthermore, the exact thermal behaviour of the IGBT power module was of concern. This task was approached by designing a thermal model. However, due to the nature of the IGBT power module, several challenges needed to be successfully addressed before obtaining all required data to properly model the thermal transient behaviour of the module. All proposed methods for condition monitoring and, essentially, estimation of remaining useful life (RUL) was designed to be implemented on an instrumentation level, with the ability to report the state of the IGBT power module to higher level systems.

3.3.1 Tech Overview

Condition monitoring includes two distinct approaches, specifically data-based and model-based. Data-based condition monitoring is a strategy in which the health and performance of the IGBT power module is assessed using data-driven techniques. The condition is then estimated by monitoring the course of information contained in one or several measurable quantities. Generally, when the end of the useful life of the monitored component is approaching, this information should react by changing the value or the course of a condition indicator. The data-based condition indicators are relying on the data analysis techniques, including statistical methods, machine learning methods, and tools of artificial intelligence to process the collected data. However, implementation of data-based condition monitoring comes with few challenges as well. Main concern lies in the tuning of the data-based condition indicators themselves, as these are developed based on the data obtained during the accelerated aging tests of the IGBT modules. Thus, it can be difficult to predict the behaviour of the trained condition indicator, e.g., when the actual operating conditions deviate from the tested ones. Furthermore, the scalability and the adaptability to different systems can be an issue, as it might potentially require generation of new datasets. Considering the wide range of possible operation conditions (e.g., temperature, current), the training process can be costly.

Another approach is based on the model of the IGBT power module. Here, the mathematical model is pre-requisite for the condition monitoring. The actual state of the IGBT, mapped by measuring selected quantities, is compared to the outputs of the model. The difference can be then used to assess the condition of the monitored system compared to the ideal state. However, in order to construct the mathematical model, it is necessary to properly understand the physics of the specific failure mode. Nonetheless, the benefit of properly constructed physics-based model is its capability to scale and adapt to different systems by reconfiguring some of its parameters.

The research presented in this deliverable focused on both approaches, where the framework for the development of both data-based and model-based condition indicators is designed. A key component in the framework is the system for acquiring data from accelerated aging tests of IGBT modules, shown in Figure 19. The system can also be configured to map the thermal transient properties of the IGBT modules, allowing the identification of the thermal model of the IGBT assembly. In the figure, major components of the data acquisition system can be seen. The system is built around cRIO, controlling the IGBT test conditions and data acquisition as well. Aiding the data acquisition is also a high-performance oscilloscope. Further, a thermal imager is used for capturing the course of temperature transients.

Additionally, the framework contains tools for the import of the collected data into the MATLAB environment and its further processing.

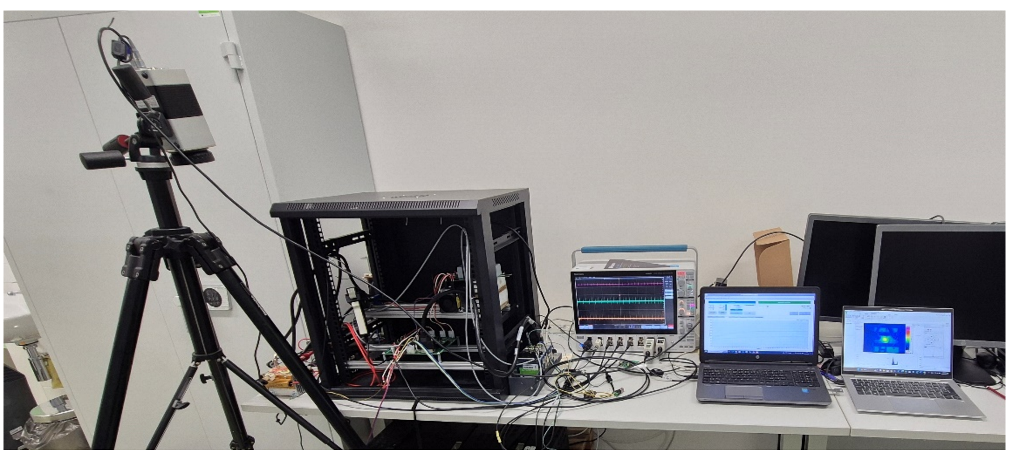


Figure 19: The system for data acquisition from accelerated aging tests and for mapping of the thermal transient properties of IGBT module

SW-034: *Power inverter transistor module health monitor*. A set of software tools usable for condition indicator methods development was created, accommodating both data- and model-based approaches. The developed tools are utilizing the structured data compatible with MATLAB Predictive Maintenance Toolbox.

3.3.2 Implementation aspects

The current implementation reflects the aim of the research on the development and validation of the methods for health monitoring. As it is necessary to obtain data describing the behaviour of the IGBT during its lifetime, a test stand allowing accelerated lifetime tests was conveniently used. The system is based on NI cRIO controlling the conditions of the test and measuring selected quantities courses during the switching states of the device under test (DUT). As the switching state is rather fast event, a high-performance acquisition system is required. In this case, a Tektronix MSO series 6 oscilloscope communicating with cRIO is integrated into the data acquisition chain. The measurement can be set up and monitored via laptop running LabVIEW Development ADE. The system architecture is shown in Figure 20.

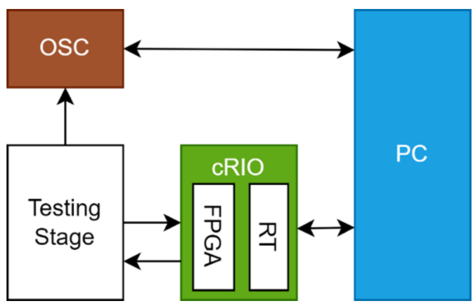


Figure 20: Architecture of data acquisition for accelerated aging tests of IGBT modules

Additionally, it was required to obtain data, which could be used for thermal model identification. The accelerated aging test setup needed to be appropriately adapted to accommodate thermal imager and new measurement setup. In this case, a step load is applied to the individual chips of the DUT and the temperature transients of other junctions are recorded. The adapted architecture is shown in Figure 21. The thermal imager communicates directly with the laptop, running appropriate software. Additionally, the DUT must be uncovered, to allow a clear sight of all chips. This requires a careful approach, as the IGBT modules are typically placed inside a polymer box sealed by silicone.

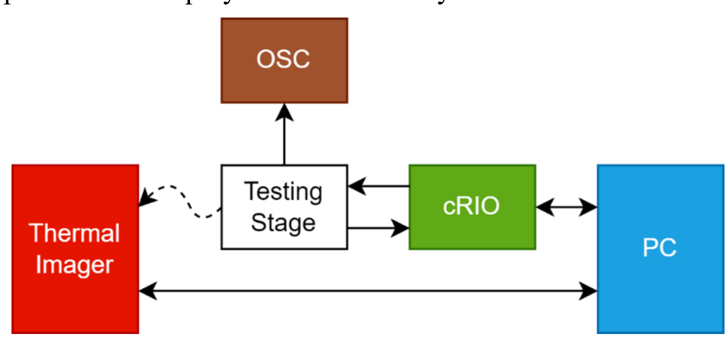


Figure 21: Architecture of data acquisition for thermal model identification

3.3.3 Results

Several accelerated lifetime tests were performed, and the data gathered were evaluated. Two condition indicators were evaluated, specifically V_{CEON}, t_{OFF}. It was shown that values of both condition indicators changed as the module neared end of its life. V_{CEON} is one of the better-known thermal sensitive parameters. When its value changes during constant load conditions during the accelerated aging test, it points to the degradation of the thermal coupling of the joint – the thermal resistance of the cooling path is increasing.

Based on the performed measurement, example of which is shown in Figure 22, it was possible to determine the prediction horizon of the condition indicators. Noticeable increase occurred about 300 minutes before the IGBT burn out. This is rather short amount of time; however, it is under the conditions of the accelerated aging and, consequently, in real operating conditions, this time can be by several factors longer. Additionally, if appropriate action of the supervisory system is taken, e.g., load reduction; the failure can be delayed even further.

In order to validate the ability of the condition indicator to estimate RUL, the gathered data were arbitrarily reduced, and the course of the condition indicator was appropriately extrapolated. This confirmed the determined prediction horizon. However, the specific values of both condition indicators depend on the load and ambient conditions.

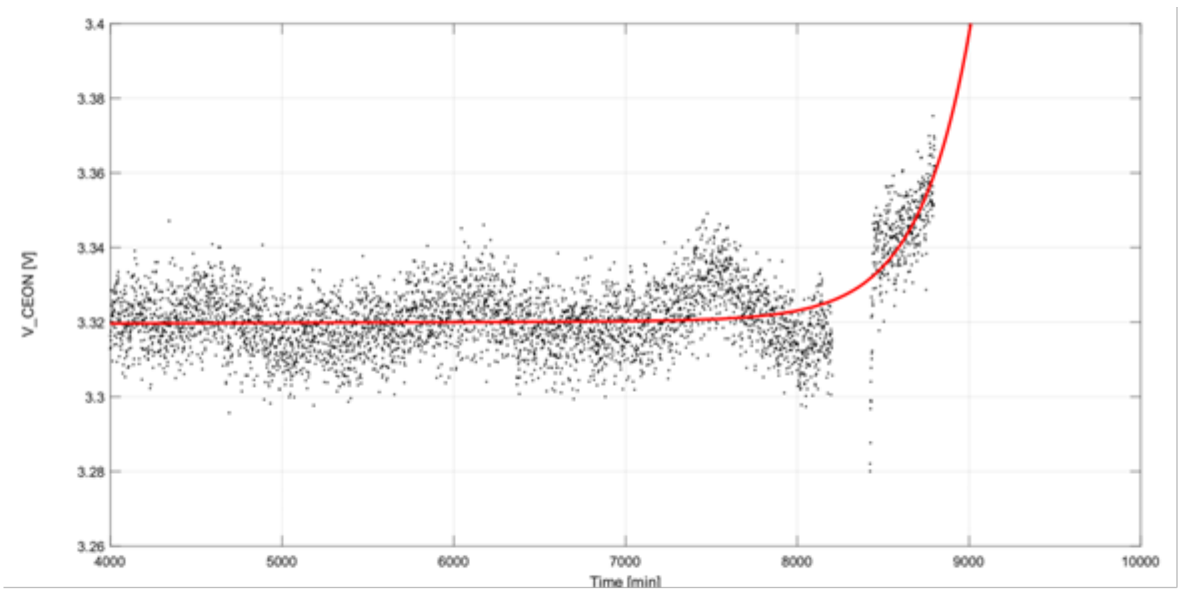


Figure 22: Course of V_{CEON} during the accelerated aging test

In order to estimate the thermal resistance independent on the operating conditions, thermal parameter estimator is proposed. The possible implementation of the thermal parameter estimator is shown in Figure 23. The thermal parameter estimator compares chip temperatures obtained via indirect measurement, i.e., utilizing dependencies of thermal sensitive parameters, and temperatures calculated by thermal model of the IGBT cooling structure. The purpose of the estimator is to adjust the parameters of the thermal model in such a way to minimize the difference between measured and calculated chips temperatures. The estimated thermal resistance can serve directly as a health status indicator of the transistor's cooling path.

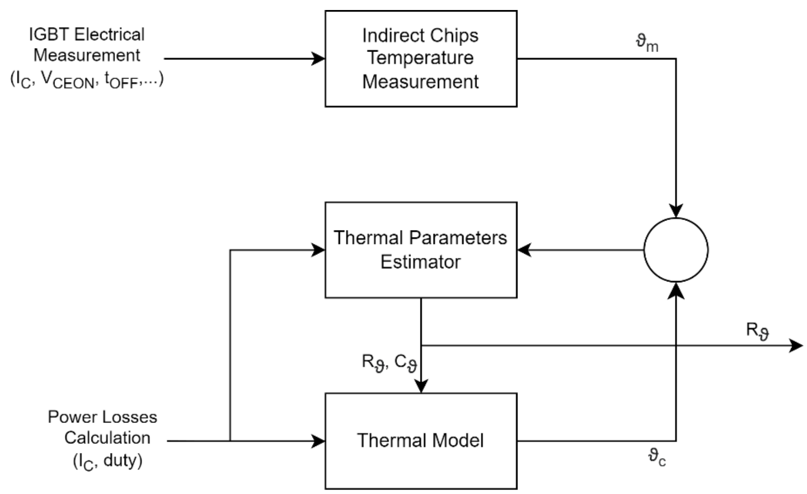


Figure 23: A scheme of the implementation of the IGBT module Thermal Parameter Estimator

To identify the mutual transfer functions in the IGBT thermal model, it is necessary to measure the thermal transients. The measurement relies on the utilization of the thermal imager, and a snapshot of the data acquisition is shown in Figure 24. The proper identification of the thermal model requires substantial number of measurements, and the data acquisition is ongoing.

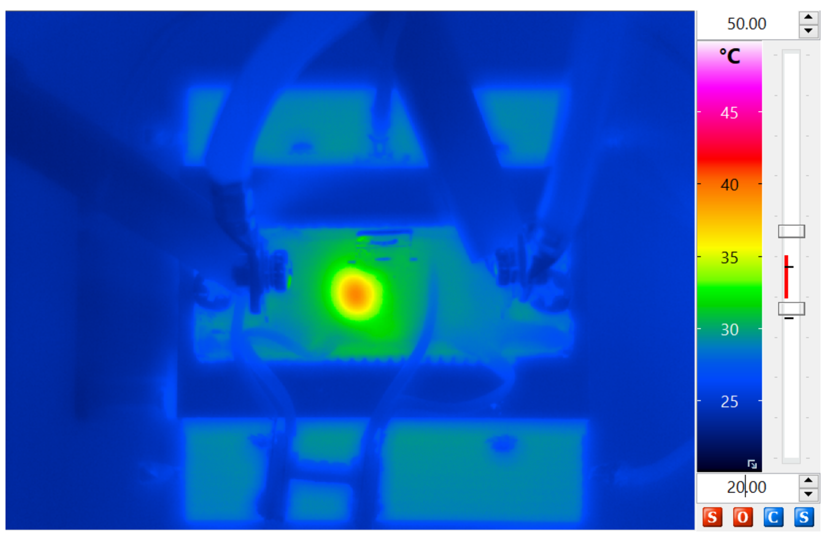


Figure 24: Snapshot of the measurement of thermal transients of an IGBT module

3.3.4 IMOCO4.E requirements

In this chapter, we discuss the fulfilment of the relevant requirements set on the system and described in detail in (D2.3): Overall requirements on IMOCO4.E reference framework.

R0 -L1: Predictive maintenance of inverter components provides the reliable message that some components prone to wear out deteriorate in performance and require maintenance. The developed condition indicators were validated and verified during the accelerated aging tests in the laboratory environment. The utilized setup allowed for evaluation of the capability of tested condition indicators to reliably predict the failure of the observed IGBT module. Thus, the setup has proved that it is possible to dependably estimate the final stages of the deterioration of the IGBT module. All measurements were conducted under well-controlled conditions, typically inducing a single failure mode. However, in real-life, the IGBT power module is rarely subjected to such well-defined type of operation and usually several failure modes are developing simultaneously. Therefore, additional mapping of varying operating modes, encapsuling a wider range of operating conditions and currents, is required to develop a method capable of monitoring and distinguishing the specific failure modes.

R139-D2.3-B6: Condition monitoring and predictive maintenance are capable of running on different layers depending on computational complexity and memory requirements. Considering the nature of the research, currently a real-time condition estimation was not integrated, as all tests were relying on offline processing. The architecture of the tests was set to accommodate rapid development, testing and validation of different condition indicators on gathered data sets. As such, the evaluation itself was typically performed with sufficient computational resources, generally well beyond the capabilities of contemporary embedded solutions. However, along with the performance testing of specific condition indicators, the system also enables definition of computational resources required for successful integration. Furthermore, depending on the type of the condition indicator used, auxiliary hardware resources might be required, as it might be necessary to measure additional quantities. Based on the obtained results, it is possible to define requirements on these measurement chains and processing hardware. The condition monitoring itself is expected to be integrated on lower layers of the system architecture, with the actual failure prognosis methods and the subsequent decision-making implemented on mid- to higher layers. This type of integration enables minimization of the data flow between layers, as the actual IGBT condition monitoring must be conducted consistently and quickly, but the condition evaluation can be performed comparatively slower or on the demand. The integration within the IMOCO4.E Reference framework was already proposed in (D5.4): Algorithms for condition monitoring, predictive maintenance, and self-commissioning of industrial motion control systems.

3.3.5 Capabilities & Limitations

The proposed environment is capable of developing, testing and validating different condition indicators and as such is closely tied to the capability of gathering data sets from accelerated aging tests of IGBT power modules. Furthermore, another modification of existing hardware was required for obtaining data for creation of thermal models of specific IGBT parts. However, this can be challenging, as it is necessary to obtain access directly to the modelled components of the IGBT, which are usually covered both in casing and by gel-like substances. Nonetheless, despite the difficulties, the developed environment for designing condition indicators proved to have the ability to both validate and evaluate the performance of selected RUL estimation methods.

Some of the tested condition indicators consistently demonstrated having the sensitivity to specific and tested types of loads of the IGBT, correctly evaluating RUL and, by inference, also accurately estimating the IGBT condition. System, leveraging the benefits of such prediction in its in-built diagnostics, is yet to be introduced into the commercial sphere. While the benefits of such solution in terms of the predictive diagnostics and on-demand prognosis for inverter power components might be obvious, the downside of the tests performed up to date is that only a single type of load was simulated. This type of loading does not accurately represent real-life conditions and the ability of the condition indicators to properly estimate the RUL under the arbitrary operating modes is yet to be explored.

3.3.6 Customizations & Adaptations

The presented method for condition monitoring and RUL estimation is leveraging the physical principles of semi-conductors and, as such, also the bond-wire construction of inverter power modules. Any customization thus depends on specific component parameters and its configuration. These parameters are, e.g., IGBT sizes and breakdown voltages, cooling path properties, number of IGBTs and diodes in the module. However, due to the nature of the developed system, there is a reserve in terms of the number of available data acquisition channels, thus an overhead exists allowing adaptation of larger power modules than tested so far. As the physical principles still apply, the developed methods could be also adapted to different IGBT sizes.

3.3.7 Methodology & Toolchains

The proposed framework for development and validation of condition indicators can serve in fact as a template for testing and development of already proposed indicators, but also new ones as well. The entire framework is facilitated in MATLAB environment. For deployment in industrial applications, MATLAB features a robust C/C++ code generator (MATLAB Coder), which can be conveniently used.

Currently, an integration of the tested methods into the use case/pilot was not considered, as the methods themselves were the aim of the research.

3.4 Lift diagnostics package for CatE device (BUT)

In this chapter, research and development on the set of diagnostics algorithms of a lift cabin is presented. The work has been done by BUT as a part of WP6 of the IMOCO4.E project. Major part of the work lies in the software package for calculating suitable condition indicators from the acceleration signal of the lift cabin during its movement. For successful algorithms development it is necessary to capture a set of the elevator acceleration data during its different operation and health conditions. Nevertheless, this procedure is quite time consuming (a lot of data must be acquired to create a complete and powerful training dataset), and hardly reachable (all possible operating modes and healthy states of a lift must be captured). From the aforementioned reasons, the work conducted in this WP gives a procedure how to extract the key condition indicators from acceleration signal and also a sample of the captured data and calculated features.

3.4.1 Tech Overview

The acceleration data of the elevator has been captured by the Smart Wireless Sensor (SWS), developed within work package WP3 (IMOCO catalog component nr. HW-022). Block schematic of the sensing system, containing sensor itself and wireless gateway, used for the data acquisition, can be seen in Figure 25.

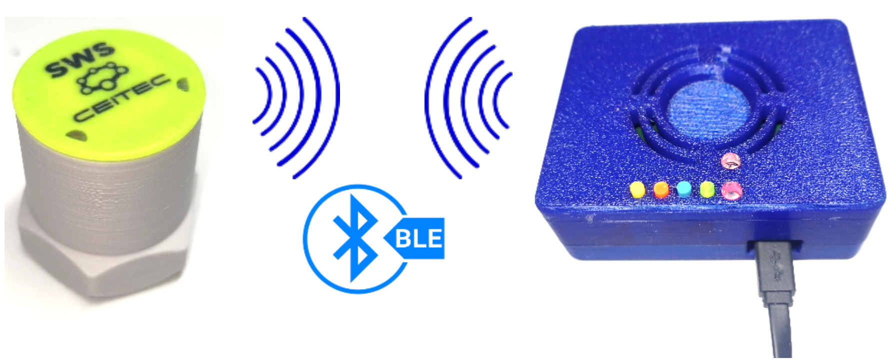


Figure 25: Smart Wireless Sensor and BLE gateway

Sensor is intended to be used in the Layer 1 and contains three MEMS accelerometers – the first for very precise single axis measurement in the high frequency range, the second for precise three axis measurement in the middle frequency range and the third for motion activation of the complete system. The sensor also contains a powerful microcontroller with ARM M4+ core running at 64 MHz with wireless transceiver and internal non-volatile memory for storing the data. Sensor is fully controlled through Bluetooth Low Energy, which is also used for transmission of captured vibration data. The power source for the sensor is provided by the internal Li-Pol battery pack with the capacity of 200 mAh, which ensures (together with motion activation feature) long operation time of the device.

For the lift diagnostics data acquisition, the sensor was located inside the elevator and measures all three perpendicular axes of the lift cabin. Only a small portion of the lift movement has been captured – a ride of duration of ca. 30 seconds, which in the measured lift corresponds to the ride from the ground floor to the third floor including opening and closing the door. A photo of the SWS inside the lift and a lift itself can be seen in the Figure 26.

Figure 26: SWS during data acquisition inside the lift

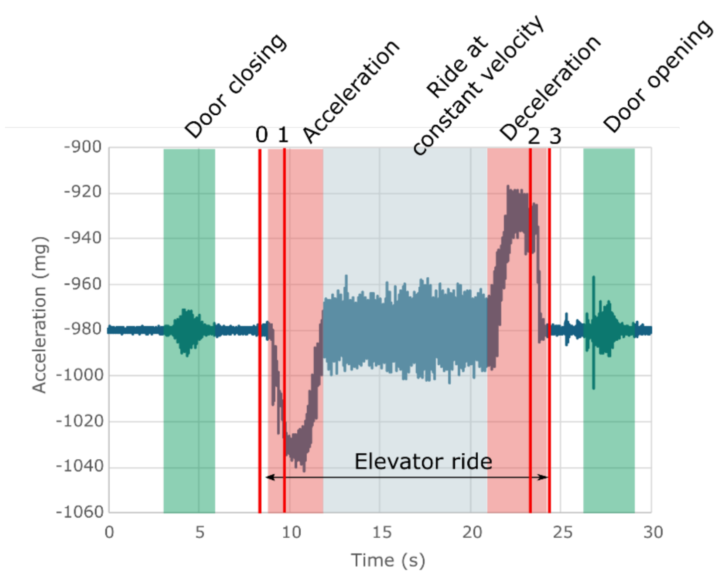


Figure 27: Lift acceleration ride profile

As soon as the data were captured, they were transferred via Bluetooth to the gateway and stored in the PC, where post-processing in LabVIEW application takes place.

Typical acceleration ride profile, acquired in the lift during its ride upstairs from the ground floor to the third floor, can be seen in the Figure 27.

This profile was captured inside the real elevator for personal use and all typical regions (represented by the filled rectangles) and boundaries (represented by the red lines) can be found in the data:

- acceleration signature representing doors movement (green areas),
- run-up and coast down of a cabin (red areas),
- ride of a cabin at a constant velocity – no breaking and no accelerating (blue area),
- boundary 0 - at least 0,5 s before commencement of door closing at the departure terminal floor,
- boundary 1 - 500 mm after commencement of lift motion from the departure terminal floor,
- boundary 2 - 500 mm before cessation of lift motion at the arrival terminal floor,
- boundary 3 - at least 0,5 s after completion of door opening, or cessation of lift motion, at the arrival terminal floor, whichever occurs last.

All the necessary features of a lift ride quality are calculated only if the lift is moving – areas represented by the door movement are excluded from the calculation. In fact, only the signal part between the lines 0-3 is taken into account.

There are two main approaches for the data evaluation and processing:

- According to (ISO18738, 2003) – data are analysed based on stages of lift movement like constant acceleration, non-constant acceleration etc. but they are not filtered according to human susceptibility to vibration.
- According to (ISO8041, 2017) and (ISO2631, 1997), where data are filtered according to human susceptibility to vibration but are analysed in whole without taking into account different stages of lift movement.

Both approaches were implemented in the LabVIEW application and are a base for IMOCO4.E component nr. SW-035 – Condition monitoring package for vibration monitoring of a lift cabin and diagnostics of a lift drive. Block schematic of signal processing and feature extraction can be seen in Figure 28.

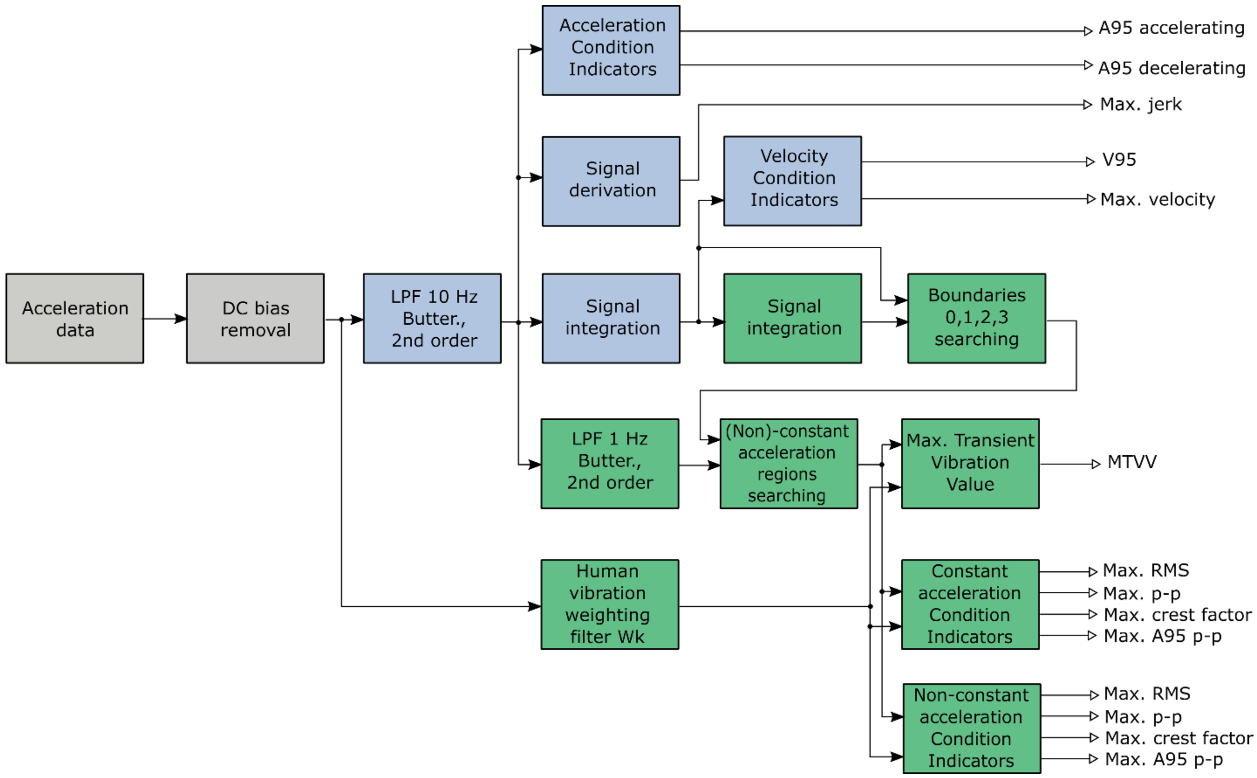


Figure 28: Signal processing chain in LabVIEW for lift ride quality evaluation

An approach based on the ISO18738:2008 standard is marked by the blue colour and appropriate condition indicators are shown in the top right part of the image. Condition indicators and their calculation based on the ISO8041 and ISO2631 are green-coloured and shown in the bottom right part of the block schematic.

3.4.2 Implementation aspects

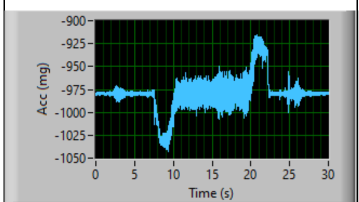
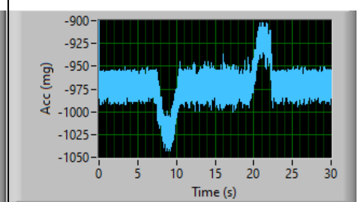
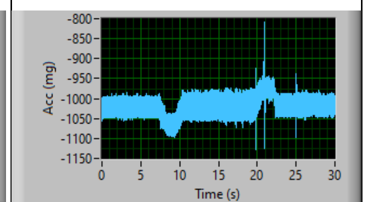
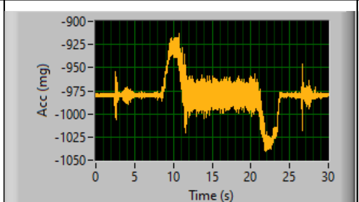
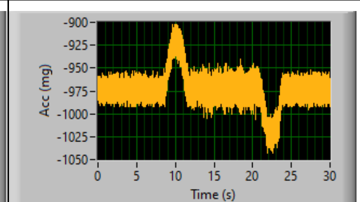
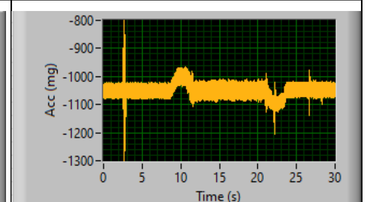
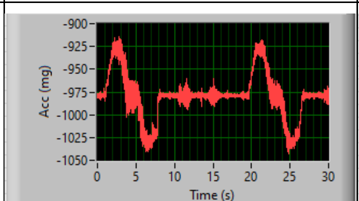
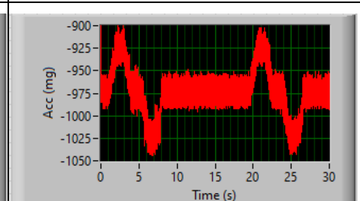
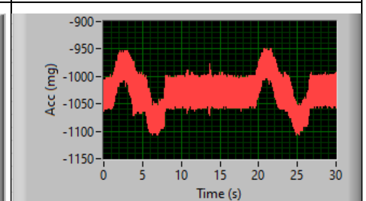
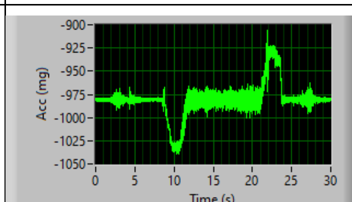
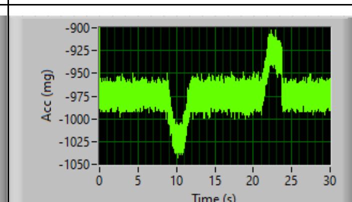
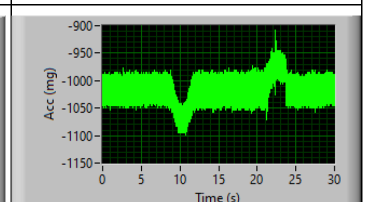
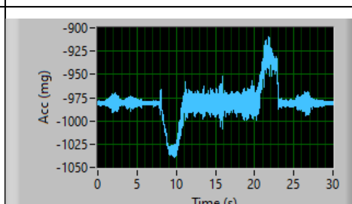
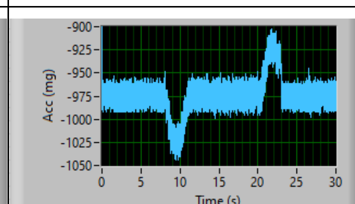
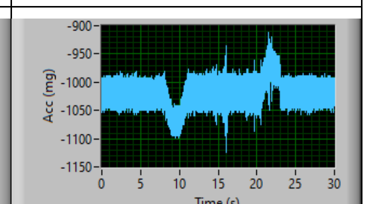
Merging the hardware component HW-022 and software component SW-035 leads into one functional entity (INT-095). Beside the basic functions, which must be implemented in the HW-022 component, and which must ensure its functionality at a low level, higher-level algorithms for diagnostics of a lift are implemented in the component as well.

The algorithms, based on the standards, were initially implemented in LabVIEW application. Tuning of the parameters and methodology has been done at this stage to obtain correct condition indicators. Final confirmation about successful data processing should be finished after the input dataset will be complete and will describe all operational and healthy states of the elevators. After that, procedures, coded in LabVIEW, will be simply re-generated into the C language and coded into the microcontroller inside the SWS. In addition, a simple neural network for condition indicators processing for lift health evaluation will be designed and implemented into the sensor as well.

3.4.3 Results

The example of the signals acquired by the different sensors during different rides in the different lifts can be found in the following Table 3. Sensor 1 represents triaxial middle range digital accelerometer, sensor 2 represents low performance, triaxial, but very low power accelerometer used for motion wakeup of the system and finally sensor 3 represents high dynamic range, high frequency range single axis accelerometer.

Table 3. Acceleration signals of a lift during different rides and conditions

Data desc.	Sensor 1	Sensor 2	Sensor 3
Lift C Up 0-3rd floor			
Lift C Down 3rd– 0 floor			
Lift C Down 3rd- 2nd – 1st floor			
Lift B Up 0-3rd floor			
Lift B Up 0-3rd floor			

According to the standards mentioned at the beginning of this chapter and according to the block schematic from Figure 28, following features have been calculated from the signals:

- In the regions with the *constant* acceleration:
  - Maximal RMS value of vibration acceleration
  - Maximal peak-peak value of vibration acceleration
  - Maximal crest factor of vibration acceleration
  - Maximal “A95” peak-peak value of vibration acceleration

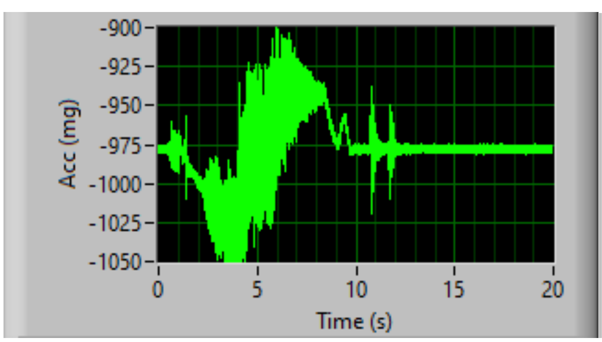
- In the regions with the *non-constant* acceleration:
    - o Maximal RMS value of vibration acceleration
    - o Maximal peak-peak value of vibration acceleration
    - o Maximal crest factor of vibration acceleration
    - o Maximal “A95” peak-peak value of vibration acceleration
- Maximal velocity of the filtered and integrated acceleration signal
- Velocity V95
- Maximal jerk of the filtered vibration signal
- Parameter A95 for acceleration of the elevator
- Parameter A95 for deceleration of the elevator

This set of features is calculated from both sensors the low frequency and the high frequency. The example of the calculated features from the low frequency sensor in the axis of elevator movement can be seen in the following table in Table 4. Please note, that the data expresses healthy state of the elevator in the small portion of the operating modes:

Table 4. Calculated condition indicators for the lift cabin

Data description	Constant acceleration				Non-constant acceleration									
	Max RMS	Max p-p	Max crest	Max A95	Max RMS	Max p-p	Max crest	Max A95	V95	A95 acc	A95 dec	Max. vel.	Max jerk	MTVV
	[m/s^2]	[m/s^2]	[1]	[m/s^2]	[m/s^2]	[m/s^2]	[1]	[m/s^2]	[m/s]	[m/s^2]	[m/s^2]	[m/s]	[m/s^3]	[m/s]
0-2nd, up, 2 person	0,04810	0,13168	2,78261	0,06810	0,04979	0,14664	4,17234	0,10675	0,99736	0,52543	-0,52857	0,99802	1,19505	0,06925
0-3rd, up, 1 person	0,04529	0,10911	2,73761	0,10099	0,03321	0,11825	3,85892	0,08414	0,99725	0,52011	-0,53214	0,99801	1,23863	0,07200
0-3rd, up, 1 person	0,04620	0,11168	2,74955	0,10843	0,04177	0,10565	3,16014	0,08767	0,99752	0,51867	-0,52827	0,99858	1,22477	0,06968
0-3rd, up, 3 person	0,04386	0,11785	3,24652	0,09235	0,04210	0,11965	4,00463	0,07730	0,99685	0,52548	-0,52680	0,99768	1,24894	0,07357
0-3rd, up, 5 person	0,04534	0,12957	3,43607	0,07206	0,04176	0,09588	3,60951	0,06085	0,99656	0,52792	-0,52358	0,99703	0,97298	0,05481
3rd-0, down, 1 person	0,05346	0,26318	4,98670	0,26318	0,03901	0,13353	3,62203	0,08784	0,99583	0,53112	-0,52906	0,99653	0,92364	0,06281
3rd-0, down, 1 person	0,06049	0,27732	4,97736	0,27732	0,04664	0,10482	3,08912	0,08104	0,99536	0,52718	-0,52765	0,99643	0,95378	0,06628
3rd-0, down, 1 person	0,05116	0,24648	4,81825	0,24648	0,04397	0,11846	3,31521	0,08883	0,99590	0,53566	-0,52819	0,99691	0,95985	0,05987
3rd-0, down, 1 person	0,05415	0,24750	4,57055	0,24750	0,03714	0,14165	3,81414	0,09514	0,99555	0,54092	-0,54783	0,99712	0,96151	0,06501
3rd-2nd-1st, down, 1 person	0,05140	0,24516	4,77006	0,24516	0,04719	0,09911	4,55683	0,01966	0,99601	0,54022	-0,52870	0,99601	0,94496	0,06093
3rd-2nd-1st, down, 1 person	0,04956	0,23377	4,71683	0,23377	0,03321	0,12369	5,32475	0,02797	0,99617	0,53502	-0,52246	0,99622	0,95866	0,06092

Another dataset was acquired on the different elevator, where it was possible to artificially induce fault frequency into the motor driving algorithm. The fault signal simulates bearing mechanical fault of the propulsion system. Time waveform, acquired at the lift with and without induced fault, can be seen in the Figure 29. There can be seen a clear difference in the data. Corresponding data are located in the Table 5 – first and last row are shown in the previous Figure 29.



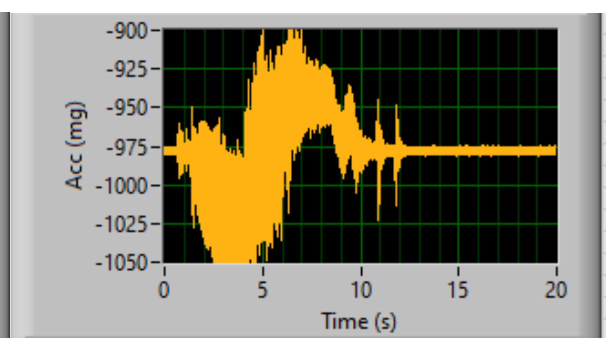


Figure 29: Elevator data without (on the left) and with (on the right) additional noise and induced fault

Selected features, representing acceleration signal from the previous table, are shown in Figure 30. It can be seen that the features with induced fault have a higher magnitude than the features for the healthy state of the elevator.

Table 5. Calculated condition indicators for the lift cabin with/without induced fault

Data description	Constant acceleration				Non-constant acceleration									
	Max RMS	Max p-p	Max crest	Max A95	Max RMS	Max p-p	Max crest	Max A95	V95	A95 acc	A95 dec	Max. vel.	Max jerk	MTVV
	[m/s^2]	[m/s^2]	[1]	[m/s^2]	[m/s^2]	[m/s^2]	[1]	[m/s^2]	[m/s]	[m/s^2]	[m/s^2]	[m/s]	[m/s^3]	[m/s]
0-1st, up, no noise	0,04974	0,18311	3,68119	0,14455	0,04994	0,13667	5,09529	0,13667	0,00000	0,58961	-0,40989	1,02670	0,56680	0,07704
0-1st, up, no noise	0,05421	0,17810	3,71448	0,13205	0,03655	0,09928	6,67297	0,02653	0,00000	0,58470	-0,41749	1,02649	0,57718	0,07552
0-1st, up, noise 1 RPM	0,10584	0,33626	3,25720	0,28825	0,12297	0,30501	2,67226	0,30501	0,00000	0,61081	-0,51929	1,02673	0,95976	0,13035
0-1st, up, noise 1,5 RPM	0,15137	0,41334	3,30257	0,39914	0,14824	0,35066	5,39861	0,35066	0,00000	0,64089	-0,58664	1,02706	1,36410	0,18654
0-1st, up, noise 0,5 RPM	0,06931	0,22820	3,43375	0,18060	0,06406	0,16844	3,46789	0,16844	0,00000	0,59724	-0,47002	1,02613	0,71670	0,08942
0-1st, up, noise 1 RPM + 40 Hz	0,07145	0,22011	3,31194	0,13443	0,05279	0,13854	2,67652	0,13714	0,00000	0,60040	-0,43394	1,02613	0,56673	0,08602
0-1st, up, noise 1 RPM + 5 Hz	0,12569	0,28653	2,67846	0,28653	0,11152	0,20338	4,60157	0,01272	0,00000	0,71031	-0,52068	1,02956	0,66195	0,13507

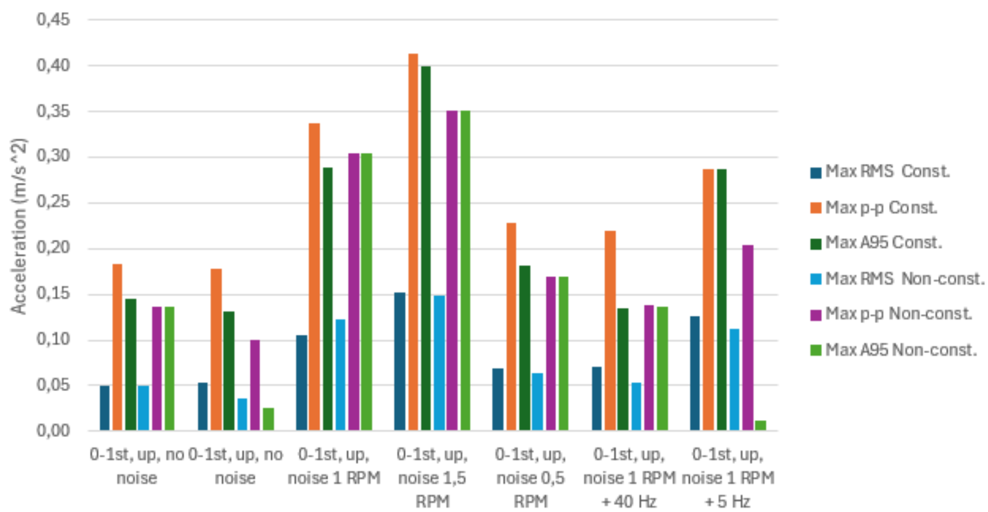


Figure 30: Acceleration signal features for the lift with/without induced fault

A set of these features will be used as an input for the artificial intelligence algorithm (probably dense neural network), which will decide about the healthy state of the lift.

3.4.4 IMOCO4.E requirements and KPIs

In this chapter, fulfilment of the relevant requirements set on the system and described in detail in the previous deliverables are shortly discussed:

Req-D2.3-L1-1 - Wireless connectivity of sensors for mechanical manifestations acquisition. The Smart Wireless Sensor uses Bluetooth Low Energy (BLE) communication interface for measured data transfer as well as for the setup of the sensor itself.

Req-D2.3-L1-2 - Sensors provide sufficient diagnostic information to evaluate their weaknesses or damages. Sensor contains three independent acceleration sensing elements. The information provided by the elements is sufficient to evaluate the weakness of the elements and trustworthiness of the acquired mechanical vibration information.

Req-D2.3-L1-3- Implemented sensors communication interface/protocol allows simultaneous connection of multiple devices. Each sensor uses its own wireless network identification – it is possible to use more independent networks in case of isolated communication areas requirement. Nevertheless, multiple sensors can be connected to the common Bluetooth network and communicate within one wireless network using TDMA/FDMA approach and unique ID of the particular sensor.

Req-D2.3-B3-1 - Wireless communication interface for sensors not located in the closed vicinity of centralized controller. Requirement is fulfilled, for the details please see previous *Req-D2.3-L1-1*.

Req-D2.3-B3-2 - Low energy communication interface for sensors with at least 500 kbit/s burst data rate and operating range with at least few tens of meters. Network star topology is preferred. SWS is equipped with the BLE communication interface allowing communication at a high speed. As a default value, 1 Mbit/s has been tested and is currently used in the sensor, but the hardware and software of the sensor also supports doubled speed of 2 Mbit/s.

Req-D2.3-B3-3 - Power supply for expected lifetime operation integrated in the sensor, e.g., battery power, wireless power or energy harvesting depending on power requirements of the device. SWS contains built-in battery pack with the capacity of 200 mAh, which ensures ca. 8 months of operation in sleep mode or ca. 20 hrs of full performance operating mode. SWS also contain the Qi charging circuits for fast wireless charging of the battery pack.

Req-D2.3-B3-4 - Sensors analyse their internal performance parameters to evaluate reliability of output data and report this information. This requirement is partially fulfilled and is still under development. However, SWS contains three independent accelerometers enabling reliability analysis of the sensed vibration information.

3.4.5 Capabilities & Limitations (including USP, strengths & weaknesses)

Smart Wireless Sensor is considered as Computing at the Edge device (CatE). Currently, the sensor acquires vibration data of a lift, stores them into the internal FLASH memory and transfers them using BLE into the PC through wireless Bluetooth gateway. Signal processing, key condition indicators extraction methods and procedures are currently being optimized. The system is capable to extract the most common features necessary for evaluation of a lift health, but more data, mainly describing faulty conditions, are strongly required. After the correctness of the selection and extraction of the condition indicators will be confirmed, an artificial intelligence algorithm design shall take place. Afterwards, proposed algorithm will be optimized for CatE device and implemented directly into the SWS.

3.4.6 Customizations & Adaptations (including possible modifications and extensions)

Thanks to the processing chain implemented in the LabVIEW environment, there is still a space for any customization and improvement of the designed methodology. All modifications can be later on transferred into the algorithms built-in the sensor itself.

3.4.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

Development of the methods has been utilized in the LabVIEW environment, where basic signal processing, automated recognition of the signal shapes and the key condition indicators extraction and calculation have been implemented and optimized.

Since sensor uses standardized BLE interface, integration into the higher IMOCO4.E components, such as Use Case 1, lies only in the implementation of the proprietary wireless communication protocol of the sensor into the any Bluetooth receiver/transmitter of the upper entity.

3.5 Lift Condition Monitoring: Utilizing Least Square Regression on Rigid Models (WEG, UNIBS)

3.5.1 Tech Overview

In recent years, there has been an increasing need for industrial devices to boost intelligence and connectivity. This demand arises from the drive to enhance overall performance and efficiency across diverse industrial applications. In motion control, controllers must effectively monitor, adapt to variations, and respond promptly to changes in system dynamics. Algorithms dedicated to estimating primary system dynamics play a vital role in diagnostics and controller autotuning, ensuring continuous fine-tuning and providing valuable insights into plant variations. Simplicity in computational design offers efficiency, reducing overhead and hardware demands, making these algorithms ideal for PLC-based controllers in diverse industrial devices. Additionally, their effectiveness ensures accurate system dynamics estimation, enhancing diagnostic and autotuning precision. Incorporating monitoring and predictive maintenance at the instrumentation level further optimizes operational efficiency, prolongs system lifespan, and boosts reliability in motion control systems.

3.5.2 Implementation aspects

High-speed/high-rise elevators aim to maximize speed while ensuring passenger comfort and addressing cabin oscillations in the process. Nonlinear dynamics involve rope stiffness/damping variations, a position-dependent resistance force, and instability from rope mass changes.

An accurate model employs lumped elements, with variable mass at the center and viscoelastic components at the rope ends. Since the controller and the motion law are designed to avoid the excitation of high-order dynamics, the elasticity effects are negligible. Thus, online estimation neglects them and focuses on low-frequency dynamics. The monitoring of the viscous-elastic components can be done by means of accelerometers (see Section 3.4).

The resulting model is a nonlinear equation derived from the torque balance involving motor torque, passenger mass, motor-cabin speed transmission ratio, gravity acceleration, cabin mass, counterweight mass, rope linear density, total inertia of the motor, and viscous/Coulomb friction coefficients. The torque balance is linear on model parameters, facilitating the application of Least Square methods.

The least-square algorithm highlights a dichotomy in its application within edge computing environments. Firstly, the intrinsic complexity of the least-square algorithm renders it impractical for execution on edge devices. This complexity arises from the algorithm's computational demands, which exceed the processing capabilities typically available at the edge. On the contrary, the estimation process governed by this algorithm does not necessitate a high frequency of data sampling since the (elastic) high-frequencies components have a negligible impact.

In light of these considerations, the operational framework involves transmitting data from the edge controller to cloud-based infrastructure, specifically layer 4, at intervals of 8 milliseconds. This approach enables utilizing cloud computing resources to perform least-square estimation periodically. By offloading the computational burden to the cloud, the algorithm can be executed efficiently without the constraints imposed by edge device limitations.

The outcomes derived from the least-square estimation process have significant applications in two primary domains. Firstly, they facilitate condition monitoring, wherein deviations from established norms' estimated parameters may signal plant degradation. This application is crucial for condition monitoring. Secondly, analysing unexplained components (correlation reality-model) within the torque measurements provides insights into anomalies such as misalignment compared to standard behavioural models. An example of such an anomaly is excessive rope slipping, which deviates from expected performance parameters.

Thus, the variation of the estimated parameters and the unexplained torque are used synthetic indicators that can be used alone or as input to the predictive maintenance module.

3.5.3 Results

Validation of this synthetic index utilizes the Amesim digital twin, conducting simulations from nominal conditions and introducing faults in each iteration.

The utilization of a digital twin serves as a pivotal strategy for validating the estimation algorithm, primarily due to the inherent difficulties in acquiring an adequate volume of data related to failures within actual plant operations. The digital twin enables the thorough testing and refinement of the algorithm under various failure scenarios that might be rare or too costly to replicate in the physical world.

Figure 31 depicts the correlation evolution between data and the model under various rope slipping coefficients. Notably, the correlation diminishes significantly as slipping becomes pronounced, as illustrated in Figure 32.

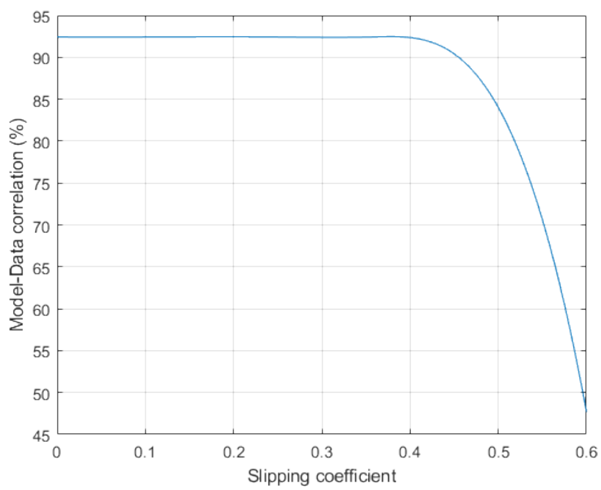


Figure 31: Correlation between data and model in a test campaign with aggravated slipping coefficients

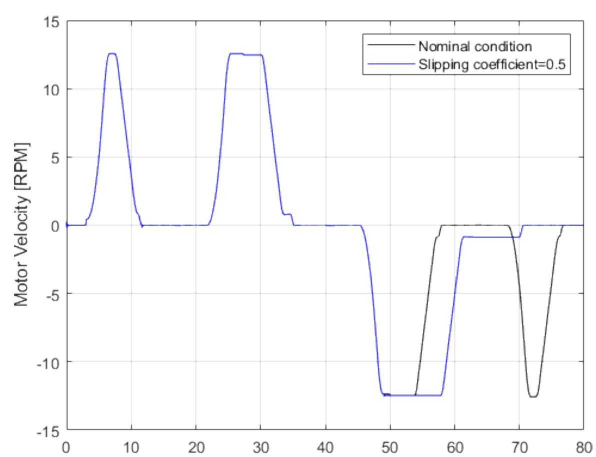


Figure 32: Velocity trend under nominal conditions (black) and with pronounced slipping

3.5.4 IMOCO4.E KPIs and requirements

The requirements of (D5.2) which apply to this technology are:

- [R010-D5.1-L2-sw] The Virtual Commissioning system will allow automatic testing of controller and plant model code: DONE. The algorithm continuously monitors the lift model's friction and inertia variations; the tracking error monitors the controller performance.
- [R168-D-UC1-1] All the modules integrated in UC1 must provide testing software in digital-twin or HIL testbed: DONE. The lift digital twin has been implemented in MATLAB/Simulink and AMESim, replicating the lift dynamics, the controlled motion law, the sensor positions, and lift logic.
- [R169-D5.1-UC1-2] All the real-time modules integrated in UC1 has to run considering the task-sample time available on the drive (i.e., routine must work considering 1 ms or 8 ms cycle time): DONE. The industrial drive uploads data at 8 ms.
- [R171-D5.1-UC1-4] Modules integrated with UC1 must communicate with MATLAB/Simulink. Real-time modules integrated with UC1 need to be compatible with IEC611311-3 Structured Text standard. A MATLAB/Simulink copy should be provided: DONE. The module has been designed in MATLAB/Simulink.

3.5.5 Capabilities & Limitations (including USP, strengths & weaknesses)

Capabilities

- Condition Monitoring for Lift Applications: A pivotal capability of our system lies in the condition monitoring of lift applications, where it leverages estimated parameters and the analysis of unexplained torque. By establishing a correlation model that accurately reflects reality, the system utilizes these insights as a synthetic index to monitor operational status comprehensively. This approach facilitates early detection of potential issues and enhances maintenance strategies by providing a detailed understanding of the lift's performance over time.
- Minimal Edge Computation Effort: Another significant strength of our system is its efficient design, ensuring almost zero computational effort is required at the edge. This efficiency is achieved by offloading complex computational tasks to the cloud, such as the least-square estimation. As a result, the edge devices can operate with minimal processing load, thereby reducing energy consumption and extending their operational lifespan while maintaining real-time data transmission capabilities.

Limitations

- Lack of High-Frequency Dynamics Monitoring: Despite its strengths, the system does present a limitation in its ability to monitor high-frequency dynamics. The current setup, focused primarily on leveraging estimated parameters and unexplained torque for condition monitoring, cannot detect rapid dynamic changes. The integration of additional sensors, such as accelerometers, is needed if high-frequency dynamics monitoring is required.

3.5.6 Customizations & Adaptations (including possible modifications and extensions)

Possible Modifications and extensions

1. **Integration of Additional Sensors**: the system can be modified to integrate additional types of sensors, such as accelerometers, to overcome the limitation of monitoring high-frequency dynamics.
2. **Advanced Data Analytics**: The system's data processing capabilities can be expanded by incorporating classification techniques, providing deeper insights into the data collected, and enabling predictive maintenance by identifying potential issues before they lead to failures.

3.5.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

The toolchain includes a digital twin of the plant in AMESim, the replication of the controller in MATLAB/Simulink, and the use of the digital twin to generate virtual failures to validate the algorithm. Each component is vital in simulating real-world conditions, facilitating thorough testing, and refining the system's capabilities.

While the toolchain provides extensive capabilities for simulation and testing, there are inherent limitations to be considered. The accuracy of the digital twin and controller models depends on the fidelity of the simulation tools and the completeness of the data used to create them. Additionally, the simulation environment may not capture all the nuances of real-world operations, particularly those influenced by unpredictable external factors.

3.6 Predictive Maintenance of Driving Motor Geometry (PEN, PMS)

3.6.1 Tech Overview

In order to enable continuous monitoring and also detect possible component degradation that can lead to a component failure for the robotic manipulator of Pilot 4, we employ two sources of data: 1) Motion Trace Data 2) Machine Log Event Data. The data collection and management of these data sources (software component SW-105) is described in deliverable (D5.3). The work described here implements part of SW-037.

Motion trace data are sensor measurements extracted from and around the motor drive that controls the movement of the device during its use. The sensor measurements include the motor drive current that is measured at 500 Hz during the movement of the C-arc, as well as motion sensors that measure the velocity of movement for the different parts of the C-arc and motor drives. In our analysis, we have explored the following failure scenarios:

- **Friction wheel slip:** The motor drive is moving the medical robot through a rotating friction wheel on a flat rail. Slip of the friction wheel can be caused by wear-out of the friction wheel, inappropriate installation and also motor drive failures.
- **Motor drive failure:** Another common failure mode for motor drives are stator coils failure/shorts and bearing failure (See e.g., (Rodriguez et al., 2015) or (Nayar, 2019)). As suggested by the relevant literature; we have explored the potential utility of signal processing techniques like the Fast Fourier Transform (FFT), Short-Time Fourier Transform (STFT) and Continuous Wavelet Transform (CWT) methods applied on the motor current measurement to identify signatures for motor drive or bearing failures.

3.6.2 Friction Wheel Slip: implementation aspects and results

In order to measure the friction wheel, slip we employ measurements from two sensors:

- Actual Motor Drive Velocity (AMDV): This is the velocity of the motor drive movement.
- End Effector Velocity (EEV): This is the velocity of the C-arm movement.

During normal operation of the C-arm movement, these two values will be almost the same. This is because the movement of the motor drive is transmitted to the C-arm using a friction wheel. When the AMDV and EEV values start to differ, then this indicates that friction wheel slip is occurring. The difference between AMDV/EEV is expected to increase during the lifetime of a system, as the friction wheel will wear off. It should be noted that slip can both increase and reduce this factor; if the slip is in the direction of the velocity or opposite (e.g., when the rails have small inclinations).

We performed a set of controlled experiments in order to identify the difference in these values that is associated with problematic behaviour of the motor drive. More precisely, we conducted controlled experiments with the following configurations:

- We assembled the motor drive using different levels of pre-tension on the friction wheel. This meant that the friction wheel did not have proper contact with the rails.
- We installed the motor drive with and without cleaning properly the rails. This meant that the friction level was different.

By cleaning the rails, we did not observe significant differences in the scale factor (AMDV/EEV), however, changing the levels of the pre-tension spring did create some observable differences in the values of (AMDV/EEV) as well as visible friction wheel slip during the movement of the motor drive. The calculated values of scale factor (AMDV/EEV) are shown in Figure 33. We can observe that the scale factor clearly increases as the pretentions spring level decreases. We can employ the results of this analysis for continuous monitoring of the difference of AMDV/EEV values to detect early occurrences of friction wheel slip and resolve the issue before it becomes severe.

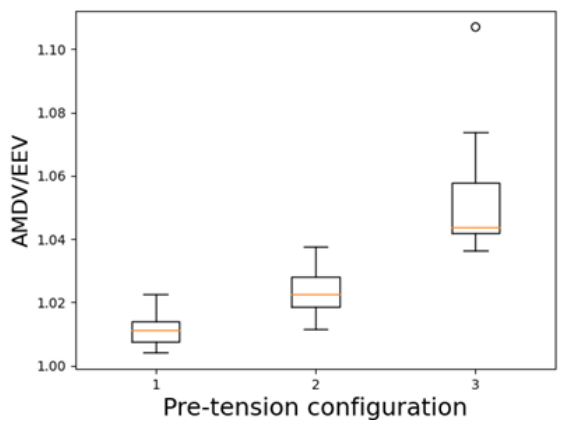


Figure 33: Scale factor (AMDVV/EEV) calculated for different levels of pretension for the pretension spring. Pre-tenson configuration=1 corresponds to the normal configuration, while pre-tenson configuration=2 and 3 correspond to less pretension applied than needed for proper contact with the rails.

3.6.3 Motor Drive Failures: implementation aspects and results

In the research literature it is common to detect motor failures using motor current analysis (see e.g., (Rodriguez et al., 2015)). This is a sensor measurement that is available to our Pilot 4 device and thus we have explored its usefulness for early detection of motor drive failures. In our analysis, we have used a relatively stationary part of the motor current signal and investigated with signal processing methods like: FFT, STFT and wavelet transform. (Nayar, 2019) provided a description about the (dis)advantages of these different methods.

Cited: 'One thing to note about these frequencies is that they are always a function of the central frequency. Further, the central frequency is a function of the motor speed and pole pairs, typically the peak with the highest amplitude. Most of the results reported are from experiments carried out at constant speeds. The amount of material present on variable speed drivers is minimal due to the relatively recent advent of wavelet transform. Using wavelet transform gives the capability of deriving an algorithm that can pick up faulty frequencies even at variable speeds.

It is clear that FFT are good transforms to map a time signal into the frequency domain only if the signal is a stationary signal. When stationarity is not guaranteed, wavelet transforms prove to be more useful.

Now, the first idea that comes to someone's mind when this problem is discussed (FFT with non-stationary signal) is why don't we perform multiple FFTs for small time periods. This is a logical thought, and this approach has been used actively (STFT) for a while until wavelet transforms were discovered. It is understandable to conclude that the shorter the time set, the better will be the results, as we don't know at the frequency the results are changing to and from. So, the higher the accuracy, the better this method works. But there is a drawback. We can never get a full idea of frequency and time. If we want a high resolution in frequency the resolution in time is compromised and vice versa. Hence, we need to decide on an optimized time interval to get a good resolution in time and frequency. STFTs use square windows of time and frequency. Wavelet transform is an upgrade to this approach. Wavelet transform gives you good resolution in time for high frequencies and good resolution of frequencies for low frequencies which proves better than a square window size in STFT.'

CWT suffers from less detail on the time scale at low frequencies, as the wavelet function is stretched out (a convolution is performed between the wavelet and the signal). Furthermore, the detail in the frequencies is reduced with high(er) frequencies. Within a CWT scaleogram with linear y-scale, like Figure 4, this will lead to horizontal blurring at low frequencies and vertical stretching at high frequencies. Hence, with CWT it is more difficult to differentiate between e.g., harmonic 9 and 10 than between harmonic 1 and 2, although the absolute difference between both sets of harmonics is identical.

The results of the FFT, STFT and wavelet analysis are shown in Figure 34 to *Figure* 36. The original current input signal was not filtered. The sampling rate was 500 Hz. The motor frequency at normal speed (150 mm/s) was 27.4 Hz.

According to (Nayar, 2019) and (Lee & Hur, 2017), stator inter-turn failures of permanent magnet BLDC (Brushless Direct Current) motors are visible in the line current of the motor in the occurrence/amplitude of the odd harmonics of the central frequency, and most specifically the third harmonic. The definition of the central frequency is somewhat vague, as being the strongest frequency in the signal; it is not clear whether this will be the rpm of the motor, or the rpm x pole pairs. Theory and practice provide the following equation:

$f = f_c(2n \pm 1)$, where n=1,2,3... and f_c = central frequency

The harmonics are also visible in the healthy signal but become stronger in case of failure or occur more often. The strongest signal in the STFT analysis is around 80 Hz, which would correspond to 3 times the motor frequency. The question is whether the central frequency should be the motor frequency or this 80 Hz signal. In the first case, the third harmonic is around 80 Hz, in the second is it around 240 Hz and only just below the Nyquist frequency. Within the FFT analysis, several harmonics of the motor speed can be distinguished and there is also a small peak visible around 240 Hz. The wavelet analyses (*Figure* 36) show very similar results in case of similar y-scales; however, the higher frequencies are pretty blurry.

Issues regarding motor current analysis:

- Current sampling frequency is relatively low (500 Hz), which results in a low Nyquist frequency. However, increasing the sampling frequency may not lead to significant improvements, as the signals are mainly of non-stationary nature and the CWT is not expected to benefit much from an increased sampling rate. The quality of FFT mainly benefits from a long stationary signal and the quality of STFT from a slowly changing signal.

- For FFT (Fast Fourier Transform) analysis, a stationary signal is needed. A longer stationary sample will result in improved frequency resolution of the FFT.
- STFT (Short Term Fourier Transform) can be used for non-stationary signals, but there a trade-off must be made between time and frequency resolution. Sudden frequency changes require reduced window sizes, which reduced the frequency resolution of the analysis.
- Wavelet analysis could prove a better option in case of non-stationary signals; this has a high frequency/low time detail at low frequencies and vice versa.

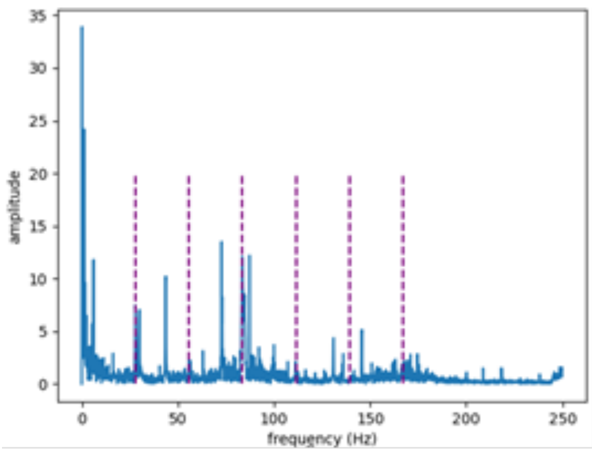


Figure 34: FFT analysis of stationary part of motor current signal (3.5s)

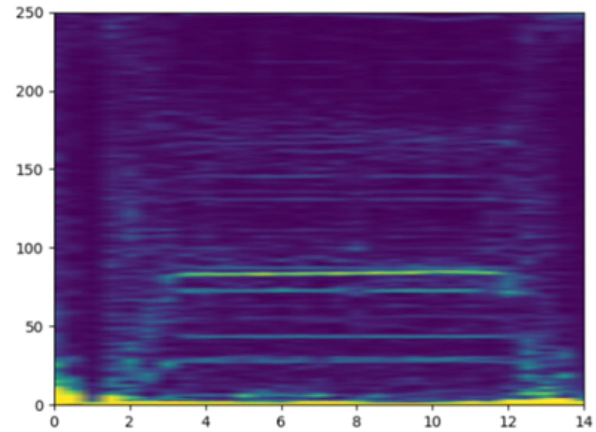


Figure 35: STFT of same signal including current ramp-up and ramp-down

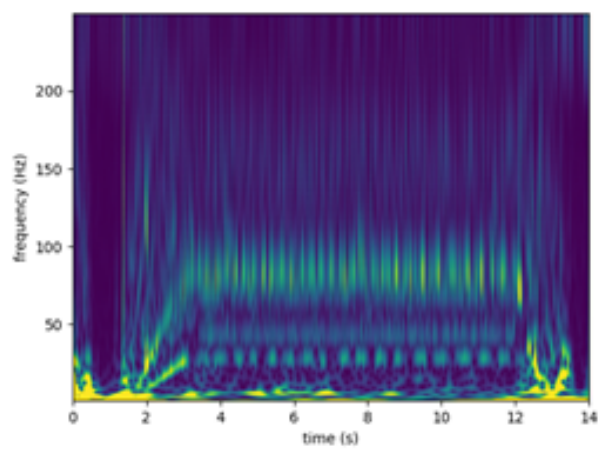


Figure 36: CWT transform using the morlet2 wavelet

- A motor with inter-turn failure is needed to prove whether the parameter changes with respect to a healthy motor.

3.6.4 IMOCO4.E requirements and KPIs

The work described in 3.6 addresses the following Pilot 4 requirements.

ID	Requirement	Priority	Verify
R134-D5.1-P4-1	Smart control algorithms, AI-components and digital twin models may use additional data or sensory input interfaces to train the model, however after completion it shall only make use of existing data and interfaces of the brown field system.	S	I
R135-D5.1-P4-2	All smart control algorithms, AI-components and digital twin models shall be testable in both simulation (e.g., by means of digital twins) and deployed on the physical target.	M	T
R137-D5.1-P4-4	Real-time (digital twin) models or algorithms require at maximum a sample rate of 500Hz.	S	D
R138-D5.1-P4-5	All smart control algorithms, AI-components and digital twin models shall be compatible and/or configurable/tuneable for different variations of similar system / robot.	S	
R140-D5.1-P4-7	If a (digital twin) model or algorithm is applicable to multiple layers (e.g., for real-time deployment and for condition monitoring) it will allow for easy adaptability/re-use across by for instance selection of variants of differing abstraction levels/complexity.	C	I
R141-D5.1-P4-8	All smart control algorithms, AI-components and digital twin models that are intended for real-time deployment shall be compatible with code generation from MATLAB / Simulink.	M	
R142-D5.1-P4-9	Any smart control algorithms, AI-components and digital twin models shall not adversely affect the safety of the system.	M	

3.6.5 Capabilities & Limitations (including USP, strengths & weaknesses)

From our analysis we were able to define appropriate threshold values that enable the early detection of friction wheel slip using AMDV and EEV sensor values. These results have been combined with the information and machine learning models, described in section 7 of this deliverable in order to enable continuous monitoring of the robotic manipulator of Pilot 4 and provide early detection of component failures. Low measurement frequency and lack of availability of motor drives with relevant motor drive failures, prevent us from deriving stronger conclusions from motor current analysis.

3.6.6 Customizations & Adaptations (including possible modifications and extensions)

Not applicable.

3.6.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

The analysis of the experiments described in 3.6.2 and 3.6.3 used the data from a test system's motion trace files directly, and was performed using custom made MATLAB scripts, independent of the Pilot 4 data and modelling pipeline. The results of the friction wheel experiments were used in the modelling pipeline. This pipeline uses the same motion trace data, after it has been loaded into the Vertica database.

3.7 Audio anomaly detection [CYBERTRON]

3.7.1 Tech Overview

Audio anomaly detection has many natural applications and has been studied within diverse research areas and application domains such as audio surveillance, machine-condition inspection, and fault diagnosis. Anomaly detection techniques can be categorized as supervised anomaly detection, semi-supervised anomaly detection, and unsupervised anomaly detection. Supervised anomaly detection requires the entire dataset to be labelled "normal" or "abnormal" and this technique is basically a type of binary classification task. Semi-supervised anomaly detection requires only data considered "normal" to be labelled, in this technique, the model will learn what "normal" data are like. Unsupervised anomaly detection involves unlabelled data. In this technique, the model will learn which data is "normal" and "ab- normal".

A large class of anomaly detection methods assumes the existence of only normal data for training. Hence, the main challenge in designing a reliable anomaly detector is to find the best representation of the data which neither generalizes to unseen anomalies resulting in false negatives, nor overfits on existing data and hence, recognize falsely unseen normal data points as anomalies which results in alerting the user too often.

Anomaly detection methods have been discussed in settings where it is assumed that the input data is either independent and identically distributed or in explicit time series settings, where the temporal dynamic of the time series is explicitly modelled, e.g., using an autoregressive model as it is often the case in traditional change point detection methods. In both mentioned cases, approaches to solve the anomaly detection problem are broadly either Generative or Discriminative.

In the generative approaches, a model is trained to generate samples of the data based on a certain latent representation. If the actual incoming measurement samples and the generated samples differ beyond a tolerable threshold, an anomalous data point is identified. Generative models include the popular autoencoder approaches and the methods based on the recent neural density estimation.

In the lack of anomalous data, the development of discriminative approaches employs the synthetic generation of anomalous data as perturbations on the normal data. Clearly, the generation of synthetic anomalous data is problem specific and being able to generate relevant anomalous data points is often not a given.

3.7.2 Implementation aspects

Reexen developed an anomaly detection application for machine states based on AI models. Our model's name is DS-CNN, and the ToyADMOS dataset is used for model training and evaluation.

ToyADMOS designed for anomaly detection in machine operating sounds (ADMOS). To build a large-scale dataset for ADMOS, ToyADMOS collected anomalous operating sounds of miniature machines by deliberately damaging them. Dataset consists of three sub-datasets for machine-condition inspection, fault diagnosis of machines with geometrically fixed tasks, and fault diagnosis of machines with moving tasks. Each sub dataset includes over 180 hours of normal machine-operating sounds and over 4,000 samples of anomalous sounds collected with four microphones at a 48-kHz sampling rate. All sounds were stored as multiple wav-files categorized into two types: individual (IND) and continuous (CNT) (Figure 37). A normal sound consists of both IND- and CNT-files, anomalous sound consists of IND-files, and environmental noise consists of CNT-files.

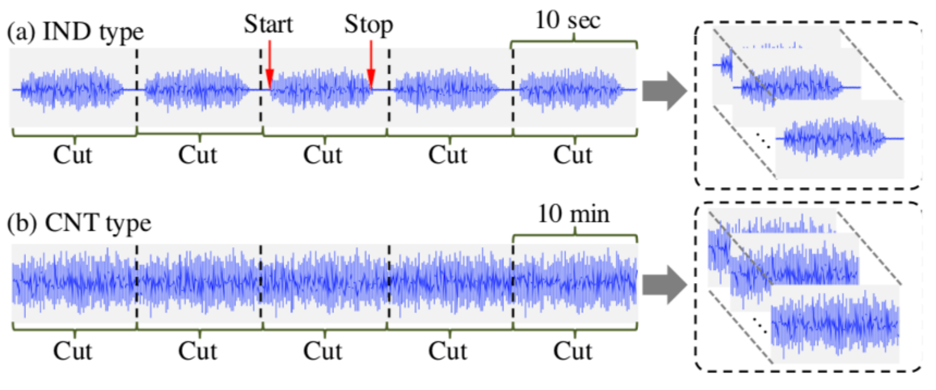


Figure 37: Individual (IND) and continuous (CNT) wav-files

The depth wise separable convolutional neural network (DS-CNN) was proposed as an efficient alternative to the standard CNN. The DS-CNN decomposes the standard 3-D convolution into 2-D convolutions followed by 1-D convolutions, which drastically reduces the number of required weights and computations. In a comparison of multiple neural network architectures for speech recognition on embedded platforms, the DS-CNN was found to be the best performing architecture. For speech recognition, the most commonly used speech features are the mel-frequency cepstral coefficients (MFCCs). In recent years, there has, however, been a tendency to use mel-frequency spectral coefficients (MFSCs) directly with neural network-based speech recognition systems instead of applying the discrete cosine transform (DCT) to obtain MFCCs. This is mainly because the strong correlations between adjacent time-frequency components of speech signals can be exploited efficiently by neural network architectures such as the CNN.

First, MFSC features were extracted based on short time blocks of the raw input signal stream (pre-processing stage). These MFSC features were then fed to the DS-CNN model, which generated latent representation in individual time blocks. Finally, we can determine whether it is an abnormal sound by comparing the cosine distance of the feature (Figure 38).

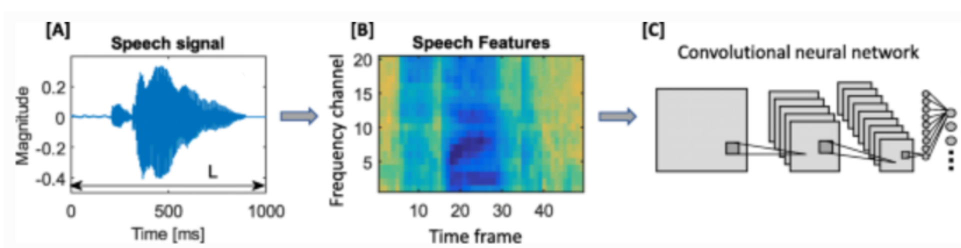


Figure 38: Overview of the different stages in the anomaly detection system

3.7.3 Results

Reexen achieved an (Area Under Curve) AUC of 85.80 on the ToyCar test data and an AUC of 68.93 on the ToyConveyor test data.

3.7.4 IMOCO4.E requirements and KPIs

No specific requirements with this task.

3.7.5 Capabilities & Limitations (including USP, strengths & weaknesses)

Strengths:

The modified method uses an AI model to anomaly detection, which effectively reduces the risk of machine system failure, enables real-time monitoring of machine health, and reduces manual maintenance costs and professional technical experience. The DS-CNN model parameters and calculations are very small and can be easily deployed on hardware to achieve real-time monitoring.

Weaknesses:

The abnormal sounds produced by different machines will be different, and environmental noise may also affect the monitoring effect. This model needs to be fine-tuned for different machines and environments.

3.7.6 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

The DS-CNN model can add more model depth and width according to actual application scenarios to achieve more robust effects. At the same time, the sound sequences can be overlapped in time, and abnormal results can be voted on to reduce the occurrence of false alarms.

4. Automatic commissioning of motion control systems

4.1 Overview of realized solutions

The aim of task 5.4 is to develop algorithms to support the control system's fast commissioning, employing model-based and data-driven methods. The algorithms should provide tools to the designer and the users to speed up the application setting time, monitor the control quality, and allow the system to be retuned if needed.

Integration with the edge, user interfaces, and digital twins (primarily digital models) are equally essential to achieving these goals. The algorithms must communicate with the edge to exchange data or run directly on the industrial drives. User interfaces simplify the user to clarify the commissioning process, which is crucial when advanced algorithms must be used by lower-trained personnel.

Autotuning and fast commissioning have been studied for decades; however, the use of the digital twin in this context empowers the designers and the users to simulate a realistic replica in both the design and the training phases.

4.2 Addressed ST objectives and KPIs

This task relates to objective ST4 (see Figure 2). To be more precise, it provides self-commissioning algorithms for motion control systems which belongs to building block BB6. It addresses Layers 2 and 3 of IMOCO4.E four layers structure.

4.2.1 KPI_BB6_2 and KPI_BB6_4 status

The KPI_BB6_4 is about Self-commissioning of industrial control loops and KPI_BB6_2 is encompassing modelling and system identification. The following subsections are showcasing autotuning of controllers, robust optimal control design, virtual commissioning of control loops, and parametric identification of multirate systems. As such, they contribute to fulfilment of these KPIs.

4.3 Autotuning of PID Controllers for Industrial Drives: Implementation and HIL Testing with Integral Performance Metrics (WEG, UNIBS)

4.3.1 Tech Overview

Examining lift applications, such as in Use Case 1, where the motor is required to follow ramp-link trajectories, an algorithm to estimate the lift's primary dynamics (total inertia, Coulomb, and viscous frictions) for each ramp is developed. This enables the controller to detect dynamic changes, prompting potential retuning. The control-oriented model approximates a first-order model with a nonlinear term for static friction. The dynamic model describes the relationship between motor torque and speed, resulting in a linear transfer function by neglecting the nonlinear friction part. The control architecture incorporates a PI controller with proportional gain, an integral time constant, and a feedforward action to compensate for static friction, aiming to reduce system nonlinearity. Controller tuning is guided by achieving a desired phase margin and gain crossover frequency, considering the mechatronic system's characteristics and disregarding transmission elasticity and damping.

To facilitate tuning, two online identification methods have been developed, utilizing integral dynamic model data during a ramp response to minimize measurement noise. These estimated parameters are crucial for obtaining the transfer function.

4.3.2 Implementation aspects

The drive has limited computation power since the lift application is a cost-driven market from the industrial drive point of view. Thus, model identification is not executed using standard least-square methods, which are powerful but require computational power and suffer from single-precision computation. Instead, a method based on the computation of integral figures allows the algorithm to cope with the measurement noise and to avoid computation complexities.

Two online identification methods have been developed to facilitate tuning, utilizing integral dynamic model data during a ramp response to minimize measurement noise. These estimated parameters are crucial for obtaining the transfer function.

The software has been designed in MATLAB/Simulink and tested with the digital twin of the lift application. The code has been converted to PLC language using the Simulink PLC Coder. Since the industrial drive has different names concerning the IEC611311 standard (for example, function *EXPT* is named *POW)*, a MATLAB function has been designed to automatically adapt the autogenerated code to the required format.

The obtained project was tested in a HIL (Hardware In the Loop) testbed, where an additional motor simulated the lift load.

Validation results on an industrial drive is shown in Figure 39 and Figure 40, where the effect of the self-tuning is demonstrated.

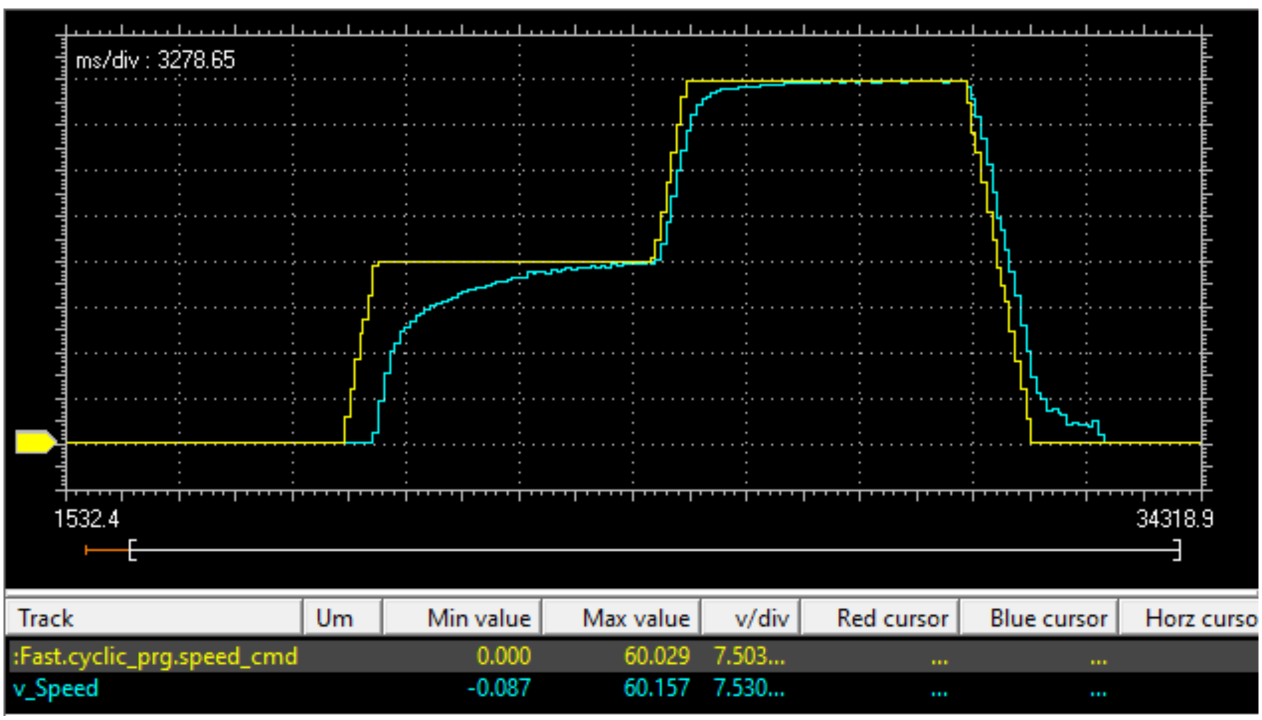


Figure 39: Reference signal and measured motor speed before the retuning, a = 30 [rad/s2]. The reference signal is marked in yellow; the measured motor speed is marked in blue.

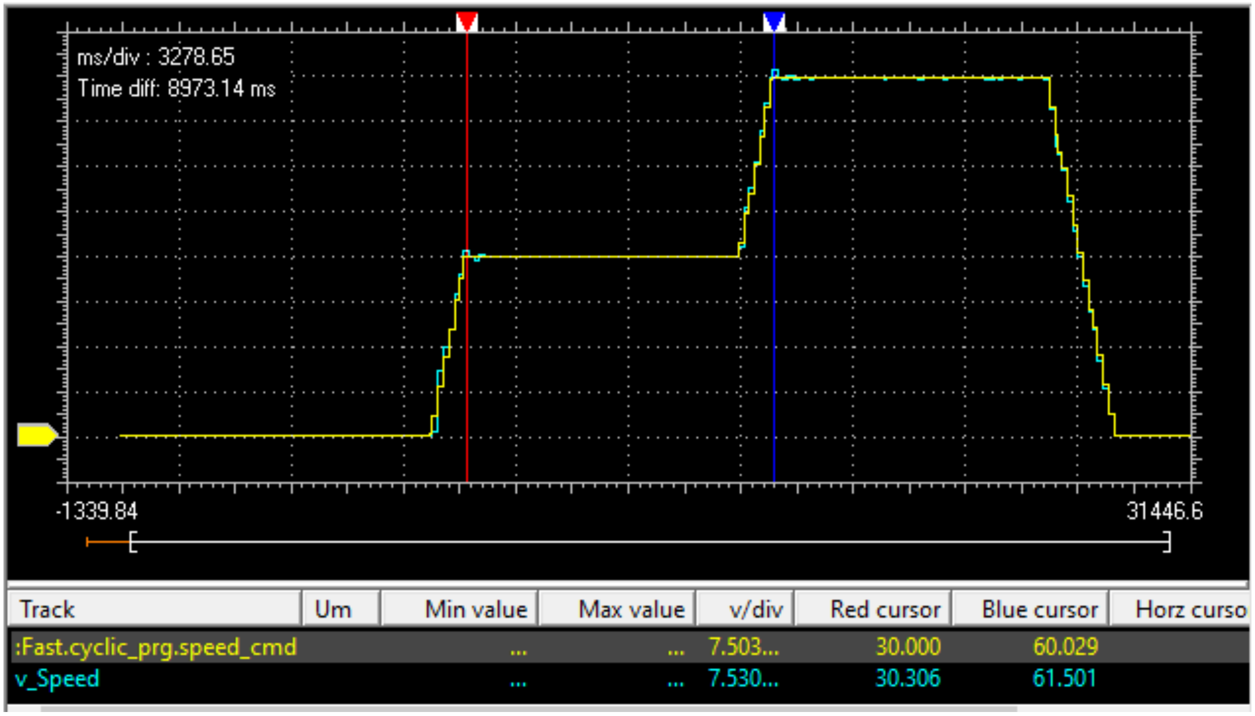


Figure 40: Reference signal and measured motor speed after the retuning, a = 30 [rad/s2]. The reference signal is marked in yellow. The measured motor speed is marked in blue.

4.3.3 IMOCO4.E requirements

The requirements of (D5.2) which apply to these technologies are:

- [R009-D5.1-L2-sw] A HIL based Virtual Commissioning solution should be provided: DONE. The autogenerated code has been tested in HIL-based architecture, using simple virtual loads (inertia+friction) and realistic lift loads (LPV elastic model).
- [R010-D5.1-L2-sw] The Virtual Commissioning system will allow automatic testing of controller and plant model code: DONE. The algorithm continuously monitors the lift model's friction and inertia variations; the tracking error monitors the controller performance.
- [R168-D5.1-UC1-1] All the modules integrated in UC1 must provide testing software in digital-twin or HIL testbed: DONE. The lift digital twin has been implemented in MATLAB/Simulink and AMESim, replicating the lift dynamics, the controlled motion law, the sensor positions, and lift logic. The Simulink code has been converted to IEC611311 ST standard and tested in a HIL test bed.
- [R169-D5.1-UC1-2] All the real-time modules integrated in UC1 must run considering the task-sample time available on the drive (i.e., routine must work considering 1ms or 8ms cycle time): DONE. The algorithm was natively designed to work in single-precision and requires few kilobytes. It can run at 1 ms in parallel with the main control application.
- [R171-D5.1-UC1-4] Modules integrated with UC1 must communicate with MATLAB/Simulink. Real-time modules integrated with UC1 need to be compatible with IEC611311-3 Structured Text standard. A MATLAB/Simulink copy should be provided: DONE. The module has been designed in MATLAB/Simulink and converted using Simulink PLC code.

4.3.4 Capabilities & Limitations (including USP, strengths & weaknesses)

Capabilities are:

1. Real-time monitoring of friction variations over time facilitates proactive maintenance, enhancing equipment longevity and reliability.
2. Enables precise assessment of controller performance metrics such as overshoot and tracking error, aiding in fine-tuning for optimal system response.
3. Adaptive retuning based on estimated plant characteristics ensures continuous optimization without manual intervention, enhancing operational efficiency.
4. Utilizes a straightforward algorithm capable of operating efficiently in single precision, bolstering computational speed and resource utilization.
5. Robustness to noise ensures reliable performance in dynamic environments, maintaining accuracy even in challenging conditions.

Weakness:

1. The algorithm's focus on primary dynamics may overlook high-frequency components, potentially limiting accuracy in systems with rapid changes.
2. While adaptable to diverse applications, rigorous testing has been confined to the Lift Hardware-in-the-Loop (HIL) model, necessitating validation across broader contexts for reliability assurance.

4.3.5 Customizations & Adaptations (including possible modifications and extensions)

Considering customizations and adaptations, the software exhibits versatility and compatibility with industry standards, being written in the widely adopted IEC611311 ST language. Moreover, flexibility could be extended by leveraging MATLAB's code generation capabilities for seamless conversion into other standardized languages like C++.

One notable feature lies in the accessibility of estimated parameters through Modbus. Extending this functionality to other communication buses could significantly broaden its applicability across various industrial settings, fostering interoperability and enhancing overall utility.

4.3.6 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

From the designer's perspective, the methodology employs a comprehensive toolchain comprising a digital twin of the plant in AMESim, controller replica in MATLAB/Simulink (motion law, control algorithm, Lift logic), and self-commissioning designed within Simulink, incorporating single-precision computation and native code generation. Leveraging Simulink PLC code generation facilitates seamless implementation of self-commissioning algorithms and plant models on PLCs, with HIL testing ensuring robust controller performance (see Figure 41). This toolchain enables us to tackle implementation challenges, such as the impact of single-precision computation, and refine strategies by identifying and addressing controller bottlenecks within the simulated environment.

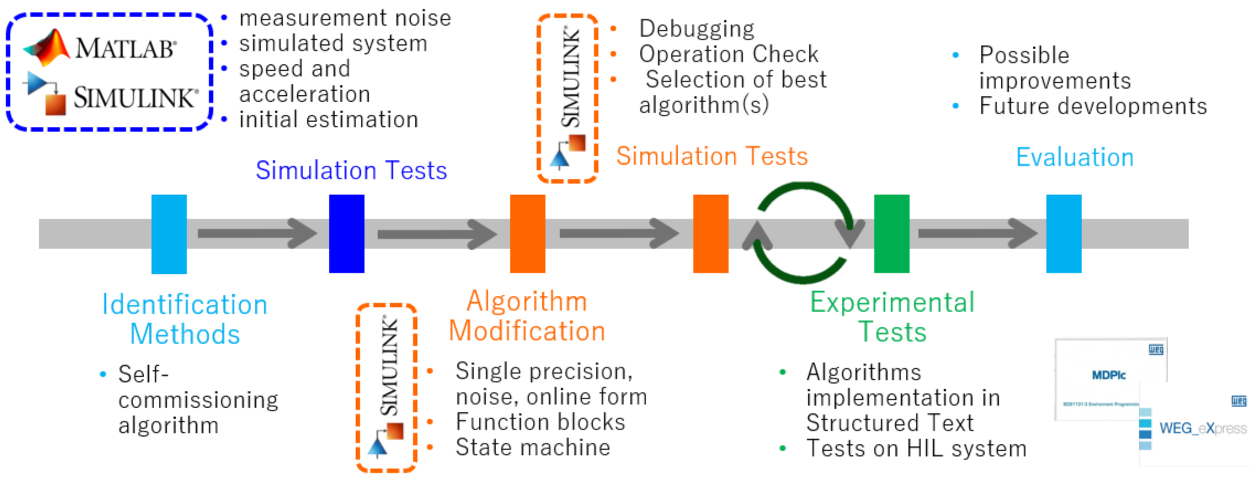


Figure 41: Toolchain used for IMOCO Use Case 1, within the DT and xIL methodologies.

Meanwhile, from the user's viewpoint, digital twin implementation offers training opportunities, enhancing user understanding and proficiency. During the application lifetime, monitoring friction variation over time provides users with intuitive insights into plant conditions, while self-tuning capabilities minimize the need for extensive domain knowledge. A key lesson learned is the importance of minimizing user intervention by reducing parameter settings, ideally zero, streamlining the user experience, and fostering usability.

4.4 Robust and optimal design of fixed structure controllers in motion systems (UWB)

4.4.1 Tech Overview

Motion control loops are crucial for modern mechatronic systems, and they need proper design and tuning. Engineers and technicians must set the parameters of motion controllers, which is a difficult task. The controlled plant has different dynamics for each machine, so the control tuning must fit the specific system. The tuning process is usually manual and involves a lot of trial and error. The results are often not optimal and depend on the people who do it.

Proper tuning is very important because industrial manufacturing systems have higher and higher performance demands that affect the control layer. For motion systems, this means strict requirements on bandwidth and tracking precision. High performance requirements can cause problems with vibrations when the bandwidth matches the resonance modes of the controlled plant. Unwanted oscillations make the control system tuning harder. Automatic tuning methods are common and successful in process control, but not in mechatronic systems. Therefore, developing systematic methods that can help with the commissioning process is very relevant for industrial practice.

4.4.2 Implementation aspects

Implementation of self-tuning algorithms can in principle be done in two ways

1. **At the edge in the lowest instrumentation layer (Level 2 of the IMOCO4.E reference framework).** This approach was pursued by UNIBS and WEG with the results described in the preceding section. The goal was to embed the algorithms directly in the servoamplifier. The advantage of such deep integration is that no additional HW/SW tools are needed to perform the

controller tuning, avoiding issues with data exchange, and potentially minimizing the necessary user intervention. On the other hand, limited memory and computational resources of the industrial control platforms used at the lowest instrumentation level do not allow direct implementation of advanced data processing, system identification and model-based design algorithms. Also, the integration is usually vendor specific, tailored to a particular HW platform and its SW environment. This is why we opted for the second approach with the goal of delivering general purpose hardware and software independent solution.

2. **Remote operation in higher control layers (Level 3 / 4 of the IMOCO4.E architecture)** – The drive itself, including its controls and sensors, serves only for the execution of system identification experiments in this scheme, collecting and sharing the data with upper layers responsible for the controller commissioning process. The commissioning can take place in a supervisory control system commanding the drives (Layer 3) or a dedicated machine in the Maintenance & commissioning Layer 4, be it a server running in the cloud or a local corporate network or a dedicated computer connected to the drive for the purpose of its commissioning. This approach requires proper handling of interfaces and data exchange between the layers but enables using vast resources of today's high-performance computers that can assist with the task.

4.4.3 Results

UWB proposed a new framework that can achieve both robustness and optimality in the closed-loop motion systems.

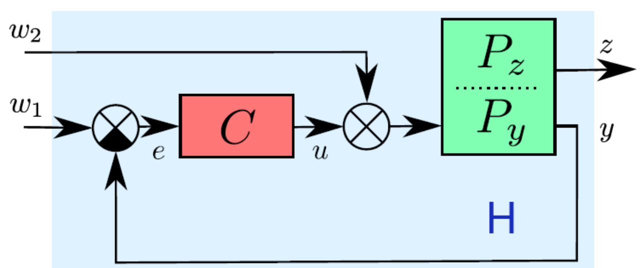


Figure 42: Assumed control setup – collocated motion system with different feedback and performance variables

We consider a feedback structure commonly designated as a "collocated control setup" (Figure 42). This setup is common in mechatronics and motion control systems. The "collocated" term means that the actuator-sensor pair is physically at the same location of the controlled plant. The output y is typically provided by an encoder installed on the rotor shaft of an electric actuator. This signal is used to close the velocity or position feedback loop. However, the objective is often to control another physical variable z; usually the position or velocity of a reference point at the load-side moving part, such as a robot end-effector or a CNC machine tool spindle. If the load is ideally rigid, the working mechanism and the actuator move in sync. The outputs y and z are the same (except for possible scaling due to kinematic transform), and the control topology is a standard single-input-single-output (SISO) feedback loop. But mechanical flexibility in the load adds more degrees of freedom. More complex behaviour with unwanted oscillations often happens because of plant bending modes. This makes the feedback control design harder, as it inherently introduces a multivariable design problem. Good performance for feedback variable y does not automatically guarantee a good response for the variable of interest z.

The design method can be summarized as follows:

1. Derive the model of the controlled plant from the experimental data using system identification or first principle / geometrical modelling, possibly including uncertainty for a robust control design.

2. Define feedback and performance variables y and z for the collocated control setup, forming a single-input-two-outputs system (the simpler SISO scenario is recovered as a special case by choosing $y = z$)
3. Find a set of stabilizing controllers of a given structure fulfilling certain robust stability condition.
4. From the admissible set, find an optimal vector of controller parameters with respect to the defined performance variable z and a chosen performance measure.
5. (optional) Provide a set of suboptimal reduced-gain controllers allowing in-situ fine-tuning of the controller during commissioning to find a suitable performance/cost trade-off.

The algorithmic details were published in (Goubej et al., 2023) where we also present validation results using the lift control problem of Use case 1.

The lift system has a specific challenge because its oscillatory dynamics depend on the position of the cabin, which changes the active length of the rope segments. The oscillations must be reduced to make the operation safe and reliable. The controller parameters must be adjusted to get good performance at the cabin side, while the feedback loop is usually closed at the motor side because most lift systems do not have easy ways to measure the cabin position and/or velocity. So, the control topology follows the collocated control setup.

The performance of the lift system is depicted on Figure 43, showing the resulting tracking performance during a rest-to-rest lift manoeuvre. Smooth velocity and acceleration response is achieved without exhibiting any cabin-side vibrations. This validates the proper design of the robust and optimal controller.

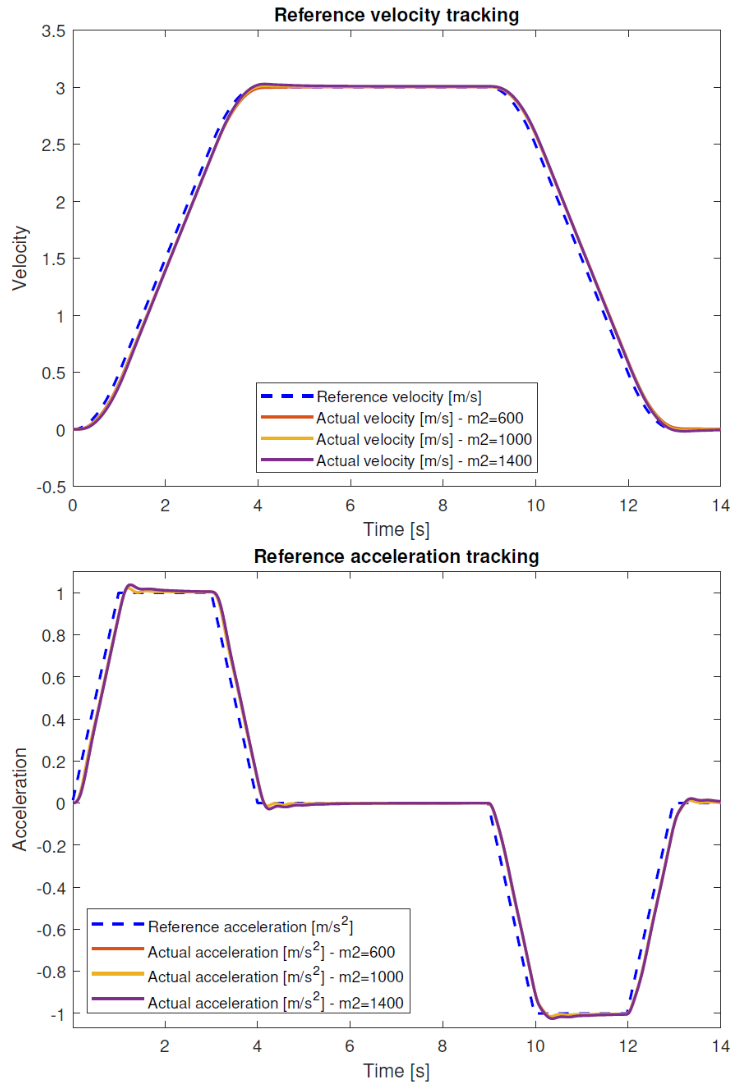


Figure 43: Closed-loop response of the lift system with the optimized tuning of the main drive control loop

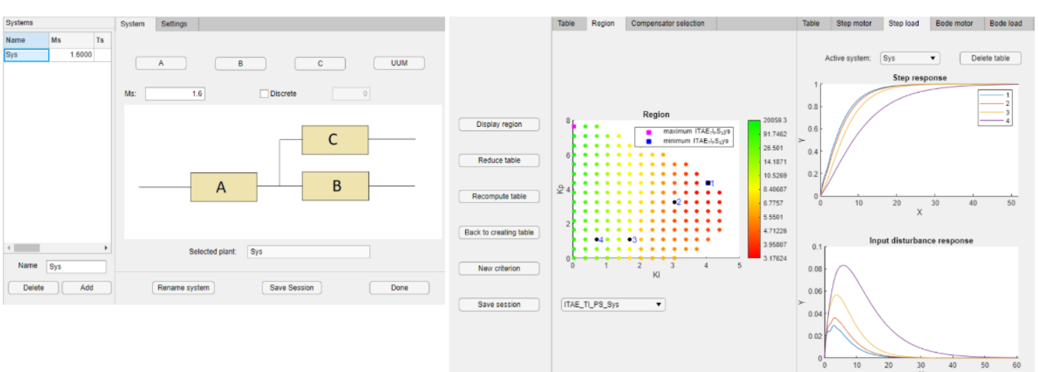


Figure 44: H-infinity electromechanical control design tool – GUI providing expert user interface to the developed control tuning methods

The control design algorithms were implemented in MATLAB scripts in the m-language. C-code can be generated to employ them to various target platforms. The software can be used by means of a command-line interface, or through a developed graphical user interface (GUI) (Figure 44) allowing the user to interfere and fine-tune the controller design process. The GUI is available in the form of a MATLAB application or as a standalone application for Windows or Linux PCs.

4.4.4 IMOCO4.E requirements

The developed results conform to the following requirements, as defined in the deliverables D5.1/5.2:

- [R008-D5.1.-L2-sw] Virtual Commissioning solution should allow real time simulation of plants (sampling < 1 ms): DONE. The developed commissioning tools allows working with plant and controller models with arbitrary sampling period as well as in the continuous time domain.
- [R009-D5.1-L2-sw] A HIL based Virtual Commissioning solution should be provided: DONE. The developed methods were tested using HIL motion testbed developed at UWB, emulating both rigid and flexible working loads. Realistic nonlinear flexible and position-dependent load was also tested in the lift control scenario as a part of UC1. The self-tuning framework was also employed to improve control performance of robotic platform developed in UC4.
- [R011-D5.1.-L2-sw] MATLAB/Simulink should be among available plant modelling environment: DONE. The control commissioning tools is compatible with MATLAB/Simulink environment allowing import of plant models and export of designed controllers.
- [R168-D5.1-UC1-1] All the modules integrated in UC1 must provide testing software in digital-twin or HIL testbed: DONE. The lift Digital twin has been implemented in MATLAB/Simulink and AMESim, replicating the lift dynamics, the controlled motion law, the sensor positions, and lift logic. The control tuning tools allow extraction of the plant model dynamics from the Digital twin model of the lift, following by model-based controller synthesis. Controller parameters are then updated in the Digital Twin, allowing performance validation by numerical simulation and automatic code generation of the control part, when needed.

4.4.5 Capabilities & Limitations (including USP, strengths & weaknesses)

The developed software tool provides a means for systematic design of low-order fixed-structure controllers. The main advantage is a possibility to mix the robustness and optimality aspects while creating simple PID controllers (and its derivatives) that are commonly employed in industrial-grade motion control hardware. The method can be adapted to other compensators with two or three parameters. Multi-objective performance optimization is achievable, using various criteria from the time, algebraic, or frequency domains, which is done by sampling and segmenting the set of robust admissible controllers set. The lift control problem is used as a benchmark example, showing the practical usefulness of the proposed approach. However, the tool may be used for any plant dynamics without prior assumptions on its order or structure.

The main limitation is a focus to single-input-single output (SISO) control loops. Applications leading to complex multivariable dynamics may require more specific MIMO (multiple-input-multiple-output) design methods. However, decent results can still be achieved with multi-loop decentralized control for plants with negligible loop interactions. Alternatively, a common strategy of decoupling the dynamics followed by multi-SISO control design can be employed. Another limitation is the assumption on linear time invariant plant models. Systems with highly nonlinear or time/configuration dependent dynamics may require employing common approximation techniques, such as local linearization or multi-model sampling.

4.4.6 Customizations & Adaptations (including possible modifications and extensions)

The provided tool is focused on tuning of PID-like feedback compensators. More complex high-order controllers are not supported in the current version. Latest results achieved at UWB show that an iterative analytical sequential design of high-order controllers is possible, giving an alternative to commonly employed techniques of non-smooth non-convex parametric optimization. This could be an interesting research direction for a future work.

4.4.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

The developed SW tool can run on Windows & Linux machine. There are relatively high-performance requirements, due to the complex nature of implemented algorithms. For the integration with specific control hardware platform, data interfaces must be established to fully automate the control commissioning process. The data exchange is always vendor specific and requires some integration effort. Partial results were achieved in integration with the servoamplifier developed as BB7, where a data exchange was designed between the drive and the commissioning software, minimizing necessary user intervention. However, the tool can be used even without making these integration steps, only requiring the data collection and control parameter changes to be done manually by the user. Even with these partially manual steps, the tool can reduce the overall machine commissioning time, avoiding tedious trial and error process commonly done in engineering applications. Moreover, the model-based design approach allows optimising the achieved control performance up to the physical limits of the controlled plant.

4.5 Virtual commissioning of complex PLC/CNC projects (TEK)

4.5.1 Tech overview

Virtual commissioning is the process of employing virtualization and simulation technologies to replicate the physical plant and/or controller in a virtual environment as a way of verifying the behaviour of the manufacturing system (see Figure 45). Through virtual commissioning techniques, it is possible to test a system virtually before it is physically constructed.

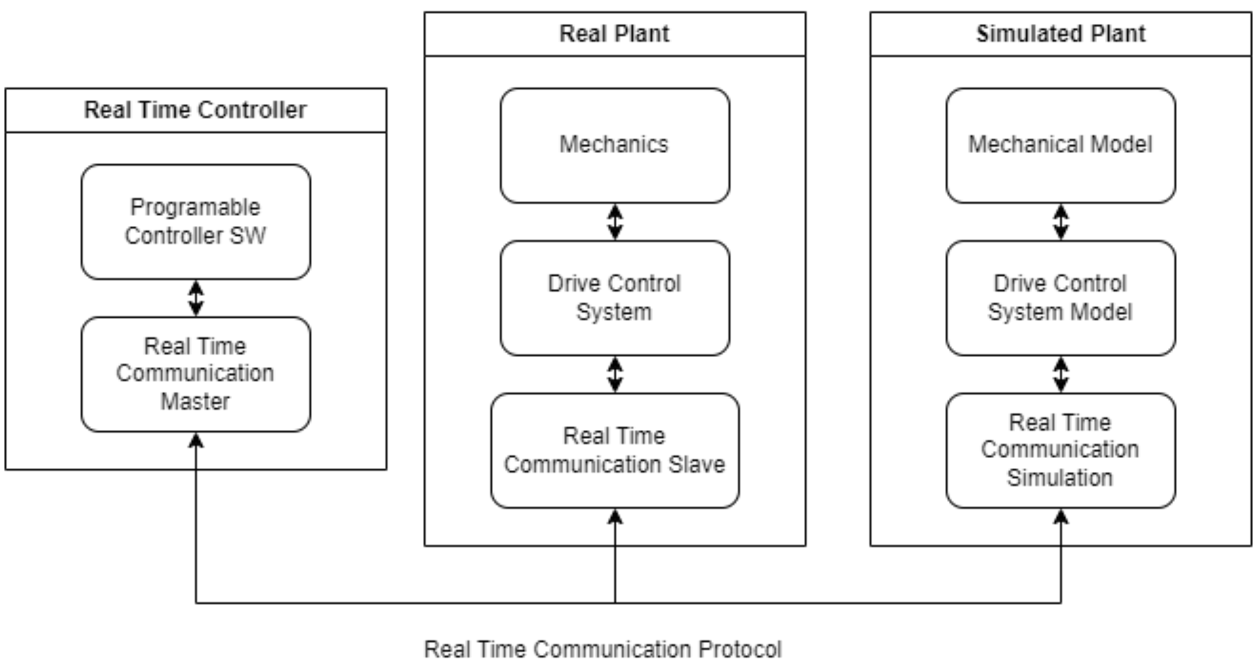


Figure 45: Virtual Commissioning approach

Although this approach is generic and aimed to any type of mechatronic system, in the IMOCO4.E project it has been implemented in the Use Case 2. Within the Deliverable 5.4, the initial development of the Virtual Commissioning approach considered by TEKNIKER was presented. The applications tested involved a Stäubli robot controlled by a Beckhoff TwinCAT control system. The robot control using real hardware is shown in Use Case 2. Advancement presented within this deliverable is the implementation over a real time platform of this Virtual Commissioning approach.

In order to utilize this Virtual Commissioning approach, an accurate model of the plant is needed to get a simulation behaviour closer to the real system. Within Task 5.5, a Stäubli robot dynamic model was developed (reported in Deliverable 5.5). Further work presented within this deliverable shows how the model C++ code generation is integrated into a drive control system model. Also, variable interfaces have been included to improve the standardization of the solution and be able to easily change the model.

4.5.2 Implementation aspects

As mentioned, this Virtual Commissioning approach has been developed for BECKHOFF/TwinCAT based controllers. This was done due to increasing usage of this system in mechatronics system control, as well as hardware and programming flexibility. The approach could be adapted to other control architectures, but it should be adapted to the corresponding limitations.

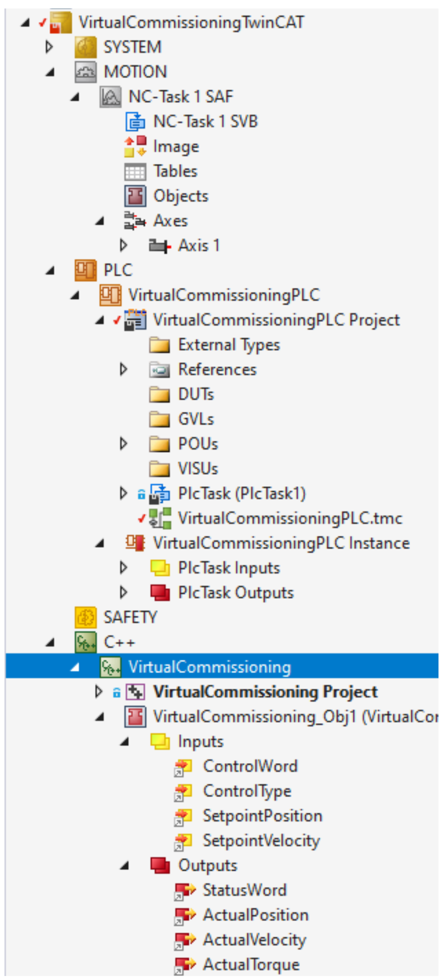


Figure 46: TwinCAT project with model objects connected to PLC program

Figure 46 shows the structure of the TwinCAT project used for Use Case 2 Virtual Commissioning. Use case 2 considers an advanced robot control system in which a PLC or a CNC commands the behaviour of the robot. Kinematics and path planning are implemented on the controller side to get better control capabilities.

The plant model, in this case the robot and its drives, is integrated through C++ code in the same TwinCAT project. This model integrates:

- Drive model: control loops (cascade type position controller) for each joint and state machine of the drives.
- Mechanical model of the plant, receiving the torque of the motors as an input and providing sensors information feedback (usually encoder position and velocity).

A generic template in C++ has been developed for the drive system modelling. This template integrates the mechanical plant model. Automatic code generation tools from MATLAB/Simulink can be used to facilitate this work. This was done in Use Case 2, where the code was generated from the model developed in MATLAB Simulink and Simscape within Task 5.5.

Once robot model was obtained, there were some more model requirements to get a reliable real time simulation representing all the layers of the real control system. Due to their favourable integration with TwinCAT environments, CoE (CAN over EtherCAT) drive have been modelled. For this purpose, both state machine and control loops have been taken into consideration. CiA 402 is the device profile specification which standardizes the communication of the drive with other elements from the control environment.

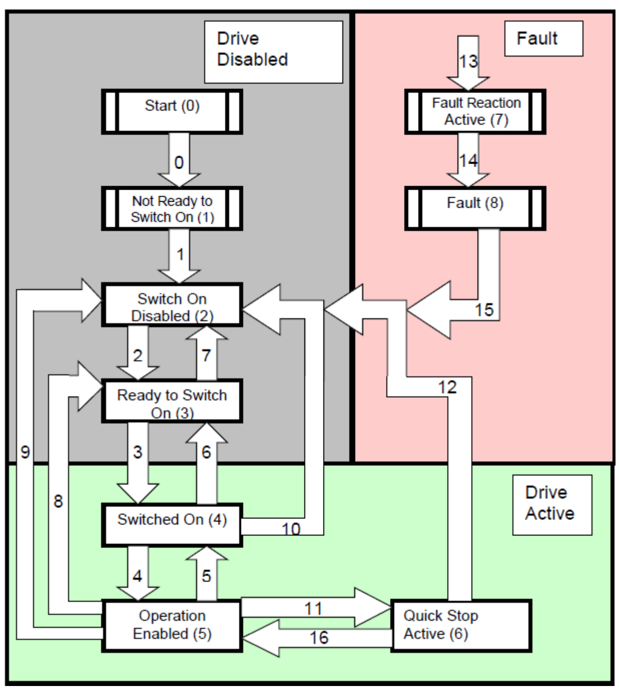


Figure 47: Drive state machine considered

Drive interaction is based on control and status words. State machine modelling (Figure 47) consists of writing the status word variable which indicates the status of the system depending on the value of the control word (representing the commands to manage the state transitions).

4.5.3 Results

A TwinCAT PLC project has been created so that it is able to control both the real and the simulated robotic system. Figure 48 show the NC axis commanding tab of the TwinCAT GUI. The commanded positions are sent to the mechanical system model integrated in the project through code generation.

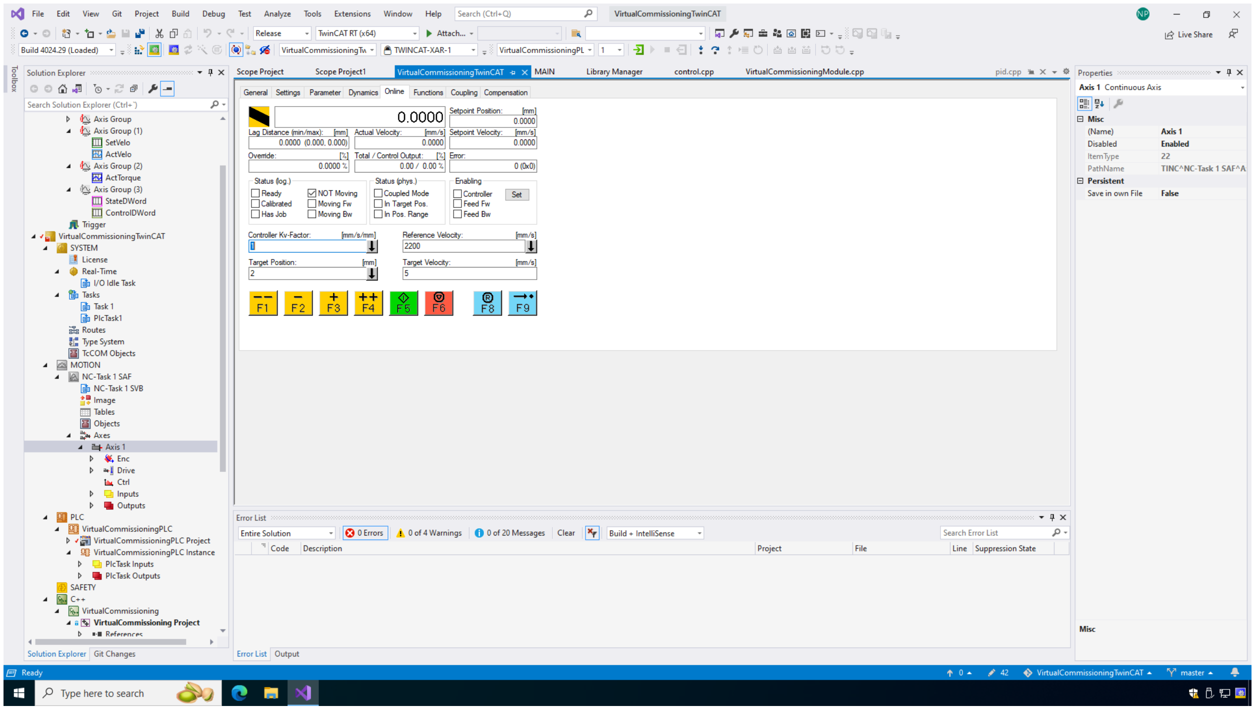


Figure 48: NC axis commanding tab of the TwinCAT GUI.

As a proof of the real time performance, step position response is presented (see Figure 49).

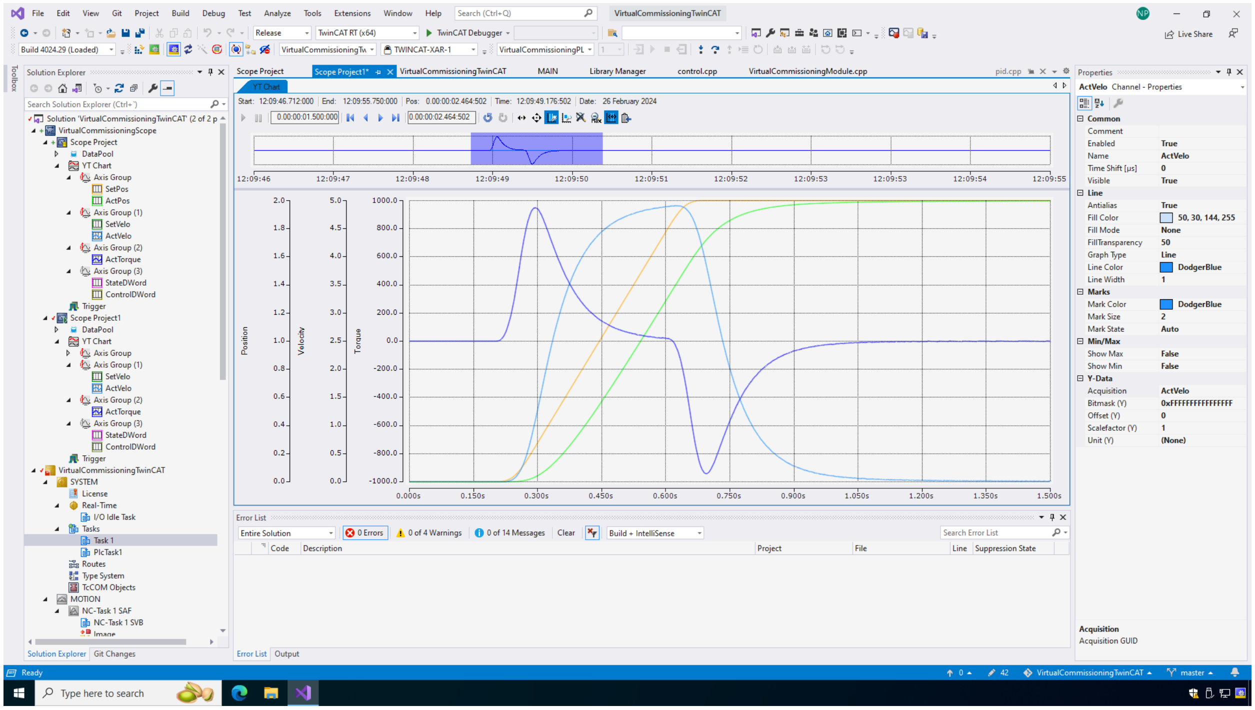


Figure 49: Screenshot of TwinCAT GUI showing the step response of a drive to a position setpoint

For visualization purposes, the same model of the robot developed in Simulink for code generation purposes can be used. ADS functions from TwinCAT3 Interface have been used to connect the PLC project with the integrated model with a second instance of these model that can be running in another computer. Through these functions, Simulink uses the positions of the controlled model to represent graphically the configuration of the robot in each time step (Figure 50).

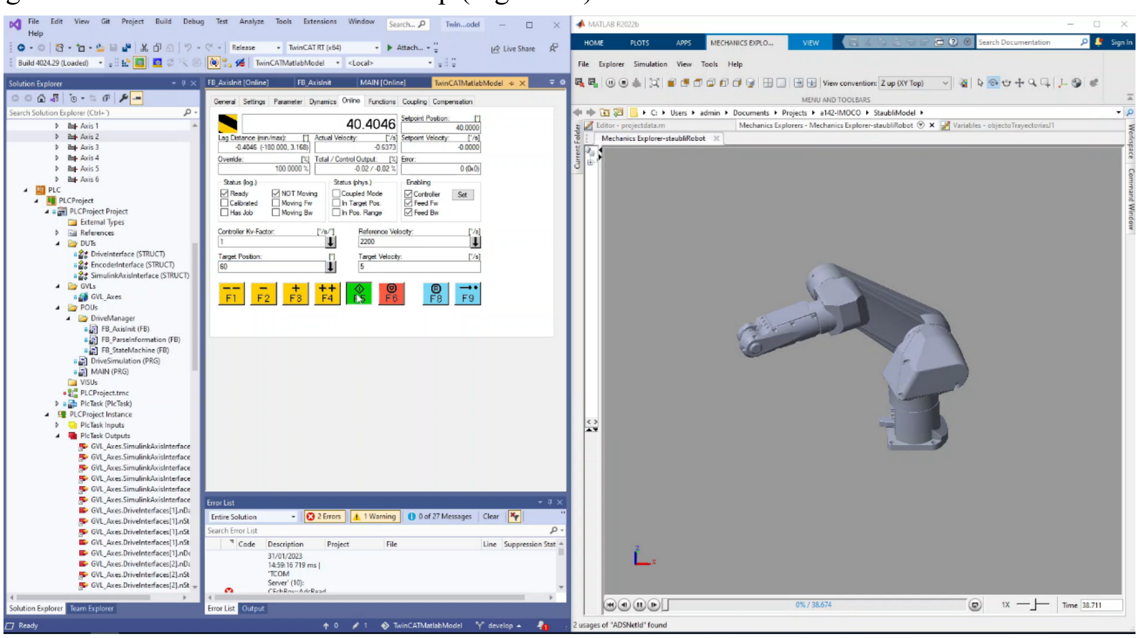


Figure 50: TwinCAT project with the robot model integrated and the correspondent visualization in Simulink

4.5.4 IMOCO4.E requirements and KPIs

The next requirements taken from (D5.2) apply to the Virtual Commissioning solution. All have been appropriately addressed:

- [R004-D5.1-L2-sw] Virtual Commissioning with, at least, PROFINET and EtherCAT communications should be supported → at this stage, only EtherCAT communication has been covered.
- [R008-D5.1-L2-sw] Virtual Commissioning solution should allow real time simulation of plants (sampling < 1 ms) → Use Case 2 requires sampling times of 4 ms. Lower rates have been also evaluated with simple drives. No problem is foreseen.
- [R009-D5.1-L2-sw] A HIL based Virtual Commissioning solution should be provided → Done.
- [R011-D5.1-L2-sw] MATLAB Simulink should be among available plant modelling environment → For Use Case 2 the required code was generated from MATLAB Simulink model of the robot.
- [R173-D5.1-UC2-3] For Synchronization of data oriented to virtual commissioning, high speed data gathering is needed → communication speed (up to 1 ms) defines the sampling rate.
- [R175-D5.1-UC2-5] Dynamic models of axes (robot and machine tool) must come in Simulink or Simulink importable formats → Done.
- [R178-D5.1-UC2-8] Reduced, dynamic only model is acceptable for virtual commissioning of MIMO loops → Done.

4.5.5 Capabilities & Limitations (including USP, strengths & weaknesses)

The proposed approach offers significant advantages for the development, validation, and commissioning of controllers of complex systems. The possibility to develop a PLC controller against a simulated plant allows reduced development times, reduced overall project duration (control programming and validation can be started in early stages), eliminating the need for real machinery, increased personnel safety, code robustness (thanks to fast testing in different operating scenarios), etc. The development of standardized approaches like this and the integration of xIL methodology in the system development as well as automatic code generation techniques allows reducing modelling development time and cost and, hence, facilitates its application.

The modelling effort in automation projects is not directly supported by the customer and, hence, it is assumed to be done by the developer. The first utilization of these approaches can take longer and causes higher expenses. This drawback is often compensated in case of reusing the models in other projects because then the required time is highly reduced – rapid prototyping. While in small projects the application of these approaches is debatable, the real power refines in big projects.

4.5.6 Customizations & Adaptations (including possible modifications and extensions)

As mentioned, the proposed Virtual Commissioning approach has been developed for BECKHOFF/TwinCAT based controllers. This was done due to increasing usage of this system in mechatronics control system, as well as hardware and programming flexibility. The approach could be adapted to other control architectures, but it should be adapted to the corresponding limitations.

Although the activity reported within IMOCO4.E has been mainly focused on the Virtual Commissioning (VC) of the robotic system forming Use Case 2, it can be directly extended to any other mechatronic system. Indeed, it was partially applied to some validation activities for the Offset Free MPC controller developed in Task 4.4 (check (D4.8)).

4.5.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

The approach itself is a new methodology for generic Virtual Commissioning of mechatronic systems. It is based on the xIL methodology and in case it is integrated in the project, Virtual Commissioning is facilitated.

For the higher-level Virtual Commissioning approach, Hardware in the Loop stage should be reached so that the control of the real system can be plug-and-play after Virtual Commissioning. For that, real time communication as well as drive system must be modelled. Automatic code generation strategies facilitate this task (see Figure 51).

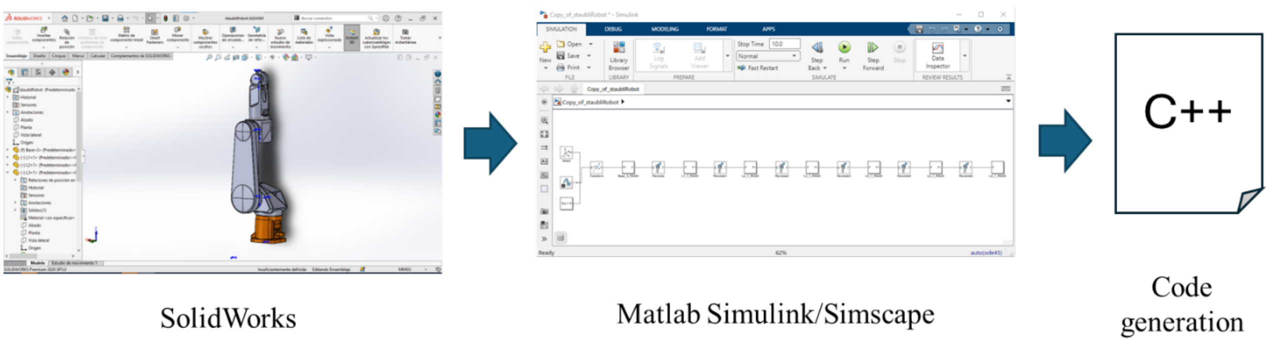


Figure 51: Automatic model generation and code generation approach used for IMOCO Use Case 2, within the xIL methodology

4.6 Parametric Identification of Multirate Systems for Digital Twinning

4.6.1 Tech overview

Digital twins that capture intersample behaviour are crucial for control design, particularly in slow-sampled systems like vision-in-the-loop applications. This deliverable addresses this challenge by proposing a novel approach for the parametric identification of fast-dynamics models that can be utilized as digital twins using only slow-sampled output measurements. The method leverages kernel-based regularization to incorporate prior knowledge and facilitate the identification of models with arbitrary order, enabling the estimation of intersample dynamics. This framework utilizes readily available fast-sampled inputs alongside the limited slow-sampled outputs and achieves accurate identification of fast-dynamics models in a single experiment. The effectiveness of the proposed method is demonstrated through simulations and experimental data, paving the way for the development of high-fidelity digital twins for slow-sampled systems.

The related SW component is SW-002 - Multi-rate (non-)parametric system identification. Furthermore, the technology is under review for journal publication.

4.6.2 Implementation aspects

The developed approach minimizes the difference between a model, i.e., in this case a digital twin, and the real system, that is visualized in Figure 52.

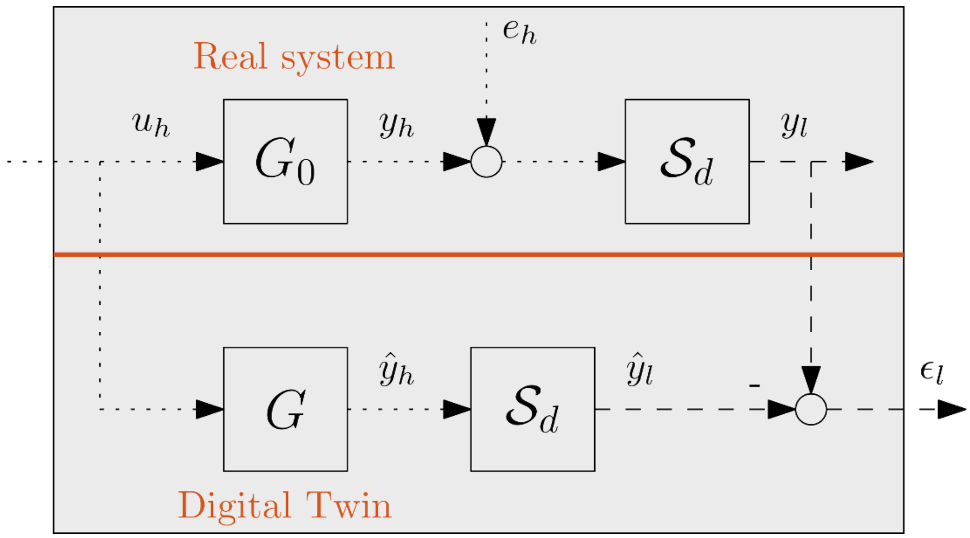


Figure 52: Setup for identification of slow-sampled systems for digital twinning, where a digital twin, or model, is identified such that the difference in output between the real system and digital twin, i.e., ϵ_l, is minimized.

The output of the digital twin is modelled using the Finite Impulse-Response (FIR) model:

$$\hat{y}_l(m) = \mathcal{S}_d G(q,\theta)u_h(mF) = \sum_{i=0}^{P-1} \theta_i q^{-i} u_h(mF),$$

Where F is the downsampling factor of downsampler $\mathcal{S}_d$ and P the FIR model order. The output can be vectorized as

$$\hat{y}_l = \begin{bmatrix} \hat{y}_l(0) \\ \hat{y}_l(1) \\ \vdots \\ \hat{y}_l(M-1) \end{bmatrix} = \begin{bmatrix} u_h(0) & 0 & \cdots & \cdots & 0 \\ u_h(F) & \cdots & u_h(0) & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ u_h((M-1)F) & \cdots & \cdots & \cdots & u_h((M-1)F-P) \end{bmatrix} \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_{P-1} \end{bmatrix} = \Phi\theta,$$

With M the amount of data points of the slow-sampled output y_l. For this setup, the following cost function is minimized to identify the digital twin:

$$\min_{\theta} \|y_l - \Phi\theta\|_2^2 + \gamma\theta^\top K^{-1}\theta,$$

Where the kernel matrix K enforces high-level correlation of the FIR coefficients, such as exponential decay, and is constructed as

$$K = \begin{bmatrix} k(0,0) & k(0,1) & \cdots & k(0,N-1) \\ k(1,0) & k(1,1) & \cdots & k(1,N-1) \\ \vdots & \vdots & \ddots & \vdots \\ k(N-1,0) & k(N-1,1) & \cdots & k(N-1,N-1) \end{bmatrix}.$$

An example of a kernel function is the diagonal correlated kernel, i.e.,

$$k_{DC}(i,j) = \lambda\alpha^{\frac{i+j}{2}}\rho^{|j-i|}.$$

The unique minimizer to identify the digital twin is given by:

$$\theta = K\Phi^\top(\Phi K \Phi^\top + \gamma I_M)^{-1} y_l.$$

Specific implementation aspects

The following aspects should be taken into account.

- The identification excitation signal u_h should be exciting of sufficient order of persistence of excitation. A white-noise or random phase multisine signal are ideal.
- Since the algorithm only uses offline data, no specific implementation aspects in terms of software or hardware must be taken into account.

4.6.3 Results

The experimental setup is the two-mass system in Figure 53, where the two masses are connected via a metal shaft.

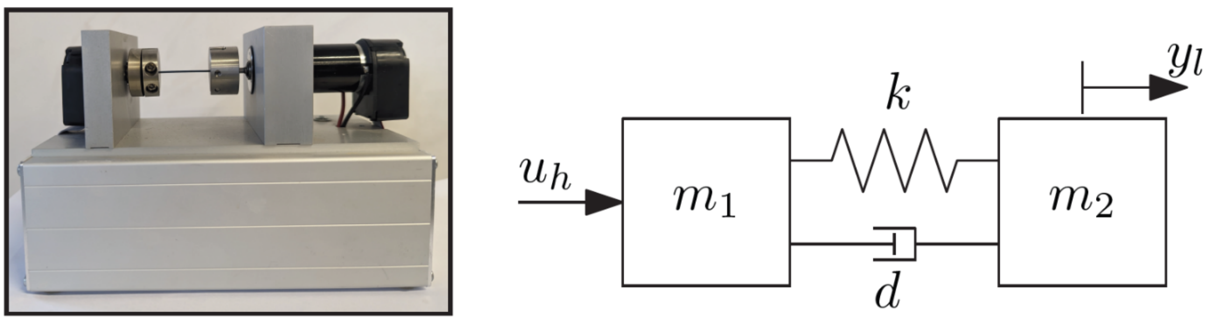


Figure 53: Left: Photograph of experimental setup. Right: Schematic overview of experimental setup.

The first mass m_1 is stabilized by means of feedback and consists of a lead filter and a weak integrator that achieves a bandwidth of 3Hz. The experimental settings are listed in the following table.

Variable	Abbreviation	Value	Unit
Fast Sampling Frequency	f_h	260	Hz
Slow Sampling Frequency	f_l	65	Hz
Downsampling Factor	F	4	-
Number of fast-sampled samples	$M \cdot F$	16640	-

The system is excited by random phase multisines u_h with variance 0.1217. Two independent data sets are created, a training set and a validation set, with a different phase realization of the multisines. The following models are identified and validated.

- Tikhonov regularized FIR, i.e., the developed approach with kernel matrix $K = I$.
- Kernel regularized FIR, i.e., the developed approach with the diagonal-correlated kernel.

For the kernel regularized estimator, the hyperparameters $\lambda = 1$, $\alpha = 0.9$ and $\rho = 0.8$ are determined using cross validation. The regularization parameter is tuned to $\gamma = 10^{-10}$. The models are evaluated using the goodness of fit:

$$\text{GoF} = 100\left(1 - \frac{\sum_{n=0}^{MF-1}\left(y_{h,v}(n) - \hat{y}_h(n)\right)^2}{\sum_{n=0}^{MF-1}\left(y_{h,v}(n) - \bar{y}_{h,v}\right)^2}\right),$$

Where subscript v is used to indicate the validation data set. The goodness of fit for different model orders P is seen in Figure 54.

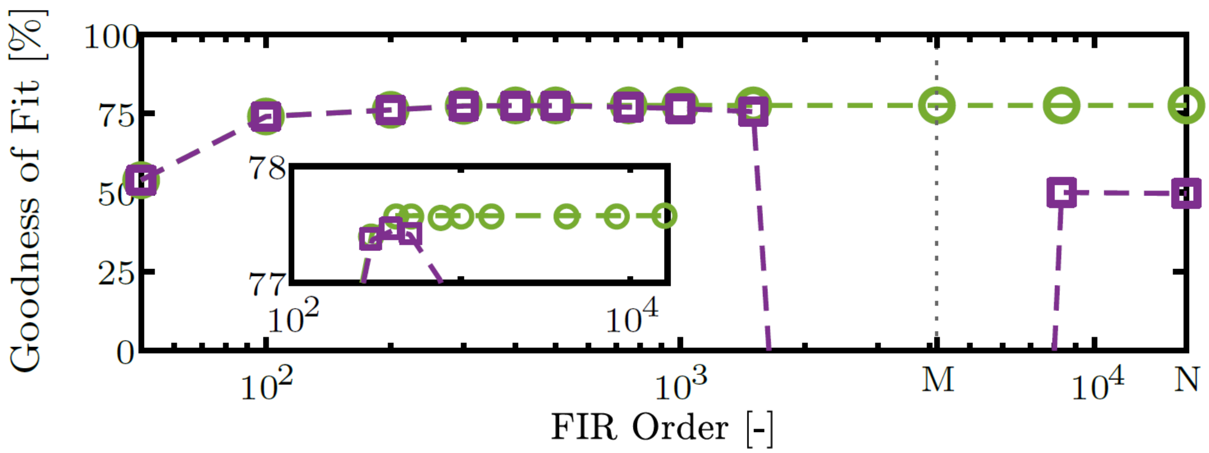


Figure 54: Goodness of fit for different orders P of Tikhonov regularized FIR (--), and kernel regularized FIR (--), that has a high goodness of fit for all model orders, including non-parametric model P = N = 16640.

Additionally, frequency response functions of the identified digital twins for a specific model order are seen in Figure 55.

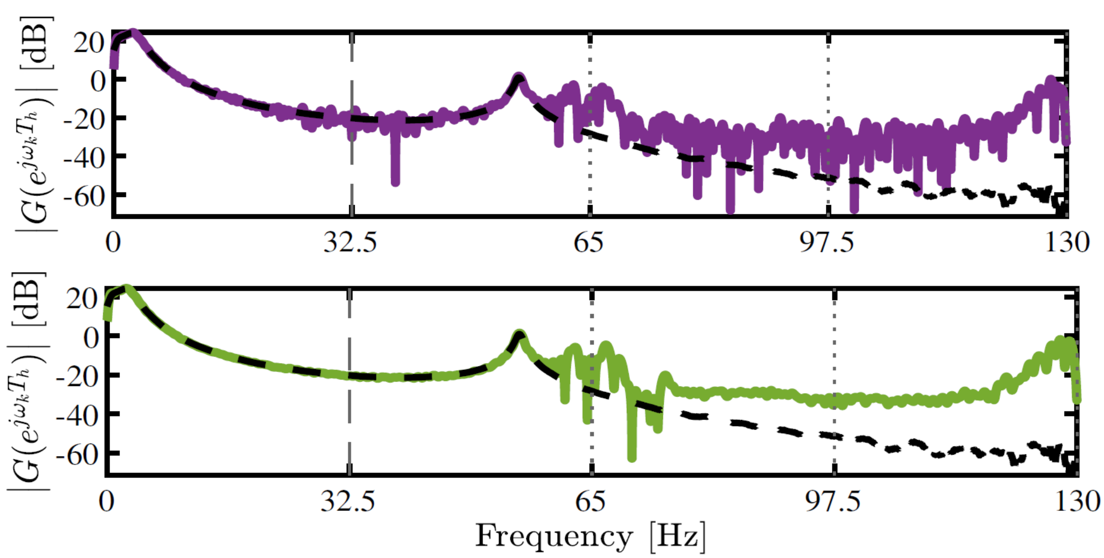


Figure 55: FRF based on y_h using an efficient identification algorithm, that is considered to be G_0(--), compared to estimated FRFs for Tikhonov FIR with P = 400 (top, -) and kernel regularized FIR with P= N = 16640 (bottom,-), that models the true system most accurately, even above the Nyquist frequency 32.5 Hz of the slow-sampled output (--).

In conclusion, the developed approach accurately identifies digital twins of slow-sampled systems, even above the Nyquist frequency of the slow-sampled sensor.

4.6.4 IMOCO4.E requirements and KPIs

The technologies comply with the following requirements. Their fulfilment will be described in follow up deliverables in WP6.

- R18-D4.2-P4-BB5, BB6, BB8-L2, L3, L4: Smart control algorithms' AI-components and Digital Twins (DT) shall only use existing data and interfaces of the brown field system after training and calibration.
- R127-D2.3-L2: Smart control algorithms and models shall be tested in simulation.
- R130-D2.3-L2-L3: All models should be compatible with MATLAB/Simulink
- R41-D4.2-P4- BB5, BB6, BB8, BB10-L2, L3, L4: Smart control algorithms, AI-components and digital twin models shall be compatible with or can be integrated into / Simulink environment.
- R137-D2.3-L2-L4: Data-driven models shall be compared to analytical models and/or validated real robots.

4.6.5 Capabilities & Limitations (including USP, strengths & weaknesses)

The main capability of the framework is to identify fast-dynamics digital twins using slow-sampled output measurements, even above the Nyquist frequency of the slow-sampled sensor. The integration of kernels allows for universal identification, including but not limited to vision-in-the-loop systems. The major limitation is that the fast-sampled input u_h should be controllable.

5. Modelling and simulation of complex multi-axis systems, complex estimators

5.1 Overview of realized solutions

The aim of task 5.5 is to develop models that can be used within a digital twin. This means that they need to be flexible and fast enough to interact with real-time measured data and to be fitted into a model that reflects the real system. The combination of the real model and the digital twin makes it possible to improve the design as well as the operation of the physical object. Data from the operation helps to improve the digital twin (refinement of the model based on measurements) and to adapt the model if, for example, the physical objects suffer wear. The degree of change can be used to predict the remaining lifetime of the physical object. The digital twin can also be used to simulate behaviour (an example is thermal expansion due to temperature changes caused by friction, cooling, external conditions) that is difficult to control based on 'real' temperature measurements.

Developing models of complex systems is not something new, there are many methods to create accurate models of complex systems. The power of these methods is that they are very generic, they can be used for many different systems. This implies a high abstraction level of the building blocks e.g., an element in a finite element method. This high level of abstraction results in a high number of degrees of freedom and a long calculation time. This makes these models not suitable for the use in a digital twin, where in general many evaluations are needed in a relatively short time.

The goal of this task is to find appropriate methods to develop virtual models and convert the full order models into reduced models that can be used in a digital twin. In this context, a reduced order model should be able to deal with parameter uncertainties, to be easily interchangeable and ensure intellectual property (IP) security.

5.2 Addressed ST objectives and KPIs

This task is exploring the methods to develop model that can be used within the digital twins as described within the work package 5 objective, ST1. This task has not directly linked to any BBs or KPIs. It is however linked to many of the use cases, demonstrators and pilots and with that there is an indirect link to some of the building blocks. An overview of all links is provided in the table below.

SIOUX	DT for thermal behaviour of mechatronic system **[P1, BB8]**
TEK	Dynamic robot model **[UC2]**
UGR	System identification of flexible Cobots **[UC4]**
REDEN	ROM and DT of C-arm **[P4]**
SIEMENS (L)	System level modelling of lifts and Extended Kalman Filter model **[P2, P4, UC1]**
UWB (ZAPUNI)	Modelling for DT for collaborative robotics **[P4, UC4, BB5]**
VTT	Model of high reach mining boom
PMS	DT framework for C-arm **[P4, BB10]**

Methods to identify models for model-based techniques in the Smart Motion Control algorithms library techniques from BB5 have been developed, as well techniques to enable online adaptivity of these models. For BB6 *Improved performance monitoring, fault detection and predictive maintenance*, A digital twin is developed. This DT offers a robust and efficient tool for enhancing the performance of semiconductor manufacturing. The digital twin models developed in this task (contributing to meet KPI_BB6_2) can typically also have a role in providing indicators or estimators of system health and support in self-commissioning (e.g., adaptive calibration) and thus contributing to accomplish KPI_BB6_3. For BB10 advances have been made in Motion/path planning, collision avoidance and navigation algorithms: for safe testing on realistic multi-axis robot models (i.e., practically all KPIs in BB10).

5.3 Healthcare robot (REDEN, UWB, PMS)

5.3.1 Tech Overview

Philips Medical Systems [PMS] owns Pilot 4, a complex multi-axis robot system for Image Guided Therapy. To integrate emerging technologies like digital twin models, AI and Big Data safely and efficiently in our complex robotic systems the challenge is to develop and test these technologies with realistic robot models in simulation. There can be many different robot configurations to support and each robot model itself consists of many sub-models.

Therefore, we aim to:

- Automate the assembly of realistic robot systems from re-usable sub-models.
- Test new technologies and models with these robot systems in a flexible simulation environment.

In addition, within this task we have two partner contributions to our Pilot 4, being:

1. Digital Twin Model for fast and reliable collision detection by REDEN.
2. Digital Twin model for adaptive feedforward control with Current Calibration Tables by UWB.

Objectives for this activity aligns with the ST1, “the development of advanced model based and knowledge-based methods for building digital twins”.

5.3.2 Implementation aspects

We have created a framework that automatically assembles the realistic robot model from the sub-models, as shown in Figure 56.

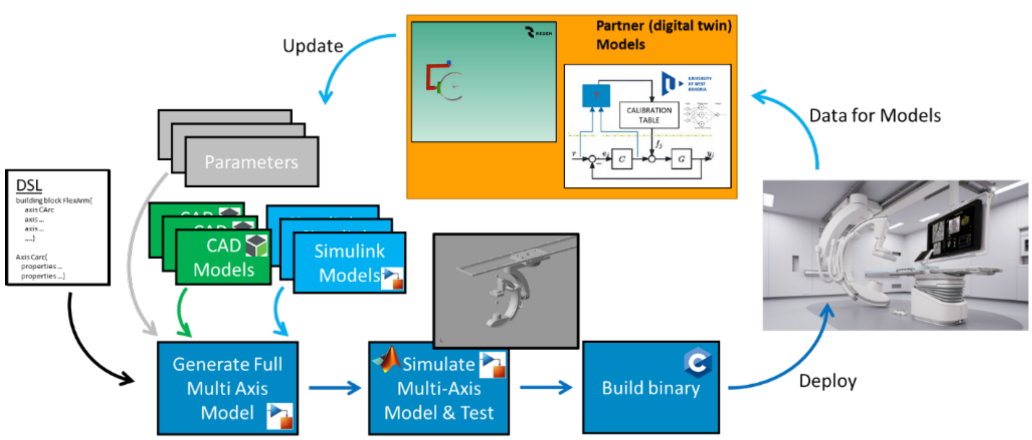


Figure 56: Automatic model assembly framework

(Sub-) model guidelines

We start with the foundational sub-models (Simulink and CAD models and their Parameters in Figure 56), which could be hardware components (e.g., motors, gearboxes) or software elements (e.g., controllers, algorithms). These sub-models are essential for building a complete multi-axis model. By adhering to the following guidelines, we can make the model generation process faster and simpler.

Develop Versatile Parameterized Sub-models: Make adaptable and re-usable sub-models with parameters where possible. This lets us use a sub-model for different robots and configurations, saving time and effort on managing models, interfaces, and versions. For example, a motor model that shows the current-torque relationship can be customized for a specific robot axis or design by changing the motor constant parameter.

Minimize Interface Quantity and allow for adaptability: Choose the division points and interfaces for your sub-models wisely, as they affect the overall model complexity. Also, sub-models may change over time and may need extending the interfaces. Take this into consideration with the chosen environment. If you for instance use Simulink, use buses to organize inputs and outputs easily.

Referenced Models/Subsystems: Simulink has a feature called referenced model/subsystem, which lets you link sub-models to the full model. This helps you reuse sub-models and keep them updated with any sub-model changes.

Establish a Catalog of Referenced Sub-models for Reusability

We maintain adaptable, reusable sub-models to create a catalog of modular components for our multi-axis robot model. This sub-model reuse strategy has many benefits, such as reducing maintenance and version control work. Also, any changes to a sub-model will update all models that reference it. These models follow the Sub-model Guidelines we mentioned earlier.

Utilizing a DSL to Define the Complete Multi-Axis Robot:

We need to find the exact sub-models, quantity, and configuration for our robot. We also need to choose the parameters to customize the generic sub-model for a specific robot axis. This process requires describing the robot's structure, components, and functionalities. A Domain Specific Language (DSL) (in Figure 56 shown as DSL next to the models and parameters and input to the model generation) helps us here, giving us a high-level language that expresses the robot's composition. This language is then converted into an output that shows the connections among all sub-models and the parameters needed to make them specific to robot or axis.

A pseudo variant of the DSL is represented in Figure 57 and indicates how that visually would result in the sub-model structure and parameters. In this example the *configuration* is a full system consisting of a set of *building blocks,* which are for instance a robot-arm (of the type *Flexarm* in this example), patient support, monitors, etc. The robot arm *building block Flexarm* in its turn consists of multiple *axis* and each *axis* is defined by a set of *properties*.

Using the DSL's input, we can generate an output that describes the sub-models, their connections, and parameters. This output facilitates the assembly of the full robot model from the sub-models. In our scenario, utilizing Simulink, we employ MATLAB to programmatically assemble the Simulink model in accordance with the DSL's output file, ensuring a seamless and accurate model construction simplified by adhering to the Sub-model Guidelines and the Catalog of Referenced Sub-models previously discussed.

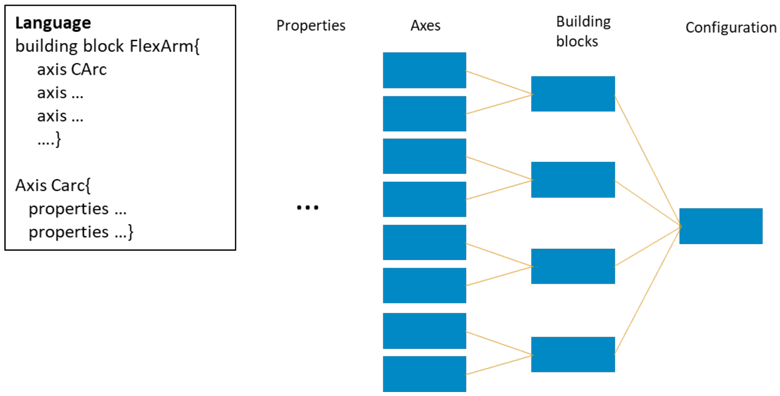


Figure 57: DSL to define the complete robot system configuration

Simulation environments

With the full model, we can simulate the robot to check motor currents or integrate partner models and test their performance. We can also visualize the robot's behaviour with trace signals and realistic visuals. We have explored different levels of visualization, from Simscape Multibody in Simulink to virtual reality in Unity.

Deployment:

Once everything is sufficiently tested in simulation the next step is the deployment to actual target hardware (in Figure 56 depicted as the build binary and deployment step). This allows to re-use of the now already verified models in simulation to be deployed to actual hardware for the final verification and validation.

5.3.3 Results

Digital Twin Model to reliably and quickly detect collisions (REDEN)

The goals of REDEN in this task are twofold:

- Find a practical way to make reduced order models that can be shared for cooperative purposes.
- Expand our knowledge on digital twins and model/data interaction in general.

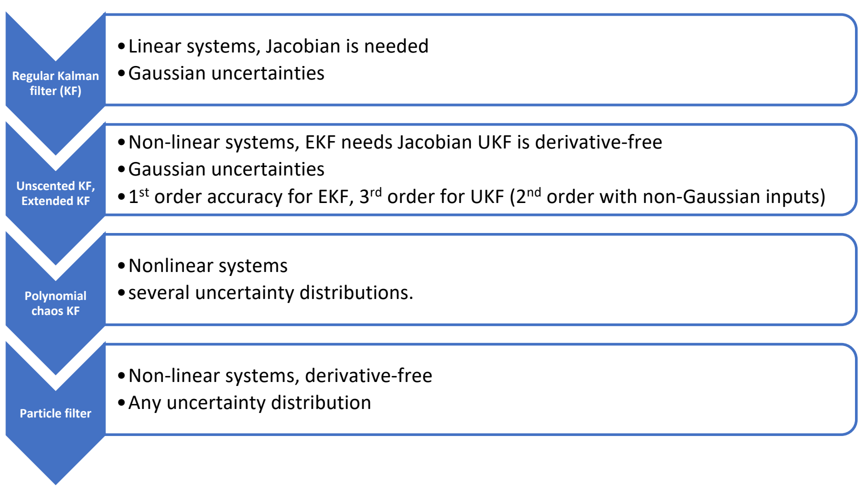


Figure 58: A few types of on-line filters sorted from fast and specific (top) to general and slow (bottom). Regular KF and particle filter have been implemented in REDEN's Digital Twin Framework.

Since digital twins operate on models and measurement data, Kalman filters are often a good fit for many digital twin services like state estimation, parameter estimation, error detection, maintenance planning. Therefore, Kalman filters (several types) are the first services that we have implemented in our digital Twin framework. Figure 58 shows several types of Kalman filters.

Next, we have been investigating how we can conveniently include system models in our Digital Twin Framework. More specifically, we investigated how we can use Amesim (Siemens software) to this end. We have found a couple of different ways to do this. It depends on the model which of these ways are available.

Co-simulation

Amesim has a Python interface that can be used to change parameter values and do time integration. This is enough to couple it to our Kalman filter. The filter sets the initial values in Amesim and then Amesim integrates the equations over a short time interval. Unfortunately, the computation in Amesim is very slow: 18 minutes with default settings and 1 minute if we select the 4th order explicit Runge-Kutta solver. That solver is not available for 3D mechanics. Figure 59shows the results of a Kalman filter coupled to Amesim using Co-simulation.

Linearization

Amesim can linearize the model at any point in time. The linear model can be exported and used in the DT framework. Time integration can then be done outside the Amesim environment. The same problem now takes 55 milliseconds to solve. The main drawback is that this is only convenient if the model is linear. Integration of non-linear models that are linearized at several points is not trivial, let alone convenient. The results in Figure 59 are also valid for coupling a Kalman filter to a model that was extracted from Amesim using linearization. State estimates (orange lines) from noisy position sensor data (green dots) and a linear model. Despite the noisy input, the prediction follows the actual state (blue dots) well and remains withing the predicted error range (3σ, gray dots). The results from co-simulation (integration in Amesim) and linearization (integration outside Amesim) are similar; therefore, only one set or results is plotted.

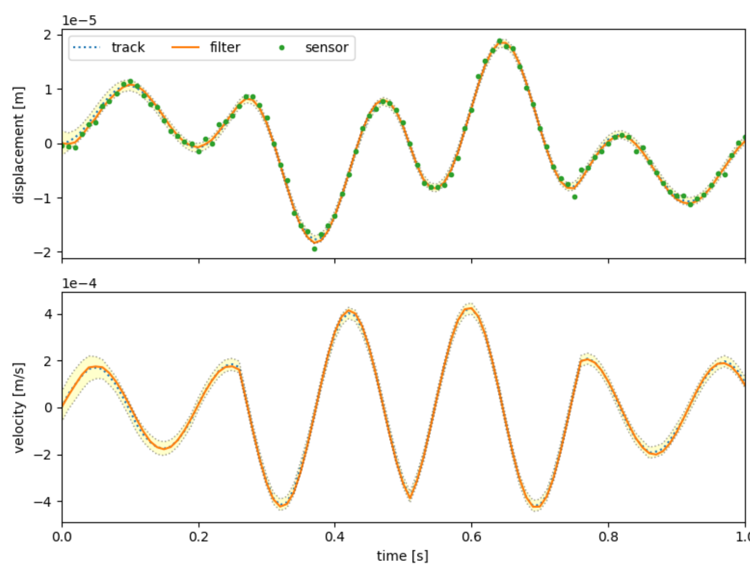


Figure 59: Results of the Kalman filter

Rom Builder

Amesim includes a useful app called ROM Builder (Reduced Order Model). This can be used in different ways to fit a neural net or another model to data from measurements or from Amesim simulations. The resulting fit can be exported in several formats which is highly appreciated. The proposed way of operation the ROM builder (on time series data) fits a Recurrent Neural Net. This is not useful for the Digital Twin Frameworks, because the resulting model is not accurate for initial transients.

Nevertheless, we found another way to use the ROM builder for exporting models. First, we create an Amesim simulation (with or without the 3D mechanical library) and do a transient simulation with output at fixed time steps (Kalman filter frequency). Next, we can train a model with the ROM builder (static data model) on the state transitions in this large data set. The ROM now becomes a time stepper that can be used in a Digital Twin; see Figure 60. Inputs are the state and forces at start of the sample interval and outputs are the state at the end of this interval (or change in state over this interval). The figures show an example of a 3D mechanical system with 2 DOFs that is exported in this way (only gravity, no forces/torques); see Figure 61 and Figure 62. This works well and is not limited to linear models.

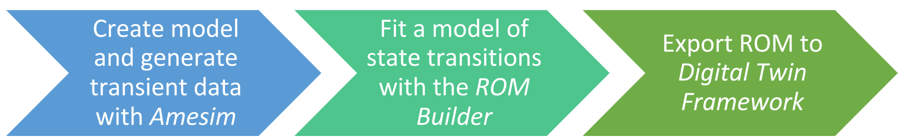


Figure 60: Our approach of using Amesim to generate models for the Digital Twin Framework

A drawback of this method is that we must be sure dataset generated for training and the fitted model contain all relevant dynamics. This requires skills that do not completely overlap with those of modelling. Nevertheless, we are happy with this approach.

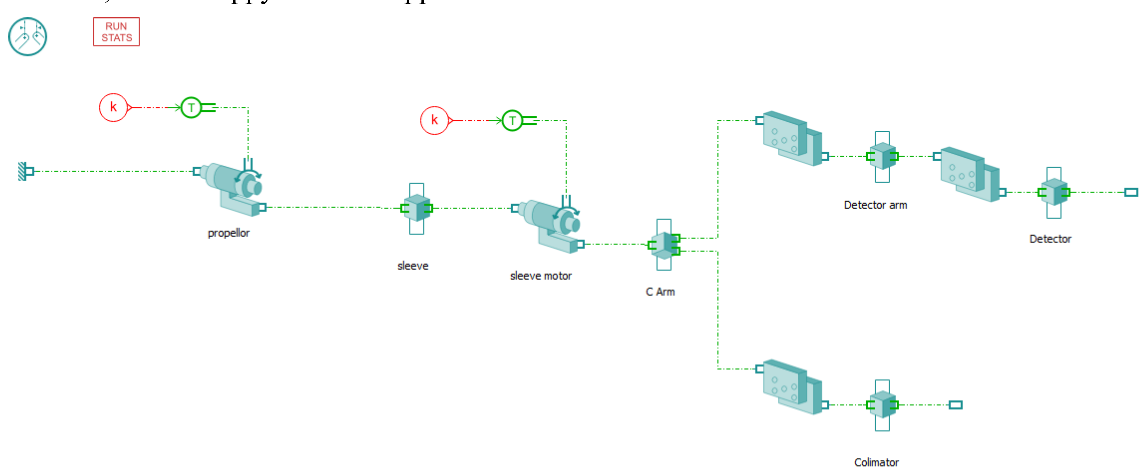


Figure 61: 3D mechanical model with two actuators

3D mechanical library: issues

The documentation of the 3D mechanical library mentions some limitations that explain issues that we have encountered and that are relevant for digital twin usage. First, the model is solved as a DAE rather than an ODE. For example: the linearization of a simple 3D mechanical model with 1 degree of freedom resulted in a linear model with 95 degrees of freedom. Second, velocity constraints are not respected at assembly.

The Kalman filter needs to prescribe the initial velocity at each integration step. So, both limitations are problematic for operation with the Digital Twin Framework.

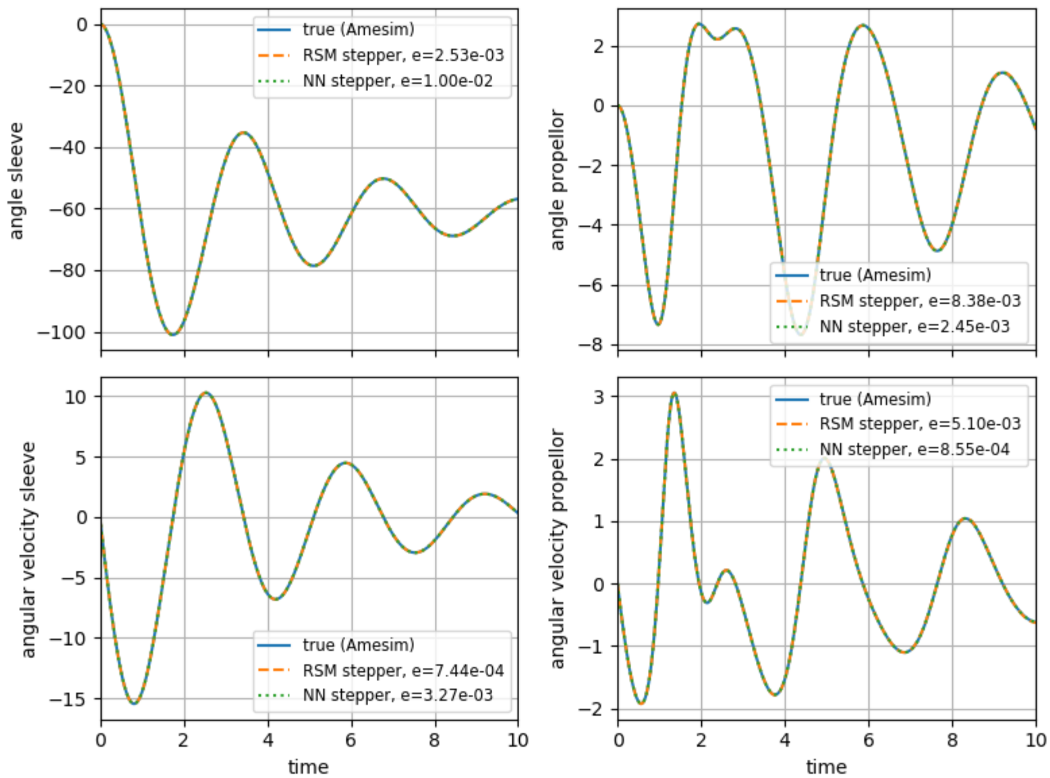


Figure 62: Amesim versus quadratic Response surface model vs Neural network model; both fitted with the ROM builder. The response of the fitted models matches the full model response well.

Digital Twin model for adaptive feedforward control using Current Calibration Tables (UWB)

The Pilot 4 medical manipulator of Philips Medical Systems relies on current calibration tables [CCTs] for precise motion control of robot axes (see Figure 63). These tables, depicted in Figure 64, are initially calculated during machine commissioning to ensure optimal performance. However, natural wear and tear, like joint and gear friction, and cable flexibility, require periodic recalibration of the tables to maintain accuracy and precision, ensuring consistent machine performance over time.

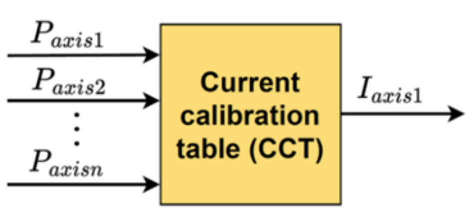


Figure 63: Schematics of a Current calibration table feedforward – a multivariate function mapping the desired robot joint positions to required actuator currents

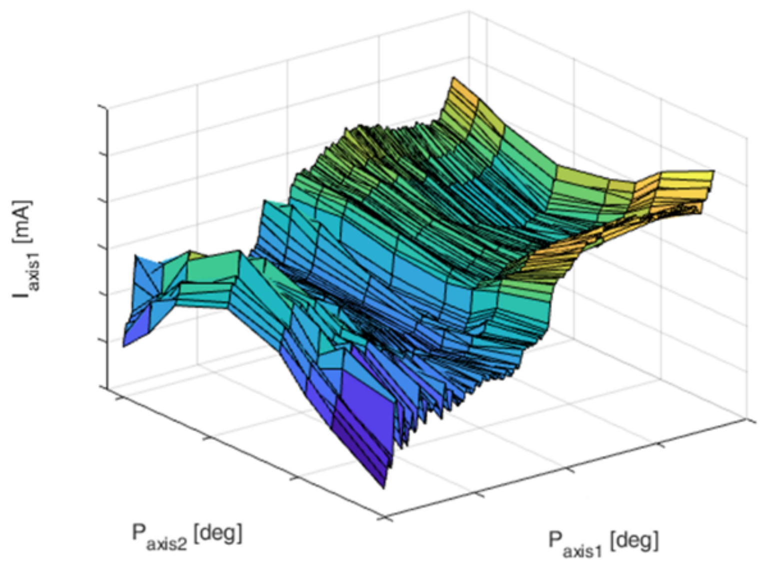


Figure 64: Visualization of a particular calibration table by means of a 3D mesh

UWB developed a Digital Twin model representing the CCT objects that can be updated with live data acquired from the robot operation, maintaining its consistency continuously over time to avoid costly periodic recalibrations.

The Digital Twin model fulfils three main objectives:

1. It provides a systematic way for storing the CCT information and retrieving it during real-time operation of the medical robot to provide current feedforward for the motion axes.
2. It provides a framework for updating the CCT by merging the actual state of the model with new dataset recorded during a period of operation of a particular machine.
3. It may provide valuable diagnostic data by storing the deviations of the adapted CCT model with respect to the initial calibration made during the machine assembly and commissioning.

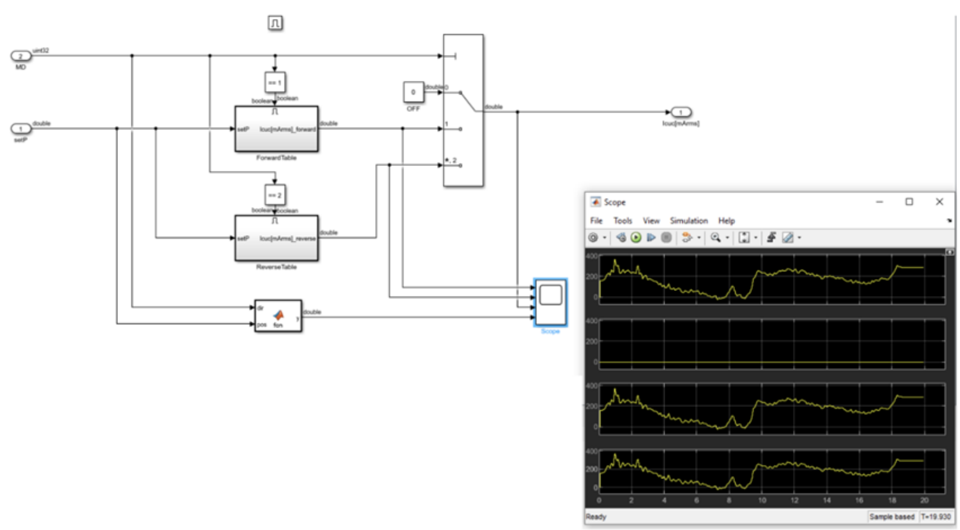


Figure 65: Validation of the real-time CCT interpolator algorithm

The real-time part of the CCT generation algorithm was implemented in a C-code and embedded in a Simulink functional block that was integrated into simulation and development environment of PMS.

The next part of the model involves the motion data postprocessor used for extraction of important motion patterns from the recorded robot trajectory.

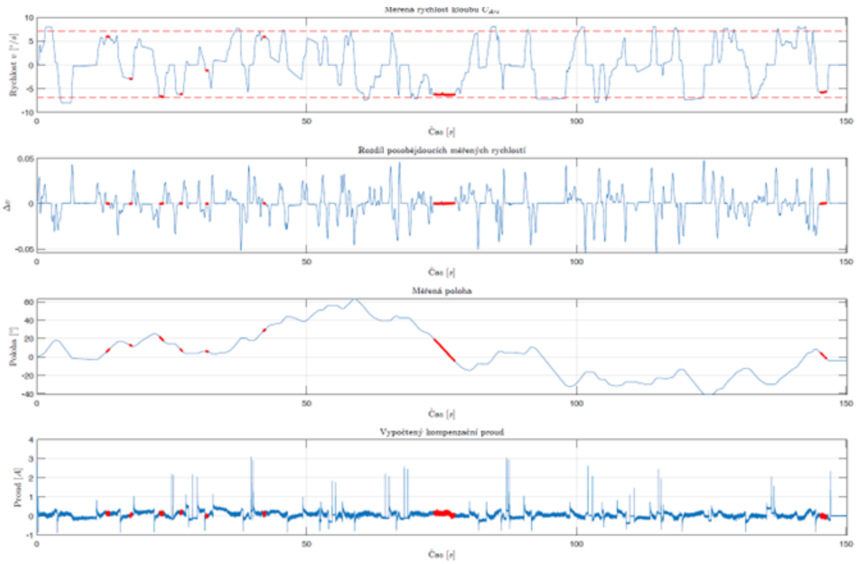


Figure 66: Automatic processing of motion trajectories (velocity, acceleration, position, motor current) – red marks designate data points used for the adaptation of the CCT model

The last submodule is responsible for merging the acquired motion signatures with the existing CCT to adapt its shape.

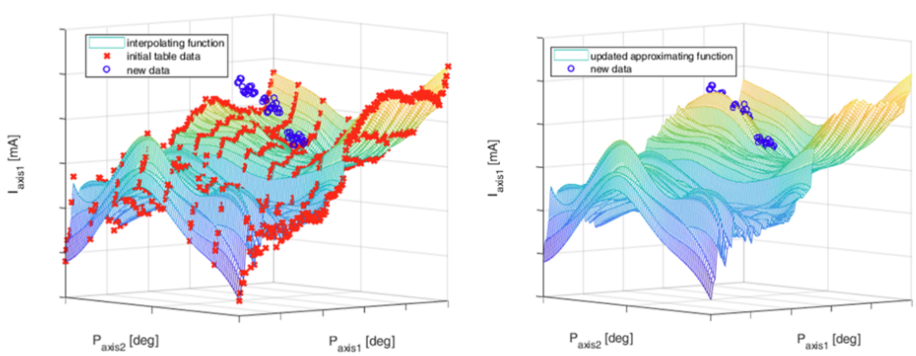


Figure 67: CCT data assimilation step – before (left) & after the model adaptation (right)

5.3.4 Capabilities & Limitations (including USP, strengths & weaknesses)

Using System simulation software like AMESIM gives great flexibility in modelling the system with different domains. However, integrating it within a digital Twin framework where the model needs updating does pose some limitations, especially when using the 3D mechanical library.

The Digital Twin model develop by UWB representing the CCT objects helps to maintaining its consistency continuously over time to avoid costly periodic recalibrations and can be updated with live data acquired from the robot operation.

The development and learnings from the work done by REDEN on a Digital Twin Model to reliably and quickly detect collisions can offer future opportunities to further enhance collision detection on our robot configurations. Similarly, the work from UWB on the Digital Twin model for adaptive feedforward control using Current Calibration Tables offers us the opportunity to investigate the future possibilities of adaptive calibrations to further reduce scheduled maintenance downtime.

5.3.5 Customizations & Adaptations (including possible modifications and extensions)

Both modelling methods have been built specifically for the medical robot use case but can be easily adapted for different applications. Special attention should go out to the speed at which the models can do their task since this will differ between applications.

5.4 Lift application (Siemens)

5.4.1 Tech Overview

WEG Italy owns Use Case 1, a lift application, where the critical performance is speed/flight time, accuracy, vibration compensation, and comfort.

The current study is performed by using the modelling and simulation capabilities of Simcenter Amesim which is a versatile system simulation platform enabling design engineers to virtually evaluate and enhance the performance of systems.

In the virtual model the elevator's drive system controls the elevator's start and acceleration, smooth operation, and braking deceleration. Traction elevators operate via a pulley system, using steel ropes and a counterweight to move the cabin up and down. The main structure the elevator is composed of five distinct parts:

- Hole: refers to the space within the building designated for the movement of the elevator car and contain various components such as the counterweight, guides, limit switches, sensors, doors, and cables.
- Machine room: inside it is located the motor, the tractor group, the protections, and all elements of controllers.
- Cabin: serves as the platform for users to travel between different levels in the building. It is designed as a closed structure equipped with automatic doors for passenger entry and exit.
- Counterweight: is positioned on the opposite side of the traction cable to balance the weight of the elevator car and its occupants, resulting in a reduced power requirement for the motor.
- Pit: is situated at the bottom of the elevator installation and houses safety dampers and other security devices.

5.4.2 Implementation aspects

The virtual prototyping of the lift system involved multidisciplinary approach, where the main implementation aspects was:

- Speed Time:
  - Optimization of the control algorithms for the drive system to achieve maximum acceleration and deceleration rates while ensuring passenger comfort and safety.
  - Fine-tuning the motor and drive system parameters to minimize response time and maximize efficiency.

    - Utilizing predictive control techniques to anticipate passenger demand and adjust elevator speed accordingly to minimize waiting times.
- Control accuracy:
    - Ensuring precise control of the virtual elevator position throughout the travel range using feedback sensors such as encoders or position sensors.
    - Implementing control algorithms to continuously adjust motor torque and speed to maintain the desired position accurately.
    - Calibrating virtual sensors and actuators regularly to maintain accuracy over time and under varying operating conditions.
- Vibration Compensation:
    - Modelling the dynamic behaviour of the elevator system, including the interactions between the cabin, counterweight, ropes, and other structural components.
    - Implementing vibration compensation algorithms to mitigate oscillations and resonances that occur during acceleration, deceleration, or changes in load.
    - Use active damping techniques such as adaptive control or vibration absorbers to suppress vibrations and ensure a smooth ride for passengers.
- Comfort:
    - Designing the elevator cabin and suspension system to minimize noise, vibrations, and discomfort during operation.
    - Optimizing the cabin layout, including interior design, lighting, and ventilation, to enhance passenger comfort and create a pleasant riding experience.
    - Incorporating ergonomic features such as handrails, seating, and floor surfaces to improve passenger comfort and accessibility.
- Traction Elevator Implementation Aspects:
    - Modelling the pulley system, steel ropes, and counterweight dynamics accurately to simulate the behaviour of the traction elevator.
    - Optimizing the traction drive system, including motor sizing, gear ratios, and traction control algorithms, to achieve efficient and reliable operation.
    - Considering factors such as rope wear, tensioning mechanisms, and safety devices to ensure the longevity and safety of the elevator system.

5.4.3 Results

The lift models (Figure 68) developed during the IMOCO4.E project simulate the lift systems dynamics behaviour and the controls.

The main aim of the activity was to model, simulate, validate, and perform a dynamic simulation of the elevator. The elevator's input command corresponds to the target floor requested by passengers where the signal is directed to a state chart, which subsequently governs the various operating phases.

The passengers' requested elevator route follows different sequence where the presented picture simulates the sequence: starting from the ground floor, it goes to the third floor, returns to the ground floor, and then proceeds to the third floor again. Throughout this route, the number of passengers using the elevator also varies. Figure 69 shows height of the floor input command, the passengers inside the elevator, elevator position request and the elevator position.

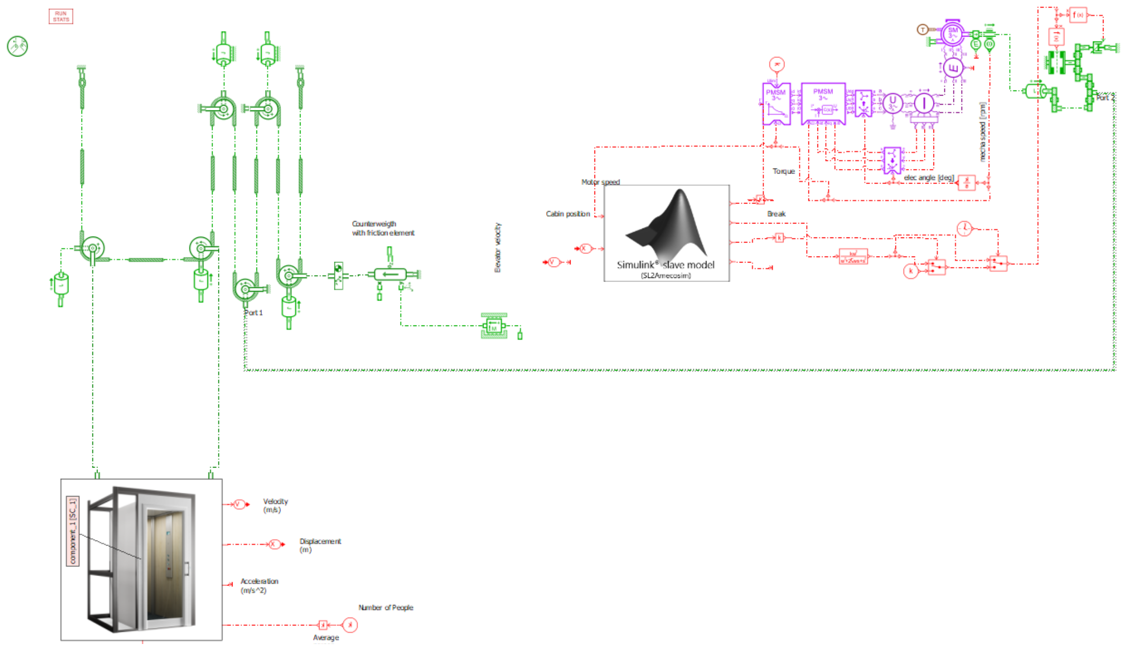


Figure 68: Simcenter Amesim 1D lift model

The dynamic behaviour of the elevator is linked to the changes in cabin speed and acceleration. The controller produces quantity u to align the actual position of the elevator cabin with the target position. In the simulation model, u denotes the torque necessary to attain the desired displacement profile within predefined limits. The speed, the acceleration and the torque required are presented in Figure 69 right.

The mode of activation of the brake depending on the evolution of the cabin speed is shown in Figure 70.

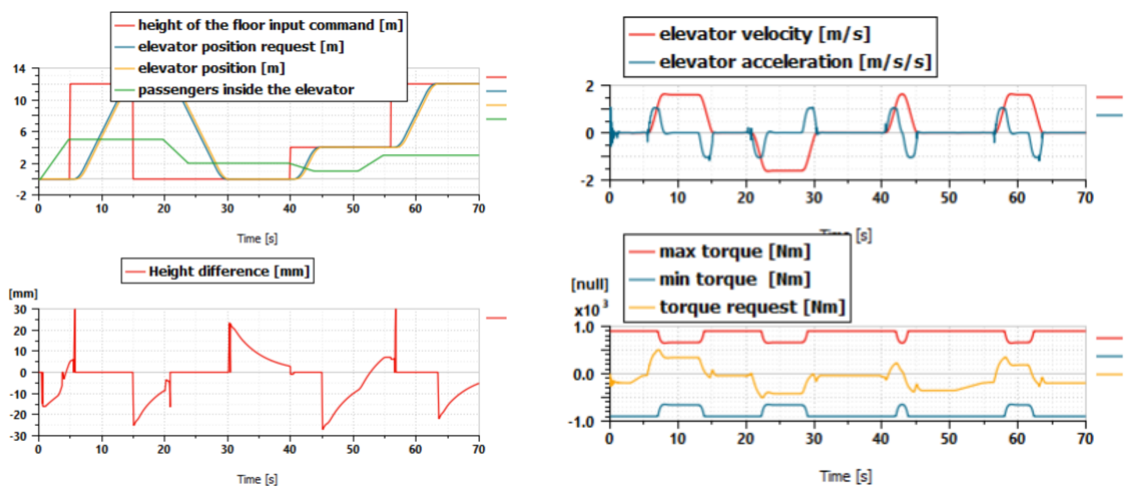


Figure 69: Floor input command and passengers inside the elevator (left) & Elevator cabin velocity, acceleration and requested torque (right)

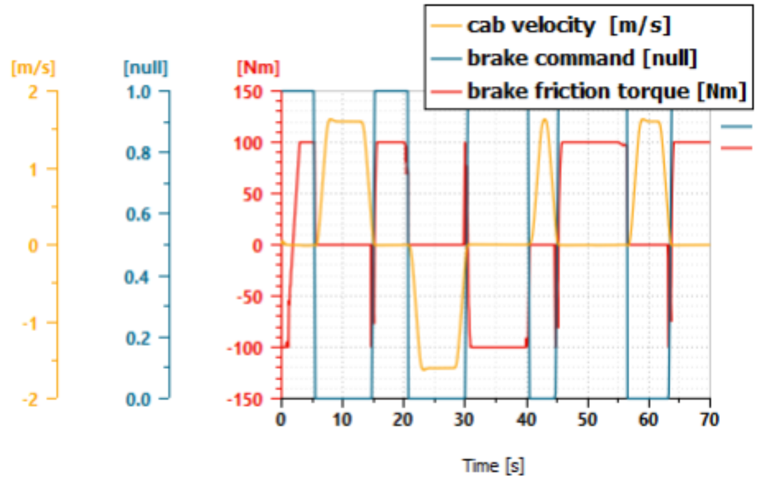


Figure 70: Brake command, car velocity and brake friction torque

In order to study the accuracy of the developed elevator model, the results obtained through simulation are compared with the measured ones where the results are obtained from the same simulation and testing conditions. Figure 71 present the influence of the variation of certain numerical parameters on the simulated drive torque for two different speed conditions.

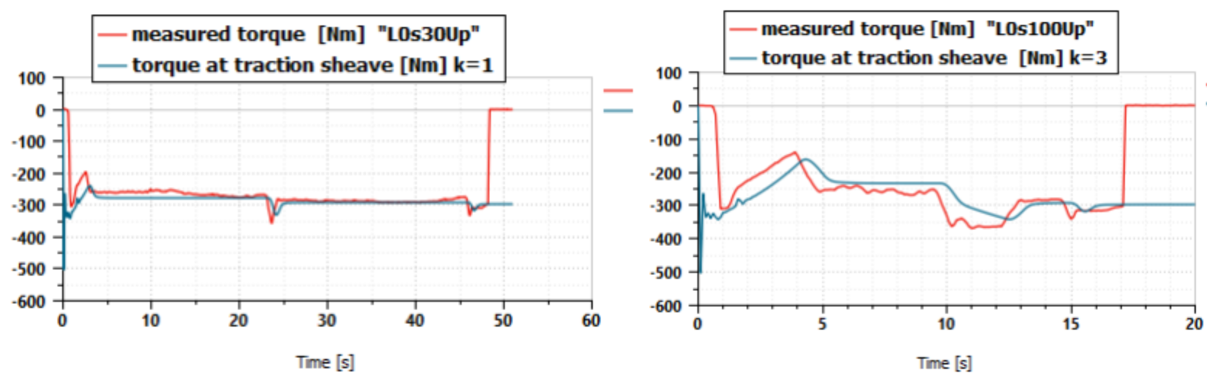


Figure 71: Measured and simulation torques, load 0, 0.3 m/s speed, Up (left) & load 0, 1 m/s speed Up (right)

5.4.4 IMOCO4.E requirements and KPIs

Key Performance Indicators (KPIs):

To measuring the performance of the elevator's drive system through simulation with Simcenter Amesim, KPIs included was energy efficiency, ride quality, system responsiveness, reliability and durability, noise and vibration levels. By monitoring these indicators, the simulated drive system not only optimizes energy consumption and passenger comfort but also shows robustness, safety, and compliance with industry standards. Through continuous analysis and refinement based on these KPIs, the virtual model was tuned to precise emulate real-world elevator operations.

Requirements for Virtual Model:

Creating a virtual model of the elevator's drive system demanded a set of comprehensive requirements to ensure its accuracy. These requirements included the integration of correct component models for motors, pulleys, ropes, and brakes, as well as the incorporation of realistic environmental conditions. To regulate speed, acceleration and braking dynamic control algorithms were implemented. To ensure the system fidelity, the virtual model was compared with measured data from physical elevator.

5.4.5 Capabilities & Limitations (including USP, strengths & weaknesses)

Simcenter Amesim user-friendly interface and extensive library of pre-built component models, helped to create rapid models and simulation setups. Additionally, its integration with MATLAB/Simulink enables the integration of control algorithms, enhancing its versatility. By using these capabilities, lift control algorithms developed by UNIBS in MATLAB/Simulink was tested in the virtual lift model developed by SISW.

Simulation platforms has also some limitations. Despite robust simulation capabilities, may lack some granularities required for modelling highly specialized components or phenomena within the elevator drive system. It was not the case in our study.

5.4.6 Customizations & Adaptations (including possible modifications and extensions)

Considerable flexibility for customizations and adaptations to meet specific needs in modelling elevator drive systems was made. Custom component models adapted to the specific characteristics of the elevator's drive system was developed. Customization options for control algorithms where provided, allowing to fine-tune parameters and logic to optimize lift performances under varying operating conditions. Simcenter Amesim's open architecture facilitated collaboration and knowledge sharing within interdisciplinary teams, facilitating the integration of expertise from different domains to support the complex challenges in modelling elevator drive systems.

5.4.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

For modelling elevator drive systems, a structured methodology encompassing model development, simulation setup, analysis, validation, and optimization where developed. This methodology involved defining system requirements, selecting appropriate component models from Simcenter Amesim's extensive library, and configuring simulations to represent real-world operating conditions accurately. Integration with MATLAB/Simulink enabled the incorporation of control algorithms, facilitating closed-loop simulations to evaluate system performance under dynamic control strategies.

5.5 Model for collision detection in hard real-time CNC application (FAG)

5.5.1 Tech Overview

The following SW catalogue components have been used to develop the model and its application for Virtual Commissioning:

id	Name	Relevant task	BB	Layer
SW-092	Robot model	T5.5	6	3
SW-093	Collision detection algorithm	T5.4	6	3
SW-094	Model reduction	T5.5	6	3

SW-095	Virtual commissionning for trajectory planning	T5.4	6	4
SW-096	Digital model for a complex system which includes a machine-tool and a robot	T5.4	6	4

5.5.2 Implementation aspects

The developed model has been successfully implemented both in Use Case 2 and in the FAGOR CNC simulator. To do this, these integration tests defined in the IMOCO4.E project catalog have been followed.

TEST-038	SW-096	Program Simulation time vs execution time.
TEST-041	SW-092	Feed the Robot with G-code programs. Unexpected errors shouldn´t arise.
TEST-042	SW-095, SW-096	Test all the manual operations and Jog movements.

5.5.3 Results

Use Case 2 focuses on the integration of an industrial robot as a subsystem of a CNC-controlled machine tool. Within the IMOCO4.E project, the development of an application for Virtual Commissioning has been addressed, specifically oriented to program the movements of the machine and robot axes, focusing the use case on collision detection, essential to ensure a correct start-up of the system. For this task, a model of the complete system (machine + robot) has been carried out, looking for a compromise between being geometrically accurate to detect collisions and not having a too high computational cost.

The starting point is the import of the 3D CAD model. From that file, each element has been simplified to get a valid model to run on the CNC hardware. Figure 72 shows the example for the CAD model of the Stäubli robot. Equivalent work has been done with the model of the 5-axis machine.

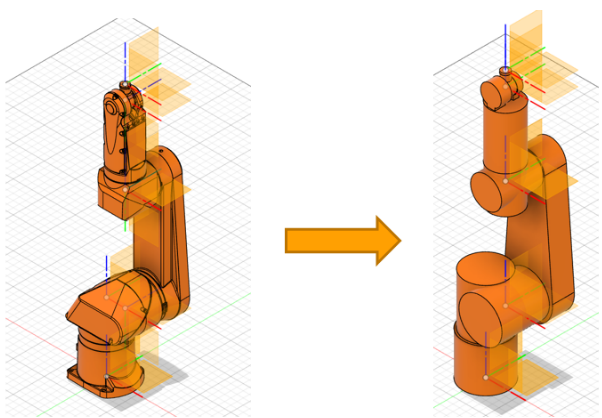


Figure 72: Reduction of model of the robot

Once this work was done, the complete model was integrated into FAGOR's CNC simulator, taking advantage of its SIL environment to program and test configurations and programs for the real system.

To validate the Virtual Commissioning application, in addition to including the robot and the 5-axis machine, collision rules have been programmed to detect invalid paths. In this way, the CNC simulator provides an environment suitable to perform Virtual Commissioning tasks. The movement of the robot axes is programmed using the ISO language, common in machine tool applications. The application is able to execute programs automatically or allow manual movement of the robot joints, verifying the validity of the trajectories and alerting about possible collisions.

Figure 73 shows how the robot executes a valid trajectory, while, in case of collision, the simulator warns of the collision.

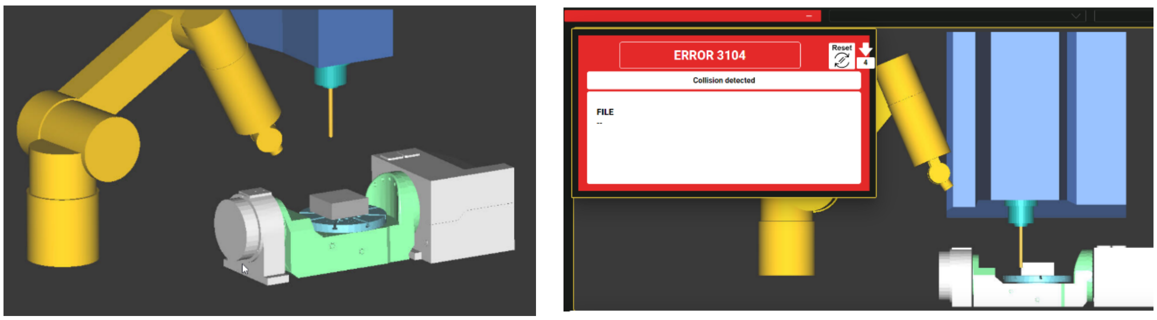


Figure 73: Example of collision detection

This approach would allow on-machine simulations and testing of new features before implementing physical changes to the axes. The developed application represents a significant advance towards the efficient coordination of complex systems in the serial production environment.

5.5.4 IMOCO4.E requirements and KPIs

This development has fulfilled the following technological and functional requirements included in the IMOCO4.E proposal:

Technological requirements / challenges (performance, precision, novel sensing, and actuation):
✓ Integration of a robot manipulator within a machine tool so that the axes of both systems are synchronously controlled by the CNC, increasing the robot functionalities, and eliminating the robot controller programming [CH6].
Functional requirements – Digital Twin (connectivity, communication, data, services):
✓ Set-up stage of the machine development process is critical and can be reduced by the application of Virtual Commissioning techniques. ✓ Virtual representation of machines in CNC HMIs is critical for operation experience and also minimizes undesired events, as collisions [CH7].

Likewise, the following KPIs have been met:

Functional KPIs (connectivity, communication, data, services):

Functional indicator	Baseline (2020)	IMOCO4.E (2022, M20)	IMOCO4.E (2024, M36)

Collision avoidance	Not available for machine + robot system	Machine +Robot system in Simulator	Machine + Robot system in real demonstrator

5.5.5 Capabilities & Limitations (including USP, strengths & weaknesses)

The development of this feature is a significant step forward for a CNC manufacturer in its offering to the end user of the machine. Considering that a system consisting of a 5-axis machine and a robot is a complex system with many degrees of freedom, this solution reduces start-up times and increases process safety.

The main limitation of the development is that it is closely linked to the CNC manufacturer. Each CNC model has its own interface and its own programming languages, so customization is unavoidable.

5.5.6 Customizations & Adaptations (including possible modifications and extensions)

The process of creating the model is readily exportable to any alternative system or application. Commencing with the 3D CAD model, one can replicate the identical procedures implemented in this instance. An expansion of the project could involve automating the currently manual task.

Regarding collision control, it becomes imperative to undertake individualized customization for each CNC manufacturer.

5.5.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt

Autodesk Fusion 360 has been used for the creation and simplification of the model. It is a general-purpose commercial tool that could be used for any other model. Once generated, the model is exported to an STL file that can be imported by the CNC for testing. From here the process becomes iterative until a model with a computational cost low enough for hard real-time execution is achieved.

It would be very interesting to have some kind of tool that can be integrated into the CNC to do this process automatically.

5.6 Deflection modelling of a high reach mining boom (VTT, Normet)

5.6.1 Tech Overview

Normet Oy manufactures mining equipment for various tasks in mining sector. In pilot 5 the main focus is on collision free path planning of a charging boom. Deflections does not play a significant role in this application. In concrete spraying on the other hand the reach of the boom is much longer, and deflections and clearances have a significant effect to the accuracy of the boom movements and a method for compensating them is a useful feature.

In Figure 74 on the left, a virtual model of the concrete spraying boom modelled with Mevea simulation software is presented. Figure 74 (right) shows a real concrete spraying boom prototype at VTT's laboratory. This boom prototype has been used as a test bed for developing various boom control features, that are currently in use in Normet products. During IMOCO4.E project the concrete spraying boom prototype was used as test case for studying the deflections and clearances of a high reach boom.

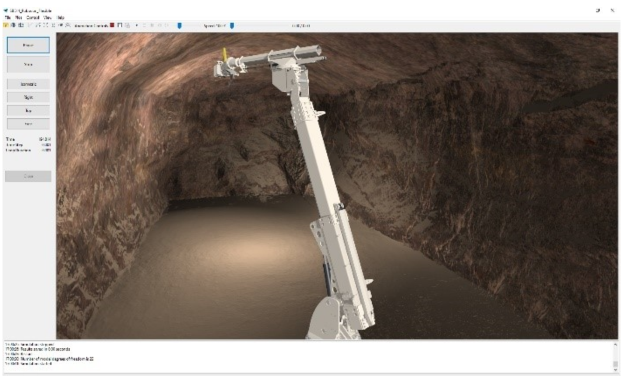
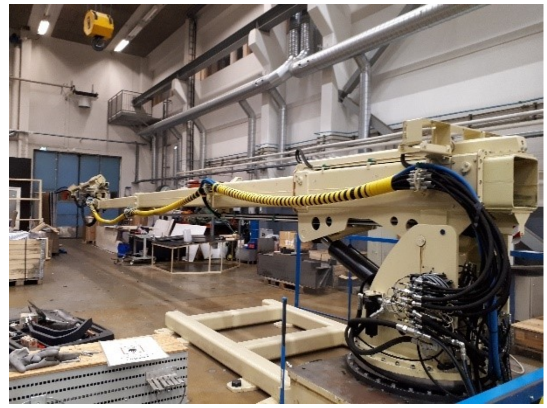



Figure 74: Simulation model of the Normet concrete blasting boom (left) and a real boom prototype at VTT's laboratory (right)

The concrete spraying boom has 9 degrees of freedom and about 16 m reach. With high reach the deflections and clearances of the boom start to affect to the positioning accuracy of the boom. Especially long linear axis cause inaccuracy. This is not a problem, when operator controls the boom manually. If target location and orientation of the nozzle is given in Cartesian co-ordinates, e.g., executing pre-planned paths, accurate boom positioning is needed. Modelling of deflections and clearances are studied to compensate their effect to the positioning of the boom. The goal is to develop a model, that can be used for compensating boom deflections in real time based on the measured positions of the joints.

5.6.2 Implementation aspects

Compensating the deflection and clearances must be possible to execute in real time with the existing control hardware of the machine. Control system consists of embedded control modules, where algorithms coded with c-language can be executed. Compensation method was in this case developed with MATLAB and it is possible to port the compensation algorithm to the control hardware of the boom. Developed method for modelling deflections utilizes existing position sensors of the boom and no additional sensors are needed. This enables that deflection compensation can be added to existing and new booms as a software feature.

5.6.3 Results

Boom deflection measurements

In order to determine the actual deflections and clearances of the concrete spraying boom, shape of the boom was measured in different boom positions. This was done by attaching 40 reflective markers to the boom and measuring the position of these markers in different boom positions with a Leica optical total station (Figure 75 left). Collected data was used as a reference data for deflection modelling.

Figure 75: Boom deflection measurements with a Leica total station (left) and markers attached to the boom (right)

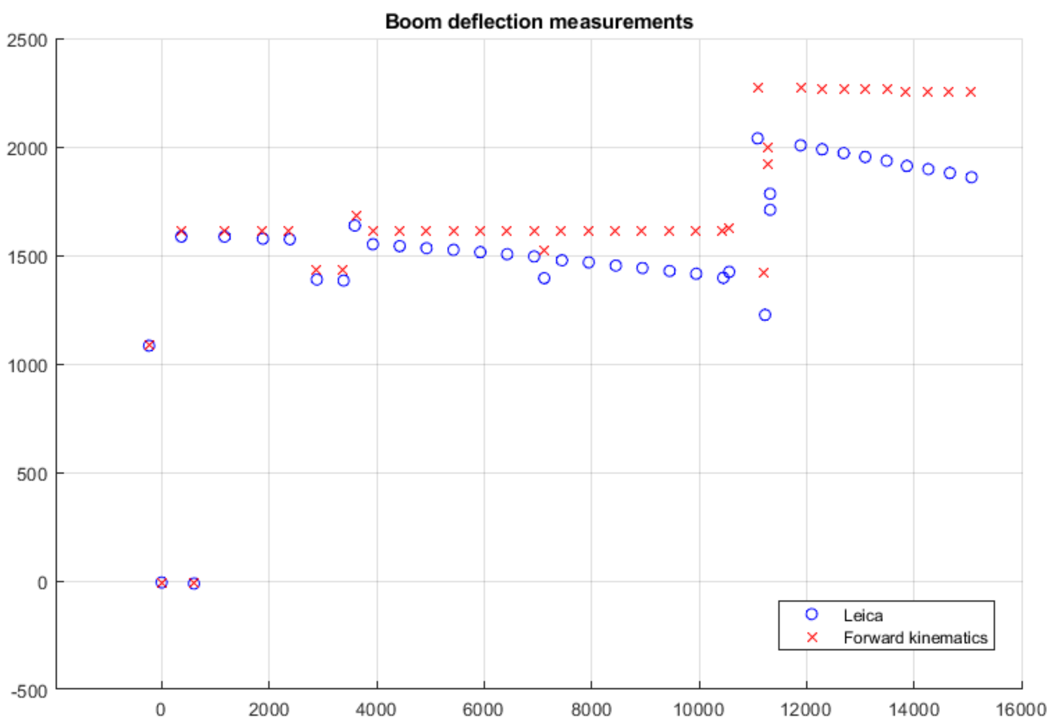


Figure 76: Deflection measurements with Leica Total station compared with position calculated based on forward kinematics, when boom is fully extended

Deflection modelling

In deflection modelling the linear movements of the boom (zooms) were modelled as a series of flexible beams with two supports. Also clearances were modelled with a geometric model. Parameters of the model were determined based on CAD models of the boom except clearances, that were determined experimentally. Inputs for the model are measured joint values and output is the calculated shape of the boom.

Deflection and clearance model calculation was implemented by utilizing MATLAB. Values were compared to measurements with Leica. When boom is fully extended, the maximum error in the end the boom is 395 mm without compensation (Figure 76). In Figure 77 the shape of the boom is calculated based on the deflection model. In this case the maximum difference of the deflection model to the Leica measurements is 32 mm in the end of the boom (measurement point 2 located at end of the boom).

When comparing measurements and calculated shape from all the measured boom positions, maximum difference in the end of the boom is 57 mm at 11.7 m reach. Although boom deflection and clearance model is a simplification, results indicate that it can be used for estimating the boom shape at the end of the boom at about 60 mm accuracy. For concrete spraying application this is sufficient. Model is applicable for other boom models with long linear zooms. Simplified calculation method can be executed in real time by the control system of the work machine.

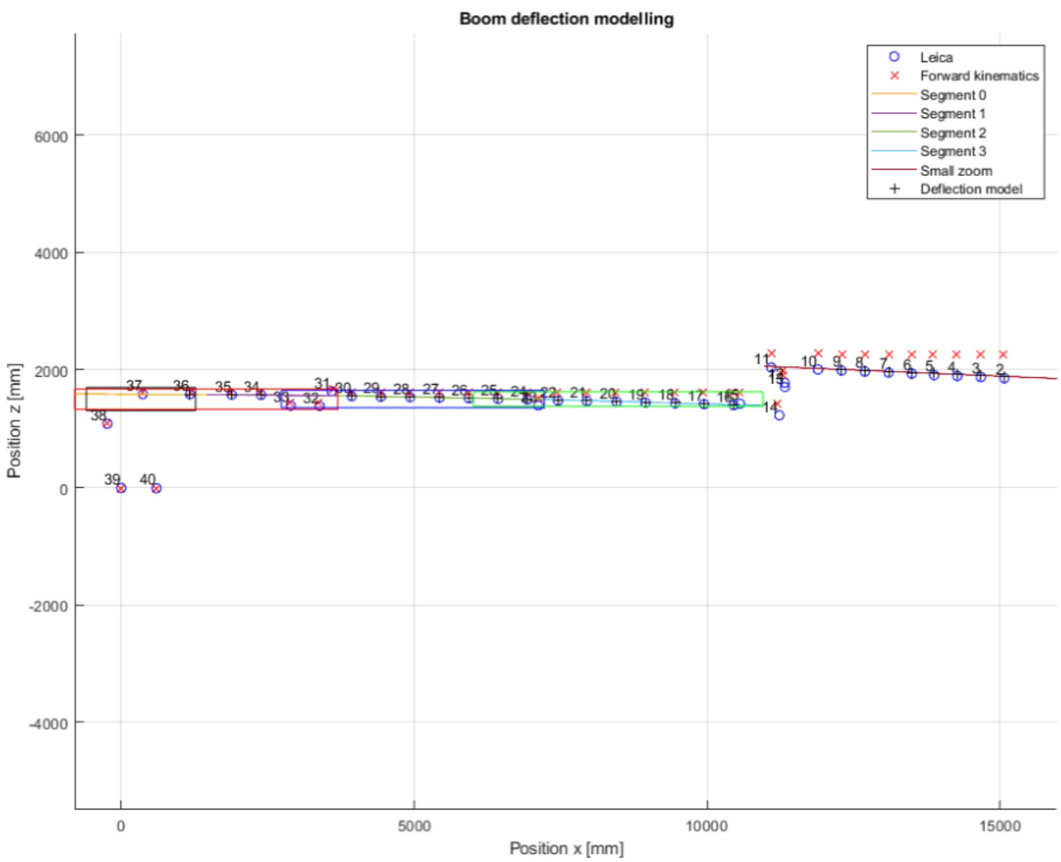


Figure 77: Boom deflection modelling simulation results, boom fully extended

5.6.4 IMOCO4.E requirements and KPIs

In technological KPIs there is one KPI relating to accuracy of deflection compensation. Deflection compensation accuracy should be ±10 cm in low accuracy booms such as concrete spraying boom. Based on the measurements developed modelling method fulfils this requirement.

5.6.5 Capabilities & Limitations (including USP, strengths & weaknesses)

Deflection model can be used for estimating the positioning error of high reach boom and compensate this error. This enables more automatic features, e.g., automatic execution of pre-planned paths. In concrete spraying application if a scanned model of the tunnel exists, boom control can be fully automated. Weakness could be, that to achieve sufficient accuracy calibration procedure is needed. Also clearances of the boom can change during the use of the boom because of wearing and calibration should be repeated periodically.

5.6.6 Customizations & Adaptations (including possible modifications and extensions

Developed deflection compensation method is suitable for high reach booms with both linear and rotational joints. Deflection model is specific to certain boom structure, but parametrization makes possible to adapt model to different size boom, if boom structure is similar. If structure of the boom is different, deflection model must be modified to correspond the structure of the boom.

5.6.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

In this case the modelling of the deflections was based on the measurements. The shape of the real concrete spraying boom was measured with Leica Optical total station. One lesson learned was, that measuring the shape of the real boom is a laborious task and it is not practical to do it for every boom. Other option is to use FEM to determine the shape of the boom with different joint values and to use this data for developing the deflection model. This is a viable approach, when designing new boom models. Simulation software can be also utilized in developing deflection compensation methods. In this case there was also a simulation model of boom available (Figure 75). In the simulation model the beams of the boom were modelled as flexible structures, but it was noticed, that deflections and clearances were much bigger in the real boom than in simulation model. If accuracy of the simulation model is sufficient, deflection model and compensation can be developed and tested in HIL environment and only final calibration is made with a real physical boom.

5.7 Data-driven dynamic modelling of a robotic arm (UGR)

5.7.1 Implementation aspects

The analytical layer is implemented in Python language, while the rest of the layers are implemented in C++ due to real-time performance limitations. The controller is deployed in simulation using Gazebosim but can be adapted to be compatible with other simulation environments, such as Simulink. Layer communication is implemented using ROS2.

5.7.2 Results

Limitations of analytical dynamic models

The dynamics of a robot can be modelled using analytical methods such as the Lagrange formulation. This formulation can be described as:

$$\tau = M(q)q'' + C(q,q') + g(q) + \xi(q,q',q'')$$

where q defines the position of the joints of the robot, τ stands for the torque vector, $M(q)$ represents the robot's inertia matrix, $C(q,q')$ computes the centrifugal and Coriolis effects. $\xi(q,q',q'')$ stands for the error of the model and gathers all the torque effects that are not taken into account into the model, such as friction and elastic components.

Most cases assume *ξ(q,q',q'')* to be close to 0, but collaborative robots (also known as cobots) count with passive safety measures, such as elastic joints, that greatly influence *ξ* (Lee et al., 2017). In these cases, the analytical methods produce inadequate models that are not accurate enough to guarantee safe human-robot interaction. This, coupled with the need for accurate torque control in human-robot interaction (HRI) generates a great need for new modelling and control methods.

Data gathering for dynamic modelling

Through the use of data-driven techniques it is possible to create models that better represent the non-linearities present in the dynamics of elastic joints in cobots (Rueckert et al., 2017). In order to obtain a high-quality model, optimal control data is necessary, and as such, the extraction methodology plays an essential role in the modelling process. This is key, because the data-driven techniques for model acquisition will capture the model that is represented in the data. Thus, if the data acquisition process assumes certain constraints or parameters such as performing specific trajectories with specific PD constants, this will be captured in the dataset and thus in the models that can be created from this dataset.

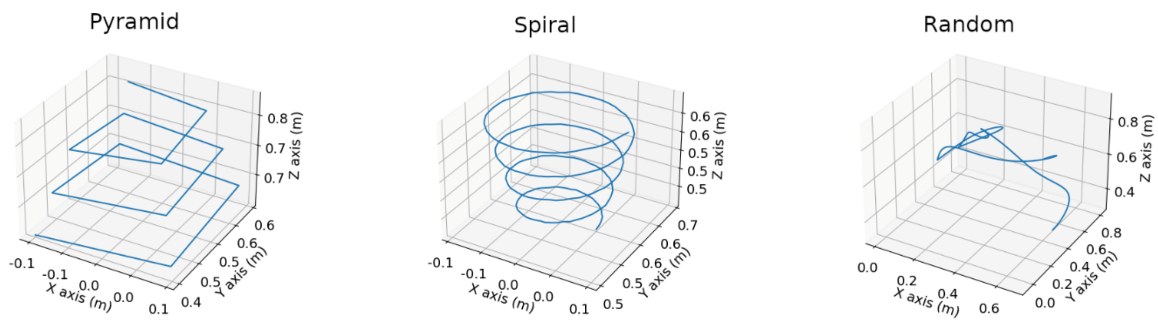


Figure 78: Examples of trajectories used for data extraction

In this extraction process, proportional-derivative (PD) controllers can be used as a way to fine-tune the motions of different trajectories (see Figure 78) from which to extract a representative dataset that will be used to model the dynamics of a cobot. PD controllers are a popular kind of feedback control due to their simplicity and their ability to perform better than other alternatives when properly fine-tuned to a specific trajectory (Kelly, 1997).

The tuning process for the PD controllers is done using genetic algorithms, due to their capacity to find optimal configurations in complex spaces. It is advisable to find algorithms able to converge in as few iterations as possible, so as to not wear down the cobot joints and motors.

It is also important to pay attention to the torque profile of the extracted data. Wild changes in torque will produce unsafe behaviour, and in some use cases safety should be prioritized over accuracy, which should reflect on the design of the genetic algorithm's objective function. In some cases, multi-objective genetic algorithms, such as NSGA-II (Deb et al., 2002), can be used to find the optimal balance between accuracy and safety, but it will usually come at the expense of a longer PD tuning process.

More information about the modelling process and its application in a feed-forward controller can be found in section 4.7 of (D4.8).

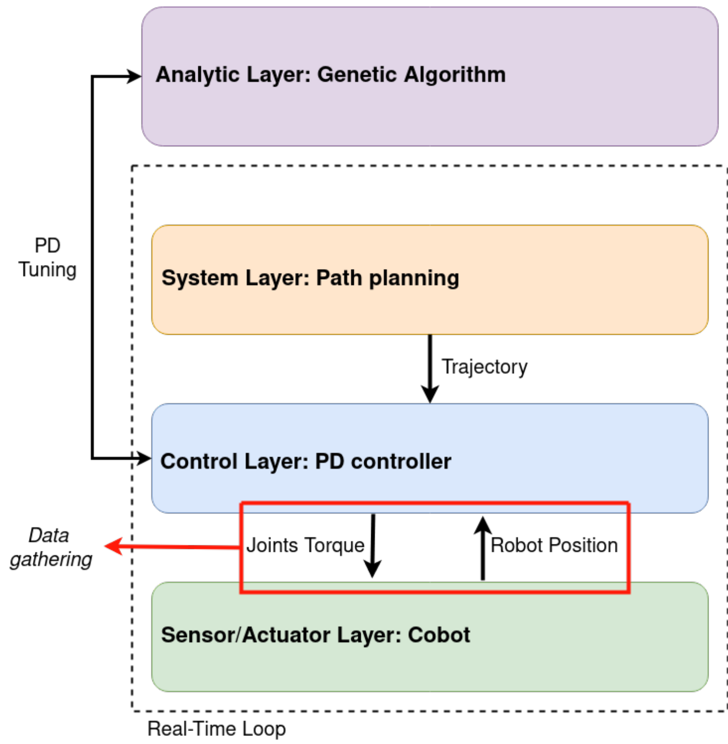


Figure 79: Layer framework of the proposed methodology

5.7.3 IMOCO4.E requirements and KPIs

R007--L2-sw: Data-driven Robot Dynamics model for compliant control should be more accurate than an analytical model, especially in fast movements.

5.7.4 Customizations & Adaptations (including possible modifications and extensions)

This methodology can easily be adapted to any collaborative robot for the purpose of creating a data-driven dynamic model as long as a rigid body dynamic model is not accurate enough.

5.8 Digital twin for transient thermal behaviour of mechatronic systems (SIOUX)

5.8.1 Tech Overview

As thermal effects cause materials to deform, accurate modelling of the thermal behaviour of mechatronic systems is a highly relevant area of interest. This is especially the case in applications where extreme positioning accuracy is critical. As it is difficult and expensive to have sensors monitoring all temperatures and/or thermal deformations, it is valuable to have a digital twin of the system that predicts these thermal effects. *SW-049 Probabilistic system identification for transient thermal behaviour* targets the realization of such digital twin.

The digital twin can serve as a set of virtual sensors that provide thermo-mechanical deformation information throughout the mechanical system structure. The virtual sensors can then be included in the mechatronic control loop, where appropriate motion corrections are applied that compensate for the thermo-mechanical deformations, as shown in Figure 80.

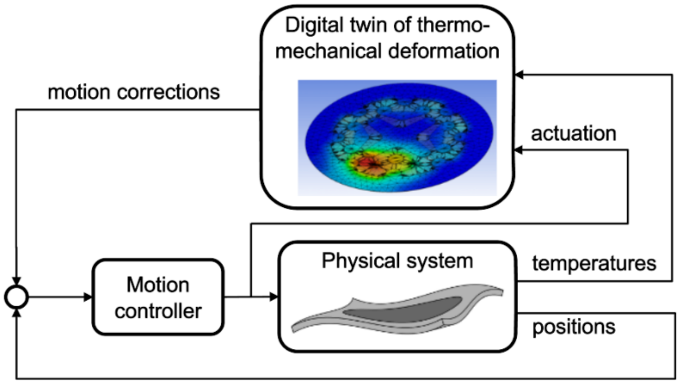


Figure 80: Digital twin for transient thermo-mechanical behaviour in a mechatronic control loop

5.8.2 Implementation aspects

We created a workflow for system identification based on thermal data, exporting an identified thermal model. A thermal digital twin, built in MATLAB Simulink, then incorporates the identified model to provide real-time virtual sensors readings that can be used in the mechatronic control loop of Figure 80. Figure 81 shows the high-level MATLAB Simulink implementation of the thermal digital twin, taking the available measurements and inferring the temperatures at virtual sensor locations. The probabilistic 'gray-box' digital twin is also compared to a deterministic 'white-box' variant, and to open-loop simulations.

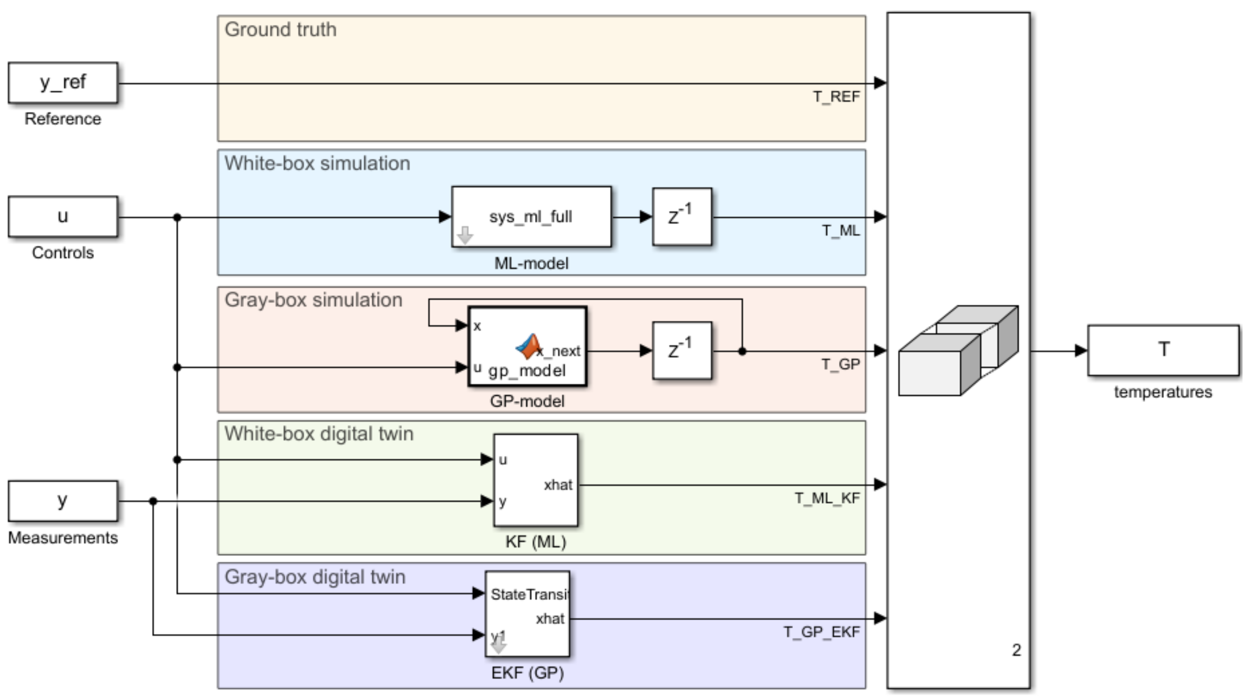


Figure 81: MATLAB Simulink implementation of the probabilistic 'gray-box' digital twin, side-by-side with a deterministic 'white-box' variation and open loop simulations for comparison

5.8.3 Results

Gray-box thermal model identification

It is challenging to obtain an accurate model for the thermal digital twin: convection is notoriously hard to model. A coarse approximation that is commonly used for lumped-element modelling of thermal systems, would be Newton's law of cooling, which states that convection h is linear in the difference between the temperature τ_i at position i and the ambient temperature τ_a. As the accuracy of this linear approximation for the convection proves insufficiently accurate for the non-linear transient behaviour, an unknown non-linear remainder term r is also added to the convection: $h = h_a(\tau_a - \tau_i) + r(\tau_a, \tau_i, i)$. This yields a gray-box state-space model for the thermal digital twin:

$$\begin{bmatrix}\dot{\tau}_1\\ \vdots \\ \dot{\tau}_3\end{bmatrix} = \underbrace{\begin{bmatrix} \frac{-k_{12}-h_a a_1}{m_1 c_{p,1}} & \frac{k_{12}}{m_1 c_{p,1}} & 0 \\ \frac{k_{12}}{m_2 c_{p,2}} & \frac{-k_{12}-k_{23}-h_a a_2}{m_2 c_{p,2}} & \frac{k_{23}}{m_2 c_{p,2}} \\ 0 & \frac{k_{23}}{m_3 c_{p,3}} & \frac{-k_{23}-h_a a_3}{m_3 c_{p,3}}\end{bmatrix}\begin{bmatrix}\tau_1\\ \vdots\\ \tau_3\end{bmatrix} + \begin{bmatrix} \frac{h_a a_1}{m_1 c_{p,1}} & \frac{1}{m_1 c_{p,1}} & 0 & 0 \\ \frac{h_a a_2}{m_2 c_{p,2}} & 0 & \frac{1}{m_2 c_{p,2}} & 0 \\ \frac{h_a a_3}{m_3 c_{p,3}} & 0 & 0 & \frac{1}{m_3 c_{p,3}}\end{bmatrix}\begin{bmatrix}\tau_a\\ u_1\\ u_2\\ u_3\end{bmatrix}}_{\substack{\text{Linear conductivity, linear convection, input heat}\\ \to \text{White box}}} + \underbrace{M^{-1}\begin{bmatrix} r(\tau_1, \tau_a, 1, t)\\ r(\tau_2, \tau_a, 2, t)\\ r(\tau_3, \tau_a, 3, t)\end{bmatrix}}_{\substack{\text{Non-linear convection}\\ \to \text{Black box}}}$$

The presented model is identified in a two-step approach:

- *White box:* The unknown parameters (conductivity k and linear convection h_a) of the white-box state-space model are determined through maximum likelihood estimation (MLE).
- *Black box:* The unknown non-linear convection functions r are introduced in the state-space model as an online Gaussian Process with a kernel covariance function designed to capture the transient effects of the non-linear contribution from convection. This GP-augmented state-space model is fit onto the measurement data, to obtain GP posteriors for the non-linear convection. As a model order reduction step, polynomials are fit through the GP posteriors, yielding the identified non-linear convection functions r.

Thermal digital twin validation

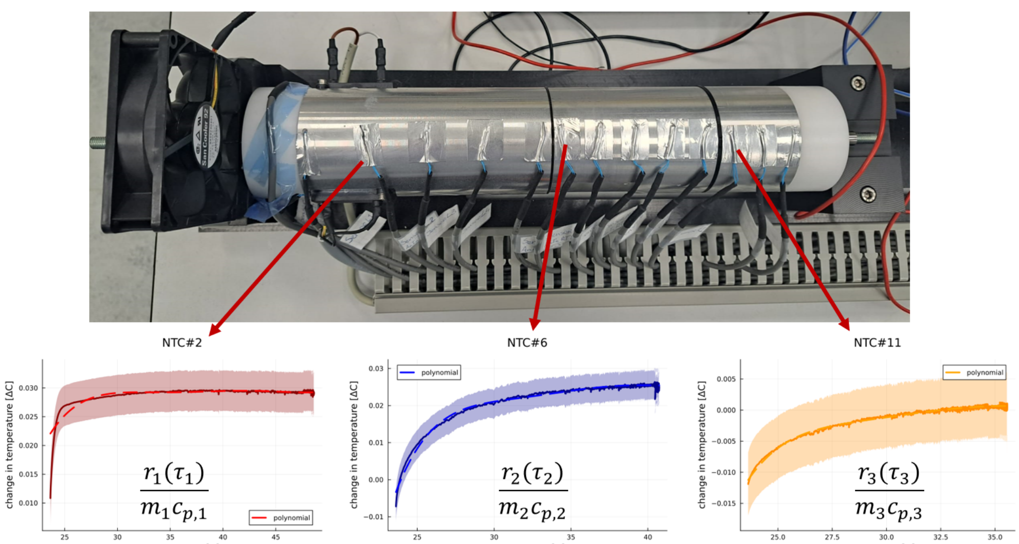


Figure 82: Thermal test rig with identified non-linear convection functions for sensor 2, 6, 11.

A thermal test rig, equipped with 13 temperature sensors, was used to validate the thermal digital twin. Using collected measurement data from each temperature sensor, the gray-box thermal model was identified, producing a non-linear convection function at each sensor location (Figure 82).

The identified model of the thermal test rig was then loaded into a digital twin in MATLAB Simulink (Figure 81). It was assumed that measurements were available from sensor 2 & 11, but not from sensor 6. The digital twin created a virtual sensor 6, using its internal model and measurement from the other sensors. The virtual measurements of sensor 6 were then compared to the real measurements. It can be seen in Figure 83 that the digital twin with the identified white-box model (only linear convection) shows a large error during the transient period (first half of the graph) while the digital twin with the identified gray-box model (linear *and* non-linear convection) has a smaller error that is also unaffected by the transients.

We conclude that the presented identification approach was successful at estimating complex non-linear transient thermal behaviour. It can be used to improve digital models and it is fast enough to be used in an online fashion. With the improved digital twin, better virtual sensors can be instantiated at locations where measurements are otherwise impossible.

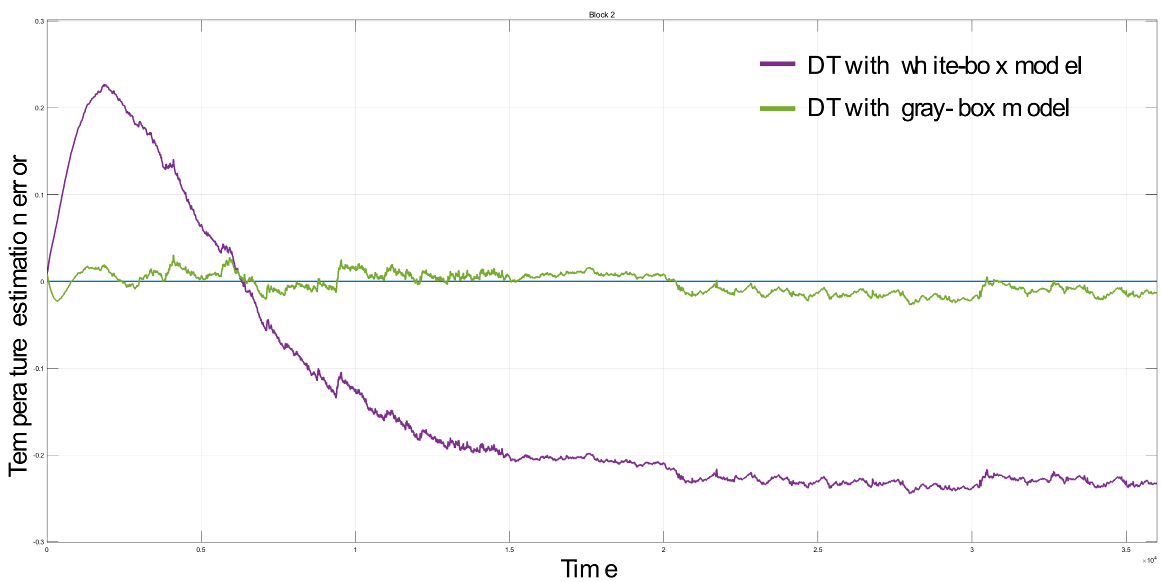


Figure 83: Performance (temperature estimation error) of the thermal digital twin using a white-box model and a gray-box model

5.8.4 Capabilities & Limitations (including USP, strengths & weaknesses)

The strength of the proposed probabilistic gray-box thermal model identification procedure is that it can accurately model fully unknown thermal effects on top of available physics knowledge, exploiting the best of both worlds. This is relevant for convection effects, as these are typically fully unknown beyond the linear Newton approximation. The only assumption that was incorporated in the GP-augmented state space model was continuity.

The most important challenge remains the GP kernel hyperparameter tuning for identifying the GP-augmented state space model. This requires a level of expertise, especially when the landscape is multi-model or in case of divergence regions (e.g., unobservable systems).

5.8.5 Customizations & Adaptations (including possible modifications and extensions)

As the digital twin uses a polynomial (non-linear) state-space model, model order reduction techniques were applied to convert the GP-augmented state-space model obtained during the identification procedure into a polynomial state-space model. It would be preferable to have an interface between the identification step and the digital twin that is agnostic of the model representation. To this end, a Functional Mock-up Interface (FMI) can be considered.

Further, the realized procedure of gray-box identification for systems with unknown nonlinear components is not limited to thermal systems. A thermal system was used as it presents a relevant use case, but it would be interesting to investigate the potential of the technique in other domains.

5.9 Digital Twin of a chip manufacture line (DTT)

5.9.1 Tech Overview

In Pilot 2, DTT developed a Digital Twin (DT) of a chip wafer table which is a part of a chip manufacturing line. This DT enhances ADAT3-XF's accuracy and production speed, predicting the performance of machines and production lines to optimize overall factory operation.

The creation of the Digital Twin for a semiconductor manufacturing line involves integrating Unity 3D and MATLAB in a software-in-loop process. This process includes constructing a virtual replica of the chip wafer table using physics-based modelling within the Unity platform. An intercommunication layer is established to seamlessly connect Unity and MATLAB applications, enabling the exchange of data. This communication interface, based on web sockets, facilitates real-time interaction between the virtual environment and MATLAB. Additionally, a MATLAB-based PID controller is engineered to function as the controller within the system, ensuring precise control and simulation fidelity. Overall, the integration of these components results in the development of a comprehensive Digital Twin of the chip wafer stage, providing insights into the performance and behaviour of the semiconductor manufacturing line.

In terms of software components, Unity 3D and MATLAB are utilized for modelling, simulation, and control. Unity 3D offers advanced capabilities for building immersive virtual environments, while MATLAB provides powerful tools for data analysis, modelling, and control design. Modern and capable computers equipped with sufficient processing power and graphics rendering capability are required to carry out the development of the Digital Twin of a chip wafer table. Overall, the integration of software and hardware components enables the realization of an effective Digital Twin for semiconductor manufacturing, facilitating optimization and decision-making in the production process.

The objectives for this activity align with the ST1, "the development of advanced model based and knowledge-based methods for building digital twins".

5.9.2 Implementation aspects

Creating a virtual replica of a chip wafer table through the integration of Unity 3D and MATLAB constitutes a software-in-loop process. This undertaking encompasses the construction of a physics-based state-machine, an intercommunication layer, and a PID controller. The Unity platform is employed for constructing the physics-based state-machine, while a web-socket oriented communication interface is established, seamlessly linking Unity and MATLAB applications. This interface facilitates the exchange of data, as depicted in Figure 84. Conclusively, a MATLAB-based PID controller is engineered to assume the role of the controller within the system. The integration of these components culminates in the realization of a comprehensive digital twin of the chip wafer stage. The conceptual framework of the interrelated closed-loop system is illustrated in Figure 84.

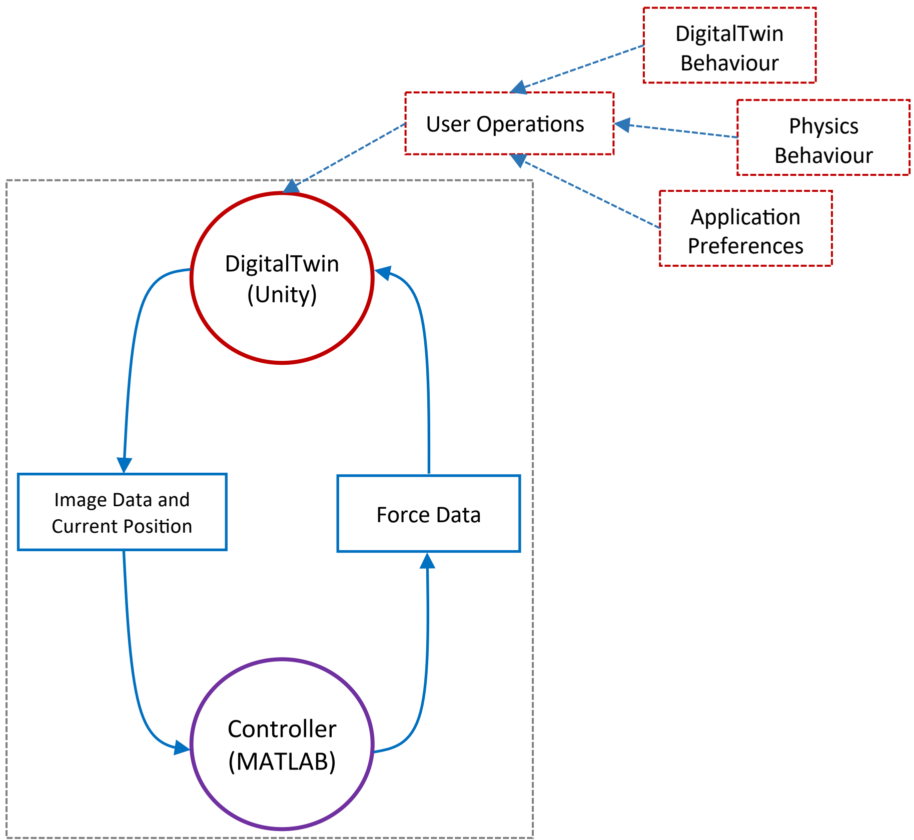


Figure 84: Operational Architecture of the Digital Twin (DT)

As depicted in Figure 84, there are two main components of the DT architecture: 1) Unity platform for physics-based state-machine construction - it contains the Digital Twin and sends position and image data to the MATLAB PID controller 2) MATLAB PID controller – it yields force values based on the incoming data from the Digital Twin and sends the force values back to the DT. The force values are integral for the DT to emulate and generate a *mechanical translation* for the movement of the DT's components. The *mechanical translation* an integral function of the DT which mirrors the physical properties and behaviour of its objects. Moreover, the DT system provides a robust and efficient solution for optimizing semiconductor manufacturing performance through features such as parameter tuning, synchronous simulation, Software-in-the-Loop, a camera-based AI controller, and an intuitive user interface.

Description of system configuration and configuration files:

Configuration files serve as predefined settings containing a set of parameters that dictate the operation of the system. These files encompass a range of parameters, encompassing system behaviours, hardware configurations, and software operation modes. An example parameter type available for adjustment is frequencies, such as the communication loop frequency between the DT and the MATLAB controller, or the update frequency in the physics loop. Fine-tuning these frequencies enables optimization of the DT to align more closely with the actual operation of the ADAT3-XF. These files play an important role in maintaining the synchronization of the simulation.

Synchronous simulation:

The DT employs a synchronous simulation approach, ensuring the coordinated operation of all system components, including communication with the external controller. This ensures consistent behaviour across various simulations, a crucial factor for accurately predicting the system's performance and optimizing its operation.

Camera-based AI controller:

The DT is designed in a way that it allows the incorporation of a Camera-based AI controller. The camera captures images of the semiconductors, which are then analysed by an AI algorithm. An example of this AI algorithm would be to perform AI training, identification of damage or misplacement of the semiconductors. If damage is detected, the semiconductor is not preserved.

User Interface:

The user interface of the DT system is designed to be user-friendly and intuitive. Leveraging Unity's comprehensive capabilities, we efficiently design, modify, and animate the user interface, creating an accessible platform for interacting with the DT and performing the aforementioned configuration operations. For instance, the ease in highlighting parts of the twin when selected or hovered over contributes to improving the user's interaction experience with the model, making it more intuitive and engaging than other platforms might allow.

Software In the loop (SIL):

Our DT aims to provide Software-in-the-Loop (SIL) functionality which involves integrating software components into a simulated environment to evaluate their behaviour and performance. It allows for the testing and validation of software functionality without the need for physical hardware, offering time and cost savings in the development process. By focusing on the software and simulating the rest of the system, SIL enables thorough testing, fault detection, and analysis, aiding in the identification and resolution of potential issues before implementation on real hardware. Example of usage training machine controllers such as PID, camera-based AI controller (as mentioned above) and so on.

PID controller:

As part of DT system development, we use MATLAB to develop a PID controller that regulates the chip wafer table. The purpose of the PID controller is to stabilize and regulate dynamic processes by constantly modifying a control variable based on the difference between the desired setpoint and the measured process variable. The three components of a PID controller work in harmony to achieve the best control performance. The functionalities of the three components can be summarized as below:

- The proportional term refers to a control action that is directly proportional to the error signal, and it is achieved through the use of the all-pass gain factor.
- The integral term is used to reduce steady-state errors by using a low-frequency compensation through an integrator.
- The derivative term improves the transient response by using high-frequency compensation through a differentiator.

In our DT, the PID controller is used to control the position of the chip wafer table. It obtains the desired position in both x and y coordinates from a DT created in Unity, as explained above. Furthermore, the DT communicates the current position of the chip wafer table to the controller via a Unity-MATLAB interface, as detailed above. The PID controller calculates a force signal based on the difference error between the current and desired positions. The DT utilizes this power signal until the desired position is achieved.

Communication interface:

We designed a communication interface using web-socket to facilitate continuous and two-way data transfer between the Unity-based DT and the MATLAB PID controller. Opting for a byte-style data format instead of JSON enables efficient exchange, particularly for handling image data. Here, the DT functions as the client, and the controller acts as the server. The connection initiates with a request from the DT, illustrated in Figure 85, leading to a complete data flow between the two.

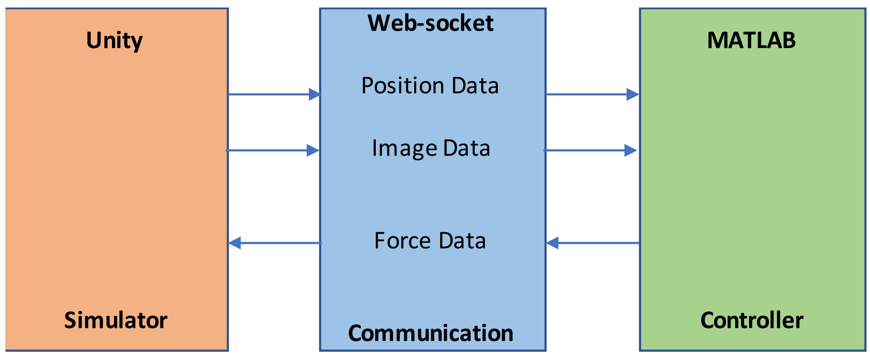


Figure 85: Communication interface between Unity and MATLAB with the type of data exchange

The DT initiates the connection, sending initial settings and the desired position, followed by continuous updates of the current position and image data. The controller uses the received position data to calculate errors and generate a force signal, which is then sent back to the DT for subsequent actions. The abstract design of the communication interface allows easy configuration or replacement of system elements for diverse use cases, facilitating not only data exchange but also synchronization between two distinct software programs.

5.9.3 Results

Throughout the development phase, a series of simulation cases were conducted, specifically focused on replicating the real-world tasks associated with the wafer pickup sequences of the target machine. These sequences were designed based on a predefined path of setpoints, each corresponding to a specific position. Using these setpoints, our PID controller produced its calculated force output, enabling an accurate simulation of the real hardware's transfer function. Our initial emphasis was on testing the synchronization of the simulation loop, encompassing both movement and communication. This phase involved encoding physics in Unity and implementing multiple setpoints to represent the positions of semiconductor dies. This test served to validate the functionality of our Unity-based physics solution.

As we progressed, we made additional adjustments to our system in accordance with end-user requirements. These modifications focused on improving the physics to better emulate the transfer function of the real hardware. This evolution resulted in the development of the DT system for the chip wafer table, featuring refined machine physics.

The following figures, Figure 86 and Figure 87, depict the behaviour chip wafer table with adjusted machine physics and also the full machine with its different components attached within.

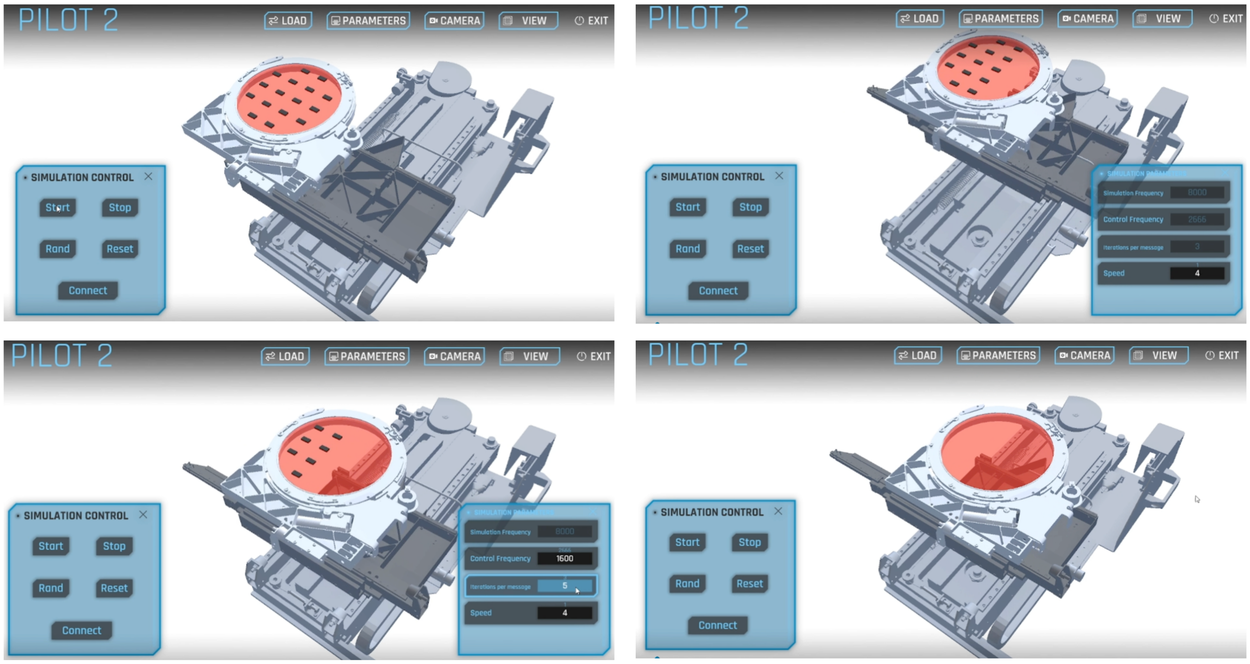


Figure 86: The DT behaviour: action sequences within the DT

In Figure 86, the first image shows the table's initial state, followed by images detailing the table's movement to reach the setpoint. The final image shows the table successfully reaching the setpoint and picking the die from the wafer. In Figure 87, we can see the full machine with different components such as the workholdr and the pickup table.

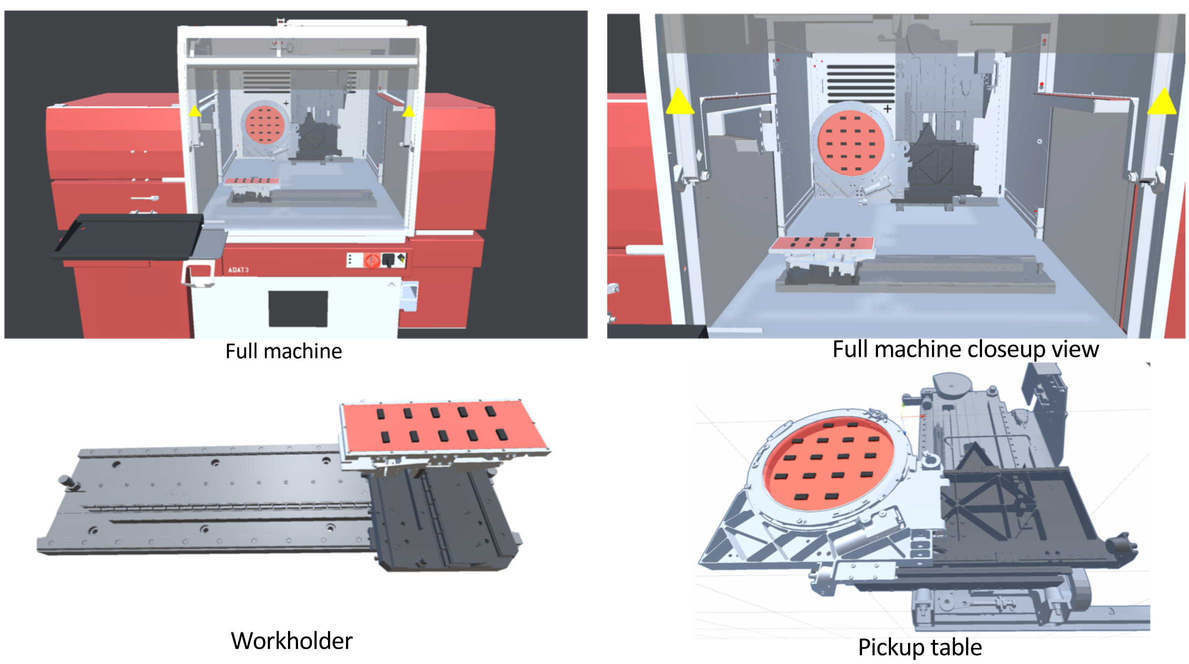


Figure 87: The DT: full machine with components - Workholder and Pickup table

Figure 88 depict the complete DT with user interface where different simulation parameters and view settings can be controlled.

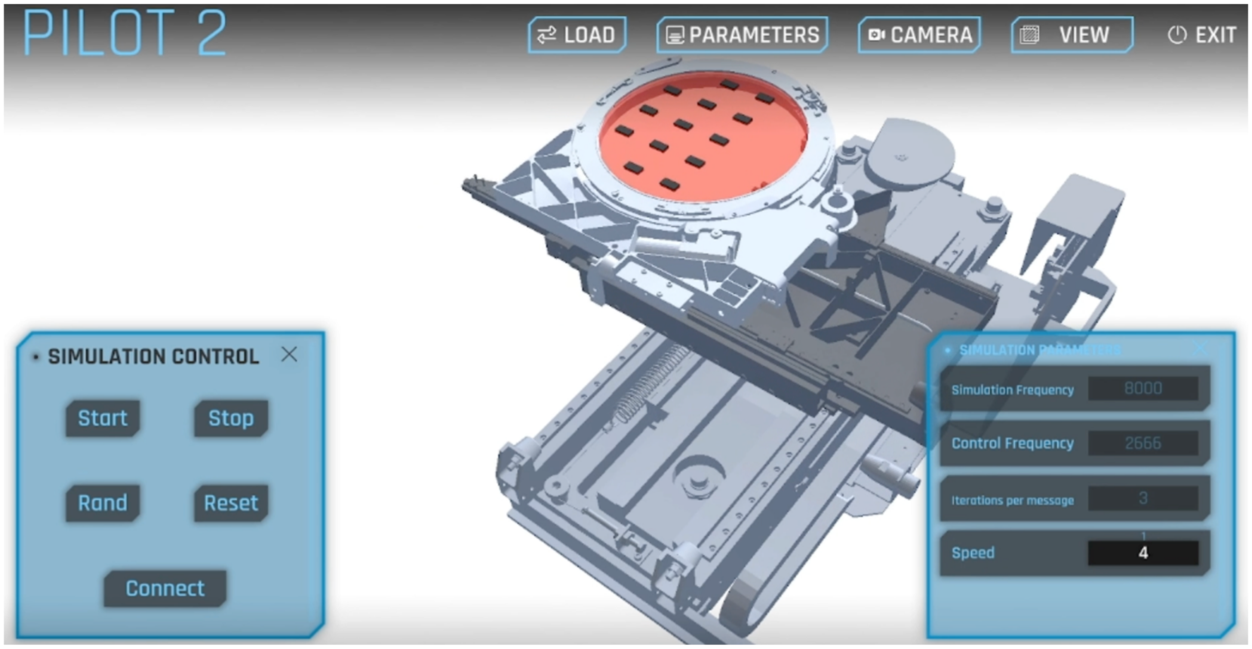


Figure 88: The DT of the chip wafer table with the user interface

5.9.4 IMOCO4.E requirements and KPIs

This Digital Twin (DT) specifically caters to the needs outlined in WP5, fulfilling the overall system requirements within the IMOCO4.E framework related to interoperability functionalities. Aligned with BB6, this DT meets the specified requirement by providing a robust tool designed to enhance semiconductor manufacturing performance, thereby contributing to the improvement of factory productivity.

This DT addresses the objectives ST1 by building a digital twin which a robust and efficient tool for enhancing the performance of semiconductor manufacturing.

5.9.5 Capabilities & Limitations (including USP, strengths & weaknesses)

This Digital Twin (DT) accurately replicates the behaviour of the chip wafer table, enabling precise simulation of semiconductor manufacturing processes. Through the seamless integration of Unity 3D and MATLAB, it facilitates real-time interaction between the virtual environment and the PID controller, ensuring continuous data exchange and accurate emulation of real hardware behaviour. Additionally, its flexible configuration options, facilitated by system configuration files, allow for fine-tuning of parameters to optimize performance, and closely align with the operational characteristics of the semiconductor manufacturing line.

The DT's unique selling proposition lies in its comprehensive simulation capabilities and flexible configuration options. By integrating Unity 3D and MATLAB, this DT offers accurate replication of semiconductor manufacturing processes, allowing for thorough testing and validation without the need for physical hardware. Additionally, its flexibility in configuration enables customization to specific manufacturing environments, providing tailored solutions for optimization and performance enhancement.

The DT's strengths lie in its synchronous simulation approach, ensuring coordinated operation of all system components and consistent behaviour across various simulations, enhancing accuracy and reliability in predicting system performance. Moreover, its user-friendly and intuitive user interface enhances the interaction experience, making it accessible for users to perform configuration operations and monitor system behaviour effectively. This contributes to improved usability and adoption of the DT in semiconductor manufacturing environments, facilitating optimization and decision-making processes.

The DT may pose challenges related to its complexity and resource requirements. The integration of Unity 3D and MATLAB, coupled with the physics-based state-machine and communication interface, adds complexity to its implementation, potentially requiring specialized expertise for development and maintenance. Additionally, running the DT may demand significant computational resources, particularly for real-time simulation and interaction between Unity and MATLAB. This could pose challenges for organizations with limited computing capabilities or budget constraints, potentially impacting the scalability of the DT solution.

5.9.6 Customizations & Adaptations (including possible modifications and extensions)

Customizations of the digital twin involve fine-tuning parameters through system configuration files, enabling adjustments such as communication loop frequencies and physics update rates to optimize performance and align closely with the semiconductor manufacturing line's operation. Additionally, modifying the user interface using Unity's capabilities allows for tailored layouts, visualization elements, and interaction features, enhancing user experience and efficiency during simulation and analysis tasks.

Adaptations of the digital twin involve integrating additional controllers beyond the PID controller, such as advanced control algorithms or machine learning models, to enhance system optimization and predictive maintenance capabilities. This extension enables the digital twin to adapt to changing process conditions in real-time, improving its effectiveness in controlling and optimizing semiconductor manufacturing processes. Additionally, adapting the digital twin to simulate a broader range of manufacturing scenarios provides insights into system performance under varied conditions, facilitating thorough testing and validation of semiconductor manufacturing processes.

Possible modifications and extensions of the digital twin may include integrating sensor data from the manufacturing environment to enhance fidelity and predictive capabilities. By incorporating real-time sensor data such as temperature, pressure, or vibration measurements, the digital twin can provide more accurate simulations and predictive analytics for equipment health monitoring and anomaly detection. Furthermore, extending the digital twin to integrate with supply chain management systems enables end-to-end visibility and optimization of semiconductor manufacturing processes, allowing for real-time monitoring of material flows, production schedules, and inventory levels to facilitate efficient resource allocation and production planning.

5.10 Conclusion

The models required by Digital Twins, which are active during the use of the physical object, are not simply the models used in the design phase. A great variety of models and modelling techniques is employed to create a Digital Twin for the use cases and pilots of IMOCO4.E. The models, in order to function in a useful way during operation, must be fast enough and must be modifiable; the model improves the operation of the physical model, and the data gathered from the operation improves the model. A very important difference is that, in a Digital Twin, the model is part of the product.

The 'conventional' model used for design and development can be used as a starting point, but it is usually too slow for the operational phase (real time calculations are needed despite limited computing capacity). Two methods to improve speed are used in the work reported here: Reduced Order Modelling (ROM) and data-based modelling, using Machine Learning (ML). These approaches are very different but have the same aim. Both result in a model that is fast enough for real time simulation. The models, in both cases, can be modified by sensor data gathered during the operation of the physical object. An interesting method to do this is presented. It uses a Bayesian filter technique. This not only brings the model closer to the physical object, but also allows the estimation of the accuracy of the model.

An important advantage of simulations over physical experiments is that all variables in a simulation can be studied without the limitations of measurements. In a Digital Twin, this advantage can be retained, as is demonstrated in the examples for the IMOCO4.E use cases and pilots. The combination of model and physical object has shown the potential to increase performance and functionality.

It is concluded that the integration of model and physical object sets new requirements for models, demands a new way of thinking, and can lead to important improvements in the performance of the project.

6. Augmented and virtual reality through digital twins

6.1 Overview of realized solutions

This section describes tools helping to create, set up and use digital twins especially in the context of IMOCO4.E applications. One main theme in this section is related to utilization of augmented and virtual reality concepts used as Human-Machine Interface (HMI). Following descriptions are based on the work done in the Task 5.6 'Augmented and virtual reality through digital twins' that is devoted to the overall development and utilization of digital twins specialized for complex multi-axis mechatronic systems, remote tactile robot teleoperation and autonomous mobile robots.

Development related to tooling and creating digital twins focuses on the algorithm development, modelling, and verification technologies for digital twins. One studied approach is to bring forward the automation in the generation of digital twin components related to the model development and code generation. Performance issues are also considered by finding effective methods to update and tune the physics based reduced order models for digital twins and analysis of platforms for performance-intensive digital twin applications.

In the following sections, we describe how digital twins are set up and used in development, testing and commissioning phases. From a system testing perspective, the use of digital twins in end-to-end testing and integration of Hardware-In-the-Loop (HIL) and Software-In-the-Loop (SIL) systems into digital twins and virtual testing environments are presented. Virtual reality (VR) and digital twin technologies are studied related to remote tactile robot teleoperation and user experience (UX) experiments in autonomous mobile robots.

6.2 Addressed ST objectives and KPIs

This chapter relates to objectives ST1 and ST4. The objective ST4 is addressed by virtual/augmented reality for real-time HMIs which is the main topic in Task 5.6 and thus also in this chapter. This task has no direct link with any BB. It indirectly contributes to KPI_BB8_3 by results from chapter6.5. These results show the implementation of different neural network structures in the FPGA (Xilinx Zynq Ultrascale+ ZCU102) and the GPU (Nvidia Jetson Xavier AGX).

The study provided by UNIMORE contributes to objective ST4 by selecting a suitable platform for the execution of AI based methods for machine predictive maintenance, which relates to software component SW-018 of building block 8. Moreover, it is in line with the objective TRL2 of KPI "High-performance embedded computing platforms with real-time capabilities" of building block 4, i.e., "Existing solutions are either limited to best-effort applications, e.g., AI edge nodes without strict timing requirements, or significantly underutilize the hardware in order to avoid interference between the computing engine".

Related to the ST1 objective described in Figure 2, the main goal for the digital twin application described in Chapter 6.9. "Comparative User Experience test of driving behaviour in encounters of mobile robots with human traffic participants in a virtual and a real environment" is a preparation for the application described in Chapter 6.10. "VR-based tool for User Experience evaluation of driving behaviour of mobile robots in encounters with human traffic participant". The overall aim is to be able to design and optimize algorithms dealing with encounters between autonomous mobile robots (AMR) and humans. For these tasks an easy and reliable evaluation method is needed. Both applications create digital twin applications for User Experience (UX) Testing of AMR with human test participants. The first application described is used to evaluate the reliability of UX-testing using the VR-applications implemented within Task 5.6. It could be shown that the VR-application can provide a suitable methodology for UX-testing of AMR-driving behaviour.

VR application contributes to the objective ST4 described in Figure 2, as a first step to implement a "Virtual/augmented reality for real-time HMIs" by creating a VR-environment as a preparation for the real-time VR-application described in chapter 6.10.

The KPI tackled in this task is from the Demonstrator 3 "Autonomous intra-logistic transportation" Autonomous functionalities tested in digital twin (TRL 4). The first step of the implementation of the capability to evaluate the User Experience of autonomous functionalities in digital twin has been described in chapter 6.9 intensively. To enable the developed setup to be able to test the autonomous behaviour in realistic traffic situations the desired setup of chapter 6.10 will be a good contribution.

6.3 Automated generation of digital twins [SIOUX]

6.3.1 Technical Overview

The Sioux Holodeck is a software solution (SW catalogue item SW-046) to model a VR digital twin from the design models of the twinned system.

The goal of Holodeck is to automate the generation of digital twins based on design data and design knowledge. A digital twin visualization tool is developed, including a pre-built library of Unity packages, and deployed for many projects. It includes virtual reality, multi-client, cloud hosting, avatars, and interaction with the system. Challenges, (future) solutions, and architecture will be shared/presented.

6.3.2 Implementation aspects

What we realized within Pilot 1 is the creation of a VR digital twin using the proprietary Sioux Holodeck. This digital twin can follow a real system or a simulated system. For this purpose, the real system contains software (partially with generated code) that supports communication to this VR digital twin.

The software decomposition is as follows (Figure 89):

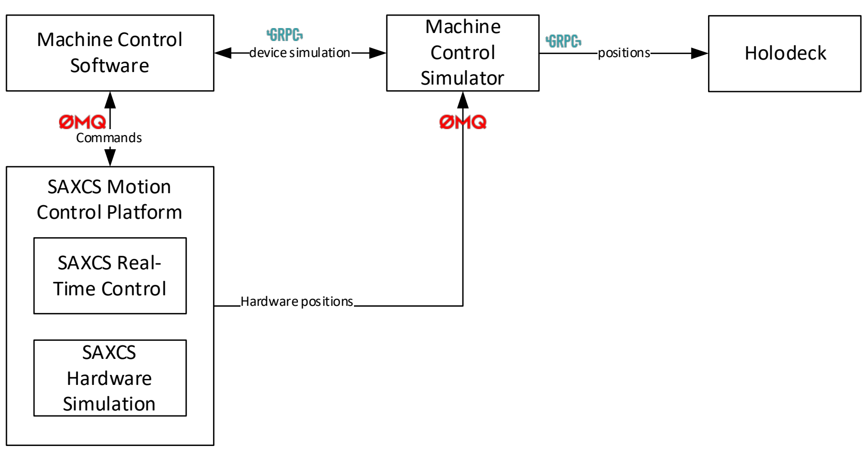


Figure 89: Pilot 1 software decomposition

Machine Control Software and the Machine Control Simulator are written in C#.

The following things are generated automatically:

- Machine Control Software state machine code is generated from a state model.
- Mapping of asset bundles to entities in the real machine / simulator.
- Motion simulator (initially used, currently replaced by real-time hardware simulation).
- Simulator UI.
- Interface proxy for the real-time control.

Additionally, the translation of CAD designs into FBX files is generated semi-automatically (tool assisted). Figure 90 shows how the tools and data are connected.

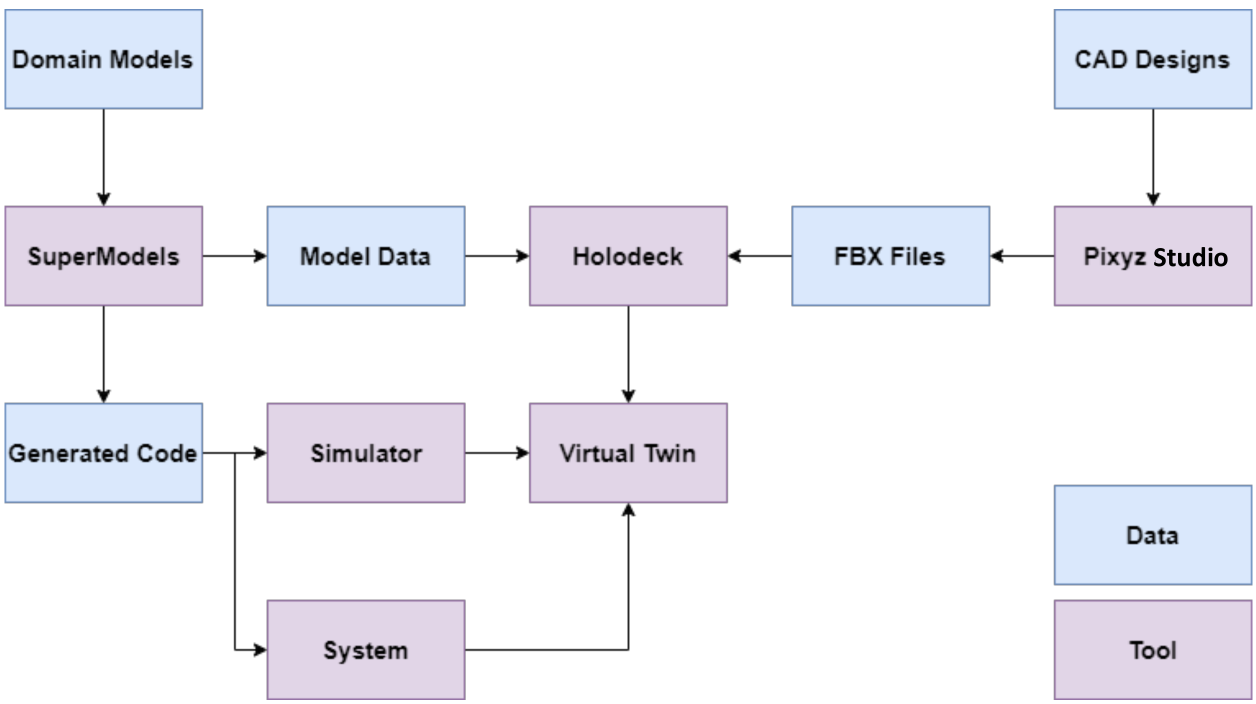


Figure 90: Pilot 1 VR digital twin development view

- Domain Models: in this case the decomposition and state models
- SuperModels: The Sioux workbench used to create and maintain the domain models, containing the generators
- Generated Code: Software classes with an executable representation of the decomposition and state models. Also includes the parts of the simulator that are generated.
- Model Data: Information about the (sub)assemblies and how it is linked to asset bundles.
- Simulator: Software that simulates the hardware controlled by 'Machine Control Software', either directly itself or connects to the real-time simulation and translates the effects to e.g., product state.
- System: In this case the 'Machine Control Software'
- Holodeck: Unity based server application that can visualize digital twins based on output of the simulated or real system.
- Virtual Twin: The concrete visualization of the digital twin, as hosted in the holodeck.
- FBX Files: 3D Assets that are used to visualize the assemblies of the system and related artefacts (e.g., carriers, disposables)
- CAD Designs: The original mechanical 3D model of the system, in step format.

- Pixyz Studio: A tool used to convert step files to asset bundles.

6.3.3 Results

In Figure 91, both the generated GUI of the control software and the generated VR digital twin are shown.

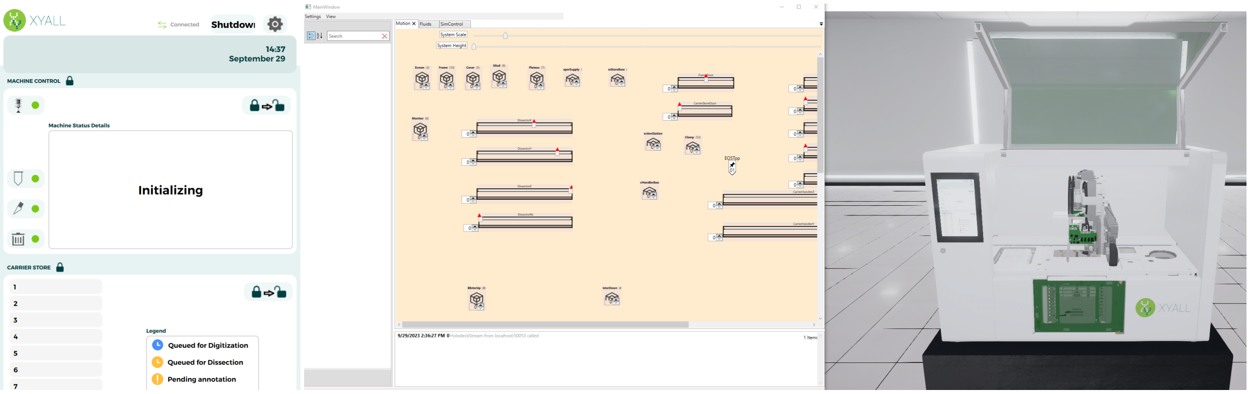


Figure 91: Control software of Pilot 1 simulator connected to Holodeck

In Figure 92, the real system and the VR digital twin are shown in action side by side.

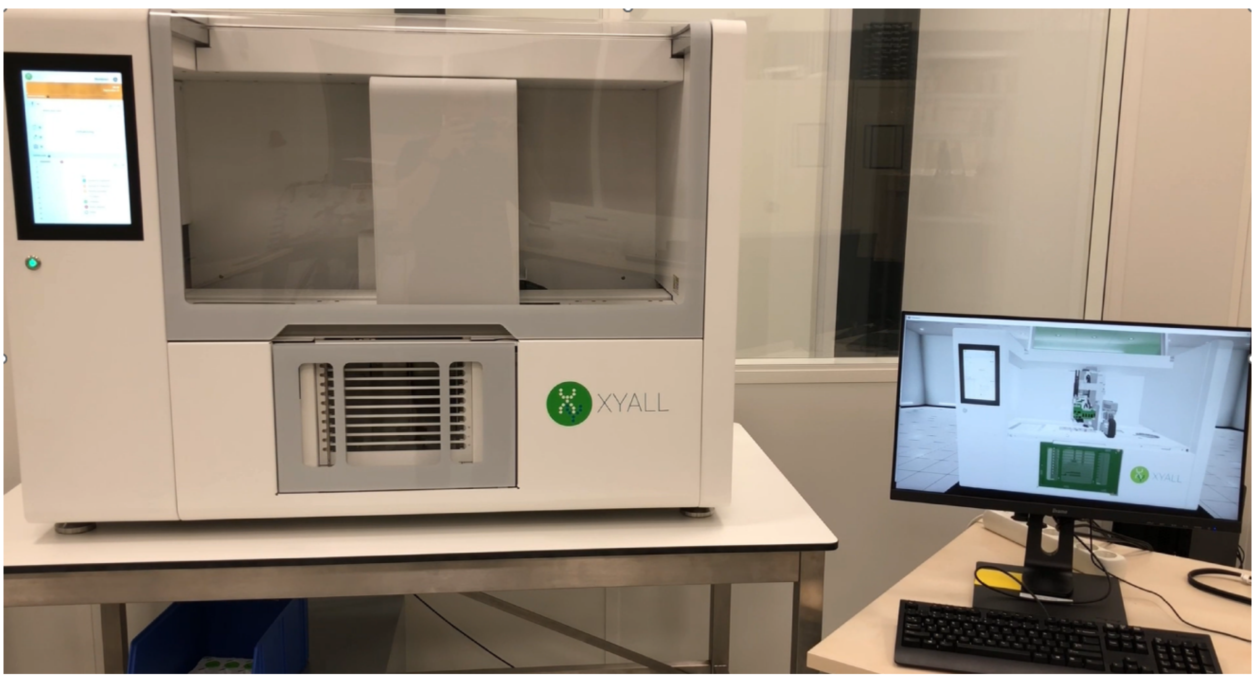


Figure 92: Pilot 1 system and its VR digital twin side by side

The VR digital twin shows the internal moving parts that are hidden behind the cover of the real Pilot 1. Making parts ‘invisible’ is one of the features of Holodeck.

6.3.4 IMOCO4.E requirements

There are no specific IMOCO4.E requirements identified for the Sioux Holodeck. The development of the Sioux Holodeck predecessor had already started before the IMOCO4.E project, and its design and capabilities were quite generic so that no specific and additional requirements were necessary to integrate the Sioux Holodeck into Pilot 1.

6.3.5 Capabilities & Limitations (including USP, strengths & weaknesses)

What is unique about the Sioux Holodeck is the seamless integration with Sioux SuperModels, which is the generic modelling tool that Sioux uses to define the structure and behaviour of the systems that are developed. Once the system has been modelled, a link can be established to the CAD data to deliver a realistic VR representation of the system, including the behaviour and simulator generated from the SuperModels model.

6.3.6 Customizations & Adaptations (including possible modifications and extensions)

Both the SuperModels and Holodeck workbenches are platforms that are easily extendable, such that generation of new components with new behaviour in new programming languages can be added according to the (future) project needs.

6.3.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

See above. Sioux SuperModels and Sioux Holodeck are strongly interoperable tools, and the integration with CAD modelling packages is done via Pixyz Studio and FBX files. The generated code is customizable and can be imported in any programming language IDE.

6.4 Building digital twins using reduced order models with measured data [SIEMENS, PCL]

In this section, the general concept of how a digital twin can be built making use of reduced order models in combination with measured data is investigated.

6.4.1 Technical Overview

Reduced Order Models (ROMs), as integral parts of the Digital Twin methodology, simplify complex mathematical computations in computer simulations. This facilitates the creation of simpler representations of system dynamics, utilizing diverse data types such as simulation results and measurements, achieved by reducing the number of variables. In the IMOCO4.E project, ROM solutions are proposed in P2, UC1 and UC2 to reduce the system model in order to be used for real-time and HIL testing (SW catalog item SW-026 & SW-094).

6.4.2 Implementation aspects

The main implementation aspects in P2/UC1 involved the following steps:

- Data Collection: Obtained measured data from physical systems via sensors.
- Import and Training: Import measured data into the virtual model developed in T5.5 (for lift model) or in IMECH project (for semiconductors machine model), perform basic formatting and pre-processing tasks, train the model using the collected data to capture essential system dynamics.
- Model Validation: Validate the trained model by comparing its predictions with additional measured data.
- Integration with Digital Twin: Integrate the validated reduced order model into the digital twin framework.

- Continuous Monitoring and Updating: Continuously monitor the digital twin's performance and update both the model and twin with new measured data or system changes over time.

6.4.3 Results

In the context of the lift application (UC 1), the implementation of ROMs within the DT framework using Simcenter Amesim simulation platform was used in accurately capturing the essential behaviours and characteristics of the system. Integration of machine learning techniques improved the DT's capabilities by allowing the prediction of faults within the lift system. By feeding the model with historical measured data the DT proactively identified potential issues before they escalate, with ROMs having an important role in improving the efficiency of fault prediction algorithms through simplified mathematical representations of the system dynamics.

In the P2 scenario, the integration of ROMs facilitated the monitoring and analysis of the 12-inch stage's performance within the DT. Trained using data collected from sensors installed on the physical system, the ROMs accurately captured the essential mechanical and electromechanical dynamics of the stage. This ensured that the digital twin provided a reliable virtual representation of the system's behaviour, enabling the analysis and optimization of existing control. The incorporation of ROMs into the DT framework significantly improved its predictive capabilities and contributed to better operational efficiency and reliability of the systems.

6.4.4 IMOCO4.E KPIs and requirements

In the IMOCO4.E project there are no specific requirements for building digital twins using reduced order models with measured data. However, general KPIs and requirements for this topic include:

- Accuracy: Ensure that digital twins accurately represent real-world systems, with reduced order models providing correct predictions aligned with measured data.
- Computational Efficiency: Reduce computational complexity to enable real-time (or near-real-time) simulation and analysis within acceptable timeframes.
- Adaptability: Design digital twin architectures that can scale to different system sizes and complexities, while also accommodating changes and updates over time.
- Data Quality: Maintain high-quality measured data through robust collection and preprocessing methods.
- Integration: Facilitate continuous integration with existing systems, databases, and tools.

6.4.5 Capabilities & Limitations (including USP, strengths & weaknesses)

The integrated approach of using ROMs with measured data offers several capabilities that distinguish it from traditional modelling techniques. These include improved accuracy, reduced computational overhead, and enhanced predictive capabilities. By involving real-time or historical data, the digital twin can adapt and evolve in response to changing operating conditions or system dynamics. However, it is essential to acknowledge the limitations and challenges associated with this approach. These may include data quality issues, model validation constraints, computational complexity, and scalability concerns.

6.4.6 Customizations & Adaptations (including possible modifications and extensions)

Customization and adaptation of the integrated approach allow for personalizing the digital twin to specific industry requirements or application scenarios. This may involve modifying algorithms, refining data preprocessing techniques, or integrating additional sensors for data acquisition. Moreover, extensions such as incorporating advanced analytics, integrating domain-specific knowledge, or expanding the scope of the digital twin to include multiple interconnected systems can further improve its utility and relevance. By accommodating diverse customization needs and adaptation requirements, the integrated approach ensures the versatility of the digital twin across various domains and use cases.

6.4.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

It's important to carefully consider the specific requirements and constraints of each project when selecting tools and methodologies. Commonly used tools and platforms such as Simcenter Amesim, MATLAB/Simulink, Python with libraries like SciPy and TensorFlow, and data visualization tools like Tableau or Power BI provide essential support throughout the development process. While these tools offer significant advantages, it's essential to remain aware of potential limitations and challenges in their usage. Regular evaluation and adaptation of toolchains based on project needs and lessons learned can lead to more efficient and successful digital twin development processes.

6.5 Platform configuration for high-speed digital twinning [UNIMORE]

6.5.1 Technical Overview

Artificial intelligence workloads are present in several layers in the IMOCO4.E toolchain for digital twinning. The role of UNIMORE in Task 5.6 is to explore different embedded platforms to enable Deep Learning algorithms in IMOCO4.E Layer 2, i.e., SW-018 of BB8. The choice of the platform is driven by the performance constraints imposed on the different AI software components developed by the partners to be used in the different Demonstrators, Pilots and Use cases in which they are involved. Two platforms were selected to be explored, the Xilinx Zynq Ultrascale+ ZCU102 (HW-004) and the Nvidia Jetson Xavier AGX (HW-003).

Presentation of the platforms

Xilinx Zynq Ultrascale+ ZCU102

This SoC is composed of 4 ARM Cortex A53 and a dual core ARM Cortex R5F designed for Real-Time applications. It also embeds a Mali-400 MP2 GPU to accelerate heavy parallel applications and an FPGA that can be used to form a Deep Learning Processing Unit (DPU).

NVIDIA Jetson Xavier AGX

This embedded platform provided by NVIDIA has been developed for autonomous vehicles. It is composed of 8 NVIDIA Carmel CPU cores, which are custom ARM cores. It also embeds a GPU composed of 512 CUDA cores and 64 Tensor Cores.

6.5.2 Implementation aspects

In the context of Pilot 3, our platforms exploration has been used to select the embedded platform which will integrate the Neural Network (NN) algorithm component of BB8. The choice was directed through the Nvidia Jetson Xavier AGX due to its performance measured during the platform exploration phase. The power consumption of both NVIDIA Jetson Xavier AGX and Xilinx Zynq Ultrascale+ ZCU102 is not constrained in Pilot 3.

6.5.3 Results

The exploration was performed on a set of Neural Networks (NN) algorithms proposed in the literature i.e., Cnet, Yolo-v3, Yolo-v3-tiny and Mv2.

Description of the NN algorithms

CenterNet

CenterNet (CNet) models objects as single points. Its uniqueness is found in the regression of all other object features, such as sizes, 3D locations, and pose orientations, as well as in the estimation of a key point that is utilized to determine centre points. In this model, an image is input into the CNN, which produces a heatmap with maximum values denoting the centres of objects found in the image. Next, each object's size and posture are regressed into features at the centre location. While there are other conceivable configurations based on project requirements, we report using CNet with two distinct backbones, i.e., ResNet101 and DLA34. We substituted the convolutional layers with deformable convolutional layers v2. Deformable Convolutional Networks (DCNs) allow the recognition of objects with geometric variations, whereas normal convolutional networks represent identified objects with a fixed square size. While the offset from the initial convolution layer (of fixed size) is learned during training, DCNs generate a deformable box.

Yolo-v3

Yolo-v3 creates grid cells from the input images. Next, it uses dimension clusters as anchor boxes to predict Bounding Boxes (BBs). Then, each BB is given an object score by means of independent logistic classifiers. BBs are then predicted on three different scales from which features are extracted. Yolo-v3 has a faster inference speed than Yolo-v2, and it also increases the overall accuracy of small-sized object detection.

Yolo-v3-tiny

Yolov3-tiny is a lighter version of Yolov3. Instead of using three scales and Darknet-53 with 53 convolutional layers as backbone, it uses only two scales and 10 convolutional layers in the backbone.

MobileNetv2

MobileNetv2 (Mv2) is very popular on mobile devices with its lightweight and high FPS throughput. It uses depth-wise convolution layers with less weight than ordinary convolution layers. It includes a one stage detector called Single Shot Detector (SSD). Similarly to Yolo, SSD divides images into grid cells. A pre-generated set of multi-scale anchors is used for each grid cell, and aspect ratios are used to discretize the BB output space. The SSD makes object predictions across multiple feature maps, each focusing on one object scale.

Training Phase

We select COCO 2017 which is widely used for object detection comparison to train all the NN explored with the same dataset. However, for CNet, we reuse the weights from its authors. Also, we used the same input size of 512x512 for each image. This size constraint comes from CNet. COCO 2017 is composed of a set of 118K images for training purposes and a set of 5K images for validation purposes.

Experiments

The worst-case end to end latency of Yolo3, Yolo3-tiny and Mv2 implementation using Single Shot Detector (SSD) is shown in Figure 93. CNet is not shown as it is not ported on the Xilinx Zynq Ultrascale+ ZCU102. Three platforms' configurations were explored:

- The Xilinx ZCU102 using only INT8 data type (ZCU9EG*).
- The Nvidia Jetson AGX Xavier using INT8 data type (Xavier AGX*).
- The Nvidia Jetson AGX Xavier using FP32 data type (Xavier AGX†).

The Nvidia Jetson Xavier AGX presents the best worst-case end to end latency [ms] in comparison to the ZCU102 in every phase, pre-processing, inference, and post-processing.

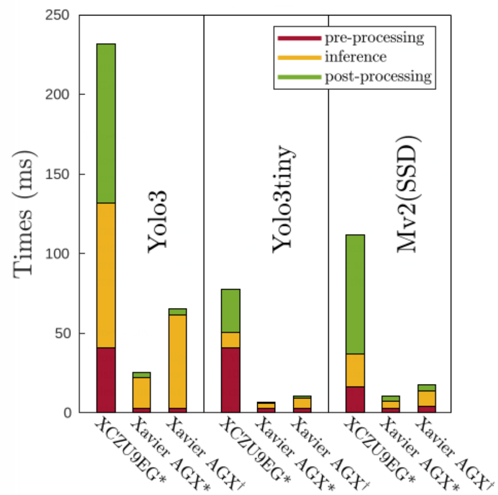


Figure 93: Worst case end to end latency of the explored NN algorithms

The efficiency [FPS / W] in relation to the mean Average Precision (mAP) [%] for the different platforms is shown in Figure 94. Several configurations have been explored: 32 bits Floating points (FP32), 16 bits Floating points (FP16) and 8 bits integers (INT8) for the Xavier AGX. However, the exploration on the Xilinx ZCU102 is limited to INT8.

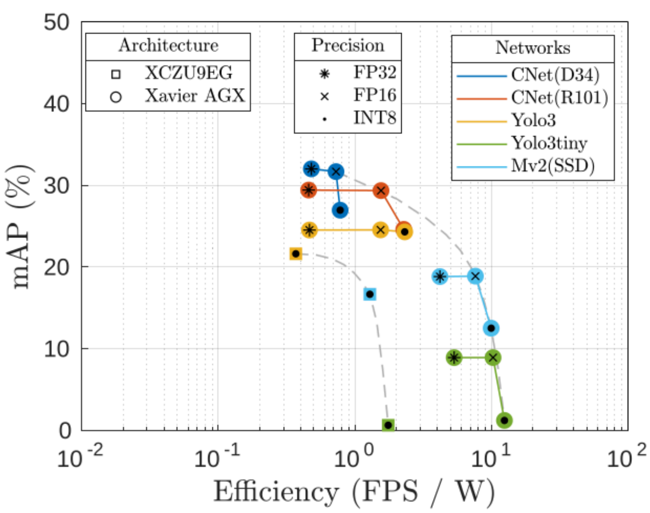


Figure 94: Efficiency in relation to the accuracy of the explored NN algorithms

We can notice that the NVIDIA Jetson Xavier AGX presents better efficiency than the Xilinx Zynq Ultrascale+ ZCU102 while also providing better precision.

The ML algorithm component from BB8 was successfully deployed on the Nvidia Jetson Xavier AGX during the integration day of Pilot 3. The deployment was performed using a docker container and the ML algorithm operated as expected. Measurements of its execution time have been performed by GNT on a

Jetson Orin Nano 8 GB. The average execution time for pre-processing and inference is 52 ms with a floor value of 42 ms and a ceiling value of 62 ms. The ML algorithm uses a single CPU core and, granted that the CPU of the Jetson Orin Nano 8 GB (Arm Cortex-A78AE v8.2 64-bit CPU) has a very similar performance with the CPU of the Jetson Xavier AGX (NVIDIA Carmel Armv8.2 64-bit CPU), the same performance levels of the ML algorithm are expected on both platforms. However, according to (NVIDIA, 2024), the Jetson AGX Xavier exposed better performance compared to the Jetson Orin Nano 8 GB for PeopleNet (v2.5 unpruned), Action Recognition 2D, Action Recognition 3D, LPR Net, Dashcam Net and Bodypose Net.

6.5.4 IMOCO4.E requirements

The study provided by UNIMORE contributes to ST4 by selecting a suitable platform for the execution of AI based methods for machine predictive maintenance, i.e., SW-018 of BB8. Moreover, it is in line with the objective TRL2 of KPI "High-performance embedded computing platforms with real-time capabilities" of BB4, i.e., "Existing solutions are either limited to best-effort applications, e.g., AI edge nodes without strict timing requirements, or significantly underutilize the hardware in order to avoid interference between the computing engine".

6.5.5 Capabilities & Limitations (including USP, strengths & weaknesses)

The study proposed by UNIMORE is limited to two Platform, the NVIDIA jetson Xavier AGX and the Xilinz Zynq Ultrascale+ ZCU102. However, several NN algorithms from the literature have been explored: Cnet, Yolo-v3, Yolo-v3-tiny and Mv2.

6.5.6 Customizations & Adaptations (including possible modifications and extensions)

The survey may be completed with other platforms and other metrics such as energy consumption. It may also be focused on a specific set of NN algorithms.

6.5.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

We chose the ONNX Runtime as it allows switching among different trained models while preserving a single inference engine, while being able to profit from the available hardware accelerators on the platform. Its C++ API provides guarantees on the performance during pre-processing and post-processing. We use PyTorch 1.4 and the operator set in version 11 on the NVIDIA Jetson Xavier AGX. Indeed, ONNX layer export is natively supported by PyTorch. Since the PyTorch framework is not supported by the Xilinx environment, we use open source code6 to convert the supported model to Caffe.

Due to its specific architecture, we did not achieve the porting of CNet on the Xilinx Zynq Ultrascale+ ZCU102. We also have to switch from ReLU6 PyTorch operator to the clip ONNX one with value ranging form [0;6] for Mv2 on the NVIDIA Jetson Xavier AGX.

Concerning the porting of Mv2 on the Xilinx platform, the final Reshape layers have been replaced with Flatten type layer as these layers are not supported by the DPU available on this platform. In addition, we converted the final Softmax layer into a software layer running on CPU and accelerated with OpenMP.

Yolo-v3-tiny has also needed some change due to the DPU architecture on the Xilinx platform. The last max-pool layer forming the NN has been removed. Nonetheless, the effect of the Yolo-v3-tiny modification on the result accuracy was barely noticeable: 0.9 % on the Xavier when comparing versions with and without the layer removed.

6.6 Digital twin toolchains for control system development [NORMET, EXERTUS, VTT]

This section describes digital twin toolchains including tools for testing and implementation of mobile machinery control system algorithms. Advanced HIL (Hardware-In-the-Loop) and SIL (Software-In-the-Loop) systems are integrated into digital twins and virtual testing environments. Development of a Digital Test Cell (DTC) concept including a partly or fully simulated control system for end-to-end testing of mobile machinery is one of the contributions to enable utilization of DTs for testing and implementing the designed motion control algorithms.

6.6.1 Technical Overview

Testing and validation of machine control system code and algorithms is one of the essential parts of the development cycle when developing new machine control algorithms. The challenge usually is that there is no real-life machinery available to test and validate the system or the situation might be that machines are not even designed and ready for testing. Also testing in real machines requires a lot of special equipment, facilities and resources. Many times, the fact when creating new concepts is that not only the algorithms but also new controls system components, advanced mechanical structures, user interfaces and safety features are developed, and all these components must be tested and evaluated as a whole. These conditions bring up the relevance of utilizing virtual environments and simulations for developing new systems. In addition to utilizing the digital environments and simulations in development phases, using these systems during the whole product lifecycle increases productivity and is more cost effective. For instance, new machine control system features including mechatronic systems can be tested and validated in the digital twin counterpart of the machine in the virtual environment prior to commissioning to production and to field machines. Control system software already in production or maintenance phase can be easily automatically tested against the existing digital twin system including pre-testing of the safety critical functionalities of the control system.

6.6.2 Implementation aspects

Within this task, digital twin toolchains are created which are based on architecture of digital twin testing and development environment for work machine control system shown in Figure 95. This architecture is referenced from the IMOCO4.E general specification and design framework. The working machine usually consists of the manipulator and its control system and the machine platform itself. The machine platform control system comprises of the main controller, distributed IO-modules, the driveline and human machine interfaces such as displays and remote joysticks. The manipulator control system has its own main controller and sub-system components for motion control purposes. These all create a complex system with many hardware modules interfaces and software components. To create an environment for developing and testing algorithms comprehensively, a digital representation of the controlled mechanical system and environment is integrated with proper interfaces to communicate in real time with the hardware and virtual software components.

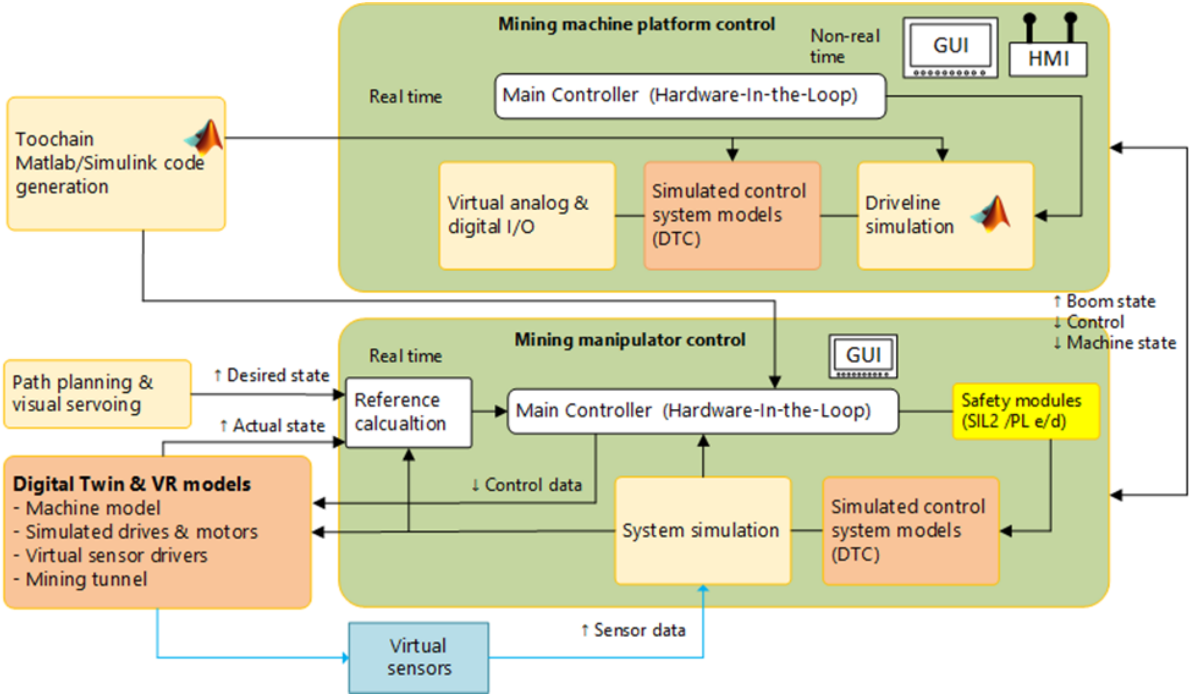


Figure 95: Pilot 5 control system integrated in the digital twin testing & development environment

6.6.3 Results

Pilot 5 Mining robotic boom manipulator is one of the use cases to utilize digital twin toolchains for control system development. For Pilot 5, the previously described digital test & development environment is an essential part of the overall solution for motion control algorithm development and testing.

The digital twins that are used in this task are so called virtual modules and they are implemented by Exertus. There is a separate virtual module for each module type. The modules contain Guitu Runtime as their firmware, and they can be configured with Guitu (Graphical User Interface tool). These virtual modules are PC programs that can be run individually or together as a system. When they are running together, they will communicate with each other with Exmebus (Exertus message bus), which simulates CANopen bus on a PC. Exmebus can deliver CAN messages, IO messages and GUI synchronization messages in real time between modules.

Exertus virtual modules are easy-to-use. They provide a logical user interface, and they can be controlled with keyboard or mouse. The virtual modules are constantly connected to Exmebus for error handling. If the connection breaks up, Guitu Runtime tries to reconnect within 30 second period. The virtual modules can be always run in the background. Parameters can be saved to modules. In the case of virtual modules, the parameters are saved to backup files. The virtual modules use the same source code as their physical counterparts. The virtual modules have been designed to work as nearly as possible like their physical counterparts.

Exertus virtual modules support Simulink programming through IOAdapter software. IOAdapter is software that can be connected to Exmebus and it delivers messages between simulator and virtual modules. IOAdapter reads logical IO signal messages from simulator and then sends IO messages to virtual modules. IOAdapter can also be used to visualize IO signal values.

Exertus virtual modules can also connect to Redi Cloud. Users can access historical data with Redi Client software from their own PC. It is also possible to connect to a machine remotely via Redi Cloud if the machine is connected to Redi Server. The Redi Client will connect to Redi Server with an SSH connection. When an SSH connection is established and authorized, the user can connect to the machine's Exmebus with virtual modules through the SSH tunnel. The virtual module then operates in monitoring mode.

Machine and manipulator control systems are integrated into the system as real hardware together with virtualized IO-controller hardware created in the Digital Test Cell concept. The subsystem components create the development environment in which the whole control system is running in real-time. This architecture enables an efficient and iterative algorithm development, parametrization and testing in a real-like, compact and safe environment before commissioning the algorithms into real mining machines.

6.6.4 IMOCO4.E requirements

The requirements presented in this chapter are also depicted and described in detail in deliverable (D5.2). Pilot 5 system level requirements are listed below along with a status update.

Interfaces and connectivity

Req143- -P5-1: *Interfacing the digital twin environment with control system is analogical with the actual physical interface*. **Status**: In Pilot 5, this is tested in the mining machine application. The digital twin main control system is similar to the real machine control system (see reference architecture).

Maintainability (modularity, analysability, testability)

Req144- -P5-2: *Configuration of the machine platform controllers' parameters can be done against the virtual environment***. Status**: The digital Test Cell concept enables parametrization of the IO-modules and machine controller in the digital twin environment. The proof-of-concept works in the Pilot 5 mining application.

Req145- -P5-3: Manipulator motion control algorithms (path planning and execution, collision avoidance, visual servoing) can be verified against the digital twin counterpart. **Status**: All motion control algorithms are evaluated and tested against the digital twin model of the mining machine in avirtual tunnel. Testing shows that the toolchain works. The same controller algorithms also work in the real prototype manipulator. This is tested in the VTT laboratory boom.

Performance

Req146- -P5-4: *Digital twin environment and xIL (hw&sw in the loop) environment runs in real time.* **Status**: Same code and control system hardware components exist in the real application and in the virtual environment. Tested within the VTT laboratory boom.

Req147- -P5-5: Robotic manipulator code can be run directly in the xIL/digital test cell control system. **Status**: Similar to Req146--P5-4.

Compatibility (interoperability, co-existence)

Req148- -P5-6: Motion control algorithms should be configurable/adapted to different types of sensors and manipulators and testable against the digital twin.

Status: Motion control algorithms are adaptable to multiple types of manipulators – configuration of DH-parameters and configuration of tuning parameters must be done. Tested with tunnelling application (spraying boom) and explosive application.

6.6.4.1 Usability (operability)

Req149- -P5-7: *Visualization of the control system I/O signals for analytics in the digital test cell environment.* **Status**: The digital Test Cell concept enables reading of logical IO signal messages from simulator IO modules and controllers. The IO Adapter can also be used to visualize IO signal values.

Req150- -P5-8: *VR capability (Unity) of the digital twin.* **Status**: The Pilot 5 mining manipulator digital twin environment consists of virtualization of the machine in MeVEA simulator with plugin to Unity, which enables integration of virtual 3D representation of the machine.

Safety

Req151- -P5-9: *Safety critical features of the mining machine can automatically be tested in the simulation environment.* **Status**: The Hardware-in-the Loop (HIL) system consists of real embedded motion controllers and real Safety IO-modules. This enables testing the safety sensors and specific safety related features in the simulator environment. For instance, all collision avoidance algorithm pre-testing is done in the digital simulation environment prior to machine testing.

Tools/toolchains

Req152- -P5-10: *Motion control algorithms can be tested and verified in simulation (MATLAB/Simulink) before commissioned to HIL environment.* **Status**: In the Pilot 5 use case, before porting the motion control algorithm, algorithms are tested and validated within the MATLAB environment with simulation tools. C functions are compiled as MEX functions and run in MATLAB. With this method it can be ensured that the C language version is working similarly to that of the original MATLAB function and can be ported to the HIL-environment embedded control system.

6.6.5 Capabilities & Limitations (including USP, strengths & weaknesses)

Having a real control system HIL-components in the simulator environment similar to the one in real machines all higher level control system algorithms can be developed and verified, and the controller's parameters can be commissioned against the virtual environment. For instance, the driveline and digital twin model of the machine (electric drives, mining environment, virtual sensors) can be simulated without the need for changing the control system component interfaces. Thus, the interface with the digital twin environment is analogous to the actual physical interface from the control system perspective. Because the digital twin environment and xIL (HW&SW) runs in real time, the execution of manipulator control algorithms can be run directly in the digital test cell control system. Moreover, immersive integration to Unity-based virtual reality is important for user experience which enables realistic real-time testing of the implemented motion control with user interfaces such as camera display and real remote joysticks for manipulator control.

6.6.6 Customizations & Adaptations (including possible modifications and extensions)

Advanced motion control algorithm development usually requires manual coding and testing in high level simulators such as MATLAB/Simulink. Currently the approach is that algorithm level testing is done in the early phase of development work and MATLAB/Simulink is utilized in algorithm development and testing. When the algorithm is working in MATLAB, it is ported to the target hardware. Before porting the algorithm, it is tested and validated within the MATLAB environment with simulation tools and after that it is usually manually coded with the C programming language. C functions can be compiled as MEX functions and run in MATLAB. With this method it can be ensured that the C language version is working similarly to that of the original MATLAB function.

After testing the C language version in the MATLAB environment, the code is ported into the microcontroller or PLC (Programmable Logic Controller) and tested against the digital counterpart, running the code on real controller hardware like that in real machines. In this phase integration testing can be done, meaning the platform and process control systems are in place, and for instance, the developed manipulator motion control algorithm with automated movements can be tested. After the whole process is tested with the digital twin system, the algorithm is tested with a real prototype manipulator in the laboratory which consists of similar manipulator control system hardware to that in the simulator. The difference between the simulated system and the real manipulator is that the simulated system boom has virtual motors, encoders, drivers, and simulated environment. In general, testing is done iteratively, and features need to pass testing in the HIL environment before they are ready to be tested with a real boom prototype in the laboratory environment. The same principles apply to both HW and SW components.

6.6.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

The current algorithm development process contains lots of manual intervention with conversions between developed code in MATLAB and target control software written in C/C++. One development target in this task has been to increase automated toolchain usage and software testing to speed up the development and testing process in the DTC concept. The idea is to maintain the algorithm codebase in simulation environment where it can be developed, tested, and then integrated as a C/C++ code with an existing codebase and finally ported to the target environment.

In this case MATLAB/Simulink has been utilized as it is a widely used development environment for the smart control algorithms, digital twin models and for AI/ML components. MATLAB has support for clear workflows for development and iteration of code and the possibility to use continuous integration pipelines. For code generation purposes, there is a MATLAB Coder that enables users to generate C/C++ code from MATLAB algorithms and similarly a Simulink Coder that does the same for Simulink models. As an add-on, there is Embedded Coder that provides additional code optimization, verification and targeting capabilities such as processor-in-the-loop (PIL) and software-in-the-loop (SIL) testing. As the mining machine platform and boom control uses algorithms developed with both MATLAB code and Simulink block diagrams, in this case both scenarios were tested by selecting existing control algorithms for the code generation process.

To meet the needs for code generation and integration for mobile machinery control systems, there are some programming considerations to be aware of. Compared to target C/C++ code, MATLAB has differences in typing, array sizing, memory allocation and some additional features in toolboxes that code generation does not support. Therefore, some effort from code generation workflow is put into planning, how the implementation and interface between generated code and external code base should be done.

To prepare the code generation process, all the used variables had to be defined in coder as C and C++ use static typing. Also, for variable-size arrays and matrices inputs, outputs, and local variables must be defined beforehand. To produce well working code that can be integrated into the existing machine control codebase, some configurations must be made for code generation. For example, function prototypes, data storage types and code optimization targets should be configured. The user can define how a function accesses the model parameters and how root level inputs and outputs are handled by the function.

For MATLAB code, used inputs, outputs and variables were defined in the coder app by generating a testcase script in MATLAB that calls the entry-point functions with sample inputs. However, the same can be done manually, but using test scripts provides also a smooth way to test the generated C code in the verification phase where the coder app generates a MEX function from C code, that can be called from inside MATLAB. The coder app runs the MEX algorithm function and reports run-time issues. This way it is possible to detect and fix run-time errors that are harder to diagnose in the generated C code. In addition, this step includes memory integrity checks.

For the MATLAB algorithm, no specific issues were found during the generation and testing process done in a PC environment. Final integration and validation in target hardware was not tested in this case. The generated code and output files may however differ in some parts from manually written C code. For example, even if the code generator preserves the function name and comments the variable names may be changed from the original. Also, code generation produces multiple files per function implementation that should be considered when integrating the code to a main codebase where simplicity and reduced number of files may be more convenient (i.e., one ".c" file and a corresponding ".h" file).

In the Simulink implementation, when code is generated, model entry point functions are defined in header file. This file can then be used in an external code base to call generated code functions. Two model entry point functions were generated for generated code specifically for initialization of the model and the model base step function. The initialization function is used to initialize model parameters and the step function has step a routine of the model and is called by the execution harness at every timer interruption.

The SIL simulation in Simulink was used to make initial verification of the generated code. The SIL manager in MATLAB compiles generated source code and executes the code as a separate process on a host computer. By comparing normal and SIL simulation results, the numerical equivalence of the model and the generated code was tested. During a SIL simulation, it is also possible to collect code coverage and execution-time metrics for the generated code.

Integration of the generated code into the existing code base was done by implementing an interface between generated code and existing code by using a wrapper function. Model entry point functions are called inside this wrapper function and the wrapper function is called at every program execution cycle when the controller functionality is requested, and it makes sure that the right initialization parameters are set for the model and then it calls the model step function to update model states.

The application code was built for the virtual embedded module (DTC component) to test the controller functionality. Tests were run in the machine digital twin testing and development environment. The controller function from generated code worked as intended within the digital twin environment and results were validated to be satisfactory by functional testing in the overall control system against the digital twin model.

Previously described tests were done using a coder application with a graphical user interface, but the code generation can be automated by using the command line scripting. This allows for a more automated development process where algorithm development done in simulation environment can be seamlessly integrated with an existing control software codebase. Based on the findings, automation in code generation can be gradually increased building up the workflow, in both practical processes and technical tools. This will certainly bring forward the development of Continuous Integration (CI) and Continuous Delivery (CD) pipeline.

Model and algorithm development together with general control software development is a central part of creating a virtual representation of the system in digital twins. Implemented tools and toolchains (for code generation) are in great supporting role of algorithm implementation and integration making the development process more efficient.

6.7 Digital twin in furnace optimization [PCL]

6.7.1 Technical Overview

Together with partner REDEN, a digital twin is developed. This model (digital twin) is needed for furnace (HW-050) optimization. The model is developed in Abacus. See SW-084. The furnace is used for the heat treatment of parts.

6.7.2 Implementation aspects

The model is still under construction. Tests are performed to further update the model and use it for improving the performance of the furnace. The model is also used to gain more knowledge about hardening and the dimensional aspects of the parts. Further implementation is waiting on more results of next tests.

6.7.3 Results

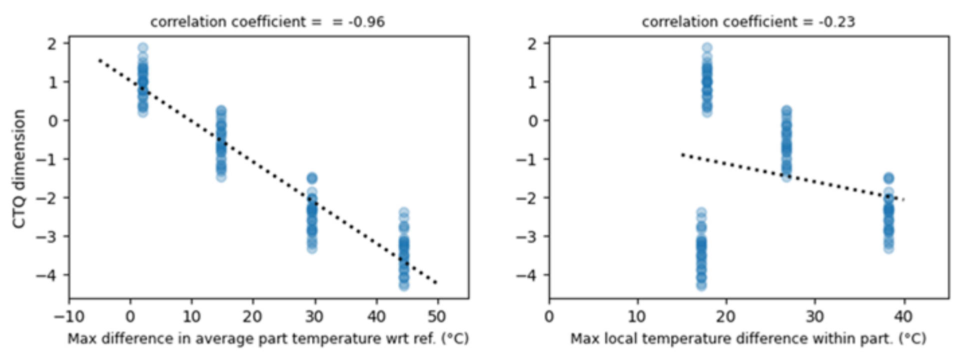


Figure 96: Temperature variation between and within parts vs. a critical dimension

A test was performed with different ways of placing the parts on the furnace belt. The concept model gave input for 4 possible ways, expecting to have the most difference. Then the measurement results of the CTQ (Critical to Quality) dimension were compared with two temperature characteristics. See Figure 96; the difference in average part temperature and the difference in temperature within parts. It is visible that the average part temperature shows the best correlation. A follow-up of this test will be to find out where in the process the difference is caused.

6.7.4 IMOCO4.E KPIs and requirements

R153 -D1-1: *The system can function disconnected from the internet*: This is true, since there is no direct interaction between the model and the production process.

R154--D1-2: *Test possibility after changes and maintenance*: after changes on the model, the model can be tested to see if it shows the same outcome.

R155--D1-3: *Maintenance possible during production (no stops needed)*: This is true, since there is no direct interaction between the model and the production process.

R156--D1-4: *Reduce human workload*: Reduction of human workload is limited, only related to the next requirement.

R157--D1-5: *Reduce machine stops*: By improving the dimensional quality of the parts, the risk of production stops is reduced.

R158--D1-6: *Real-time decision-making functionalities*: Because there is no direct interaction between the model and the production process, real-time decision-making is not applicable.

R159-D5.1-D1-7: *Continuous learning systems*: Changes of parts and new data is input for further (continuous) improvement of the digital twin.

R160-D5.1-D1-8: *Enable sensor-controlled functions*: The temperature measurements in the furnace are done by sensors and checks of the overall temperature-time profile are done by sensors too. When settings are not as expected, adjustments are done.

R161-D5.1-D1-9: *Store data from various sources*: Measurement data of parts is stored in the factory data system. Gathered knowledge is stored in a knowledge storing system at PCL.

R162-D5.1-D1-10: *Autonomous or semi-autonomous operations for quality checks*: The digital twin can help finding possible semi-autonomous operations for quality checks.

R163-D5.1-D1-11: *New approaches for data correlation extraction*: The found correlation between part quality and process characteristics is new.

R164-D5.1-D1-12: *Minimize structural costs*: Quality issues are linked to structural costs, so the digital twin helps to lower structural costs.

R165-D5.1-D1-13: *Fit for different processes*: Although the model is made for one specific process, there are possibilities to fit it for other processes too.

R166-D5.1-D1-14: *Must be safe for humans, products, machine/system and environment*: There are no safety issues related to the digital twin.

6.7.5 Capabilities & Limitations (including USP, strengths & weaknesses)

Limitations for the model are that it is (initially) developed for one production process.

6.7.6 Customizations & Adaptations (including possible modifications and extensions)

A customization is to make the model fit for comparable processes.

6.7.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

The development of the digital twin and the collaboration with a partner has shown that a digital twin can improve processes and gain knowledge. Process and part quality data can be used to improve the digital twin.

6.8 Virtual reality digital twins, enabling human-computer interaction [UCC, ADI, EMD]

This section presents the viewpoints of digital twins (DTs) and virtual reality (VR) applied to Use Case 3 (UC3) research conducted in relation to human-computer interaction in the context of tactile robot tele-operations.

6.8.1 Technical Overview

The tactile robot constitutes the next generation of soft collaborative robots, equipped with sensing capabilities to process humanlike tactile sensation. Human safety and labor/skill shortages in 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. Remote robot tele-operation is the core focus of UC3, which has the Tyndall National Institute, (UCC) as the lead partner and two industrial partners, Analog Devices Ireland (ADI) and Emdalo Technologies (EMD).

This section will provide a general overview of UC3 with reference to DT and VR research conducted and the potential for DT advancements as UC3 moves up the TRL scale in future research projects outside of IMOCO4.E.

UC3 research and development will contribute to future safe remote tele-operation for many evolving forms of tactile robotics. Critically, humans in the loop will be considered through complex human machine interface (HMI) technologies that can be coupled with future digital twin innovations along with virtual and augmented reality developments. The UC3 is also heavily edge focused and is enabled with high performance AI embedded close to the edge. Figure 97 provides a graphical overview of UC3 that has already been presented in selected IMOCO4.E deliverables.

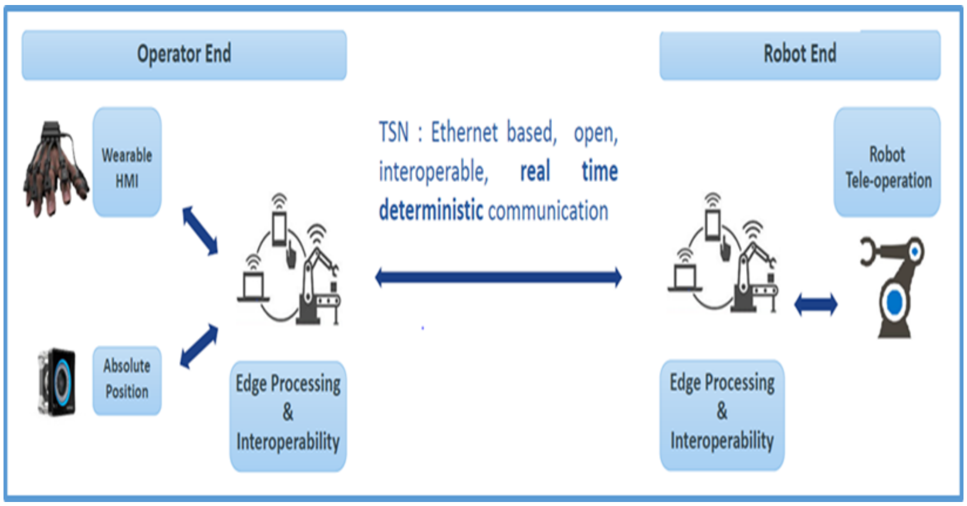


Figure 97: UC3 Tactile Robot Teleoperation Platform Overview

6.8.2 Implementation aspects

UC3 IMOCO4.E Digital Twin Concepts: The UC3 incorporates both a local user end and a remote robot end in terms of an overall platform. The architecture was developed with reference to the four-layer IMOCO4.E architecture and selected building blocks BB1, BB3 and BB8 are most applicable across the platform research activities. Figure 98 provides a non-technical overview of both the local and remote ends of the UC3 platform.

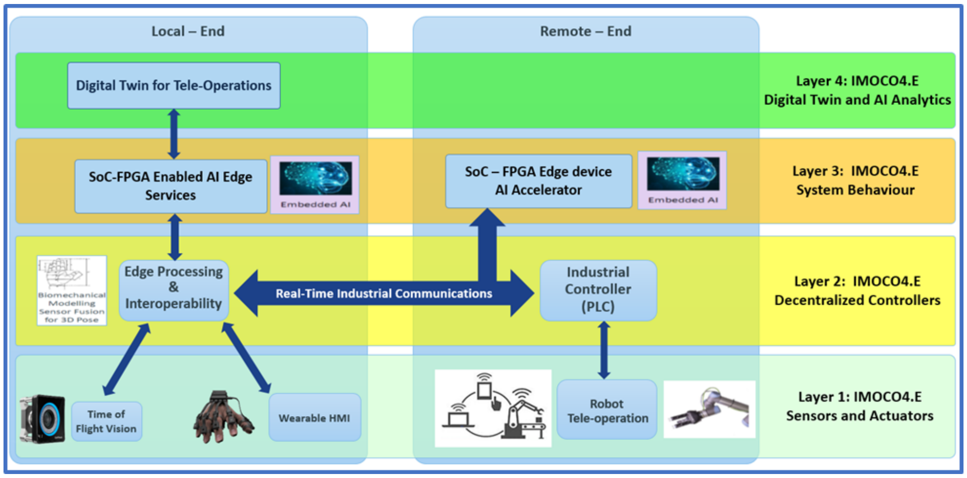


Figure 98: UC3 Tactile Robotic Tele-Operation Platform: IMOCO4.E Architecture

With a focus on the DT and VR aspects, the current version of the architecture has the DT and VR components located at the local end in layer 4 "Digital Twin for Tele-Operations" as per the above diagram.

As reported in (D5.6), the overall DT research, engineering, and development work in relation to UC3 incorporated the following concepts and requirements:

- The DT will also integrate virtual reality (VR) concepts and functionality and will be referred to as DT/VR from now on in this description.
- The DT/VR will primarily exist at the user/operator end of the platform.
- In the architecture specification, the remote robot end of the UC3 platform will be responsible for the communications of data streams back to the local end for the updating of the DT/VR environment.
- The data streams from the remote end should represent live robot activations/movement co-ordinates along with data to represent both objects and scene/environment representation back to the user/operator local end.
- The DT/VR should ideally be developed as a separate PC/Server platform such that future advanced engineering of the DT/VR may be envisioned. UC3 researchers believe that in the coming years, core DT functionality components will be engineered and be available for selected edge devices, but these innovations are also most likely to require interfacing with the main backend DT/VR platform.
- The DT/VR should be engineered to interface with the edge communications devices at both the local and remote ends of the UC3 platform.

DT/VR UC3 Technical Hardware (HW)

As discussed, the DT/VR systems are proposed to be deployed on a separate server system that is ideally provisioned for the user at the local end of the platform. User interaction may incorporate limited application of the DT/VR capabilities up to full total immersion using VR/AR headset interfacing for robotic tele-operations. Research this far on the UC3 has been primarily proof of concept which has mainly been conducted during year one and early year two of the project.

While a full technical discussion on the UC3 HW is outside the scope of this section of the report deliverable, the below diagram provides an insight for the reader into the various HW components at the local end of the UC3 platform where the DT/VR is provisioned for in the platform architecture. Please note that the DT/VR proof of concept developments in UC3 are all provisioned for in a PC/Server, which also connects with the Vive Trackers and ADI ToF sensor in Figure *99*.

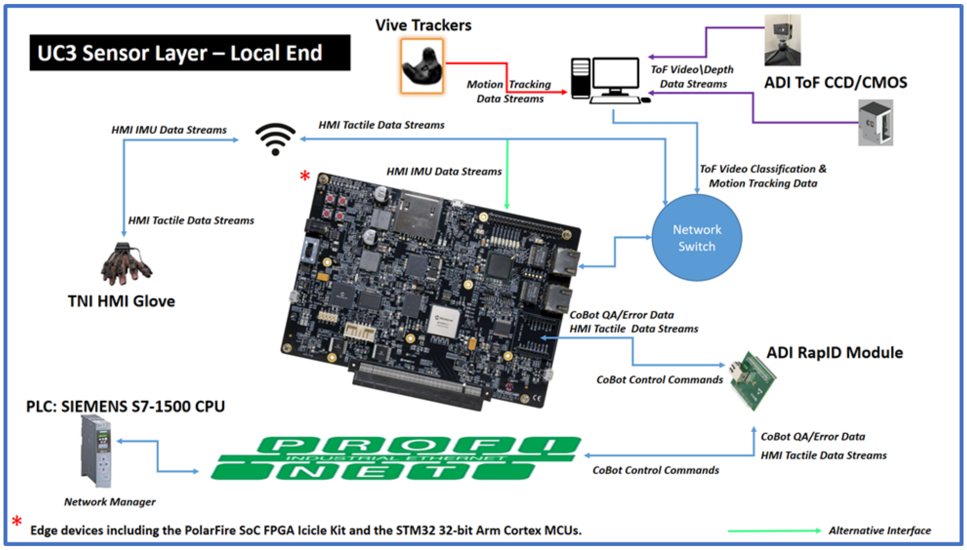


Figure 99: UC3 Sensor Layer - Local End for DT/VR Activations

For further details on the hardware of both the local and remote ends of UC3, the reader is advised to refer to selected deliverables of WP3 "*Perception and instrumentation layer based on AI at the edge*". Please refer to (D3.3), (D3.4) and (D3.5) for additional details.

DT/VR UC3 Technical Software (SW)

UC3 Digital Twin and the IMOCO4.E Architecture: Figure 100 is reproduced with reference to WP6, deliverable (D6.1) and is an important figure to understand and position the UC3 DT/VR. The figure explains the relationship between physical objects and the varying stages of evolution to a full DT.

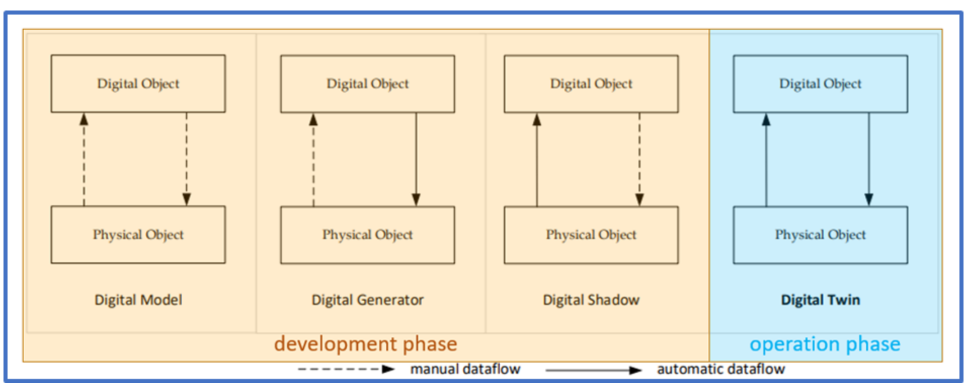


Figure 100: Relationship between Physical Objects and Digital Objects

At present, the UC3 partner team considers that the DT/VR is proposed as a "**Digital Shadow**", and this is not expected to change until there are major advancements in DT development and deployment technologies. In the digital shadow scenario, the DT is receiving and processing near real-time data streams fed by the edge device(s) in relation to the robot physical representation and must also represent the scene physical objects and remote end world environment.

As the UC3 moves up the TRL scale, then a full DT platform will be required with the potential for direct control of the robot from a DT/VR world environment. Future services from such a DT/VR platform will also incorporate synthetic dataset creation for machine learning and AI based performance, prediction, and condition monitoring functionality.

UC3 Digital Twin Development: User Interface and Functionality: This section provides a general discussion on the user interface and functionality in relation to the development of the DT/VR in UC3.

During the main proof of concept work for the UC3 DT/VR infrastructure, all project partners contributed to the various WP5 deliverables. The Unity platform was selected as the development environment and was integrated with HMI and motion tracking sensors at the local user end of the UC3 architecture. Virtual reality headsets were also investigated in terms of user experience and interaction with the DT/VR world.

As part of the research and development conducted, the DT/VR infrastructure was also engineered to provision a virtual representation of the actual robot and part-environment that would typically exist at the remote end of the UC3 platform. In experiments conducted at the Tyndall test labs, the DT/VR prototype was also used to drive the physical arm and gripper on the UR16e robot. While the practical research and development proof of concept work fully finished early year two, the partners continued to research the next development phase of the DT/VR UC3 infrastructure by contributing theoretical and technical insights for related WP5 deliverables.

6.8.3 Results

Figure 101: UC3 DT/VR Proof of concept research and development

Figure 101 captures the results of the proof-of-concept work conducted mainly in year one of the IMOCO4.E project. In the left part of the image, the research and engineering work built a prototype tele-operations platform based on the IMOCO4.E UC3 architecture. Here the user is wired up with sensors on their right hand and this represents the local end of the UC3 platform. Actions by the user at the local end are then communicated to the remote end to be activated by the robotic infrastructure. In the case of the left part of the image, the user is operating their arm and hand virtually to stack several blocks (block world example) physically at the remote end of the platform.

Moving to the image at the right-hand side, there is a prototype DT of the robot. On inspection, the three random objects that appear to be floating are an attempt to represent the remote physical world virtually in the DT at the local end of the UC3 platform. This was complex to engineer, and the researchers proposed attaching motion trackers to the physical objects at the remote end such that they could be represented (in near real-time) virtually in the DT at the local end. This research and development work very quickly identified the complexities of representing a remote world of objects that is under robotic manipulation within a DT.

The proof-of-concept prototype work described above was primarily conducted to identify complexities and risks in the overall approach to the UC3 research, before building out the edge-to-edge platform. With reference to the DT/VR year one set-up, this contributed to the identification of many challenges and futures, which are discussed in the WP5 deliverables and below for the reader.

The work conducted for WP5 confirms that the development of truly flexible multi-user capable robotic tele-operation in the future will most certainly require highly sophisticated DT/VR/AR platforms. These platforms will be required to provide a complete range of services to robot tele-operators/users that will perform their work using such direct digital interfaces.

6.8.4 IMOCO4.E KPIs and requirements

The below presents selected UC3 DT/VR requirements already presented in full detail in deliverable (D5.2). The UC3 partners have compiled the requirements below, as an IMOCO4.E contribution to any future development of a full-scale DT/VR platform for robotic teleoperations. The associated status updates provided are further insights into how critical the requirements are for any future DT/VR engineering.

Interfaces and connectivity

- **Req-D5.2-U3-1-com:** Communications of both local and remote data streams from edge devices and PC/Server systems to continually provide updates to the world/objects represented in the DT/VR platform. **UC3 status update:** Tested in proof-of-concept research.
- **Req-D5.2-U3-2-com:** Communications control software (SW) such that the sensor and robotic tele-operations data can be reliably and securely communicated to the DT/VR platform and that all updates are fully quality assured (QA) in reflection and association with the actual real-life infrastructure at the local sensor and remote robotic ends. **UC3 status update:** Core to the IMOCO4.E UC3 architecture and for a future research project.

Maintainability (modularity, analysability, testability)

- **Req-D5.2-U3-1-hw-sw:** The DT/VR must be engineered and developed as a series of structured HW and SW modules capable of being seamlessly interfaced with the other modular HW and SW components of the overall tele-operations platform. The DT/VR platform should also offer a suite of standard, and AI powered analytical services to assist in the running, adaptability, and optimisation of the tele-operations tasks between the local user and the remote robot ends. **UC3 status update:** Core to the IMOCO4.E UC3 architecture and for a future research project.

Performance

- **Req-D5.2-U3-1-hw-sw:** An overall requirement in relation to performance is to achieve near real-time edge-based updates to the server-based DT/VR platform to be provisioned at the user local sensor end in the current architecture of the UC3 platform. **UC3 status update:** Tested as part of the DT proof of concept work conducted.

- **Req-D5.2-U3-2-hw-sw:** There is a mandatory requirement in the case of latency (performance and impacts) that novel approaches using AI based prediction techniques must be researched in the case of the DT/VR platform. This is to ensure that any processing requirements for the DT/VR platform do not severely impact on latency between the local and remote edge processing devices. **UC3 status update:** Critical for future research and building the DT platform.
- **Req-D5.2-U3-3-hw-sw:** Efficient use of 3D ToF scene and object data from the physical remote robot end for the processing, representation, and compilation of parts of the remote physical end as a virtual scene/world in the DT/VR platform. **UC3 status update:** Explored during the proof-of-concept work conducted for UC3.
- **Req-D5.2-U3-4-hw-sw:** Efficient use of 3D ToF object detection algorithms to identify objects in space at the remote robot end, and to communicate ToF object classifications to the corresponding VR objects in the DT/VR virtual scene. **UC3 status update:** Can be integrated into the DT/VR in a future project.

Usability (operability)

- **Req-D5.2-U3-1-hw-sw-com:** The DT/VR should have a clearly defined UI with options for on-screen visual representation of the DT/VR world scene and also total VR immersion into the DT/VR world scene with the engineering of headset options for the user.**UC3 status update:** Mandatory for any future research project work.
- **Req-D5.2-U3-2-hw-sw-com:** The DT/VR must also be engineered with an expanding set of features that will better assist users in conducting tele-operation tasks. For example, to easily stop and start, backtrack on a task, safety control features, etc. **UC3 status update:** The UC3 developments in gesture recognition will assist with this requirement in any future work.
- **Req-D5.2-U3-3-hw-sw-com:** With reference to other listed requirements for UC3 in this section, overall engineering of the DT/VR platform for low latency is also a critical requirement from a usability/user perspective. **UC3 status update:** Addressing latency has been a core requirement throughout UC3.

Reliability (fault tolerance, availability)

- **Req-D5.2-U3-1-hw-sw-com:** An overarching requirement for usability is to aim for near video image quality for the internal data driven DT/VR representation of the scene/world in order to realistically (as much as possible) reflect the actual physical robot remote end of the tele-operations process. **UC3 status update:** Advances in DT technology will benefit from this requirement, and this was found to be vital during the DT/VR research conducted.

Security (cyber-security, integrity, confidentiality, authenticity)

- **Req-D5.2-U3-1-hw-sw-com:** For real world implementation of a DT/VR for robot tele-operation there is a requirement to implement selected services of the security building block BB9. Such services as cyber threat detection, malicious attacks and denial of service prevention components are relevant requirements for the overall platform and also the DT/VR platform components and services. **UC3 status update:** Critical for real-world deployment of robotic teleoperations.

Portability (adaptability, replaceability)

- **Req-D5.2-U3-1-hw-sw-com:** In terms of DT/VR platform adaptability and replaceability there is a requirement to be able to use different robots and environments, starting with the Universal Robots family and possibly expanding to other types of robots which can be controlled by passing a set of coordinates and can return feedback for near real-time rendering of the robot and the scene in the DT/VR platform. **UC3 status update:** Any future DT platform must be adaptable to different robot interfaces with selected engineering.

Tools/toolchains

- **Req-D5.2-U3-1-hw-sw-com:** Interfacing and near real-time updating of the DT/VR must be a core component and also be seamlessly integrated into the UC3 toolchains at both the user and development levels. **UC3 status update:** Tested during the proof-of-concept work.
- **Req-D5.2-U3-2-sw-com:** Various SDKs and digital representation formats both open source and publicly available digital assets should be researched and investigated for the representation of the physical remote world in the virtual world viewed by the user at the local end. Selected tools such as Unity and others will thus form an overall integral part of the tele-operation platform with DT/VR features and services. **UC3 status update:** Unity was applied in the proof-of-concept work conducted.
- **Req-D5.2-U3-3-sw:** Research and development is required into how the DT/VR platform can be used to support the compilation of AI/ML datasets. Such datasets may be actual live tele-operation data or be synthetic/mixed datasets generated by conducting tasks/processes exclusively in the DT/VR world without the need of the physical world set-up.**UC3 status update:** This work was explored during the year 1 work efforts.
- **Req-D5.2-U3-4-sw-com:** World/Scene representation: In a DT/VR toolchain context, there is a requirement to investigate the human in the loop and to identify how the human tele-operation task is formally represented and re-created in the DT/VR platform. More than likely this will be represented for the user at the local end. **UC3 status update:** This aspect was investigated during the proof-of-concept research work conducted.
- **Req- D5.2-U3-5-hw:** Requirement to identify if the DT/VR world representation alone is sufficient in order for a user to carry out their task/process or if there is also a requirement to have live video stream over a separate channel such that the user can see a video of the robot's performance as the task/process is carried out by the tele-operator (local user). **UC3 status update:** Until there is sophisticated VR the requirement for a live video feed will be mandatory.

Safety

- **Req-D5.2-U3-1-hw-sw-com:** In built safety functions are required at the user end in order to handle emergency situations involving the robot at the remote end. **UC3 status update:** The UC3 research on gesture control can be of benefit here.
- **Req-D5.2-U3-2-hw-sw-com:** Requirement for the evaluation and testing of safety and health in the context of the user's immersive experience when using a VR headset in conjunction with the DT/VR world. **UC3 status update:** This will be relevant as VR interfaces advance in the future.

As a summary, the research partners for UC3 have highlighted a selection of key requirements/services for any future DT/VR/AR platform:

- AI dataset generation capabilities.

- User training for increasingly complex robotic tele-operation tasks.
- Representing the remote world back to the user for high-precision tasks.
- Creating a real-time VR world for the user to work in via a headset display.
- Creating AR features and services to assist users with ever-complex robotic manipulation tasks.
- Testing and evolving the AI capabilities within the tele-operations platform.
- Updating, testing and maintenance of ML models deployed on the platform and synthetic dataset generation.
- Preventative maintenance and condition monitoring of the remote tele-operation components.
- Development of future aggregations of DTs for multi-interactions in remote tele-operations.

6.8.5 Capabilities & Limitations (including USP, strengths & weaknesses)

The following discusses limitations/challenges that lie ahead for the development of a true digital twin platform for remote tactile robotic tele-operations.

Remote Robot/Machine DT Implementation: One of the main issues is to research how the real robot can be fully implemented as a true DT. This will require communications advancements in latency minimisation with reference to sensors, edge devices and local to remote communications. The intrinsic nature of distance communications in teleoperations is an extreme challenge and innovative techniques are required in order to create a true DT in the case of UC3 and other similar approaches.

Remote Scene Representation: Another major limitation relates to how the real-world remote tele-operations scene can be formally represented in the DT/VR world. With this in mind, is there a need to re-create the whole world scene at the remote end or should this only be the part of the real-world that directly relates to the actual task the user is driving from the local end to the remote robot end. To build up and maintain a real-time image representation of the real-world in the DT/VR is exceptionally challenging and even outside the latency aspects, currently the technologies do not exist to easily construct and maintain such a representation. Tools to assist real-world re-creation in DT/VR worlds are required and will need to be investigated as part of any future research and development efforts in relation to UC3, outside the scope of the IMOCO4.E project.

If the remote world is not to be represented in the DT/VR, then at a minimum the true-life objects under manipulation by the remote robot infrastructure should be represented as virtual objects within the DT/VR world. Such object representation is not as complex as whole world scene re-creation and could be engineered using advances in various sensor technologies and algorithms, while also directly addressing latency demands to represent the remote world state in near real-time, back at the user local end.

6.8.6 Customizations & Adaptations

DT/VR & Datasets: With reference to synthetic dataset generation, a DT can be used to capture live data streams from the remote end. This will involve additional computation on the remote end and then a communications overhead of sending the actual robot movement/activations back to the DT. An alternative approach is populating the local DT directly from the sensor feeds to create a dataset, but such an approach may not accurately reflect the actual robot activations at the remote end. Regardless of the approach taken the resulting DT/VR platform should offer major opportunities for task learning, AI and ML developments.

AI Capabilities: The DT/VR platform will also enable the research and testing of advanced AI features before they are formally deployed to the tele-operation platform. Opportunities for the platform include latency reduction, prediction capabilities, AI reinforcement learning potential, and AI based assistance as the user conducts a series of pre-defined tele-operations tasks.

- System health: Condition monitoring and predictive maintenance of tele-operation infrastructure.
- Creation of virtual sensors that may not be physically possible in the real world.
- Deployment and testing of simulated sensors and system simulations.
- Re-training and optimisation of AI models.
- Synthetic and real-world dataset generation.
- User training and evaluation before live activity.

In terms of customizations, adaptations, and with reference to UC3, there is clearly a significant number of future technological advancements required in DT/VR/AR toolchains. This is in order to engineer highly sophisticated virtual worlds such that the true global potential of the Internet of Robotic Things (IoRT, 2024) and the Internet of Skills (IoS, 2024) can be fully provisioned and realized over the coming years.

Note: To learn more about the IoS and the IoRT the reader is referred to WP8 Task 8.3 IMOCO4.E education deliverables.

6.8.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

Conclusions and lessons learnt:

- The complexity of developing a true DT for robotic teleoperations is a complex process and is a dedicated project in itself.
- DT development tools are evolving and are expected to become more sophisticated and easier to use for the development of teleoperations platforms.
- For the foreseeable future latency and real-time visualization will remain as the major challenges for true DT in robotic teleoperations.

As a final review, this section "*Virtual reality digital twins, enabling human-computer interaction"* started with a high-level overview of the tele-operations platform under development in UC3. Next, the overall concept of the DT/VR was presented and positioned in the context of the local and remote ends of the UC3 platform. The section on the UC3 HW presented an insight into the sensor and communications infrastructure of the local end where typically a DT/VR platform should be provisioned. This was then followed by a more detailed SW focused discussion which covered the "digital shadow" model and explanations of the practical proof of concept work conducted in relation to the UC3 DT/VR. Finally, the section concluded with thoughts from the UC3 partners on limitations/challenges, customizations, adaptations, and futures for DT/VR/AR in relation to two key future Internet evolutions, in the form of the IoRT and the IoS.

6.9 Comparative User Experience test of driving behaviour in encounters of mobile robots with human traffic participants in a virtual and a real environment [STILL, NURO, DTT]

6.9.1 Introduction [STILL]

The use of Automated Guided Vehicles (AGVs) in industrial contexts and warehouses has strongly increased in recent years. In these contexts, AGVs usually are in operation in mixed traffic with pedestrians and manually driven trucks. Recently AGVs are equipped with increasing autonomous functionalities and shall in the following text be referred to as Autonomous Mobile Robots (AMR). With the increase in autonomous functions, the AMR's behaviour becomes less predictable for the people on the work floor.

This makes it important to evaluate their effect on the humans encountering them and draw design implications from the interaction between AMRs and people on the work floor. Evaluation of the User Experience during encounters between humans and AGV (Howey & van der Anker, 2023) has shown that both explicit visual and auditory signals and implicit information mainly via driving behaviour of a vehicle is very important to foster trust into AGVs and their acceptance by people working next to them. Prior work has dealt with external signals and lead to a VDMA recommendation about "Design of visible and audible signals of driverless industrial trucks" (VDMA, 2023). Within IMOCO4.E, the use of implicit communication via movement behaviour should be evaluated.

User experience testing of movement behaviour of AMRs using real robots is very elaborate. AMRs must be set up as well as the appropriate environment, for example, a part of a warehouse. Furthermore, all potential risks of encounters of people with possibly large and heavy AMRs must be eliminated. Testing in a simulated world would significantly reduce effort and would eliminate the risk of collisions of AMRs with the test participants (Wiczorek & Protzak, 2022). The question if findings of User Experience (UX) testing within a simulation using VR/AR-technologies can be transferred to real world setups is currently investigated in numerous research projects, as, for example, in the automotive industry and for delivery robots (Maruhn, 2021). (Schneider & Bengler, 2020) states that a general trend can be transferred for many research questions, a quantitative transfer of outcomes from VR to reality must be considered cautiously because of moderate correlations in many studies and the still limited number of studies aiming explicitly towards the evaluation of comparability. Because of this and since the conditions in the setup in material handling domain are not comparable to others (size of robot, speed, environment), a study was conducted within IMOCO4.E for AMRs in material handling.

6.9.2 Technical setup overview [STILL]

The technical setup is divided into two streams, one dealing with the VR-part, one dealing with the real truck encounter (see Figure 102).

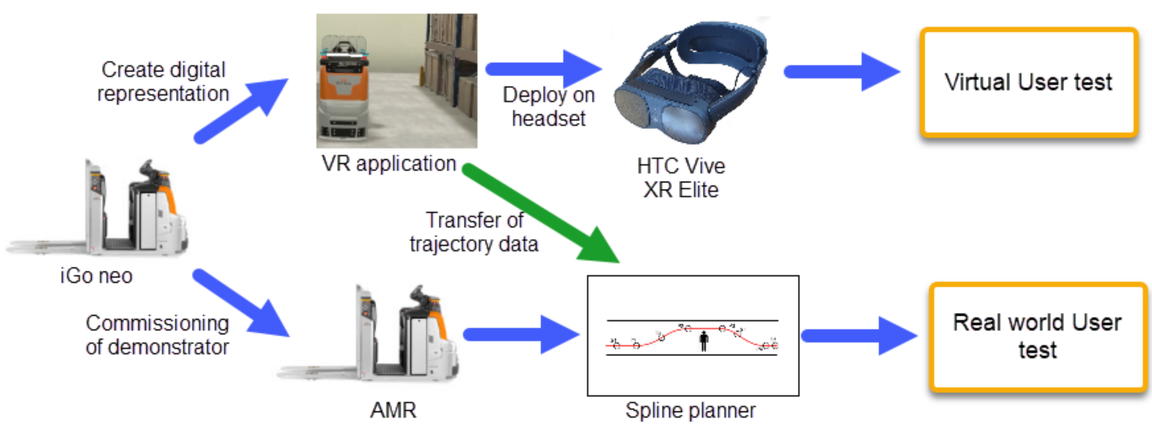


Figure 102: Technical setup

Starting with the series product iGo neo OPX, the test was prepared in the two streams. In the VR stream a digital representation of the truck was generated and the VR-application (SW-098) was developed by Nuromedia. Two possible headsets (HW-044 and HW-045) were evaluated, the HTC Vive XR Elite was chosen (HW-045). The VR application is running on a PC and the headset was connected to it via Wi-Fi 6.

The VR-application generates a set of points for each trajectory to be tested which can then be deployed onto the real truck.

In the real-world stream, the functionality of the series truck needed to be enhanced (Demonstrator 3 by Working Package 7, HW-026). To be able to execute the planned tests the truck had to be enabled to follow exactly the same trajectory as presented in VR. To realize this a new spline had to be developed since all existing ROS planners only use autonomously generated trajectories. This task was taken over by DTT.

6.9.3 Test Concept

A typical simple traffic situation was set up in a VR application, to conduct UX experiments with test participants and let them rate the driving behaviour of the autonomous truck. In a second step the same driving behaviour could be experienced by the same test participants with a real truck (demonstrator 3). The ratings of the driving behaviour could then be compared. In the experiment, the AMR is driving around a stationary test participant (see Figure 103). Points P1 through P4 define the trajectory, points S1 and S2 define speed-changes from normal travelling speed to a reduced speed during the encounter.

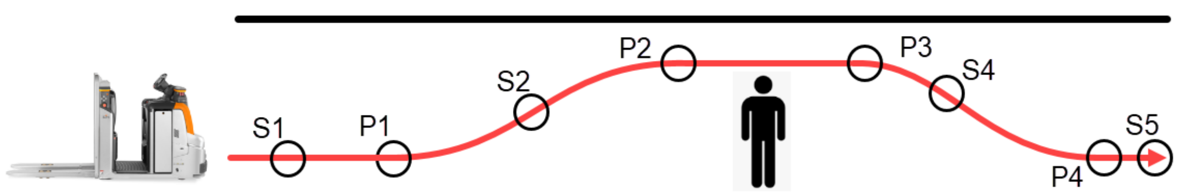


Figure 103: Schematic of the traffic maneuver

6.9.4 VR-application for virtual test run [NURO]

STILL provided a model of the iGo Neo (demonstrator 3), which was used to build up a VR simulation of the test using Unity. Unity is a simulation and gaming platform which is fully scriptable and thus is perfect for the task. The vehicle model was shaded by NUROs designers to enhance the immersive experience. To create a realistic setting, the test hall was built up and racks were added.

In order to closely simulate the behaviour of the iGo neo, STILL measured the acceleration and deceleration curves of the vehicle, and these were implemented in the digital twin. To further improve the immersive experience of the user sound was added to the simulation. STILL recorded different sound assets which can be heard when the iGo neo is steering, starts and comes to a halt. These sounds are processed by NURO's sound module, which receives only speed and wheel angle to intelligently determine when to play each sound, possibly in parallel. NURO also added background noise to the simulation.

In order to give the test participants, the possibility to get habituated to the VR-environment a training environment was created, where skills needed for the experiment can be trained. For the planned test scenario, a driving simulator was developed which is able to follow the three versions of the spline path described above including deceleration and acceleration (see Figure 104). A user-specific path can be set up by the user by shifting the 4 trucks representing the control points of the trajectory. During the evaluation runs, the users can rate their discomfort by pressing the trigger of the controller. The level of discomfort is visualized by a changing bar at the lower end of the field of view.

After the virtual evaluation runs, the driving path created by the users according to their individual preferences, can now be passed on to the robots operating system ROS. ROS, in turn, uses this data to drive the real physical iGoNeo to repeat this user-specific trajectory. In this way, users can compare the virtual experience with the actual experience of an iGoNeo passing them by.

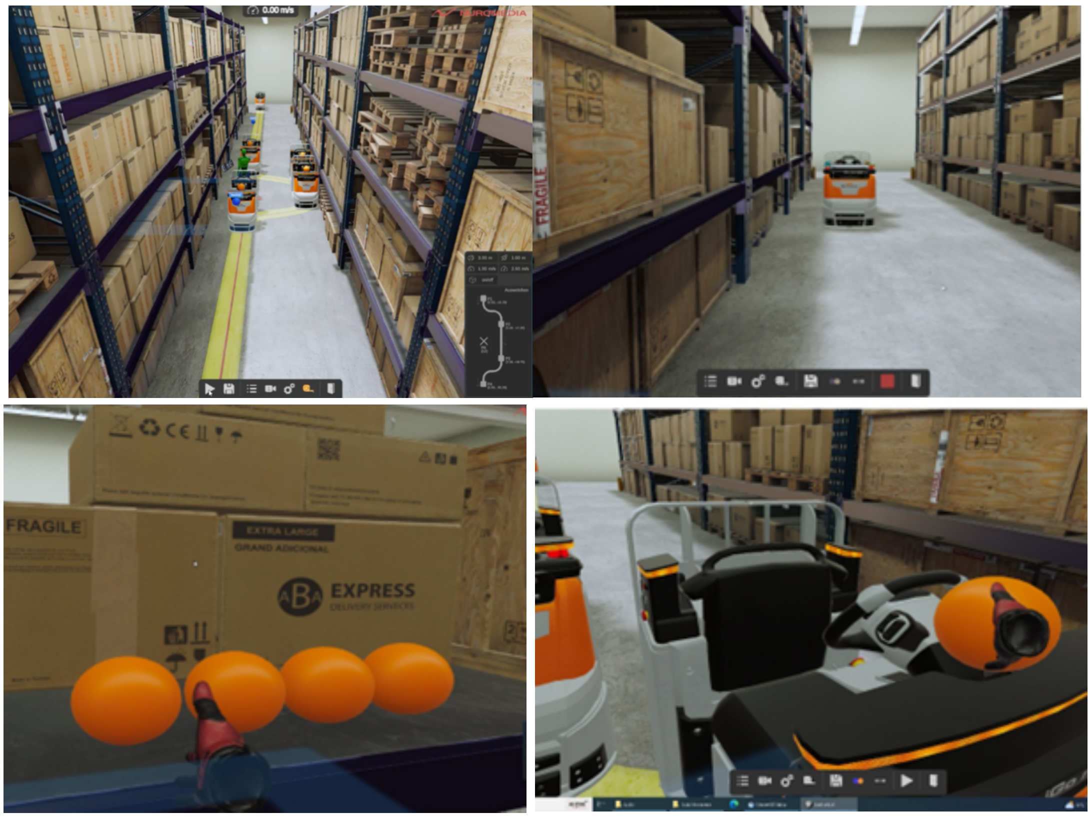


Figure 104: Screenshots of VR application

6.9.5 Selection of headset and deployment of VR application [STILL]

The main requirement regarding the headset was that the test participant should be able to move freely in a rather large area (minimum 10m x 5m). Additionally, the headset should be comfortable to wear and be capable of creating a highly immersive experience. Since the test conductor needed to be able to interact with the VR-application to guide the test participant through the course of the experiment the application had to run on a stationary PC and the video had to be streamed to the headset. For the above reasons (movement area and comfort) a wireless solution was required. For the second application planned (see 6.3.3) the headset should also be capable of working in a stand-alone setting without a stationary PC rendering the images on the device. A secondary requirement was the availability of a professional software version that did not require signing up and connecting with an external service running on a server of the headset provider. The optimal headset at the time of purchase was the HTC Vive XR elite. It does not use base stations for tracking (inside-out tracking), it is capable of running applications stand-alone and it comes with high-performance video streaming using Wi-Fi 6. Ensuring a video streaming with low latency occurred to be a major problem, the solution was a high-performance Wi-Fi 6 router (ASUS RT-AX86U Pro) connected to the PC via LAN-cable.

6.9.6 Vehicle integration for real-world test run [STILL]

The demonstrator 3 is based on the STILL iGo Neo, a warehouse truck with robotic capabilities. In contrast to a classical AGV, the iGo Neo is capable of sensing its environment and reacting to it to support commissioning processes in the warehouse. Within these processes, the warehouse worker performing the picking-operation for commissioning the pallet (called "picker") is supported by the iGo neo with an autonomous drive-by functionality. For this functionality, the iGo Neo senses to position of the picker and operates the drivetrain autonomously. For more information on this vehicle concept, see OPX-L 12 iGo neo | STILL Germany.

As is common for industrial trucks, the demonstrator 3 based on the STILL iGo Neo is built up on a layered control system architecture sketched in Figure 105. Main robotic capabilities developed within the IMOCO4.E context are located in the Robotic subsystem, using an IPC as its main control unit. The system controlling the HMI-components is located in the Vision- & AI-subsystem, which uses a Nvidia Jetson NX as the main control unit. The IPC as well as the Nvidia Jetson NX are running on ROS2 using Ubuntu Linux as OS.

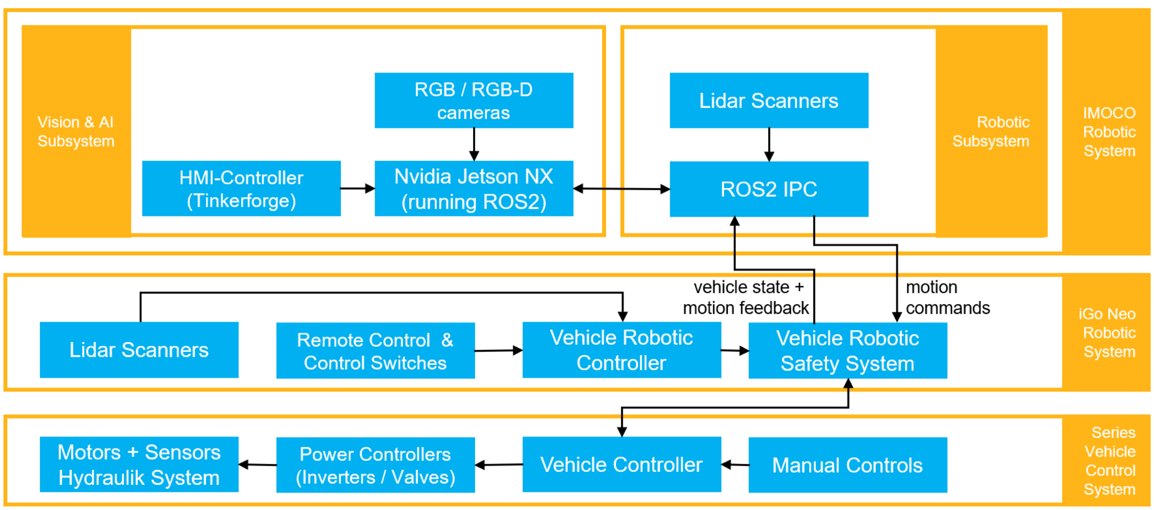


Figure 105: Control architecture of demonstrator 3

The navigation system of the IMOCO-ROS2-system is based on the ROS2-navigation stack Navigation2. For more details about this stack, see Nav2 — Nav2 1.0.0 documentation (ros.org). Quite often in research and autonomous robotics, robots are smaller and show a more centric kinematic shape, like the turtlebot3, which has a nearly circular footprint. This enables the robots to do turn operations nearly on spot without consuming additional space for these operations.

In contrast to, the iGo Neo has a tricycle kinematic structure and rotates around the rear axis, which makes the vehicle much more space-demanding when doing turn operations. In addition, the vehicle is performing a two-directional drive with different drive speeds and task dependence. Whereas the sensor field-of-views make the "driver-direction" to the preferred drive direction for long-distance drive task, docking operations into pallets have to be in "fork-direction" which is the rear drive direction. Consequently, the ROS2-navigation-stack offers a quite good base but must be extended to be able to operate the vehicle. Figure 106 sketches the extensions made to the ROS2-navigation-stack for demonstrator 3.

We extended the navigation-stack by a vehicle specific behaviour tree and designed vehicle specific planner and controller extensions to take the untypical vehicle shape into account. These extensions are capable of including planner and controller plugins, so that the reuse of the planners and controllers offered by the plugin-architecture of Navigation2 is possible.

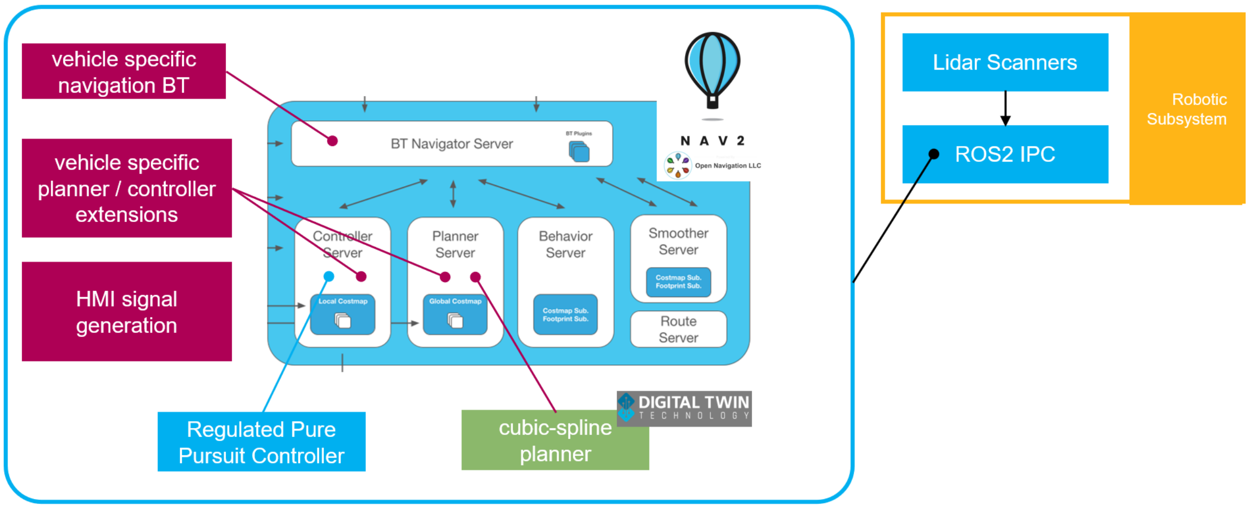


Figure 106: Extension of ROS2-navigation stack by vehicle and test specific components

For performing the tests, we use a cubic-spline-planner developed by DTT in combination with the Navigation2-RPP-plugin (RPP = Regulated Pure Pursuit) with a vehicle-adapted calibration parameter set.

During the user tests the demonstrator truck should display optical and acoustical signals according to a new harmonized external HMI (eHMI) concept (VDMA, 2023). Since the base truck was not capable of displaying multicolour visual and extended acoustical signals a prototypical setup was implemented. Four RGB-LED strips and a broadband speaker were added to the truck (see Figure 107). To address these new components a small microcontroller system was added (HMI-controller) which can receive the demand for a certain signal (e.g., driving forward or blinking left) via CAN-bus from the IMOCO4.E Robotic subsystem of the truck and then create the necessary commands to display the corresponding signals. The HMI components need to be articulated in coordination with the vehicle movement, which is done by the "HMI signal generation". This module is running on the ROS2-IPC (which is part of the IMOCO4.E Robotic subsystem) and uses ROS2-service calls to articulate the HMI components. These service calls are handled by a ROS2-node running on the Nvidia Jetson NX, which controls communication with the HMI-controller using CAN bus.

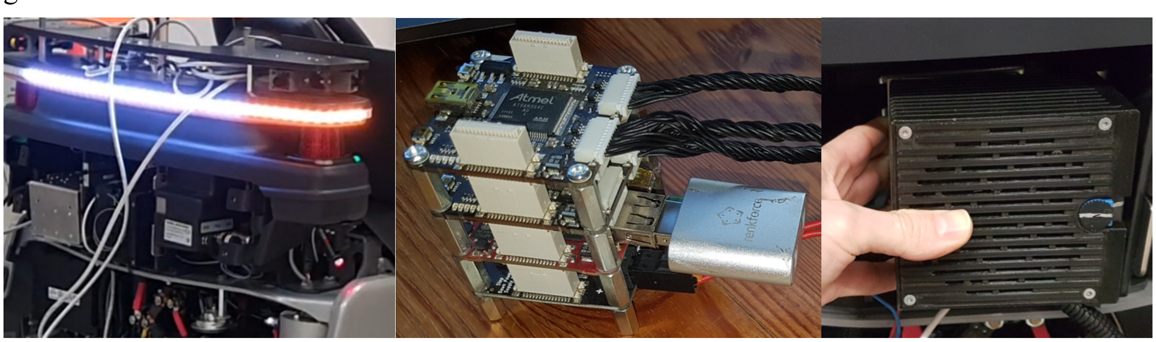


Figure 107: LED strip, microcontroller, Speaker for eHMI setup

6.9.7 Spline planner for real truck to generate given path from VR-setup for mobile robot [DTT]

To describe a path along which the iGo Neo should drive, a spline function is used for the STILL's iGo Neo, a warehouse truck with robotic capabilities. In contrast to a classical AGV, the iGo Neo is capable of sensing its environment and reacting to it to support commissioning processes in the warehouse. In this particular case, the spline describes an evasion pattern that can be adjusted but the basic pattern is always the same. The spline here is always defined by 6 control points; these 6 points are generated in the Unity simulator. Those points are as follows - the start position, the point where the evasion begins (changes lanes), the starting point on the second lane, the endpoint on the second lane, the point where the initial lane is resumed, and the endpoint of the simulation. Also, speed hints can be placed over the spline which tells the iGo Neo to slow down before the evasion curve and when to speed up again.

This data is generated by the VR application described above and then stored in files, which are subsequently supplied to the ROS2 system in CSV format. ROS2, by default, lacks the capability to interpret a spline. To align with ROS2 compatibility, DTT has developed a spline path generator. This generator reconstructs the spline path based on the user-defined data file, ensuring seamless integration with the ROS environment.

The spline path generator is a ROS C++ package to generate 2D c2-splines with ROS2-Nav2 eco-system. The splines are generated by reading the user-defined CSV files. Table 6 shows a sample CSV file snapshot of the 6 control points.

Table 6. Sample CSV file snapshot of the 6 control points from Unity Simulator

5	spline points								
6	6								
7	TangentIn.x	TangentIn.y	TangentIn.z	TangentOut.x	TangentOut.y	TangentOut.z	Position.x	Position.y	Position.z
8	-1	0	0	1	0	0	0	0	0
9	-1	0	0	1	0	0	15.84379196	0	0
10	-1	0	0	1	0	0	17.95703506	1.070687175	0
11	-1	0	0	1	0	0	19.12044525	1.070687294	0
12	-1	0	0	1	0	0	21.21334648	0	0
13	-1	0	0	1	0	0	30	0	0

The file, mentioned above, provides 6 control points, which are provided as input to the package to generate the spline. Each point provides the following information in 2D space:

- The position of the point in *X* and *Y* coordinate.
- The Inward and Outward tangents of the point in *X* and *Y* coordinates.

The package then generates a spline with N number of points which matches the standard of the ROS2-Nav2 global path planning plugin. Each point consists of the position of the point in X and Y coordinates and its yaw angle. The number of points can be adjusted by the user to generate less dense and more dense splines. Finally, before passing the spline to the Nav2 package, a smoothing logic is applied to the points to generate smoother spline curves to match the robot dynamics.

The package is tested with STILL ROS2 workspace. Figure 108 shows the STILL ROS2 environment. The developed package is seamlessly integrated into the NAV2 stack without any need for modification of the workspace. Figure 109 shows the generated spline path by the developed package in the STILL ROS2 environment.

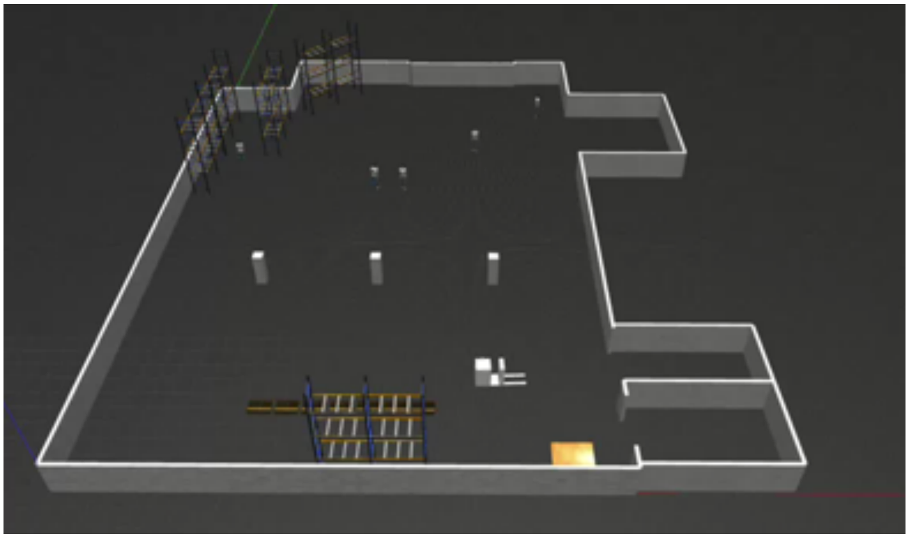


Figure 108: The STILL ROS2 environment in Gazebo

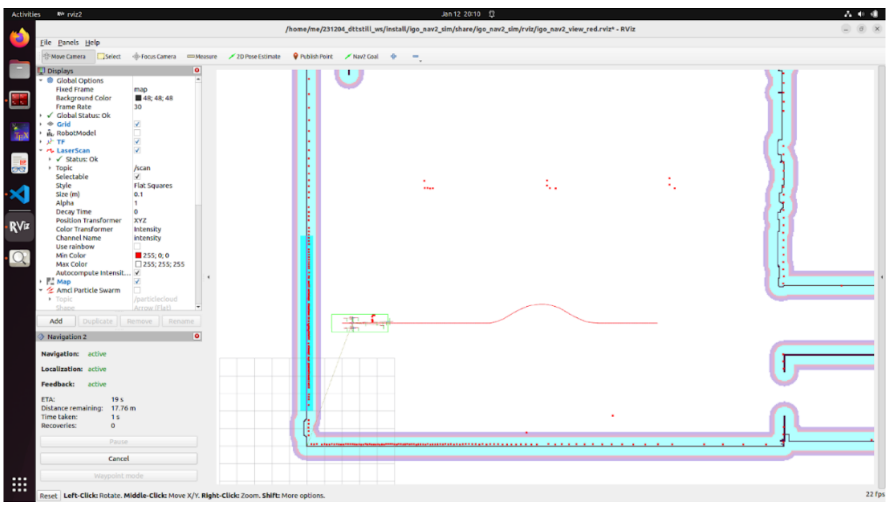


Figure 109: The re-generated spline path (in line in red colour) by the developed package from the user-defined CSV file

6.9.8 Procedure of UX-test [Still]

The main objective of this task is to find out to what extent the findings of the user experience test in a virtual setup can be transferred to a real-world encounter of participants with AMRs. Therefore, the test participants should be enabled to experience the same encounter situation as similar as possible in VR and the real world. The secondary aim is to generate first findings about how a good behaviour should look like, e.g., what the lateral distance should be while driving past a participant and how far ahead of a participant the evasion should start.

In the experiment, the AMR is driving around a stationary test participant (see Figure 103).

- Point P1 is the turn-in point from which the evasive manoeuvre is started. It is always located on the straight line originally driven by the AMR (x-direction). However, it can be placed closer to the participant (swerving starts later) or further away from the passer-by (swerving starts earlier).

- Points P2 and P3 represent the parallel lane to be swerved to. They can be moved laterally to the passer-by (y-direction) to increase the distance of the lane to the participant and longitudinal (x-direction) to vary the point when the parallel lane is reached and left again.
- Point P4 is again on the X-axis of the original lane and can also be moved closer in the X-direction (dodge ends earlier) or further away (dodge ends later).

The points are interpolated, so that a smooth curve is created (Bezier curve). To indicate the upcoming lane change the AMR will give a signal by a blinking orange light at the side where it will start the evasion.

At the start of the manoeuvre the vehicle will drive at its maximum speed (2 m/s). For the evasion the speed will be reduced to 1 m/s. The acceleration and deceleration behaviours simulate the real vehicles behaviour when the setpoint for speed is changed:

- At Point S1 the reduction of speed to the evasion speed will be finished.
- At Point S5 the acceleration to the initial speed will be started.

Three predefined trajectories were generated: in the aggressive version the distances to the participant were as small as possible, in the calm version they were very generous, and the medium version was in between. Additionally, to the predefined trajectories the test participants were able to define their own version in a way that they just would feel comfortable enough. The self-defined version could not be executed by the real truck because quite quick turning manoeuvres would cause a safety function limiting lateral speed to be triggered causing the truck to abruptly slow down. Since this behaviour would feel awkward to the test participants it was decided to only use this version in VR.

The experiment was executed in the following procedure:

1. Habituation: Short introduction to VR to give the test participants the chance to get used to the virtual environment.
2. One predefined trajectory (calm) was presented to the test participants as a base line for their own optimal version.
3. Test participants optimize the trajectory according to their needs with the instruction to make it just comfortable enough.
4. The three different predefined and self-defined trajectories were conducted and rated continuously by the test participants. Half of the test participants experienced the real-world version first, the others the VR-version (see Figure 110 and Figure 111). The sequence of the trajectory versions was permuted to reduce effects of habituation. The test participants were not told which trajectory was their self-defined one, this version could only be executed in VR. After each trajectory the overall experience was rated on a 5-point-likert scale regarding the perceived safety between “much too safe” and “much too unsafe”.
5. Intermediate interviews.
6. The second set of test-runs was executed.
7. Intermediate interview.
8. Direct comparison: The aggressive version was experienced twice directly following in VR- and real-world (sequence permuted).
9. Final interviews including questions about the comparison of VR- and real-world.

Ten test participants were recruited from the Still forklift truck production line and the internal transport division. They were chosen to vary in age, gender and experience with driverless industrial trucks (see

Figure 112). Since mostly male personnel work in this field at the company only one female test participant could be found. The tests were performed on the test premises of Still in Hamburg.

Figure 110: Test execution in VR

Figure 111: Test execution with real truck

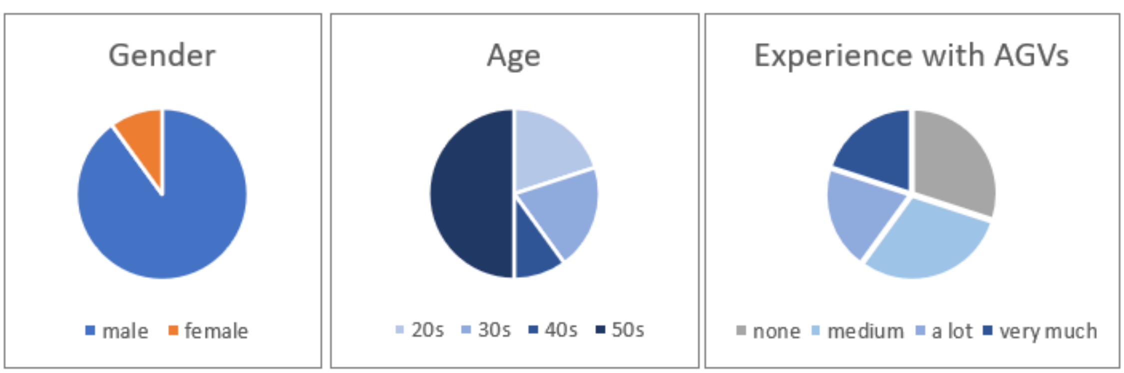


Figure 112: User test sample

6.9.9 Transferability of VR experiences to real-world

To measure the perceived safety the test participants were pressing a button on the VR-controller continuously during each test run. They could press the button only slightly when feeling a little unsafe up to all the way when feeling most unsafe. After each test-run they were asked to rate the overall experience with a 5-point Likert scale ranging from much too unsafe (Lee et al., 2020) to much safer than necessary (Maktoubian et al., 2021).

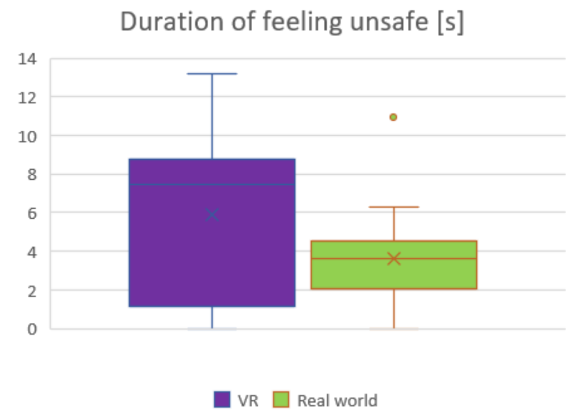


Figure 113: Continuous threat level rating of the 10 test participants

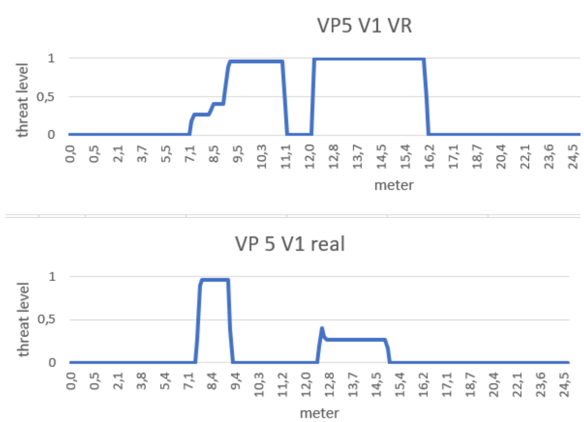


Figure 114: Example of continuous threat level rating (test participant 5)

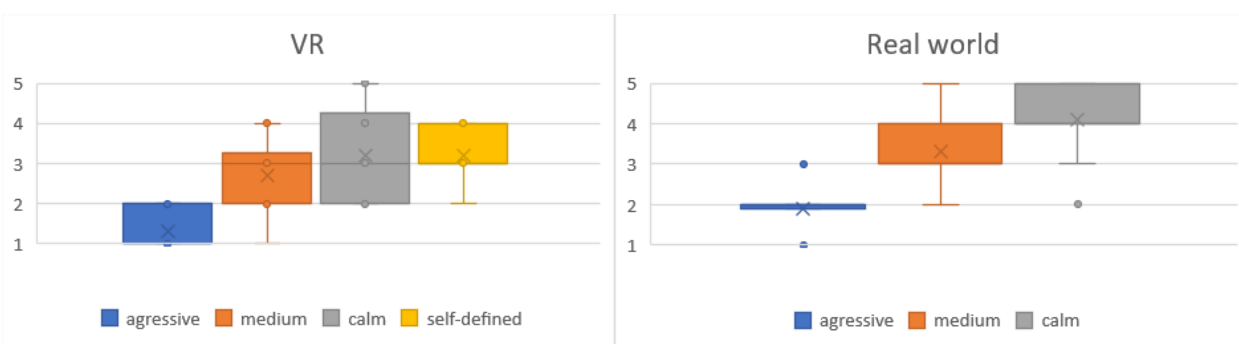


Figure 115: Discrete threat level rating

To relate the ratings between VR- and real-world experience the duration of the button being pressed was compared (Figure 113). Since most test participants only pressed the button during the aggressive version only this version can be compared. The duration that the test participants pressed the button varied widely across the test participants, in average the button was pressed 5,9 s in VR and 3,6 s in the real world, thus 2,3 s shorter. On average the button was pressed as also the maximum amount was similar on average. Figure 114 also shows one example of a diagram of threat-level rating over travel distance of the AMR. The evasion starts at 9,5 m and ends at 16 m, the test participant is located at 12,5 m. The second source of data used for this relation was the Likert scale rating. Figure 115 shows these results. The average rating over all test participants and the three predefined trajectories were 0,7 points lower in the VR-world experiments thus giving a tendency that test participants feel safer in the real-world experiments. This result is consistent with the continuous rating with the button and also with subjective opinions stated in the interviews where 7 test participants stated that the results would be well transferable, one said there would be some differences and 2 test participants stated that comparison would be difficult.

6.9.10 Preferred behaviour of AMR during encounter

To extract preferred characteristics of the encounter the ratings of the different trajectory versions were compared, the self-defined trajectories were analysed, and interviews were conducted.

In the real-world experiments, the medium version was generally preferred by most test participants being rated just fine on average (3,3 points). In the interviews, this finding could also be verified.

The main source of discomfort was the lateral distance during the evasion. In the self-defined trajectories during the VR-experiments this value was mostly changed. The average distance defined was 1,3 m (see Figure 116). Since the test participants stated distances would seem closer in VR compared to real-world and given the ratings of the real-world medium version a distance of 1,1 m from the centre of the participant to the edge of the AMR seems appropriate. One test participant demanded a much larger distance of 2,1 m.

The second characteristic often mentioned in the interviews was the distance to the participant when the manoeuvre would start. In the self-defined trajectory, an average of 2,3 m was defined (see Figure 116). Given again the difference in perception of distances, the good ratings of the medium real-world version and the comments in the interviews the distance of 1,8 m seems appropriate in this case. This result is only valid for a standing test participant, an AMR that activates the blinker about 2 s prior to the start of evasion and finishes the deceleration process before starting to evade.

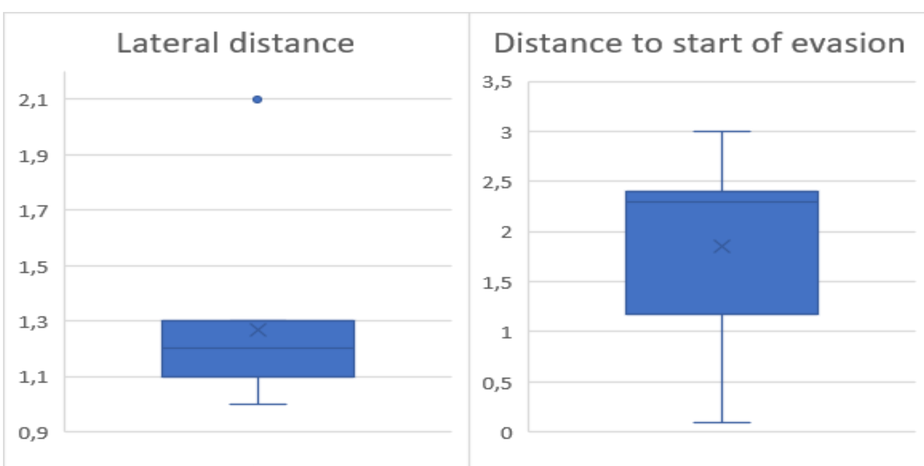


Figure 116: Self-defined trajectory characteristics

Test participants were also asked about the speed of the AMR and whether the deceleration timing was fitting their expectations. The vehicle speed was 1,6 m/s before the manoeuvre, and it decelerated to 0,8 m/s just before starting the evasion. This behaviour was generally rated as appropriate. Two test participants had the wish of a slightly earlier deceleration. In the interviews it was stated that the deceleration and the blinker signal would be a very important hint that the AMR has perceived the participant in its way.

6.9.11 Evaluation of the virtual experience

To evaluate the immersion of the participants in the virtual experiment, the iGroup presence questionnaire (IPQ) was conducted (Schwind et al., 2019). The questionnaire was executed with a 5-point Likert scale, but since it is common to use a 7-point scale values have been adjusted accordingly to span from 0 to 6. Spatial presence (SP): 5,55; Involvement (INV): 5,06; Experienced realism (REAL): 2,01; General (G): 5,58. Maruhn (2022) used the IPQ to evaluate the effect of an avatar on immersion. The ratings on average over the setups are: SP: 4,9; INV: 3,2; REAL: 3,1; G:4,7. A comparison to these results shows high values in all of the subscales, apart from Experienced realism, which is due to the difference in image quality. In the experiment of (Maruhn, 2022) photorealistic environments were used. Since values in spatial presence and involvement are quite high and it was stated in the interviews that the test participants felt very much involved and present in the VR environment the quality of the given VR application can be seen as sufficient.

6.9.12 IMOCO4.E requirements

Objective ST1: The main goal for the digital twin application described in 6.9. is a preparation for the application described in 6.10. The overall aim is to be able to design and optimize algorithms dealing with encounters between autonomous mobile robots (AMR) and humans. For these tasks an easy and reliable evaluation method is needed. Both applications create digital twin applications for User Experience (UX) Testing of AMR with human test participants. The first application described in this chapter is used to evaluate the reliability of UX-testing using the VR-applications implemented within Task 5.6. It could be shown that the VR-application is capable of providing a suitable methodology for UX-testing of AMR-driving behaviour.

Objective ST4: The VR application contributes to ST4 as a first step to implement a "Virtual/augmented reality for real-time HMIs" by creating a VR-environment as a preparation for the real-time VR-application described in chapter 6.10.

6.9.13 Capabilities & Limitations (including USP, strengths & weaknesses)

The VR-application developed by NURO and used in the UX-experiments can be seen as a very good basis for evaluation of the user experience in encounters of pedestrians with AMRs in material handling contexts. The results of the transferability of VR experiences to real-world (see chapter 6.9.5) show that the results of VR-studies can generally be transferred to real-world applications. Regarding the determined distances and speeds, a slight offset should be considered. Results should be fine-tuned and verified with real-world experiments. Since the VR application used is only capable of showing one specific situation further work is needed to enable testing of realistic scenarios. This work has been started and is described in chapter 6.10.

The replication of the AMR behaviour implemented in the VR-application to real AMR proved to be a big challenge. The main problems were the localization of the AMR, the accuracy of following the given path and the required functional safety. The safety system of the base truck iGo neo used in the experiments limits the lateral speed of the truck to 300 mm/s which resulted in unintentional deceleration in the evasion trajectory. Reducing the speed of the AMR in the experiment and choosing softer curves reduced this effect to an acceptable level for the predefined trajectories. The self-defined trajectories could not be driven by the real truck because we did not want to limit the possible trajectory settings for the test participants too much to enable a wide range of preferred trajectories to be developed by the test participants. Small errors in the localization of the AMR resulted in an offset of the desired distances to the test participant. The complex kinematics of the AMR used resulted in small errors in the accuracy of the path driven. The S-curves from the VR-world were driven a little less smoothly by the real truck. This was noticed by two test participants and described as disturbing.

The results regarding the preferred behaviour of AMR during encounter (see chapter 6.9.10) can only be seen as a starting point for further evaluations. Since the test participants needed to stand at a specific spot and were not allowed to walk the transferability of the finding to real warehouse traffic is limited.

6.9.14 Customizations & Adaptations (including possible modifications and extensions)

The VR headset HTC Vive Pro 2 was extended by a wireless adapter attachment kit to enable wireless operation.

6.9.15 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

The VR headsets evaluated (HW-044 and HW-045) were both providing sufficient video quality. The main limitation proved to be the wireless video streaming. The HTC Vive Pro 2 (HW-044) used a proprietary wireless extension which did not meet the requirements of the experiments at all since the reliability of the streaming was insufficient and even worse the receiving component at the headset side was overheating after about 15 minutes leading to a black screen in the headset. Also, the localization of the headset in a large area of 10x10m using 4 base stations was not working well enough. The HTC Vive XR Elite (HW-045) used Wi-Fi 6 for video streaming, which was still the main challenge but could be solved using a high-performance Wi-Fi router and a short distance to the headset. Inside-out localization without the need for base stations worked very well. In future projects the aim is to eliminate video streaming by using an application running on the VR headset itself to compute the visualization which is also possible with the HTC Vive XR Elite.

6.10 VR-based tool for User Experience evaluation of driving behaviour of mobile robots in encounters with human traffic participants [STILL, NURO, DTT] Overview

6.10.1 Overall concept [Still]

In order to be able to conduct various tests regarding mixed traffic with AMRs and pedestrians a very flexible solution for the integration of a test participant into a robotic simulation would be useful. Testing user experience using a VR environment would be a safe and easy way. Behaviour of AMRs is implemented and tested using simulations extensively. Today the simulations mainly do not provide a photorealistic image and there is no possibility to use VR headsets for visualization. The aim of this second part of our work is to develop an interface between the (a) ROS ecosystem in which robotic simulations take place and (b) the Unity ecosystem which is used widely for VR-visualizations. Both applications shall run independently on their respective devices and shall exchange information about the position and orientation of all assets including the VR headset itself as a representation of the participant wearing it. This will be realized using a ROS messaging service. Using this setup every desired testing environment could be generated in the simulation and a test participant could enter the simulation using the VR-headset. An avatar of the test participant can be implemented inside the robotic simulation to enable the simulated AMR to react to the participant's behaviour. Using this setup, multiple participants inside the simulation should be possible.

6.10.2 Technical setup overview [DTT]

The Ros-Unity bridge is a TCP-based communication protocol developed as part of the Unity-Robotics-Hub project by Unity-Technologies. The interface provides a Bi-directional communication interface between Unity and ROS 1/2. Figure 117 shows the working architecture of the communication interface.

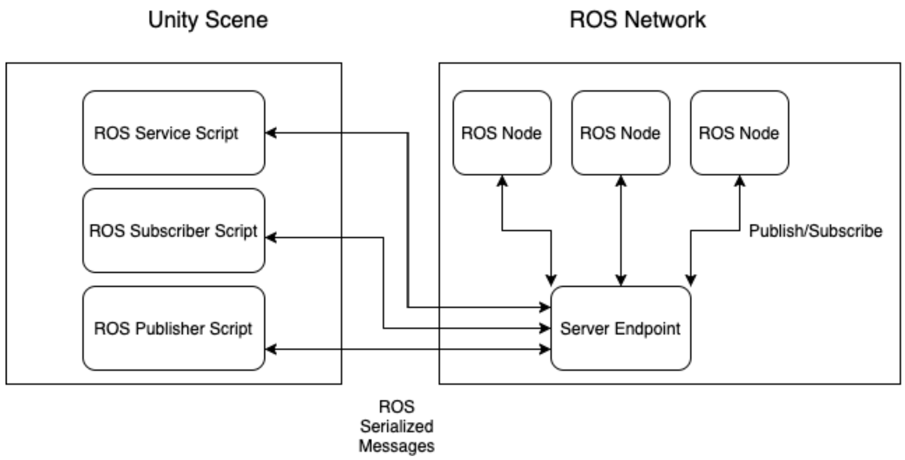


Figure 117: The working architecture and the components of the ROS-Unity bridge.

In the current use case, the ROS-Unity bridge is directly connected between the Unity Simulator and the STILL ROS ecosystem. Figure 118 shows the placement and direction of communication between the Unity Simulator and the STILL ROS ecosystem.

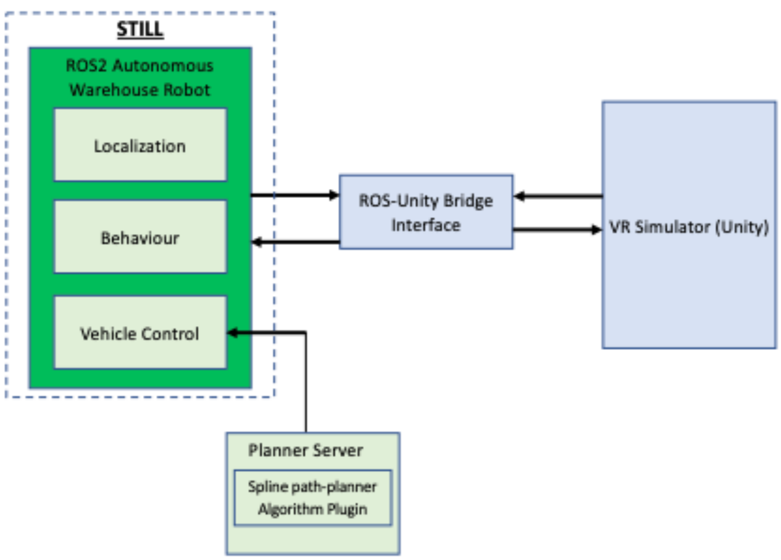


Figure 118: Block diagram showing the placement and bi-directional communication of the ROS-Unity Bridge between the Unity Simulator and the STILL ROS ecosystem.

The VR Simulator (SW-099) that is visualizing the scene of the STILL ROS2 simulation was contributed by NURO and will be deployed on the VR-headset HTC Vive XR Elite (HW-049) (Figure 119)

Figure 119: VR-Headset HTC Vive XR Elite

6.10.3 ROS-Unity bridge interface (DTT)

The data is exchanged in the format of ROS messages. Each message type holds different properties. Figure 120 shows a sample message that helps to send the position of the 3D object in the Unity Simulator space to the STILL ROS ecosystem. Although the sample message, in the figure below, shows only the position information, the message also contains orientation among other properties.

```
float64 posx
float64 posy
float64 posz
```

Figure 120: A sample message that helps to exchange the position of the 3D object in Unity simulator space to the STILL ROS ecosystem.

6.10.4 Unity VR-application for visualization of simulated setup [NURO]

To be able to enter a robotic simulation as a human, we created a VR application to create enhanced visualizations of the simulation, as the first application described in chapter 6.9 is based on Unity.

The visualization app accepts commands from the ROS system. It can be run in VR Mode (wearing a headset) or Screen Mode (on any PC or notebook screen) simultaneously. There are 2 types of commands. The first group is used to create racks, boxes, AMRs and other such objects inside the hall. The second group accepts position and orientation commands to move these objects around in real-time. The app uses a plugin from ROS to process the communication between Unity and ROS. In a later stage, the app will also report the position and rotation of users wearing headsets, so each user can see each other user and the ROS system can treat these users as obstacles in the system.

6.10.5 Results

Since the applications are not fully ready yet only limited tests could be performed. The first test shows that communication between the ROS part and the Unity part is possible and that objects can be spawned in both applications. Intensive testing of the VR application to ensure its usability and the capability of conducting user tests with it will be carried out within the timeframe of IMOCO4.E

6.10.6 IMOCO4.E requirements

The KPI tackled in this task is from the Demonstrator 3 (Autonomous intra-logistic transportation): Autonomous functionalities tested in digital twin (TRL 4). The first step of the implementation of the capability to evaluate the User Experience of autonomous functionalities in digital twin has been described in chapter 6.9 intensively. To enable the developed setup to be able to test the autonomous behaviour in realistic traffic situations the desired setup of chapter 6.10 will be a good contribution.

6.10.7 Capabilities & Limitations (including USP, strengths & weaknesses)

Because of challenges in the implementation of the functionalities of the Demonstrator 3 (iGo neo) needed for the subtask described in chapter 6.9 it was delayed considerably. Work on the ROS-Unity interface (subtask described in chapter 6.10.3) was already started but could not be implemented sufficiently in time to be ready before the due date of this deliverable.

6.10.8 Customizations & Adaptations (including possible modifications and extensions)

Customizations and adaptations for the VR-based tool involve tailoring simulation environments to replicate real-world scenarios encountered by mobile robots, adapting for realistic user interaction, extending support for multi-user collaboration, simulating human behaviour, and considering possible extensions for data analysis and reporting. These enhancements aim to improve the tool's functionality, usability, and versatility, enabling more accurate evaluations of driving behaviour in encounters with human traffic participants and facilitating training, testing, and research in various simulated environments.

Moreover, customizations and adaptations for the ROS-Unity communication involve optimizing data exchange for seamless communication between Unity and ROS ecosystems, enabling real-time interaction and dynamic adjustments within the simulation environment. These modifications extend to supporting multi-user scenarios, simulating user behaviour, and enhancing visualization features for a more immersive experience. By refining message formats, accommodating multiple users, and integrating user input data, the tool becomes more versatile and effective for evaluating driving behaviour of mobile robots in encounters with human traffic participants.

6.10.9 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

Tool Integration: Here, we leverage the ROS-Unity bridge interface to integrate Unity and ROS ecosystems to facilitate seamless communication between the VR simulation and the robotic simulation environment. This integration enables bi-directional data exchange, allowing real-time interaction and dynamic adjustments within the simulated environment.

Tool Limitations: Despite advancements in VR technology and simulation capabilities, limitations may include hardware constraints, such as processing power and graphics rendering capabilities, which may impact the complexity and fidelity of simulations. Additionally, limitations in data exchange protocols or compatibility issues between Unity and ROS ecosystems may pose challenges in achieving seamless integration.

Generic Usability: The tool’s usability can be ensured across different scenarios and user groups by designing intuitive user interfaces, providing clear documentation and tutorials, and offering customizable features to accommodate varying user preferences and requirements. Generic usability considerations aim to enhance accessibility and user satisfaction while minimizing the learning curve for new users.

Lessons Learned: Reflecting on the development process to identify key insights and lessons learned, such as best practices for tool integration, optimization techniques for simulation performance, and strategies for addressing technical challenges and limitations. During the development phase, we've come to understand that while considerable effort was necessary to attain the current level of performance for this tool, there is still scope for enhancements to tackle the remaining challenges. These improvements and enhancements can be performed via future iterations of the tool to enhance functionality, usability, and effectiveness.

7. AI methods for monitoring and predictive maintenance at higher IMOCO4.E layers

7.1 Overview of realized solutions

In this section, we present a summary of the advancements achieved in Task 5.7, focusing on the design and implementation of Artificial Intelligence techniques dedicated to monitoring and predictive maintenance within the higher layers of IMOCO4.E structure. Subsequently, an array of solutions for various tasks related to this specific task is outlined, showcasing the multifaceted approaches undertaken in addressing the objectives set forth.

7.2 Addressed ST objectives and KPIs

Task 5.7 addresses mainly ST4 (Figure 2). It aims to develop robust and AI-based condition monitoring and predictive diagnostics algorithms and AI-base methods for machine predictive maintenance. Task 5.7 contributes with the results mainly to BB6 and marginally to BB8.

KPIs of BB6 were already listed inTable 2 in chapter 3. KPIs of BB8 are listed in following Table 7.

Table 7. KPIs of BB8

KPI #	KPIs / smart functionality	IMOCO4.E target (TRL 5) MS6 / M30
KPI_BB8_1	The current performance of AI- based vision-in-the-loop functionality driven by such edge computing technologies as Embedded GPUs, FPGAs MPUs, ASICs.	Developed AI-based ASICs, accelerators, SW/HW implementations with a focus on parallel and determined execution. Proof- of-concept integration of the components into the use-cases, pilots, and demonstrators.
KPI_BB8_2	The possibility and correctness of applying the reinforcement learning techniques for realizing control algorithms and transferring learned models to physical setups.	Proof-of-concept (TRL4) pick & place system in a simulated environment with the ability of transferring learned operation from virtual environment to the actual physical setup.
KPI_BB8_3	Hardware (ASIC) realization of Deep Learning Techniques and Neural Networks, and the length of achievable real-time control loop.	New topologies and architectures will be explored in the FPGA prototyping medium and tape-outs of ASICs are anticipated throughout the project.

The KPIs in table above are shared with WP3 which contributes with hardware related goals and ANN implementation aspects. The KPI_BB8_1 and KPI_BB8_2 are both addressed in Tasks 3.3 and 3.4 in WP3. Here, in WP5, ANN algorithms and software related goals were targeted by the training, optimization, and implementation of different types on ANNs. Realization of deep learning and neural networks on FPGAs embedded GPUs dealt with in chapter 6.5 contributes to fulfilment of KPI_BB8_3.

In the following we report a detailed description:

- [UNISS] The solution is tailored to address the requirements delineated in WP5, thereby fulfilling the overarching system requirements within the IMOCO4.E framework. In accord with BB6, this solution aligns with the specified objectives of WP5 by providing a robust tool intended to optimize the timely maintenance of mechatronic systems, consequently bolstering their reliability. KPIs for these NN models are gauged in terms of accuracy and Mean Squared Error (MSE) test loss. On the simulated data provided by WEG and SIEMENS (UC1), our most proficient model attained an impressive overall accuracy of 97 % and a test loss of 7×10^{-3}, while on the provided One Year Industrial Component Degradation (OYICD) Dataset for P3 (https://www.kaggle.com/datasets/inIT-OWL/one-year-industrial-component-degradation), our most proficient model attained an impressive test loss of 5.72×10^{-7}.
- [DTT] This solution as part of a Digital Twin (DT) in Pilot 3 specifically caters to the needs outlined in WP5, fulfilling the overall system requirements within the IMOCO4.E framework related to interoperability functionalities. Aligned with BB6 and BB8, this solution meets the specified objectives of WP5 by providing a robust tool designed to enhance overall bottle manufacturing process and performance, thereby contributing to the improvement of factory productivity.
- [GNT-ITML] SW-044 contributes to the achievement of the ST4 project objective by contributing to the achievement of a BB6 KPI_BB6_3 and associated TRL: A software library of data/model/digital twin - based condition indicator calculation for mechatronic systems, algorithms for fault identification, classification, and prediction'. SW-044 contributes to the achievement of the target of this KPI since it extends the existing Predictive Maintenance Toolbox functionality by introducing an LSTM Autoencoder model and applying to a case of component degradation (blade) for anomaly detection and RUL estimation in the context of predictive maintenance, reaching TRL 4 as well as notable performance metrics (100 % for precision, accuracy, recall and F1 in the case of the analysed labelled dataset) without any dependency upon the MATLAB framework, as its development is based on open-source solutions and is integrated with other IMOCO4.E sub-systems.
- [INTRA] ST4: SW-097 constitutes a predictive diagnostics module incorporated in BB6 (contributing to meet KPI_BB6_2 and KPI_BB6_3) and communicates with BB9 on the cloud for retrieving data and for sending back results for visualization. SI1: Through SW-097, edge-to-cloud integration is implemented starting from monitoring the HW components of the Tissector and resulting in meaningful insights on the cloud via BB9. SO1.1: SW-097 is involved in condition monitoring and predictive maintenance of Pilot 1. KPI-Smart functionality 2: SW-097 implements a diagnostic model for motion control systems.

7.3 AI for anomaly detection and cause prediction (UNISS)

In industrial automation, maintaining the smooth operation of critical machinery, such as elevator control systems, is vital. However, maintenance challenges arise due to system complexity. Anomaly detection, traditionally managed by rule-based methods, struggles with real-world anomalies. Artificial intelligence (AI), particularly Neural Networks (NN), offers promising solutions. In this scenario, UNISS presents its solution, developed as part of Use Case 1 Task 5.7, which focuses on detecting anomalies and identifying their root causes within mechatronic systems through the utilization of NN technologies. It is noteworthy that this solution undergoes formal verification throughout its design and training phases, thereby guaranteeing a heightened level of reliability.

7.3.1 Tech Overview

The UNISS AI for Anomaly Detection and Cause Prediction models are designed to identify and recognize anomalies within mechatronic systems by analysing sensor data. While initially developed for UC1, the versatility of these models allows for potential adaptation across other IMOCO4.E deployments. At present, they focus on anomaly detection within elevator control systems, leveraging NN models for this purpose. The development and training of these NN models have been executed using frameworks like PyTorch.

In essence, our solution comprises two distinct NN models, each tailored to fulfil a specific task. Both models are fed with identical sensor data as input. The first model is dedicated to predicting a numeric performance index, enabling the identification of anomalous behaviour within the mechatronic system. Meanwhile, the second model is designed to discern the root causes of such anomalies, such as the aging of pulleys or the unwanted sliding of mechanical components.

Within UC1, the integrated components facilitating this solution are SW-108 and SW-110. Notably, SW-110 plays a crucial role in providing formal verification for the models slated for deployment within SW-108 throughout their design and training phases.

As part of the deployment strategy, the solution is set to be implemented within the Fog environment, ensuring optimal performance and seamless integration within the existing infrastructure.

7.3.2 Implementation aspects

The development process for UNISS AI Sensor Anomaly Detection models is conducted in Python, utilizing frameworks such as PyTorch and pyNeVer. Specifically, pyNeVer is employed for designing, training, and verifying the NNs of interest, with PyTorch serving as the backend for training purposes. To facilitate deployment in the Fog environment, the models are converted into the standard ONNX format and saved as .onnx files.

As previously mentioned, the data utilized for training and testing the models consists of simulated data provided by WEG and SIEMENS. This data is subjected to preprocessing by domain experts at WEG, who extracted 187 pertinent features for each elevator journey between two different floors. Consequently, the resulting dataset comprises a single data point with 187 features for each journey. The target measurements encompass the classification of anomaly types (such as aging, bearing, or sliding) and the computation of the performance index (PI) by WEG.

The performance index serves as an indicator of the degree to which the elevator's behaviour deviates from its nominal state, with higher values indicating greater deviation.

7.3.3 Results

We evaluated three distinct NN architectures for our tasks, maintaining a consistent design for both anomaly classification and performance index (PI) prediction. These models are fully connected NNs with ReLU activation functions, consisting of two hidden layers. The first set of models comprises 32 hidden neurons in the first layer and 16 in the second layer. The second set includes 64 neurons in the first layer and 32 in the second layer, while the third set consists of 128 neurons in the first layer and 64 in the second layer.

It is noteworthy that the networks used for classification and PI prediction differ in their output layers. Classification models have three outputs corresponding to the three classes of interest, while PI prediction models possess a single output representing the predicted PI. We designate regression models as EA [n1-n2], where n1 indicates the number of hidden neurons in the first layer, and n2 denotes the same for the second layer. Similarly, classification models follow a parallel naming convention denoted as EAC [n1-n2].

All models underwent formal verification to ensure their local robustness. For classification models, we assessed their resilience against adversarial examples, that is, inputs subtly perturbed to mislead machine learning models. Regression models were evaluated for the extent of numerical output changes in the presence of input data noise.

All NNs evaluated in our experiments displayed satisfactory precision for their respective tasks. Table 8 presents the Mean Squared Error (MSE) calculated on the test set for the regression models, with even the highest MSE remaining below 1.5×10^{-2}. **Model ID** corresponds to the identifier assigned to specific NNs. **Test Loss** represents the Mean Squared Error (MSE) obtained by each model on the test set. **Epsilon** and **Delta** denote the maximum input perturbation values and their corresponding acceptable output perturbations, respectively. Lastly, **Result** and **Time** indicate the outcome of the verification query and the time (in seconds) required by pyNeVer to resolve the query. "True" indicates that a counterexample was found, rendering the model's robustness uncertainty. In Table 9, we provide overall accuracy on the test set along with specific accuracies for each class. Relative accuracy indicates the proportion of correctly predicted samples within a particular class, addressing class imbalance in the dataset. **Model ID**, **Epsilon**, **Result**, and **Time** are consistent with Table 8. **Accuracy** denotes the percentage of samples classified correctly by each model on the test set. **SLD**, **BEA**, and **AGE** respectively represent the percentage of samples belonging to sliding, bearing, and aging anomalies that are classified correctly by each model on the test set.

Table 8. Overview of our experimental evaluation of Mean Squared Error (MSE)

Model ID	Test Loss	Epsilon	Delta	Result	Time
EA_[32-16]	0.012	0.001	0.002	TRUE	11.56
		0.01	0.02	TRUE	11.52
		0.1	0.2	TRUE	11.55
EA_[64-32]	0.015	0.001	0.002	TRUE	13.81
		0.01	0.02	TRUE	12.88
		0.1	0.2	TRUE	13.86
EA_[128-64]	0.007	0.001	0.002	TRUE	14.79
		0.01	0.02	TRUE	16.30
		0.1	0.2	TRUE	27.09

Interestingly, model complexity does not consistently correlate with performance; even smaller networks exhibited satisfactory performance, underscoring their relevance in industrial applications alongside larger NNs.

Table 9. Overview of accuracy on the test set of our experimental evaluation.

Model ID	Accuracy	SLD	BEA	AGE	Epsilon	Result	Time
EAC_[32-16]	92%	95%	95%	90%	0.001	FALSE	8.21
					0.01	FALSE	8.50
					0.1	TRUE	8.76
EAC_[64-32]	89%	86%	97%	85%	0.001	FALSE	10.01
					0.01	TRUE	10.57
					0.1	TRUE	13.08
EAC_[128-64]	97%	100%	100%	95%	0.001	FALSE	12.85
					0.01	FALSE	12.46
					0.1	TRUE	24.53

Regarding formal verification, Table 8 and Table 9 illustrate that pyNeVer successfully verified properties for all models within a reasonable timeframe. The size of the input space and model complexity appear crucial in determining verification time. Regression models showed less robustness to adversarial perturbations, with pyNeVer unable to ensure their safety due to stringent perturbation thresholds. Conversely, classification models demonstrated greater resilience to adversarial perturbations, with pyNeVer certifying their safety across various magnitudes of perturbations.

7.3.4 IMOCO4.E requirements

The activity of this solution meets the following requirements:

Requirement		L/BB/P-D-UC	Ver. type (I/D/T/A)
ID	Description		
R028-D5.1-B6	Predictive maintenance components should be integrated with relevant monitoring systems to create alerts and recommendations.	BB6	I
R031-D5.1-B6	ML predictive maintenance components should be able to process incoming data and apply trained models in real time.	BB6	I

7.3.5 Capabilities & Limitations (including USP, strengths & weaknesses)

Leveraging advanced NN technologies, our solution ensures robust anomaly detection and root cause analysis, enhancing operational efficiency and minimizing downtime in industrial settings. This enables proactive maintenance and minimizes disruptions to critical machinery.

Furthermore, formal verification throughout the design and training phases instils confidence in the reliability of our solution.

While initially developed for elevator control systems, our solution is adaptable to various industrial environments, offering flexibility and scalability across different deployments. Seamless integration within the Fog environment enhances effectiveness, allowing for efficient data exchange and processing.

Challenges include managing the complexity of NN architectures and ensuring the availability and quality of sensor data. Resource-intensive development and training of NN models may impact scalability and deployment timelines.

Further, scalability of formal verification presents challenges, particularly with large NNs, raising questions about ensuring reliability at scale.

Finally, if the reliability of a model cannot be certified, determining how to make it reliable remains an open problem.

7.3.6 Customizations & Adaptations (including possible modifications and extensions)

AI for Anomaly Detection and Cause Prediction solution presents opportunities for customizations and adaptations tailored to specific industrial needs. These modifications could enhance the solution's effectiveness across diverse mechatronic systems.

Firstly, refining existing NN models or developing new ones could allow for adjustments in architecture and optimization of hyperparameters. This fine-tuning process aims to improve anomaly detection accuracy by capturing a wider range of anomalies.

Augmenting the dataset with real-world sensor data is another avenue for enhancement. Techniques such as noise injection or synthetic data generation could diversify training data, improving model generalization and robustness.

Feature engineering plays a crucial role in enhancing anomaly detection capabilities. Continual refinement of feature extraction methods, including exploring new sensor inputs and adapting feature selection algorithms, could prioritize relevant information for improved anomaly detection accuracy.

Integrating real-time monitoring capabilities could enable proactive anomaly detection and immediate response to emerging issues. By implementing streaming data processing techniques and deploying lightweight inference models, organizations could facilitate rapid detection and intervention, minimizing downtime.

In conclusion, extending the solution beyond elevator control systems would requires domain adaptation. Transfer learning techniques could facilitate knowledge transfer from pre-trained models to new environments, accelerating adaptation efforts. Moreover, edge computing integration is essential for localized anomaly detection and decision-making at the device level. Deploying lightweight NN models optimized for edge devices could enhance scalability and responsiveness while reducing reliance on centralized processing infrastructure.

7.3.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

The development of UNISS AI for Anomaly Detection and Cause Prediction models follows a systematic methodology, incorporating various tools and frameworks to facilitate efficient model development, training, and deployment.

Python serves as the primary programming language for model development, leveraging libraries such as PyTorch and pyNeVer. PyTorch provides a flexible and scalable platform for building and training neural network models, while pyNeVer offers formal verification capabilities critical for ensuring model reliability.

The integration of PyTorch and pyNeVer within the development workflow allows for seamless transitions between model design, training, and verification phases. This integration streamlines the development process and ensures consistency in model design principles and verification procedures.

However, it's important to acknowledge the limitations of these tools. PyTorch, while powerful, may pose challenges in terms of resource requirements and scalability, particularly when dealing with large datasets or complex neural network architectures. Similarly, pyNeVer's formal verification capabilities, while valuable, may encounter scalability issues when applied to larger and more complex neural network models.

Despite these limitations, the generic usability of Python, PyTorch, and pyNeVer makes them accessible to a wide range of users with varying levels of expertise in machine learning and formal verification. The availability of comprehensive documentation, tutorials, and community support further enhances their usability and adoption.

Throughout the development process, several lessons have been learned. First and foremost, the importance of iterative development and testing cannot be overstated. Regular testing and validation of models against real-world data are essential for identifying and addressing potential issues early in the development cycle.

Additionally, collaboration and communication among interdisciplinary teams are critical for success. Effective collaboration between data scientists, domain experts, and software engineers ensures that the developed models meet the specific requirements and constraints of industrial applications.

Lastly, continuous learning and adaptation are fundamental in the rapidly evolving field of artificial intelligence and machine learning. Staying informed about the latest advancements, tools, and methodologies enables teams to leverage cutting-edge techniques and technologies to continually improve the effectiveness and reliability of anomaly detection solutions.

7.4 AI for Sensor Anomaly Detection (UNISS)

In the realm of industrial automation, ensuring the continuous operation of critical machinery is paramount. However, the complexity of these systems poses significant maintenance challenges. While traditional anomaly detection methods, often rule-based, struggle to handle real-world anomalies effectively, artificial intelligence (AI), particularly through the application of autoencoders, emerges as a promising avenue for solutions. Within this context, UNISS introduces its solution, developed as part of Pilot 3 Task 5.7, aimed at identifying abnormal behaviour within industrial systems by leveraging autoencoder technologies. Notably, this solution undergoes rigorous formal verification throughout its design and training phases, ensuring a heightened level of reliability.

Implemented within the Fog environment, the solution seamlessly interfaces with pertinent data streams via Kafka and circulates its findings through the same system. This cohesive strategy fosters integration and synergy within the overarching framework, thereby enhancing the solution's effectiveness in monitoring and addressing anomalies within industrial processes.

7.4.1 Tech Overview

The UNISS AI models for Sensor Anomaly Detection are designed to identify anomalies in the behaviour of industrial systems by analysing sensor data. Initially developed for P3, these models possess adaptability for potential integration into other IMOCO4.E deployments. Currently, their focus lies on detecting anomalies within industrial packaging systems, employing autoencoder models for this purpose. The development and training of these models have been conducted using frameworks such as PyTorch.

Essentially, our solution consists of an autoencoder model that, when provided with input sensor data, endeavours to replicate them using its internal lower-dimensional representation. If appropriately trained, the autoencoder can replicate the sensor data corresponding to the standard behaviour of the system accurately. However, it will exhibit a significant error, termed Vector Reconstruction Error (VRE), when attempting to recreate anomalous data. By monitoring the magnitude of the VRE, we can identify when the behaviour of the packaging system deviates from the norm.

In the context of P3, the integrated components supporting this solution include SW-40, SW-107, and SW-109. Notably, SW-109 plays a key role in offering formal verification for the models slated for deployment within SW-107 throughout their design and training phases. SW-40 provides the data stream upon which SW-107 applies our models.

As part of the deployment strategy, the solution is primed to be implemented within the Fog environment, ensuring optimal performance and seamless integration within the existing infrastructure.

7.4.2 Implementation aspects

The development process for UNISS AI Sensor Anomaly Detection models is carried out using Python, employing frameworks like PyTorch and pyNeVer. Specifically, pyNeVer is utilized for designing, training, and verifying the autoencoders, while PyTorch acts as the backend for training purposes. To facilitate deployment in the Fog environment, the models are converted into the standard ONNX format and saved as .onnx files.

As mentioned earlier, the data used for training and testing the models are sourced from the OYICD Dataset. This data undergoes analysis to ensure that only information corresponding to the standard behaviour of the packaging system is utilized during the training phase of our models. The dataset comprises 1,062,912 samples across 519 sampling sessions, each lasting 8 seconds, spanning a whole year of operations of an OCME Vega shrink-wrapper. Each sample includes 9 different sensor measurements as features.

7.4.3 Results

In our experimental evaluation, we examined four distinct autoencoder architectures, all sharing a similar design consisting of three hidden linear layers, each followed by a ReLU activation function. In particular, the output layer employs an identity activation function. These autoencoders differ in the number of neurons within each hidden layer: 32, 50, 64, and 128 in the first layer; 8, 10, 16, and 32 in the second layer; and 32, 50, 64, and 128 in the third layer, respectively. As is common in autoencoders, there is an expansion of the output space in the first and third hidden layers and a contraction in the second hidden layer. For simplicity, we refer to these autoencoders as A1, A2, A3, and A4, respectively. We did not explore alternative activation functions beyond ReLU, as it remains one of the most widely used activation functions in practical applications.

Table 10. Results of our experimental assessment

Model ID	Architecture	Test Loss	Epsilon	Delta	Result	Time
A1	32, 8, 32	5.72×10^{-07}	0.1	0.2	FALSE	27.339
			0.01	0.02	FALSE	27.743
			0.001	0.002	TRUE	30.798
A2	50, 10, 50	7.54×10^{-07}	0.1	0.2	FALSE	27.858
			0.01	0.02	FALSE	28.405
			0.001	0.002	FALSE	28.559
A3	64, 16, 64	3.60×10^{-06}	0.1	0.2	FALSE	29.309
			0.01	0.02	FALSE	30.411
			0.001	0.002	FALSE	32.403
A4	128, 32, 128	8.05×10^{-06}	0.1	0.2	FALSE	32.339
			0.01	0.02	FALSE	32.882
			0.001	0.002	TRUE	31.897

Table 10 displays the results of our experimental assessment. **Model ID** represent the identifier assigned to a certain autoencoder. **Architecture** represents the number of neurons in each hidden layer.

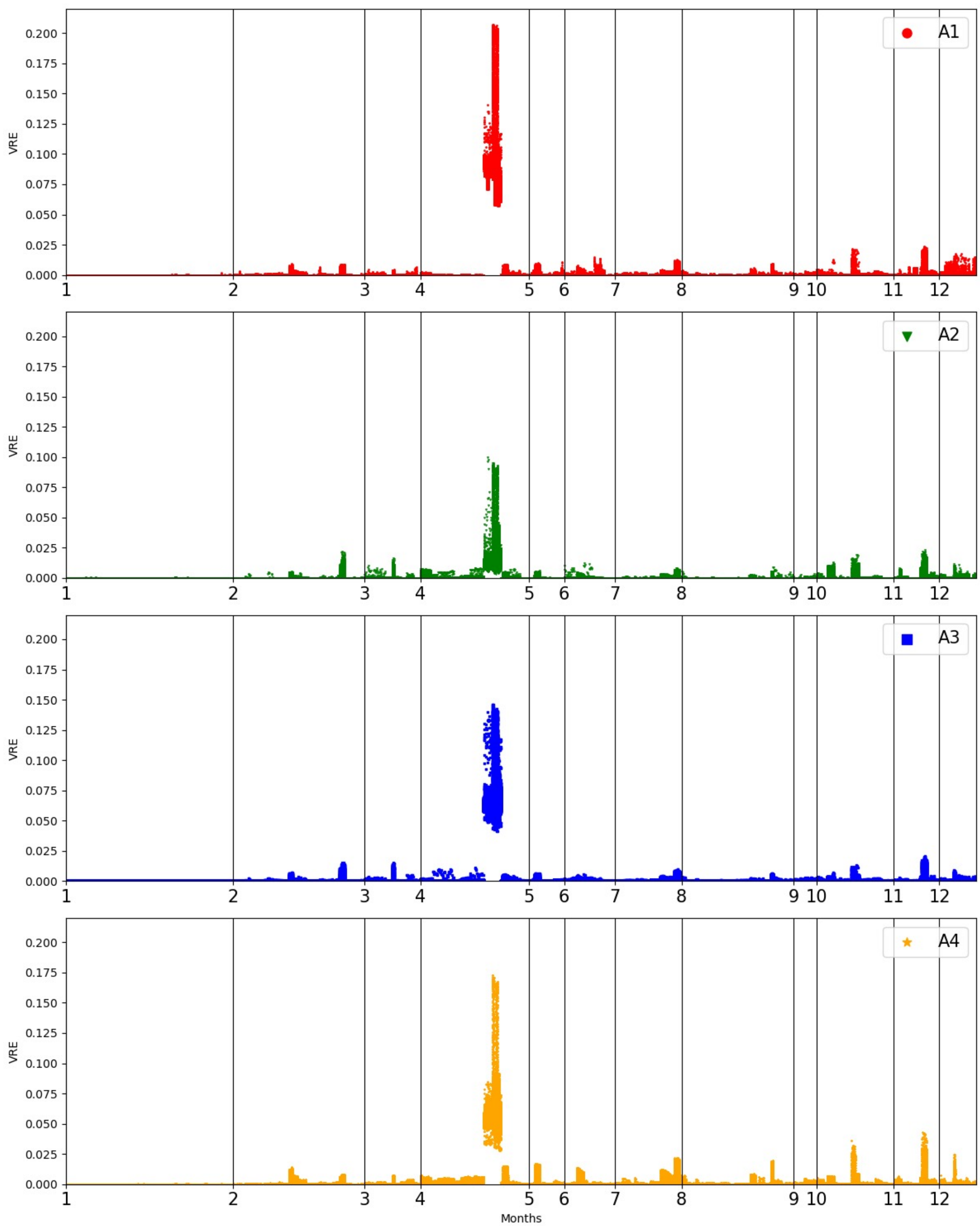


Figure 121: Graphical representation of the Vector Reconstruction Error computed over the entire dataset using the autoencoders of interest.

Test Loss reports the MSE obtained for each model on the test set. **Epsilon** and **Delta** denote the maximum input perturbation values and their corresponding acceptable output perturbations, respectively. Lastly, **Result** and **Time** indicate the outcome of the verification query and the time (in seconds) required by pyNeVer to resolve the query. "True" indicates that a counterexample was found, rendering the model's robustness uncertain. It is evident that all autoencoders achieve a satisfactory level of performance in terms of test loss. However, it is interesting to observe that the simpler autoencoders, specifically A1 and A2, exhibit lower test loss compared to their more complex counterparts.

Figure 121 showcases scatter plots illustrating the Vector Reconstruction Error (VRE) for our autoencoders. Specifically, an anomaly emerges in the data during the fourth month, and it is most effectively detected by A1. Even the smallest Mean Squared Error (MSE) value computed during the anomaly significantly exceeds the maximum MSE among non-anomalous data points. It is noteworthy that while A1 demonstrates superior anomaly detection capability, all autoencoders display the ability to accurately identify anomalies. Following A1, the second, third, and fourth best performers are A3, A4, and A2, respectively. As observed earlier regarding network accuracy, the size of the autoencoders does not seem to have a direct correlation with their performance in anomaly detection.

In terms of verification results, as shown in Figure 121, pyNeVer successfully verifies the property of interest in a reasonable time using its over-approximate algorithm. Furthermore, it appears that A1 and A4 exhibit a slight vulnerability to adversarial perturbation (i.e., small perturbation of the inputs causing unexpected behaviour in the output) in the strictest case, where Epsilon is 0.001 and Delta is 0.002.

7.4.4 IMOCO4.E requirements

The activity of this solution meets the following requirements:

Requirement		L/BB/ P-D-UC	Ver. type (I/D/T/ A)
ID	Description		
R028-D5.1-B6	Predictive maintenance components should be integrated with relevant monitoring systems to create alerts and recommendations.	BB6	I
R031-D5.1-B6	ML predictive maintenance components should be able to process incoming data and apply trained models in real time.	BB6	I
R111-D5.1-P3-8	Real-time decision-making functionalities (on-cloud) GA type: Functional (AI) BBs: BB4, BB6, BB8 Layers: SYS WP5 sub-type: Capability Parent REQ: [Goal Req-D7.10-P3-1]	P3	I
R125-D5.1-P3-22	AI algo must work on synthetic data or those provided as machine log GA type: Functional BBs: BB6, BB8 Layers: SYS WP5 sub-type: Need Parent REQ: [Requirement Req-D7.10-P3-sw-110]	P3	I

7.4.5 Capabilities & Limitations (including USP, strengths & weaknesses)

Utilizing cutting-edge autoencoder technologies, our solution robustly detects anomalies, improving operational efficiency and reducing downtime in industrial settings. This facilitates proactive maintenance, minimizing disruptions to vital machinery.

Moreover, rigorous formal verification during the design and training phases instils confidence in the dependability of our solution.

Although initially tailored for an industrial packaging system, our solution could be adapted to diverse industrial environments, providing flexibility and scalability across various deployments. Seamless integration into the Fog environment enhances effectiveness, facilitating swift data exchange and processing.

Challenges include navigating the intricacies of autoencoder architectures and ensuring the availability and quality of sensor data. The resource-intensive nature of developing and training autoencoder models may also impede scalability and deployment timelines.

Further, scalability of formal verification presents challenges, particularly with large autoencoders, raising questions about ensuring reliability for more complex models.

Finally, if the reliability of a model cannot be certified, determining how to make it reliable remains an open problem.

7.4.6 Customizations & Adaptations (including possible modifications and extensions)

One possible avenue to tailor the UNISS AI for Sensor Anomaly Detection solution to specific industrial contexts, could involve modifying autoencoder architectures to better suit the intricacies of different machinery or environments. By adjusting the number of layers, neurons, or activation functions, the model could be fine-tuned to capture relevant features and anomalies effectively.

Furthermore, exploring feature engineering could open possibilities for enhancing detection capabilities. Incorporating domain-specific features or engineering new ones from sensor data could refine the model's understanding of system behaviour. This tailored approach to feature selection might enhance the model's sensitivity to subtle deviations indicative of anomalies.

Beyond architecture and features, exploring alternative activation functions could present another avenue for improvement. While ReLU is widely used, experimenting with functions like sigmoid or hyperbolic tangent might reveal better-suited options for certain scenarios, potentially improving the model's ability to capture complex patterns.

In terms of extensions, incorporating temporal analysis techniques could provide deeper insights into system behaviour. By leveraging methods such as LSTM networks, which are adept at capturing sequential dependencies in time-series data, the model could detect anomalies that manifest over time, enhancing its predictive capabilities.

Another promising avenue could be multi-sensor fusion, which involves integrating data from diverse sensors to gain a comprehensive understanding of system dynamics. Fusion algorithms, such as Kalman filters or Bayesian networks, could combine information from different sources to improve anomaly detection accuracy.

Looking ahead, implementing incremental learning techniques could enable the model to adapt to evolving system dynamics. Online learning algorithms could continuously update the model based on new data, ensuring its ability to detect novel anomalies and maintain effectiveness over time.

Finally, enhancing the explainability and interpretability of the model's decisions could be important for user trust and troubleshooting. Techniques such as SHAP values or LIME could provide insights into the factors influencing anomaly detection outcomes, empowering users to understand and act upon the model's insights effectively.

7.4.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

The methodology employed in developing the UNISS AI for Sensor Anomaly Detection solution encompasses a combination of advanced techniques and robust toolchains. At the core of this methodology lies the utilization of Python as the primary programming language, facilitating seamless integration of various frameworks and libraries essential for model development and deployment.

Key components of the toolchain include PyTorch and pyNeVer, which serve as fundamental frameworks for designing, training, and verifying autoencoder models. PyTorch, renowned for its flexibility and ease of use, acts as the backend for training purposes, while pyNeVer complements it by providing tools for formal verification—a crucial aspect ensuring the reliability of the solution.

Furthermore, integration with other tools and platforms is essential for seamless deployment and integration within industrial environments. Leveraging frameworks like Kafka for data streaming and ONNX for model conversion enhances interoperability and ensures compatibility with existing infrastructure, such as the Fog environment.

While these tools offer significant advantages, they also present certain limitations and challenges. For instance, managing the complexity of autoencoder architectures within PyTorch may require careful consideration and expertise to ensure optimal performance. Similarly, while pyNeVer provides robust formal verification capabilities, scalability issues may arise with larger autoencoder models, necessitating careful resource management and optimization.

In terms of generic usability, the chosen toolchain offers a balance between versatility and complexity. Python's widespread adoption and extensive ecosystem make it accessible to developers with varying levels of expertise, while frameworks like PyTorch provide powerful capabilities for advanced model development. However, navigating the intricacies of tool integration and optimization requires a certain level of proficiency and understanding, highlighting the importance of continuous learning and skill development.

Through the development and deployment process, several lessons have been learned. Foremost among these is the importance of thorough testing and validation at each stage of development. Rigorous testing not only ensures the reliability and robustness of the solution but also helps identify and address potential issues early on. Additionally, effective collaboration and communication among multidisciplinary teams are essential for successful tool integration and deployment, emphasizing the importance of clear documentation and transparent workflows.

7.5 Image Anomaly Detection AI model (DTT)

Predictive maintenance is crucial in enhancing manufacturing by employing machine learning models on vision data. DTT's Image Anomaly Detection ML model, developed under Pilot 3 Task 5.7, achieves predictive maintenance by analysing vision data, utilizing an AutoEncoder NN model for early fault detection in manufacturing process. Leveraging frameworks like TensorFlow and Keras, this solution, part of a Digital Twin (DT), optimizes manufacturing reliability and minimizes downtime. Specifically designed for the bottle packaging process, this model can be deployed across IMOCO4.E deployments, showcasing its versatility and efficacy in predictive maintenance strategies.

Deployed initially on a Huawei Edge device, the ML/NN model swiftly identifies anomalies in bottle images captured by camera sensors, presenting findings through a user interface (UI). From there, images are then streamed in JSON format via Kafka for further processing in the Fog environment, where the image anomaly detection ML/AI algorithm operates, enabling proactive maintenance interventions. This integration of advanced ML techniques on EDGE and Fog environments ensures efficient anomaly detection, facilitating timely maintenance actions for enhanced manufacturing reliability.

7.5.1 Tech Overview

DTT's Image Anomaly detection ML model is designed to achieve predictive maintenance for manufacturing process through vision data. Although, this solution was developed in the context of Pilot 3, it can be used in other IMOCO4.E deployments. For the anomaly detection, the bottle packaging process was considered as a use case where predictive maintenance is planned to be achieved through deploying an Image Anomaly detection ML/NN model. The NN model development and training were conducted using NN framework such as TensorFlow and Keras.

At the beginning of the process when bottle images are generated via camera sensors which are then fed into the ML/NN model, it identifies any irregularities as Anomalies within the images; the model is deployed on a Huawei Edge device. The final output of the identification of irregularities is displayed through a user interface (UI). From there, these images are then converted to JSON format and streamed using Kafka. Further processing is executed in the *Fog* environment where the image anomaly detection ML/NN algorithm operates. This data flow is illustrated in the block diagram depicted in Figure 122.

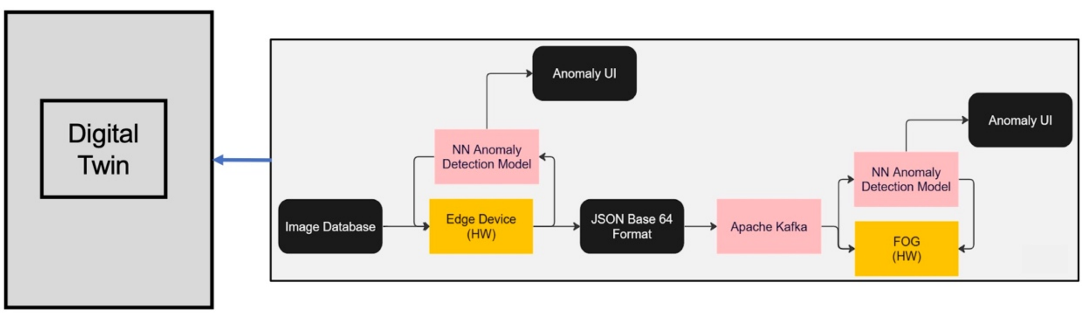


Figure 122: Block diagram of the ML/NN model showcasing the data flow among different components of the solution.

The SW/INT components integrated in Pilot 3 for this solution relates to both BB6 and BB8 and are depicted in Figure 123. For this solution, we have one ML/AI solution for the *Fog* environment (SW-101) and one ML/AI solution for the edge environment (SW-100). All together the implementation of these two solutions serves to showcase the flexibility of the IMOCO4.E approach in different contexts and with different objectives. The development of an ML/AI solution on the edge, in fact, responds to the need of having quick responses, e.g., identification of damaged products triggers whilst ML/AI solutions on the *Fog* enable the possibility of accessing data coming from different sources (e.g., all the machines in a shopfloor) and thus building more reliable and comprehensive models to, e.g., tracing the overall quality of the production.

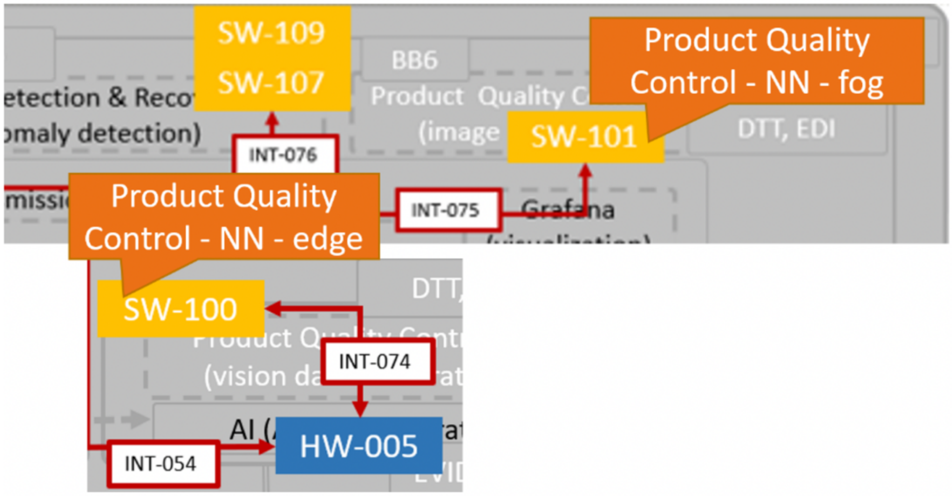


Figure 123: SW/INT components of BB6 and BB8 for the product's quality control (bottle check application)

Both the solution implemented to be installed and run on the edge (i.e., onboard the BB4 board) and the one implemented for the fog environment are image anomaly detection AI algorithms based on Neural Networks. Whilst the on-edge solution is part of BB8, the on-fog algorithm is a solution developed within BB6.

7.5.2 Implementation aspects

The development of DTT's Image Anomaly Detection NN model is carried out in Python by leveraging frameworks like TensorFlow and Keras. The model is trained on the "bottle" dataset (Bergmann et al., 2019). To deploy the ML/NN model on the Edge device, the ATC tool from Huawei was used to convert the model into suitable format. Also, a C++ application was developed to run the interface on the Huawei's Atlas device. The Kafka consumer was configured properly to receive the streaming data in the cloud. Moreover, a Python GUI was developed to visualize the cloud steaming data with the inference output.

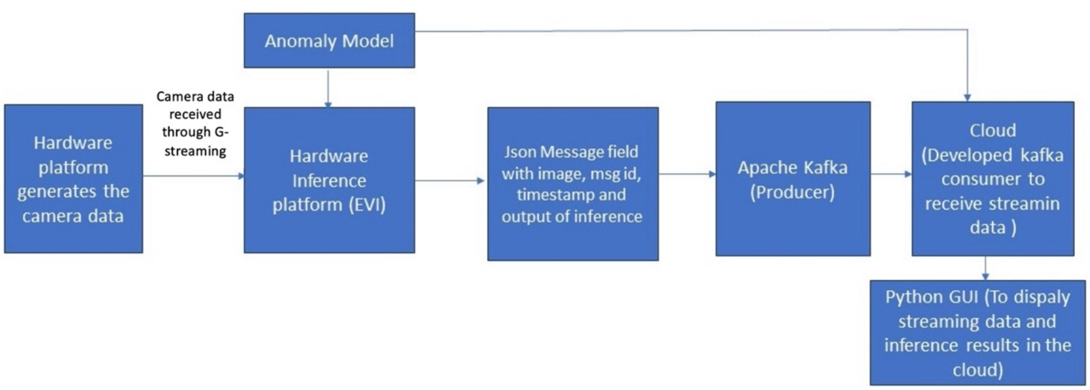


Figure 124: Deployment of the Image Anomaly model ML/NN on the Edge and Fog environment.

After the initial test and validating the performance of this model on the "bottle" dataset, the ML/NN model was deployed on Huawei Edge device, and the integration of this model with other components were carried out by EVI. The diagram in Figure 124 depicts the model's deployment on Edge and *Fog* environment.

7.5.3 Results

DTT's model is trained on the "bottle" dataset. Here, we achieved an overall accuracy of 88 %. The precision and recall values of this model are 75 % and 85 % respectively.

At an initial test (mentioned in previous section), we designed a user interface to select single image and feed this image into the trained ML/NN model; the model then provides the prediction whether image contains any anomaly or not. The Figure 125 shows two images where the left image contains no visible anomaly, and the right image contains anomaly. Our ML model predicts the correct results as shown in Figure 125.

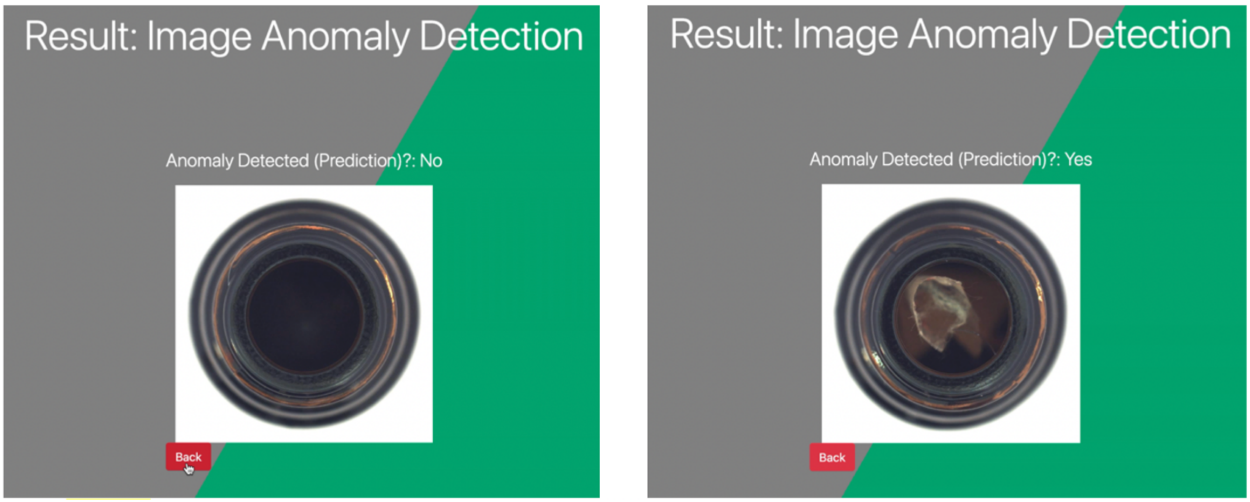


Figure 125: ML model prediction on anomaly: the ML/NN model predicts that - the left image has 'no' anomaly and the right image has anomaly; hence it shows 'yes'. The right image shows visible faults.

After deploying the ML/NN model on the Edge device and on *Fog* environment, we used a Python GUI (mentioned in the previous section) to visualize the cloud steaming data with the inference output.

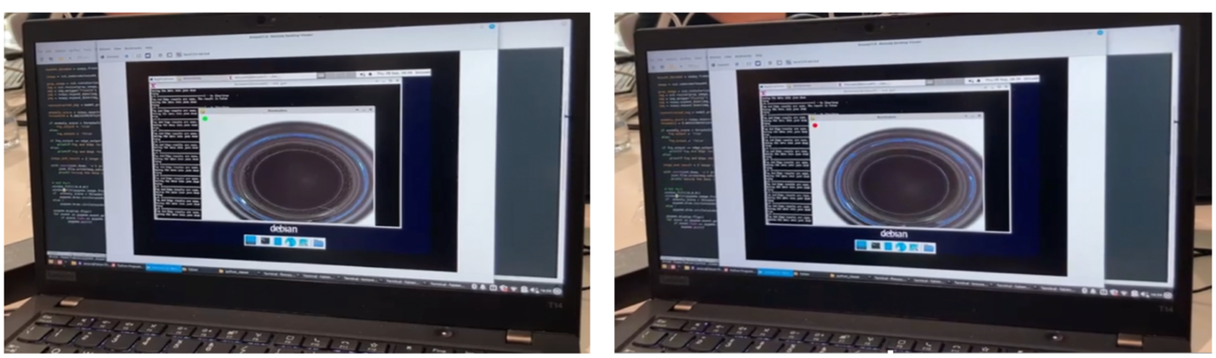


*Figure 126: Visualization, in Python GUI, of ML/NN model's prediction output on the cloud (*Fog*) platform for streaming data*

Figure 126 depicts the visualization, on Python GUI, of the bottle image data which are streamed within the deployment platform; here we can see the model provides prediction whether a bottle image has anomaly or not by a red (with anomaly) or green (no anomaly) circle. A checking was also conducted to check the compatibility of the Edge ad *Fog* results; Figure 127 depicts the compatibility checking execution results for the streaming data.

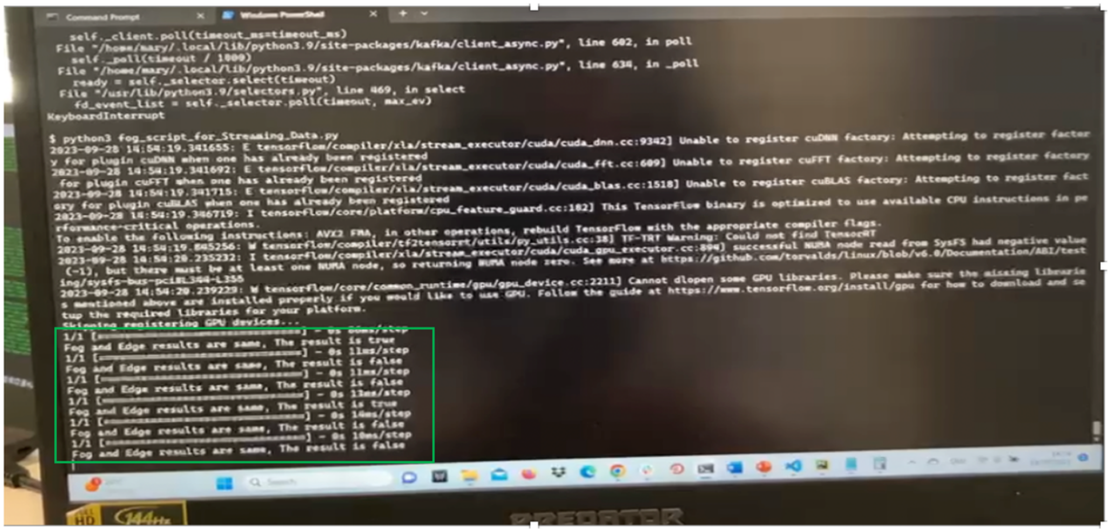


Figure 127: Compatibility checking of Edge and Fog *results on the streaming data. The green rectangle marked area shows the messages "Fog and Edge results are same. The result is false"; "the results is false" indicates* no anomaly *and the result is true indicates* anomaly.

The development and implementation of the Image Anomaly Detection AI model within the framework of Pilot 3 and Task 5.7 have yielded significant strides towards achieving predictive maintenance in the manufacturing process through visionary data analytics. The integration of this solution into the broader context of IMOCO4.E deployments showcases its versatility and applicability beyond its original scope, demonstrating the adaptability of the IMOCO4.E approach in diverse contexts with varied objectives. The AI model, built on Neural Networks using TensorFlow and Keras, successfully addressed the anomaly detection needs within the bottle packaging process. The deployment on Huawei Edge devices and subsequent integration with *Fog* environments exemplifies the seamless operation of the IMOCO4.E framework in real-world manufacturing scenarios. The solution spans both BB6 and BB8, emphasizing the synergies between edge and fog computing in optimizing manufacturing operations.

7.5.4 IMOCO4.E requirements

This solution as part of a Digital Twin (DT) in Pilot 3 specifically caters to the needs outlined in WP5, fulfilling the overall system requirements within the IMOCO4.E framework related to interoperability functionalities. Aligned with BB6 and BB8, this solution meets the specified requirement by providing a robust tool designed to enhance overall bottle manufacturing process and performance, thereby contributing to the improvement of factory productivity. The activity of this solution meets the following requirements:

Requirement		/L/BB/P-D-UC	Verification type (I/D/T/A)
ID	Description		
R133-D5.1-P3-30	Cloud infrastructure be able to retrieve data and run AI SW (algorithms for quality check automation and/or alarm detection & classification algorithms) GA type: Functional BBs: - Layers: L4 WP5 sub-	P3	D

	type: Requirement Parent REQ: [Capability Req-D7.10-P3-sw-12]		
R118-D5.1-P3-15	Autonomous or semi-autonomous operations GA type: Functional (AI) BBs: BB6, BB8 Layers: L3, L4 WP5 sub-type: Capability Parent REQ: [Goals Req-D7.10-P3-1, Req-D7.10-P3-2, Req-D7.10-P3-3]	P3	D
R111-D5.1-P3-8	Real-time decision-making functionalities (on-cloud) GA type: Functional (AI) BBs: BB4, BB6, BB8 Layers: SYS WP5 sub-type: Capability Parent REQ: [Goal Req-D7.10-P3-1]	P3	D
R131-D5.1-P3-28	New modelling approaches to highlight interdependences among independently designed machine parts GA type: Technical (Digital Twin) BBs: BB6, BB8 Layers: L3, L4 WP5 sub-type: Requirement Parent REQ: [Capability Req-D7.10-P3-fw-18]	P3	I

7.5.5 Capabilities & Limitations (including USP, strengths & weaknesses)

With an overall accuracy of 88 % during model training on the "bottle" dataset, and precision and recall values of 75 % and 85 %, respectively, affirm the model's capability and reliability in detecting anomalies. The development of user interfaces, deployment on Huawei Edge devices, and compatibility checks further validate the robustness of the Image Anomaly Detection AI model.

The model boasts advanced capabilities in early fault detection, versatility across manufacturing environments, and efficiency on Edge devices. By leveraging AutoEncoder NN models and frameworks like TensorFlow and Keras, it excels in swiftly identifying anomalies in bottle packaging processes, enabling proactive maintenance interventions to optimize reliability and minimize downtime. Developed under Pilot 3 Task 5.7, its versatility allows for deployment across various IMOCO4.E deployments, showcasing its adaptability to diverse manufacturing use cases. Additionally, its integration into existing infrastructures and seamless deployment on Huawei Edge devices underscore its efficiency in real-time anomaly detection, crucial for time-sensitive manufacturing operations.

The model's unique selling proposition (USP) lies in its ability to integrate seamlessly into existing manufacturing infrastructures, providing a cost-effective solution for predictive maintenance. By utilizing advanced ML techniques such as AutoEncoder NN models, TensorFlow, and Keras, it offers state-of-the-art anomaly detection capabilities, empowering manufacturers with actionable insights for optimizing reliability and minimizing downtime.

The model's strengths lie in its high accuracy and scalability. With its ability to achieve reliable anomaly detection in bottle packaging processes, it ensures targeted and effective maintenance interventions, minimizing false positives and negatives. Developed as part of a Digital Twin (DT), its scalability accommodates the evolving needs of manufacturing environments, future-proofing predictive maintenance strategies and ensuring continued effectiveness as processes evolve.

The model's limitations include its reliance on the quality of training data, which may affect its accuracy and generalization capabilities. Additionally, the resource-intensive nature of training using NN frameworks like TensorFlow and Keras may pose challenges for organizations with limited computational resources or expertise.

7.5.6 Customizations & Adaptations (including possible modifications and extensions)

Customizing the model involves implementing feature engineering techniques and fine-tuning hyperparameters to enhance anomaly detection performance. Feature engineering methods like edge detection or texture analysis can extract relevant information from bottle images, while adjusting hyperparameters such as learning rate and batch size can optimize the model's performance for specific manufacturing environments. These customizations aim to improve the model's ability to accurately identify anomalies in the manufacturing process.

The model's adaptations include the implementation of transfer learning and domain-specific training techniques. Through transfer learning, pre-trained neural network architectures can be fine-tuned with limited annotated data, accelerating training, and improving anomaly detection accuracy. Additionally, domain-specific training involves training the model on datasets that closely represent the target manufacturing environment, incorporating specific features and characteristics to better generalize and detect anomalies inherent to the manufacturing process. These adaptations enhance the model's flexibility and effectiveness in diverse manufacturing settings.

Modifications and extensions to the model involve implementing advanced techniques such as temporal analysis and multi-modal fusion. Temporal analysis allows the model to consider sequential patterns in image data over time, enabling it to anticipate anomalies before they occur by analysing trends in bottle packaging processes. Multi-modal fusion enhances the model's capabilities by integrating additional data sources such as sensor data or metadata, providing a more comprehensive understanding of the manufacturing environment and improving anomaly detection accuracy, particularly in complex scenarios. These modifications and extensions enable the model to adapt to diverse manufacturing contexts and enhance its predictive maintenance capabilities.

7.5.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

The methodology involves employing an AutoEncoder neural network model for anomaly detection in bottle images. Preprocessing techniques such as normalization and augmentation are applied to enhance model performance. TensorFlow and Keras serve as primary toolchains for model development and training. Integration with Edge and Fog environments, facilitated by Huawei Edge devices and Kafka, enables real-time anomaly detection and efficient data processing. Key considerations include scalability, usability, and efficient integration for successful deployment in manufacturing environments.

The primary toolchains utilized are Python TensorFlow and Keras, providing extensive support for neural network development and training. These frameworks enable efficient model customization and optimization. Integration with Edge and Fog environments is facilitated by Huawei Edge devices and Kafka, ensuring seamless deployment and real-time data processing for anomaly detection in manufacturing processes.

The tool limitations include potential requirements for significant computational resources when using TensorFlow and Keras, which may pose scalability challenges for organizations with resource constraints. Additionally, the interpretability of the AutoEncoder models employed in the methodology may be limited, making it challenging to understand the underlying reasons for detected anomalies and hindering the ability to take meaningful corrective actions based on the anomaly detection results.

Generic usability and lessons learned highlight the methodology's scalability and user-friendly toolchains, facilitating adoption across diverse manufacturing environments. Key lessons include the importance of efficient data transmission and processing for real-time anomaly detection, as well as the significance of seamless integration with Edge and Fog environments. Additionally, lessons emphasize the usability of TensorFlow and Keras for model development, underscoring the importance of tool integration aspects for successful deployment and operation of AI models in industrial settings.

7.6 ML Algorithms for data analytics and predictive maintenance (GNT-ITML)

Predictive maintenance is considered to be the most recent and prominent type of approach for the maintenance of industrial equipment and is the result of a series of maintenance paradigm shifts that have historically evolved in parallel with the industrial revolutions realized so far (Lee et al., 2020). The most notable forms of maintenance are the following (Poór et al., 2019a)

Reactive maintenance. During the First Industrial Revolution in the 18th and 19th centuries, the dominant maintenance practice was “reactive maintenance” (also known as “breakdown maintenance” or “corrective maintenance”), where repairs were only performed when a machine broke down.

Preventive maintenance. This type of maintenance was introduced during the Second Industrial Revolution (from late 19th century until mid-20th century). Preventive maintenance relies on performing maintenance activities according to a schedule or when early signs of wear are identified upon regular inspection. With this approach, the risk of unforeseen breakdowns was minimized, and the total equipment lifetime was prolonged.

Productive maintenance. This maintenance form was introduced in the second half of the 20thcentury, along with the advent of computers and the Third Industrial Revolution. In this case, reactive maintenance and preventive maintenance were combined along a systematic data collection and statistical analysis with the aim of increasing overall productivity and maintenance cost-efficiency.

Predictive maintenance. This is the most recent type of maintenance and is associated with Industry 4.0 (Achouch et al., 2022). Even though the concept of predictive maintenance has only been introduced in recent years, it has already been adopted in multiple and diverse industrial sectors (Jimenez-Cortadi et al., 2019), (Maktoubian et al., 2021). In the context of predictive maintenance, sensing technologies can be installed on or near the industrial equipment to constantly monitor its health and performance levels (Tiddens et al., 2022). Remote access to the sensors is enabled by IoT technologies and the data they produce can be accumulated and potentially combined with data from other sources (e.g., environmental). The resulting body of big data can be analysed using Artificial Intelligence techniques to detect specific patterns and early warning signs in order to predict problems before they occur. Therefore, failures can be prevented and associated maintenance requirements can be anticipated, while the necessary parties can be alerted accordingly, and useful insights can be presented to them. As a result, predictive maintenance provides improved planning and prioritization of maintenance activities, where potential downtime can be scheduled to avoid an impact on production, while unforeseen downtime and unscheduled repairs are minimized (Ran et al., 2019). Such a practice leads to a more efficient, sustainable, and cost-effective use of resources, a reduction in repair costs and an increase of the equipment’s reliability and total service lifetime (Pech et al., 2021) and (Poór et al., 2019b).

7.6.1 Tech Overview

The work reported here corresponds to the IMOCO4.E component SW-044 that is jointly developed by GNT and ITML in the context of T5.7 and as part of BB6.

The aim of SW-044 is to assist in predictive maintenance tasks by analysing sensor data from industrial equipment that are aggregated in the cloud. The analysis aims at detecting anomaly patterns in the data that suggest that the equipment is showing early signs of health or performance degradation so as to forecast equipment maintenance needs, improve maintenance planning, avoid possible future equipment downtime or failure and issue corresponding alerts and notifications.

SW-044 relies on Machine Learning (ML) techniques for anomaly detection in order to identify early signs of equipment degradation and predict forthcoming problems in the context of predictive maintenance. It is conceived as a cloud-based component, delivered in the form of containerized microservices. It operates in two distinct modes, i.e., ML training mode and inference mode. SW-044 receives data that have been collected from equipment sensors through the IMOCO4.E BB9. BB9 serves as a central data management infrastructure that accumulates data from multiple sources and makes them available to other IMOCO4.E components though suitable interfaces.

In the SW-044 training mode, the SW-044 training module accesses the BB9 Elastic Search data repository via the provided native Elastic Search API to receive historical data that have been aggregated from sensors over a period of time by performing suitable data queries. The returned data are submitted to pre-processing and are then used for ML training that result to updated ML models. When a new ML model is available, it can be pushed to the component inference module for subsequent use in the inference mode.

In the SW-044 inference mode, the SW-044 inference module needs to receive new incoming data points from sensors that are collected and relayed in real time. For this purpose, the inference module accesses the BB9 Kafka messaging system by implementing a Kafka consumer client. Upon receiving new data points, SW-044 performs any necessary data pre-processing and then analyses them to detect anomalies. Anomaly detection results are structured as JSON messages and published to the BB9 Kafka messaging system by implementing a Kafka producer client within the inference module. The results can be monitored in real time and used to trigger alerts via a Grafana-based interface that accesses the results published on the BB9 Kafka messaging system. Results published to the BB9 Kafka messaging system are also automatically archived within the BB9 Elastic Search repository so that they can be subsequently accessed and visualized using a Kibana-based interface.

The SW-044 internal architecture and connectivity to other components are illustrated in Figure 128.

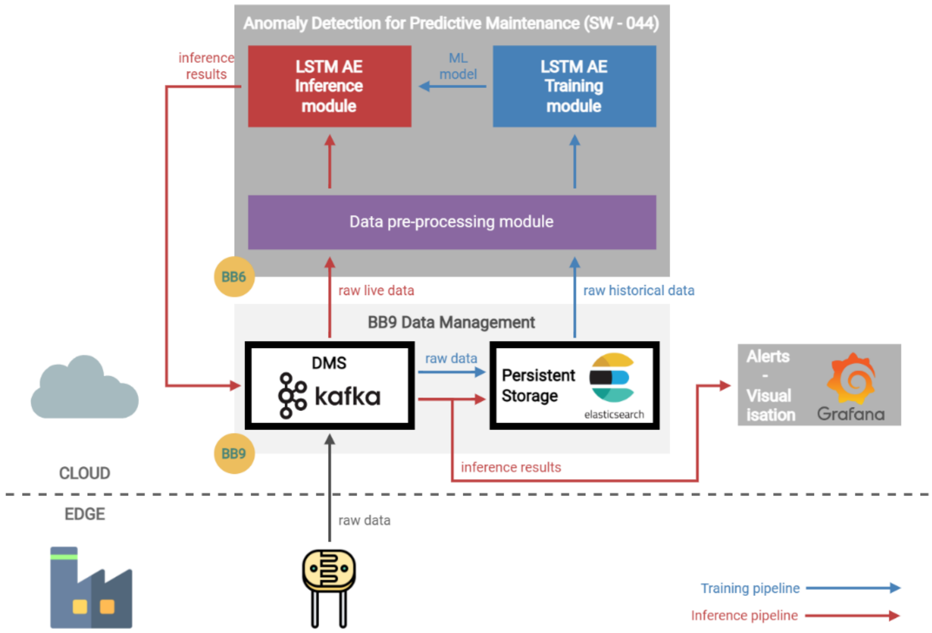


Figure 128: Block view diagram of SW-044 operation, including interconnection with other IMOCO4.E BBs and positioning on edge and cloud layers.

7.6.2 Implementation aspects

Application in Pilot 3

Pilot 3 refers to the High-Speed Packaging industry and is focused on improving current machine monitoring activities and quality control processes. It was decided to apply SW-044 in Pilot 3 for the purpose of predictive maintenance of industrial equipment. In this context, SW-044 is meant to analyse input sensor data streams that are obtained at the edge and relayed to the cloud in order to detect anomalies that are associated with early stages of industrial equipment performance loss and indicate the need to perform equipment maintenance.

Datasets

The datasets that were provided in the context of Pilot 3 refer to the Vega shrink-wrapper machine. Such machines are deployed in large production lines within the food and beverage industry. The Vega shrink-wrapper groups loose bottles or cans, wraps them in plastic film, and heat-shrinks the film to create packages. The cutting assembly, a crucial component, requires proper setup and maintenance, with degradation monitoring vital for machine reliability and minimizing unexpected downtime. Visual inspection during operation is challenging due to the enclosed metal housing and high rotation speed of the blade. Therefore, assessing the state of the blade of the cutting assembly based on sensor readings and predicting maintenance needs is of paramount importance. More specifically, the following two datasets [A] and [B] were used.

Dataset [A]: New-Worn Dataset (labelled)

Initially, the "Vega shrink-wrapper component degradation" dataset published on Kaggle (https://www.kaggle.com/datasets/inIT-OWL/vega-shrinkwrapper-runtofailure-data) was used. The dataset focuses on the degradation of the cutting blade in the Vega shrink-wrapper machine, specifically comparing the performance of a new blade to a worn-out blade. The dataset includes six (6) data snippets as separate CSV files, each of which is 8 seconds long and consists of 2048 data points. The dataset compares a new blade to a completely worn-out one, with a focus on deviations in the cutting process. The dataset is labelled, as three (3) of these files clearly indicate that they contain data obtained from a new cutting blade, while the other three (3) contain data obtained from a worn blade. We use the following naming conventions for these 6 files: NewBlade001, NewBlade002, NewBlade003, WornBlade001, WornBlade002, WornBlade003. Each data point in each file includes values for eight (8) parameters, which correspond to columns in the tabular dataset format. A visualization of the each of these time-series parameters in each dataset was produced for the needs of analysis in IMOCO4.E, which is presented in Figure 129.

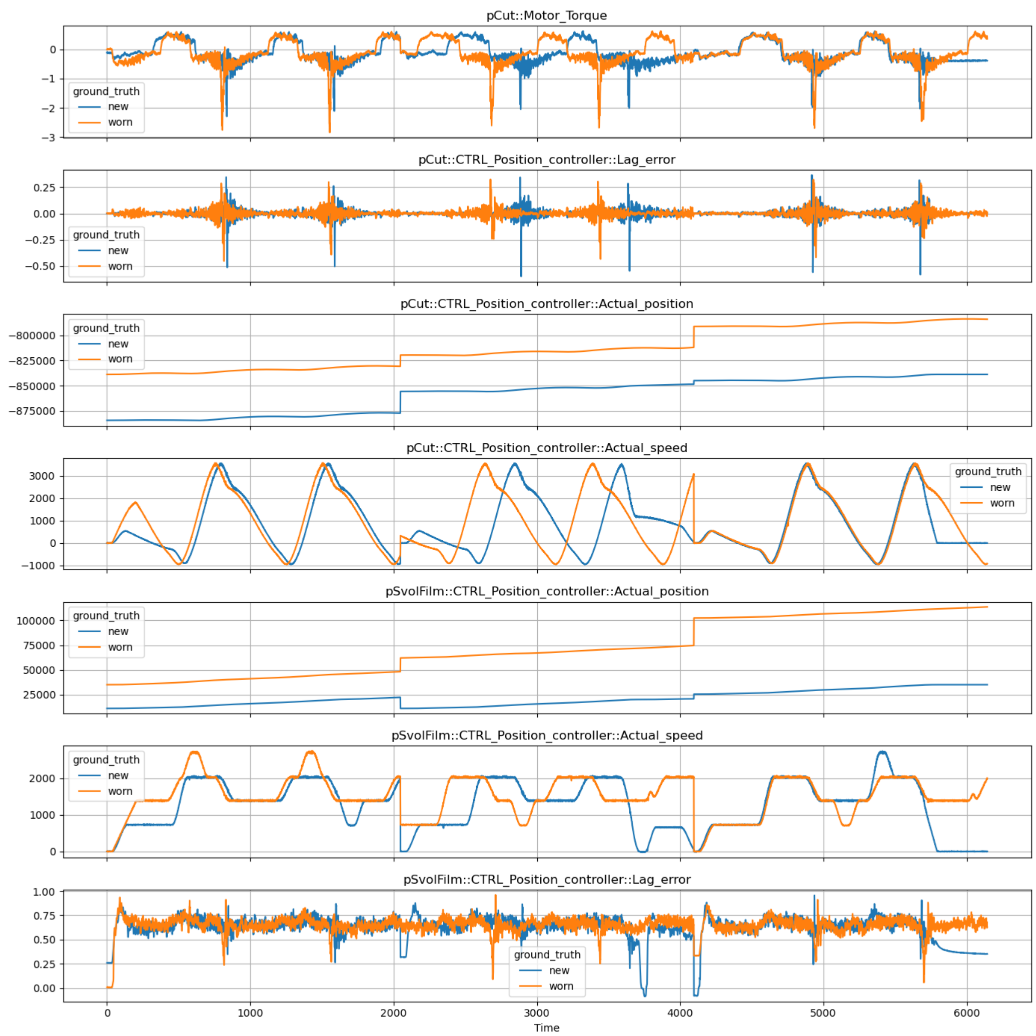


Figure 129: Vega Shrink-Wrapper new-worn blade data, six dataset, three new and three worn. Time is measured in number of samples – Each dataset is 2048 samples long.

Dataset [B]: One-Year Dataset (unlabelled)

In a later stage of SW-044 development, we also used another dataset for verification purposes. This second dataset was also collected from the same shrink-wrapper machine. It is the "One Year Industrial Component

Degradation" (https://www.kaggle.com/datasets/inIT-OWL/one-year-industrial-component-degradation) dataset, which provides machine data recorded over a 12-month period. The dataset comprises 519 files, each representing an approximately 8-second sample with a 4ms time resolution, totalling 2048 time-samples per file. The file naming convention includes information about the month, day, start time, sample number, and machine operation mode, ranging from 1 to 8. Figure 130 presents the 7 features of the dataset from all 519 files. Each feature is plotted in a separate graph as a continuous time series over the period of 12 months (X axis). The colour coding denotes the respective operation mode that is active in each case.

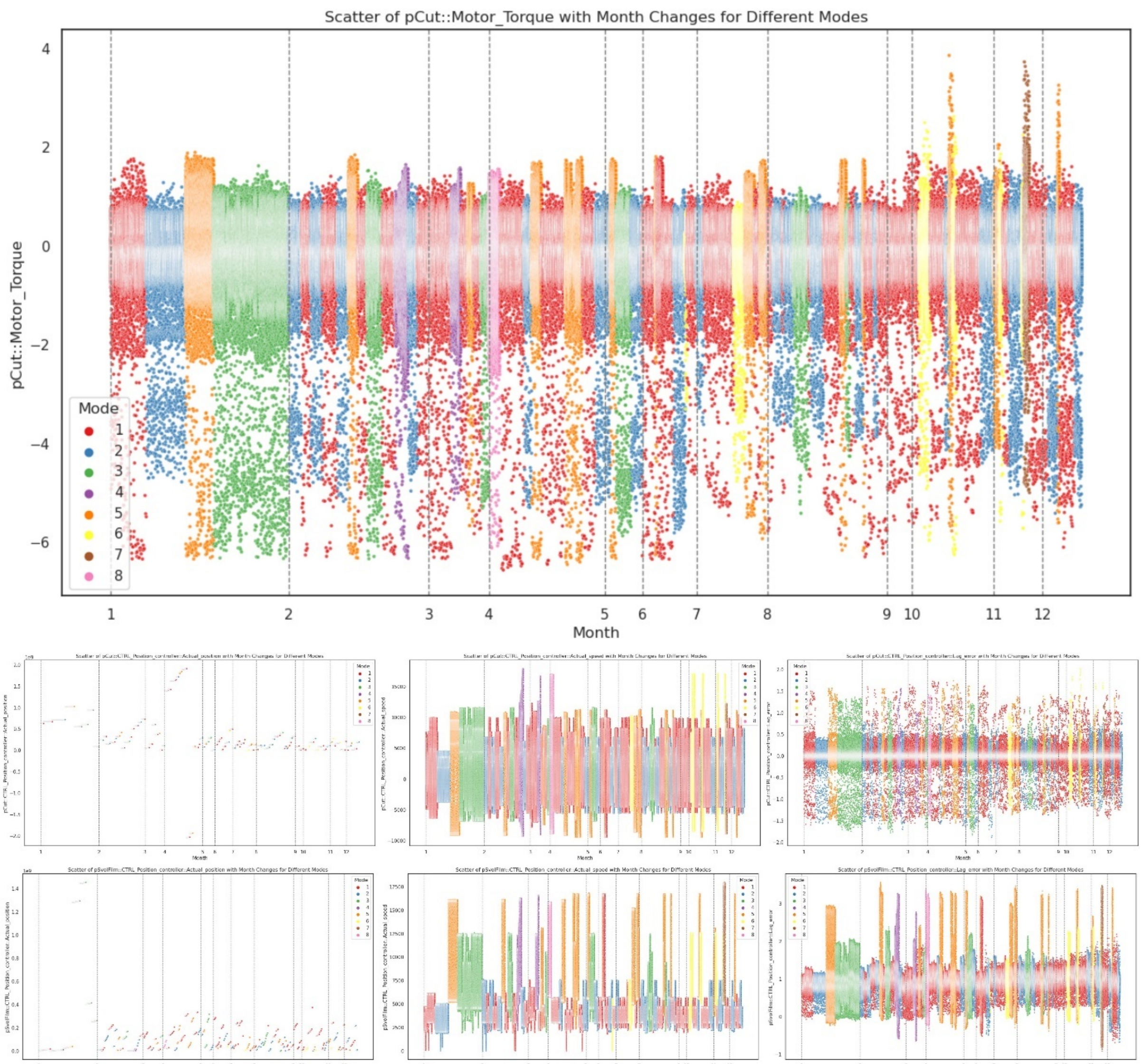


Figure 130: The seven features in one-year dataset split by month and colour-coded by operation mode

Comparison of the two datasets ([A] and [B])

Both datasets present distinct advantages and limitations. Dataset [B] spans a 12-month duration, offering a substantial amount of data for robust training and evaluation. However, a drawback is the absence of explicit labelling, leaving uncertainty about whether each file corresponds to a new or worn cutting blade.

Conversely, dataset [A] comprises six labelled datasets, clearly indicating whether the snippets are from a machine with a new or worn cutting blade. An additional consideration arises from the mode labels embedded in the filenames of the One Year dataset, denoting the operational mode during each snippet's recording (ranging from 1 to 8). Notably, this mode information is missing from the labelled datasets used for training and evaluation (3 new and 3 worn).

ML approach

Our goal is to utilize state-of-the-art deep learning techniques to implement an anomaly detection module which can be used for online inference. The employed deep anomaly detection technique utilizes a Sequential Autoencoder framework for anomaly identification. The Autoencoder (AE) architecture comprises an encoder, a decoder, and a middle layer bottleneck. The AE is trained to reconstruct its input by compressing it into a low-dimensional encoding and then restoring it to its original dimensions. The sequential properties of the AE are enabled by using Long Short-Term Memory (LSTM) units to implement the encoder and decoder layers. LSTMs can capture long-term dependencies in temporal data.

The computation flow involves passing a multivariate input vector through the encoder, consisting of one or more LSTM layers, generating a compact representation fed into the decoder. The decoder reconstructs the original input size, and the objective function minimizes the reconstruction error between input and output. After training on normal instances, reconstruction errors are used to define the anomaly scoring function. More concretely, the approach models normal reconstruction errors as a Gaussian multivariate distribution, with parameters μ and Σ estimated using Maximum Likelihood Estimation. The anomaly score for a new data point is calculated as the Mahalanobis distance from the Gaussian distribution defined using the normal reconstruction errors, allowing for continuous anomaly scores, offering more informative insights than binary labels.

Experiment design and dataset usage for ML model training and validation

Considering the two available datasets, **only the labelled dataset [A] (new-worn) could be reliably employed for supervised or semi-supervised training and evaluation** due to the libelling ambiguity in dataset [B] (one-year). A more robust strategy for model training and evaluation was designed that was based **on a re-organisation of the available data inside the 6 files of Dataset [A] into 7 sets,** as described below.

NewBlade001 and **NewBlade002** were split into the following sets:

- **(1) Normal Train (s_N)**: This set is used for training the LSTM-AE. The model uses it to learn the patterns and characteristics of the normal behaviour.
- **(2) Normal Validation-1 (v_{N1})**: This set is used for early stopping during the training of the network weights. Also, it is used for estimating the Gaussian parameters μ and Σ of the normal reconstruction errors' distribution (Mahalanobis params).
- **(3) Normal Validation-2 (v_{N2})**: This set is used to learn the threshold τ by maximizing the F_β-score.
- **(4) Normal Test (t_N)**: This set is used for evaluating the performance of the trained anomaly detection model. It should contain normal sequences that are independent of the s_N, v_{N1}, and v_{N2} sets.

WornBlade001 and **WornBlade002** were split into the following sets:

- **(5) Anomalous Validation (v_A)**: This set is used to learn the threshold τ by maximizing the F_β-score.

- **(6) Anomalous Test (t_A)**: This set is used for evaluating the performance of the anomaly detection model on unseen anomalous sequences. It should contain anomalous sequences that are distinct from the v_A set.

NewBlade003 and **WornBlade003** were combined into a single dataset:

- **(7) Mixed Validation (v_M)**: This set was reserved for the final model validation.

On the other hand, **dataset [B] (one-year) was exclusively utilized** to carry out more informal experiments for **verification purposes** and to investigate the feasibility of a system health calculation method. To assess the effectiveness and generalization capability of the model, we conducted a new semi-supervised training using the final model architecture on dataset [B]. We assumed that the dataset represents a full lifecycle of a cutting blade, with month 1 reflecting the use of a new (healthy) blade and month 12 indicating a worn-out blade needing replacement. For this implementation, we considered month 1 data as normal and month 12 data as anomalous. Following this approach, we utilized the same dataset sub-splits as in previous experiments: Normal Train (sN), Normal Validation-1 (vN1), Normal Validation-2 (vN2), Normal Test (tN), Anomalous Validation (vA), and Anomalous Test (tA). However, this time, we employed the entire dataset for verification to evaluate both unseen data and the data used for training and anomaly thresholding.

ML model improvements

In the preliminary phase reported in the previous (D5.7) deliverable, a vanilla LSTM-AE model was trained and evaluated. In that phase, we ran multiple experiments to decide the reconstruction error threshold that distinguishes normal and anomalous instances and empirically adopted a threshold of 0.4 which appeared to have the highest F1-score (Accuracy: 70.5 %, Precision: 72.1 %, Recall: 67.5 %, F1-score: 77.3 %). **In the second phase reported in this deliverable (D5.8), several improvements were applied to the model** that focused on the areas that are elaborated in the following sub-sections.

Anomaly scoring thresholding function

In the preliminary phase (D5.7), a threshold was empirically selected and was applied directly on the anomaly scores (i.e., the Mahalanobis distance of a reconstruction error from the distribution of the reconstruction errors of the normal training set, Normal Train (sN)). This approach, however missed elasticity, i.e., the threshold would not adapt dynamically and in an automated way to the data distribution or generalize well across different data distributions. Our currently improved elastic scoring function provides a more dynamic threshold determination mechanism using the F1-score maximization technique. To compute the threshold for anomaly detection, we first determine the lower and upper percentiles (25th and 75th) of the normal and anomaly scores (computed on the validation sets Normal Validation-2 (vN2) and Anomalous Validation (vA)). Then, we divide the range between these percentiles into intervals. For each interval, we calculate the F-beta score, which considers the trade-off between precision and recall. The threshold with the highest F-beta score is selected, and if there are multiple such thresholds, we choose the median among them to ensure robustness. This approach enables adaptive threshold determination based on the distribution of normal and anomaly scores, enhancing the effectiveness of anomaly detection.

Hyperparameter tuning

The hyperparameter space defined for tuning the LSTM-AE encompasses several key parameters that are crucial for optimizing model performance. These parameters include the learning rate, number of layers in the encoder and decoder, units in each layer of the encoder and decoder, batch size, number of epochs for training, dropout rate, choice of activation function (either ReLU or tanh), and regularization techniques such as L1 or L2 regularization. The range and granularity of each parameter were selected to explore a diverse set of configurations efficiently. By searching through this parameter space, the goal is to find the combination of hyperparameters that maximizes the performance of the LSTM-AE model in terms of its ability to reconstruct input sequences effectively while generalizing well to unseen data. To implement this process, we used Bayesian optimization and more specifically, the Hyperopt library (Hyperopt, 2024).

Sequence length

In an LSTM network, data is typically organized into sequences, where each sequence consists of a series of time steps. The sequence length hyperparameter determines how many timesteps the LSTM will consider at once during training and inference. When designing the model, choosing an appropriate sequence length is a crucial but challenging task, since balance must be achieved between computational efficiency and model performance. Longer sequences allow capturing more temporal dependencies but increase computational complexity and memory requirements. Conversely, shorter sequences might overlook critical patterns in the data.

In the context of LSTM-AE for detecting contextual anomalies online, with streaming data arriving every 4ms according to the employed datasets, selecting an optimal sequence length involves balancing efficiency with capturing relevant historical context. For a sequence length of N, queuing N datapoints is necessary to infer on the (N/2)th datapoint in the past, following Backpropagation Through Time (BPTT) principles. Hence, prioritizing efficiency is essential, as larger sequence lengths may only allow inferences for distant datapoints, potentially affecting real-time decisions.

Given these constraints, to define a search grid for the sequence length hyperparameter, we further inquired potential values to test using a signal processing approach. First, we conducted frequency domain analysis to identify the dominant frequency present in the data, providing insights into the underlying temporal characteristics. Next, autocorrelation analysis was performed to assess the presence of repeating patterns or correlations at specific lags, shedding light on the temporal dependencies within the time series. Finally, we analysed the significant lags identified through autocorrelation to gauge their relevance in capturing relevant historical context.

The results from this analysis were collected per dataset and feature. We only present the features with positive dominant frequency and those for which significant lags were detected. To choose a sequence length that allows for efficient inference without introducing significant latency, we decided on the following grid search: 6 (24 ms), 16 (64 ms), 20 (80 ms).

F-Beta

The choice of the beta parameter value in the F-beta score calculation affects the balance between precision and recall in an anomaly detection model as follows:

- Beta < 1.0: Emphasizes precision more than recall. Suitable for minimizing false positives even if it means sacrificing some recall.
- Beta = 1.0: Balances precision and recall equally. Useful when precision and recall are equally important.
- Beta > 1.0: Emphasizes recall more than precision. Suitable when it is crucial to capture as many true anomalies as possible, even at the cost of more false positives.

In the context of anomaly detection and choosing a threshold, the beta parameter indirectly influences the selection of the optimal threshold. A higher beta value will result in a preference for higher recall, potentially leading to a lower threshold that captures more anomalies. Conversely, a lower beta value will favour precision, possibly resulting in a higher threshold with fewer false positives. **Three different values for beta were investigated to determine the most effective model: 0.6, 1, 1.6.**

7.6.3 Results

Testing on dataset [A] (new-worn)

The first testing iteration assessed the impact of the three different sequence lengths on the F1 metric. At the same time, hyperparameter tuning was enabled for 200 different setups to determine best architecture and hyperparameters for the model (Figure 131).

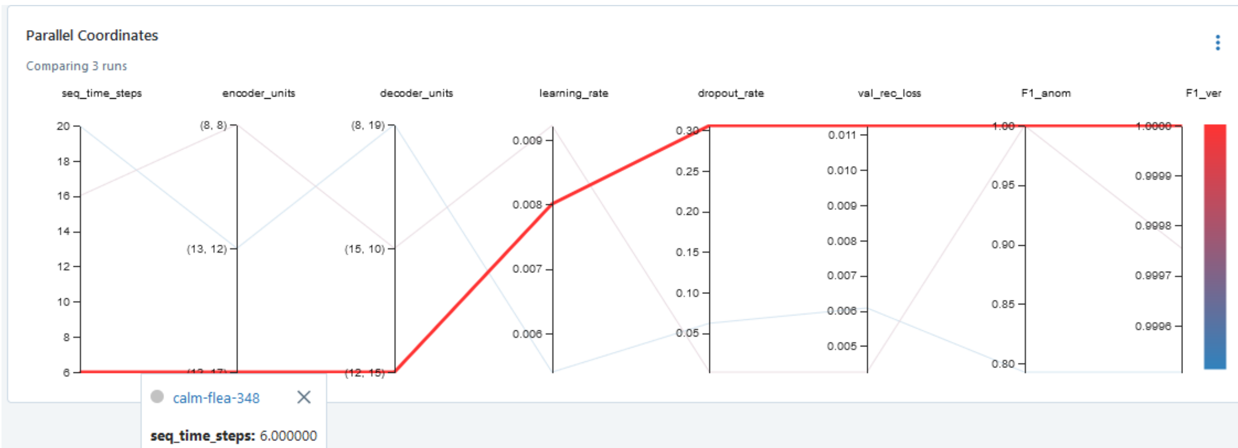


Figure 131: Evaluation of different combinations of sequence length and hyperparameter values.

The optimum sequence length was 6 timesteps, which provides a great balance between accuracy and efficiency. Table 11 presents the optimal hyperparameters that were determined for the selected sequence length.

Table 11. Optimal hyperparameters values for selected sequence length 6

encoder_layers:	**2**	decoder_units:	**(12, 15)**	batch_size:	**32**
encoder_units:	**(12, 17)**	seq_time_steps:	**6**	dropout_rate:	**0.30603**
decoder_layers:	**2**	activation:	**tanh**	learning_rate:	**0.008004**

With regards to the metrics evaluation, as can be seen in Table 12, the model had excellent performance on recognizing anomalies in both the anomalous data set vA and the verification dataset vM, which was essentially a mix of normal and anomalous instances. The results are presented in the table. Please note that we excluded F1, Precision and Recall metrics for the fully normal dataset testing, as these metrics are undefined since there cannot be any true positives in this data partition.

Table 12. Effectiveness metrics for selected sequence length and hyperparameters.

Accuracy_anom (v_A):	**1.0**	Accuracy_ver (v_M):	**1.0**	Accuracy_norm (v_{N1}):	**1.0**
Precision_anom (v_A):	**1.0**	Precision_ver (v_M):	**1.0**	Threshold:	**128.3**
F1_anom (v_A):	**1.0**	F1_ver (v_M):	**1.0**	train_rec_loss:	**0.006**
Recall_anom (v_A):	**1.0**	Recall_ver (v_M):	**1.0**	val_rec_loss:	**0.01**

The next iteration explores beta as a parameter that can influence the determination of the anomaly thresholding. Based on the results that can be seen in the parallel coordinates graph in Figure 132, all three beta value experiments lead to excellent F1 score in the verification dataset. However, the value Beta= 0.6 is selected, as it provides the lowest rec_error_loss, while favouring less true positives which we expect due to noisy data and data drifts in the analysed datasets.

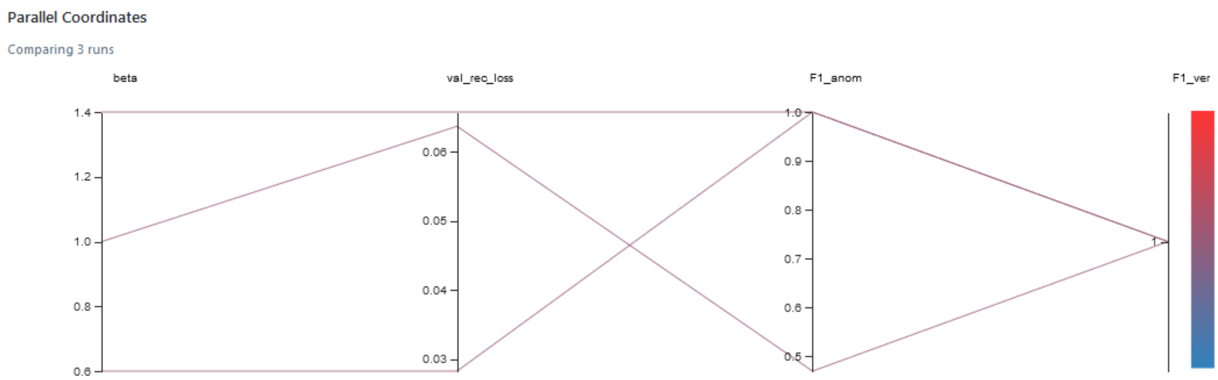


Figure 132: Influence of F-beta to reconstruction error and F1.

Figure 133 presents the distributions of detected normal and anomalous data points for each value of F-beta as well as the associated threshold. Both beta values of 0.6 and 1.4 appear to lead to a clear differentiation of the two distributions. However, given that the dataset is very noisy, **we selected a beta equal to 0.6** aiming to have a less strict threshold that favours false positives.

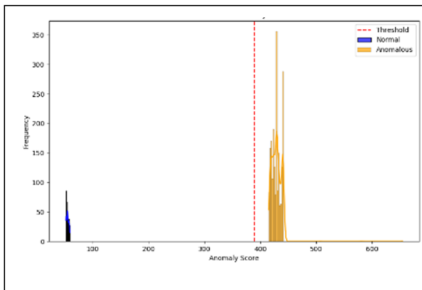
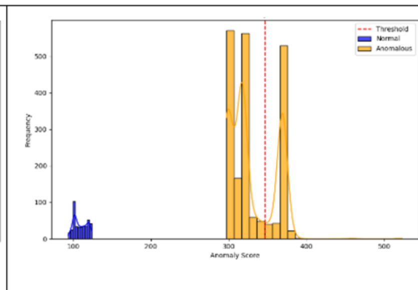
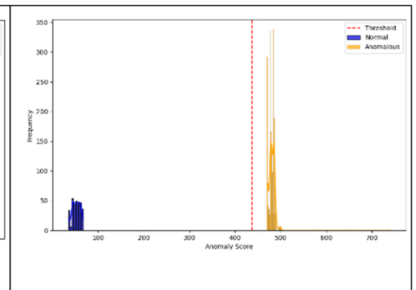
		
beta=0.6	*beta=1*	*beta=1.4*

Figure 133: Influence of F-beta to anomaly threshold determination.

The following Figure 134 demonstrates the result of the anomaly detection on the verification set vM that is composed of 2048 initial samples (timesteps) that contains normal instances from NewBlade003, followed by another 2048 samples (timesteps) that are all anomalous instances from WornBlade003.

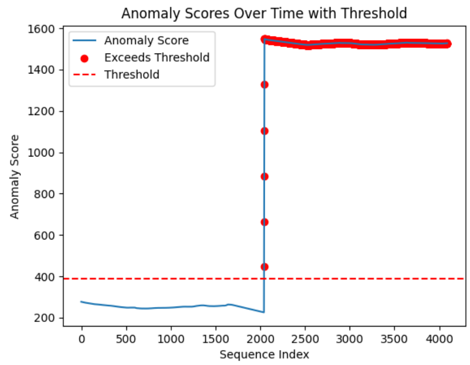


Figure 134: Anomaly detection results in verification dataset (v_M) and calculated threshold

The anomaly scores threshold learned using the F1-score maximization for the final model was equal to 389.4. There seems to be to a clear separation between normal and anomalous instances, indicating a consistent operation of the model, which is confirmed by the excellent effectiveness metrics that are presented in the following Table 13.

Table 13. Effectiveness metrics for final model application to verification dataset v_M.

Accuracy_anom (v_A):	**1.0**	Accuracy_ver (v_M):	**1.0**	Accuracy_norm (v_{N1}):	**1.0**
Precision_anom (v_A):	**1.0**	Precision_ver (v_M):	**1.0**	Threshold:	**389.4**
F1_anom (v_A):	**1.0**	F1_ver (v_M):	**1.0**	train_rec_loss:	**0.003**
Recall_anom (v_A):	**1.0**	Recall_ver (v_M):	**1.0**	val_rec_loss:	**0.028**

Testing on dataset [B] (one-year)

Figure 135 presents the anomaly detection results of a model that underwent semi-supervised training on Dataset [B] (one-year), where month 1 was assumed to refer to normal and month 12 to refer to anomalous. In this case, anomaly scores express the reconstruction errors' distance from normal reconstruction errors computed for month 1. The plot also includes a regression line that has been fitted over the resulting data, which shows an increasing anomaly score trend as time progresses. The extreme peak in month 4 agrees with the outliers detected in feature "pCut::CTRL_Position_controller::Actual_position".

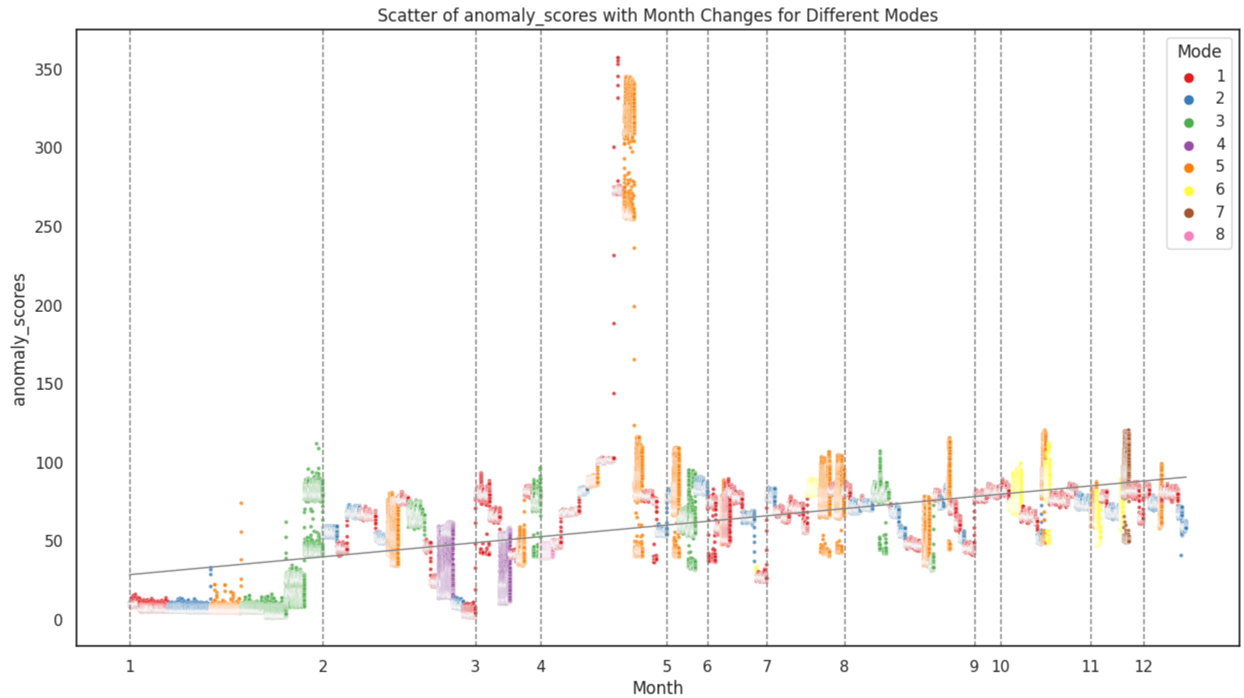


Figure 135: Results of semi-supervised training on One Year data, using month 1 as normal and month 12 as anomalous: the raw anomaly scores inferred by the LSTM-AE based on the reconstruction error. X-axis follows the timesteps of the raw data with the tick labels indicating the month of operation.

System Health Calculation

To assess the cutting blade's health in the shrink-wrapper machine, we utilized anomaly scores, which indicate deviations from normal behaviour. Higher anomaly scores suggest a higher likelihood of abnormal system behaviour, potentially indicating deterioration in the cutting blade's health. Our goal was to develop a function that converts anomaly scores to a 0 to 100 scale, representing system health.

To achieve this, we normalized the anomaly scores to ensure consistency across datasets and time periods. Normalization involves scaling the scores between 0 and 1, where 0 signifies normal operation and 1 signifies the most anomalous behaviour. These normalized scores were then used to compute the system health index. We employed a logistic function to map the normalized anomaly scores to the system health index. A system health index of 100 denotes optimal health, with minimal deviations from normal behaviour. Decreasing index values indicate declining cutting blade health, with lower values suggesting increased likelihood of anomalies or performance deterioration. A system health index of 0 indicates a critical state requiring immediate maintenance or replacement to prevent further damage or operational disruptions.

In Figure 136, we plot the anomaly scores (blue) and system health index (red) over one-year data timesteps on the x-axis. Despite noise in the dataset, observable trends reveal a decreasing system health index with increasingly frequent and severe declines over time.

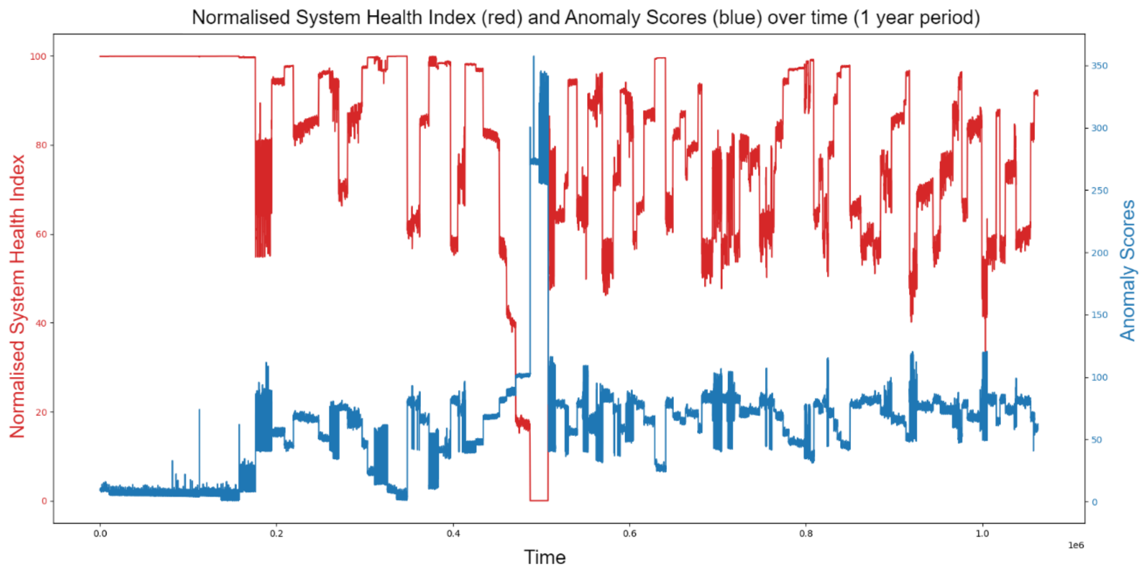


Figure 136: System Health Index (red) calculated from anomaly scores (blue) over the period of one year.

7.6.4 IMOCO4.E requirements

SW-044 satisfies the following BB6 requirements: R028- D5.1-B6, R031-D5.1-B6

SW-044 satisfies the following Pilot 3 requirements: R107-D5.1-P3-4, R108-D5.1-P3-5, R111-D5.1-P3-8, R125-D5.1-P3-22.

7.6.5 Capabilities & Limitations (including USP, strengths & weaknesses)

Advantages of Deep Learning approach for anomaly detection

The goal of anomaly detection, also known as outlier detection or novelty detection depending on the context, is to identify rare events that are significantly different from the rest of the data or do not match an anticipated pattern. There is intense ongoing research activity around the topic of anomaly detection and a growing demand for it by multiple industries and application fields.

While typical ML algorithms such as distance-based methods, ensemble-based algorithms, and statistical algorithms are widely used in anomaly detection, they underperform when applied to IoT data (Al-amri et al., 2021). Factors that contribute to the presented challenges include scalability, heterogeneous and high-dimensional feature spaces, interdependencies between features, the cost of feature extraction and sparsity and imbalance of classes.

Conversely, Deep Learning (DL) has been found to show great potential in developing useful representations of high-dimensional, heterogeneous, and large data of spatial or temporal nature. DL models can handle heterogeneity and big data volumes without the burden of feature engineering by domain experts. This allows for end-to-end optimization across the entire task pipeline (Al-amri et al., 2021), (Pang et al., 2021) and (Basora et al., 2019). In addition, deep learning methods can learn representations of normality and abnormality, as opposed to conventional unsupervised methods that only manage to estimate statistical deviations and cannot acquire any preceding knowledge of expressive representations, which could be applied to a generalisation and detection of new types of anomalies (Pang et al., 2021). Deep sequential models are also very good at capturing the long-term dependencies and dealing with temporal complexity. This is an advantageous property for IoT applications where time-series are more closely linked to collective or context anomalies rather than point anomalies (Basora et al., 2019) and (Chalapathy et al., 2019). Finally, DL training can occur under an unsupervised or semi-supervised scheme, which is particularly significant in cases where high-quality labels are absent in available datasets.

The implementation of DL in anomaly detection tasks consists of applying neural networks for learning either feature representations or anomaly scores. When separate models are used for feature extraction and anomaly scoring, they can be combined in a sequential application, although the decoupled nature of learning has a negative impact on the method's overall efficiency in anomaly detection. A more recent trend is the learning of feature representations for normality (Pang et al., 2021), where a single model can be used to learn both features and the representation of normality. For this purpose, the autoencoder model has been found to show great promise. This model learns low-dimensional representations of features and uses its data reconstruction errors to define an anomaly scoring function. This model presupposes that normal instances are more accurate than anomalies. Autoencoders can be easily implemented and trained, but they are also quite generic. They allow the integration of deep learning networks of all types, such as Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), etc., depending on the type of data (e.g., sequential, tabular or images). However, the model does not perform well when the dataset used for training is not pure and contains anomalies. In such cases, the model could learn a representation of normality that is contaminated with several irregularities, resulting in a lower performance for detecting deviations from normality.

Rationale for promoting a Long Short-Term Memory (LSTM) Autoencoder (AE)

For the needs of SW-044, following a careful consideration of possible DL approaches and techniques for anomaly detection, the approach of **learning feature representations for normality** was chosen. The designed architecture combines the autoencoder approach (AE) with a sequential network, the Long Short-Term Memory Network (LSTM). The rationale for this choice was based on the following aspects:

- Sequential models are able to capture the temporal dependencies in time series data.
- Implementation steps are clear and manageable.
- There are no significant stability problems associated with model training (unlike methods based on Generative Adversarial Networks).
- It is not necessary to label anomalous instances in the training dataset.

The deep autoencoder framework (AE) forms the higher-level architecture of the module for anomaly detection. The AE is a neural network that learns how to reconstruct input by passing it through a lower-dimensionality middle layer. The architecture is shown in Figure 137. It integrates an encoder, a decoder and middle layer bottleneck. In the encoder, input x is initially sent and gradually compressed into a low-dimensional version called encoding. The encoding then passes through the decoder, which slowly restores it to its original dimensions. The AE can be optimized to produce faithful reproductions x' of original inputs by penalizing reconstruction errors, which are the distances between x and x'. The AE is a by-product of this process. It can be used to reduce the dimensions of inputs, as the encodings created by the bottleneck layer are low-dimensional representations.

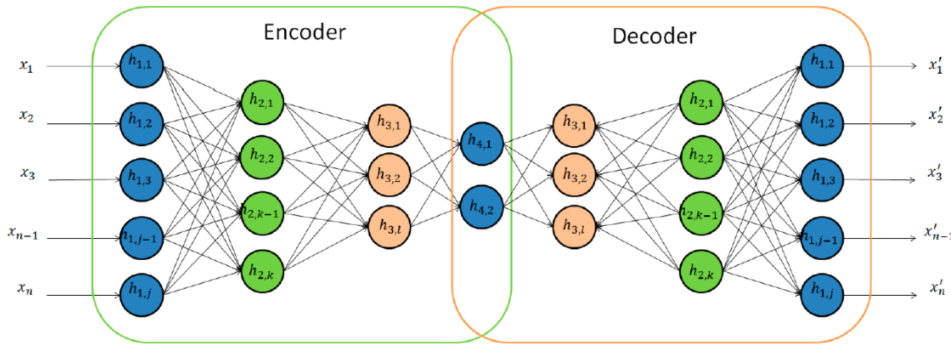


Figure 137: The autoencoder architecture (Berman et al., 2019)

Long Short-Term Memory Networks (LSTMs) allow information to be persistent. This is particularly valuable in cases of time-series data, where temporal dependencies can be important and past information may help to better understand the present. LSTMs have a similar underlying concept to Recurrent Neural Networks (RNNs). However, LSTMs are designed to overcome RNNs' main weakness of being unable to track long-term dependency (vanishing gradient problem). LSTMs maintain data in memory over long time periods, which makes them highly applicable to a wide range of tasks that require sequential data processing, such as analysis of video, sound, and sensor data streams. LSTMs use the same chain structure of RNNs as shown in Figure 138, where each network instance is sending a message to the next instance at each timestep. The LSTM unit (Figure 139) is more complex, with three multiplicative units - the input, output, and forget gates - that control the information flow. It also has a memory block called the cell-state, which stores past information until a forget gate decides to discard it. LSTMs can be applied in both predictive and generative tasks.

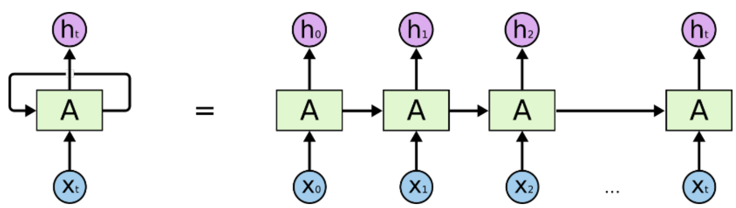


Figure 138: An unrolled RNN unit.

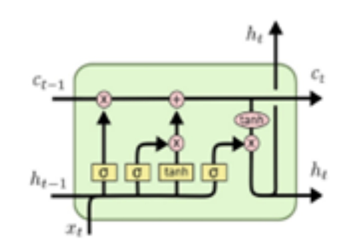


Figure 139: An LSTM unit (Olah, 2015)

The LSTM AE implementation for anomaly identification in multisensory time-series (Malhotra et al., 2015) and (Malhotra et al., 2016) adopts the approach of learning feature representations as normality (Pang et al., 2021) as a way of detecting anomalies. This paradigm combines feature learning with anomaly scoring into a single model, as has already been noted. It is done by using generic objective functions to learn representations that are forced into capturing the underlying regularities of normal data. The anomaly score is calibrated using the reconstruction error for the normal representation. Values that exceed this threshold are then marked as anomalies. A more robust approach is available in (Malhotra et al., 2015) and (Malhotra et al., 2016), where the reconstruction errors e of the normal data is modelled as a Gaussian multivariate distribution and their parameters μ and Σ are calculated using the Maximum Likelihood Estimation. Once the parameters are estimated, the anomaly score for new data points can be calculated using the Mahalanobis distance. The Mahalanobis distance is a measure of the distance between a point and a distribution, and it takes into account the correlation between the different features of the data. Subsequently, anomaly scores for new data points are calculated while a threshold above the probabilities is determined by maximising the Fβ , which is a weighted average of precision and recall. It should be noted that continuous anomaly scores are generally more informative than a simple binary label.

Strengths and limitations of specific implementation and datasets

We applied the LSTM-AE architecture for deep anomaly detection in a predictive maintenance scenario. The model learns a normality representation and uses the reconstruction error of new samples to infer anomaly scores, which serve as a proxy for equipment state estimation. The module incorporates a thresholding function that dynamically adjusts based on the training data distribution of reconstruction errors to determine anomalies. Initially, we applied these techniques to the new-worn dataset and later verified on the one-year data. The implemented module, including the integrated model and thresholding function, yielded excellent results on the new-worn dataset. In the verification process using the one-year dataset, we confirmed the utility of reconstruction errors as a proxy for cutting blade wear, as they consistently increased over time. Additionally, we introduced a System Health index as a quantitative measure of wear, bounded between 100 (new blade) and 0 (needs replacement). The work carried out in relation to the System Health Index can serve as the foundation for a future implementation of Remaining Useful Life (RUL) estimation in cases where more complete, labelled and truly run-to-failure datasets are available.

Our approach aims to offer a comprehensive AI-based predictive maintenance solution. However, we encountered limitations, such as incomplete data labelling and observed data drifts in the datasets. Regarding labelling, we could not confidently establish whether the new-worn dataset data were captured under the same conditions as the one-year data, lacking operational mode information. For the one-year dataset, we couldn't ascertain if it was truly run-to-failure or included intermediate blade replacements. Moreover, noticeable outliers were detected in the measurement of the cutting blade's (month 4) and film's actual positions, where expert opinions were unavailable. Notably, significant data drift was observed between the two datasets for the cutting blade's (order of magnitude 106 for new-worn, 109 for one-year) and film's actual positions (order of magnitude 103 for new-worn, 109 for one-year). This data drift suggests potential discrepancies or shifts in the operating conditions between the two datasets, which may impact the model's performance and generalization capability. Efforts to mitigate these challenges will be crucial for enhancing the robustness and reliability of the predictive maintenance solution. Finally, even though we managed to achieve excellent effectiveness metrics in the model validation, it should be noted that the underlying new-worn dataset that was employed was very limited in size and therefore the results may not be easy to transfer and generalise when considering other larger and more diverse datasets.

7.6.6 Customizations & Adaptations (including possible modifications and extensions)

In the context of application in Pilot 3, SW-044 was deployed on a fog server using a Docker container. It was configured to receive its input time series data from the BB9 Kafka broker (SW-040) hosted on the same fog server by implementing a Kafka consumer client. It was also configured to implement a Kafka producer client so as to send its inference results (classification predictions) as messages to the Kafka broker. The inference results were also permanently stored in an Elastic Search repository (SW-041) and were visualised using a Kibana dashboard and via a Grafana dashboard for a real-time view (Figure 140). The interactions of SW-044 with other components in the Pilot 3 environment are illustrated in Figure 141.

Figure 140: BB9 Grafana dashboard created to visualize the Pilot 3 sensor data and inference results by SW-044

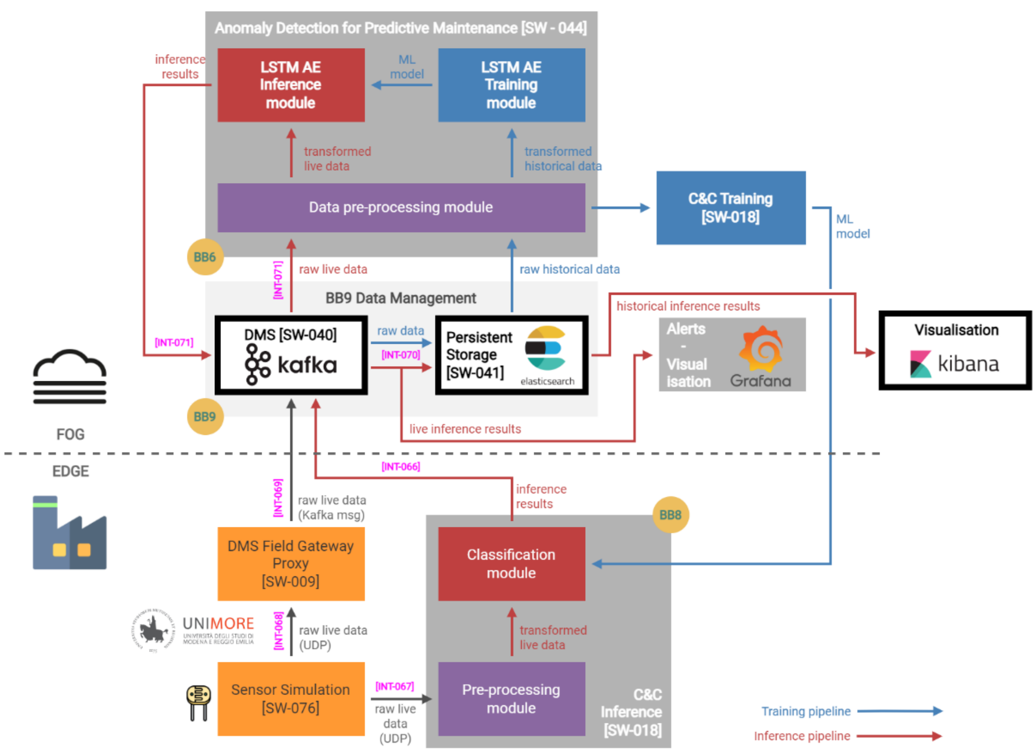


Figure 141: Implementation of SW-018 in Pilot 3 and interactions with other components

The containerisation of SW-044 allows it to be easily **deployed and maintained on different host environments** at the fog or cloud layers. In future implementations, SW-044 can potentially adopt **different protocols for receiving input data and sending its inference results**, as a modular design of the component has been adopted internally. Furthermore, depending on the use case and available time series dataset, SW-018 can be re-trained and customised by implementing other sets of LSTM AE hyperparameters that may prove to be more suitable for other use cases and other datasets; also in this case, the modular internal design and exposure of critical re-configurable variables allows for an easy adaptation of such aspects.

7.6.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

The following tools and toolchains have been used for the development and evaluation of SW-044:

- **Jira** (Atlassian Jira, 2024): Used for Issue Tracking.
- **Bitbucket GIT Repository** (Atlassian Bitbucket, 2024): Used for source code version control.
- **Docker** (Docker website, 2024): Used for containerisation and deployment.
- **MLFlow** (MLflow, 2024): Used for ML automations, ML model management and experiment tracking.
- **TensorFlow** (TensorFlow, 2024): Used for ML model development.
- **PyCaret** (PyCaret, 2024): Used for ML model development and ML automations.
- **Quix** (Quix, 2024): Used for handling data streams in ML components.
- Third-party open-source **Python libraries**, including **pandas** (pandas, 2024), **NumPy** (Numpy, 2024), **Keras** (Keras, 2024) and **Hyperopt** (Hyperopt, 2024).

7.7 Predictive maintenance using Remaining Useful Life estimation (INTRA)

SW-097 “RUL estimation for feedforward calibration” of the IMOCO4.E software catalogue aims at predicting the remaining useful life (RUL) of the Tissector system of Pilot 1 at any operational time. RUL prediction is particularly useful in motion control systems, to proactively estimate in a timely and cost-effective manner when a maintenance action on a monitored machinery is required.

In particular, the software developed in the scope of this task monitors data collected from the Tissector and “learns” the healthy state of the machine, as well as its degradation scheme reflected by trajectory errors, a main cause of which often appears to be friction. Thus, a health indicator (i.e., the state of operation in relation to the monitored parameters that is considered as “healthy”) for the system is built, which is based on measurements of friction compensation applied on the system and its efficacy in the accuracy of the motion control of the system. Based on this knowledge, the ML component developed in the scope of SW-097 estimates the remaining operational time of the Tissector before it needs recalibration to bring feedforward compensation back on track.

The data collected is related to the operational time of the system in cycles, where each cycle corresponds to a single usage of the Tissector. For RUL prediction, various ML techniques are employed to continuously estimate the remaining cycles of healthy operation of the Tissector, at any point in time, before it needs calibration to nullify feedforward compensation degradation.

7.7.1 Tech Overview

SW-097 is implemented in Python. It comprises a training and an inference component. The training of the component is performed on an initial dataset extracted from the Tissector or its simulator. RUL predictions are produced by the inference component in real time whenever the Tissector outputs its state information. The results of the prediction are visualized, so that appropriate actions can be decided. Both training and inference pipelines incorporate appropriate pre-processing pipelines to curate the input datasets.

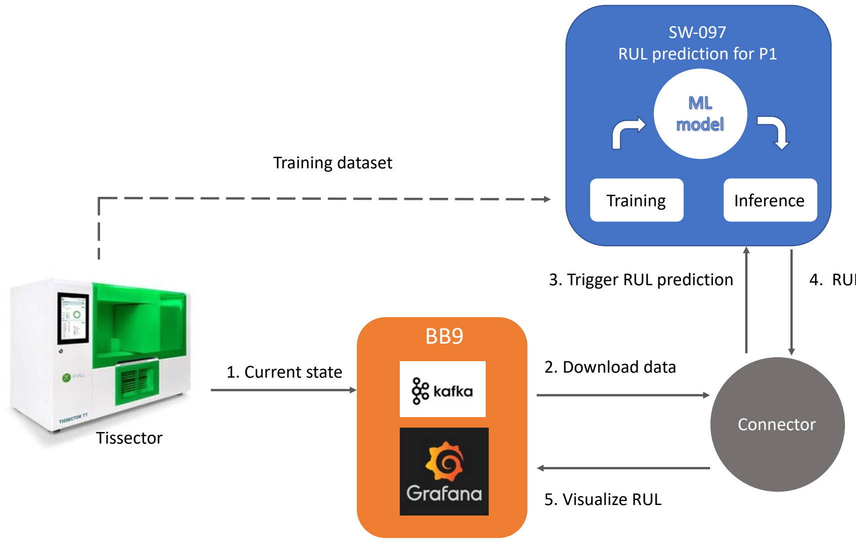


Figure 142: Training and prediction process flows of SW-097

Figure 142 illustrates the integration of SW-097 with P1 and BB9. The Tissector (P1) generates data that is published through BB9. The Connector component mediates the communication between SW-097 and the device for modularity and security reasons. The Connector is responsible for detecting when a change in state of the Tissector is published and is granted with access for downloading the relevant dataset of arbitrary size to feed the RUL prediction process.

7.7.2 Implementation aspects

The process flows of both training and inference processes are depicted in Figure 142. When a proper training dataset is extracted from the Tissector, it is given as input to the training component of SW-097, which results in building the ML model that will be used for the inference cycle, from that point on. Each time the Tissector sends new state information (step 1) to BB9, a Kafka message is created. The message is detected by the Connector, which downloads the state dataset from an address designated in the message and is stored in a location accessible by SW-097 (step 2). The RUL prediction process is triggered (step 3). When the inference process is complete, RUL prediction is sent back through the Connector (step 4) to BB9, so that it can be visualized in P1's dashboard (step 5).

The dataset, used for the training phase of SW-097 at the time of this report, included measurements from simulating the operation of the Tissector in 6 cycles. The different cycles of operation demonstrated gradual degradation of the Tissector condition from the healthy state to the state when calibration is required. Each experiment corresponding to a single operation cycle, produced friction data for 5 different operation velocities of the Z-stage of the machine. From the 30 data points produced as a result, 24 were randomly selected for training and 6 for testing.

Several ML algorithms Figure 143 were tested with various hyperparameters' settings. Ideally, more data reflecting the degradation scheme of the Tissector is required to represent not a single but a multitude of healthy-to-unhealthy state transitions. This would lead to increased accuracy and generalization of the methodology. Nevertheless, even with the data currently available, various ML approaches prove to perform well as it will be shown next.

7.7.3 Results

Figure 143 depicts the test results of different algorithms that were examined for the prediction. Each plot demonstrates the RUL in cycles computed by the adopted ML algorithm for each test data point, in comparison to the actual RUL reflected by the data. The detailed test results are listed in Table 14. Based on these results, Adaptive Boosting and Decision Trees regressors seem to outperform the rest. However, it remains to be seen in the final stages of the implementation and testing, if enriched datasets that better represent actual states of the Tissector are available, whether this conclusion is robust, or another methodology will prove prevailing. The final results will be reported in (D6.6).

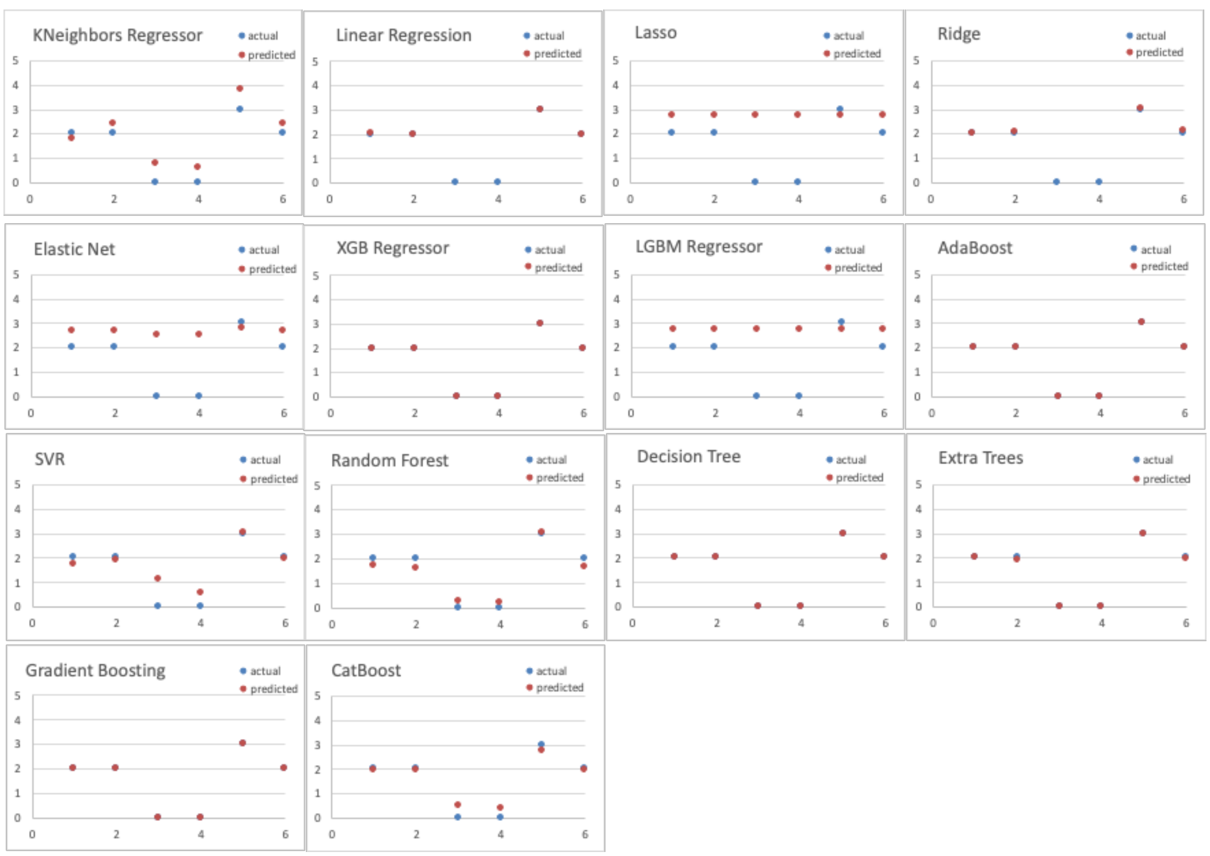


Figure 143: ML algorithms tested for RUL prediction of the Tissector (Pilot 1)

Table 14. ML algorithms effectiveness for RUL prediction of the Tissector (Pilot 1)

Algorithm	r2	MSE	RMSE	MAPE	MAE
AdaBoostRegressor	1	0	0	0	0

DecisionTreeRegressor	1	0	0	0	0
XGBRegressor	1	1.09E-07	0.00033	8.47E+11	0.000232
GradientBoostingRegressor	0.999922	9.79E-05	0.009893	1.1E+11	0.004071
ExtraTreesRegressor	0.99924	0.00095	0.030822	0.010556	0.021667
LinearRegression	0.99892	0.00135	0.036737	7.55E+13	0.025774
Ridge	0.993201	0.008498	0.092187	1.49E+14	0.079836
RandomForestRegressor	0.941627	0.072967	0.270123	3.38E+14	0.246667
CatBoostRegressor	0.935945	0.080068	0.282964	6.83E+14	0.213251
SVR	0.774383	0.282022	0.531057	1.28E+15	0.340057
KNeighborsRegressor	0.733333	0.333333	0.57735	1.05E+15	0.533333
ElasticNet	-0.84715	2.308934	1.519518	3.74E+15	1.208902
Lasso	-1.25	2.8125	1.677051	4.13E+15	1.333333
LGBMRegressor	-1.25	2.8125	1.677051	4.13E+15	1.333333

As explained earlier, the output of SW-097 is sent to BB9 for being visualized in time. Figure 144 show the measurements extracted from the Tissector (top plot), and the results of RUL prediction when the latter is invoked (bottom plot).

Figure 144: RUL prediction by SW-097 in P1 dashboard

7.7.4 IMOCO4.E requirements

The relevant to SW-097 requirements are partially related with BB6 and the nature of SW-097 as a predictive maintenance ML piece of software, and partially with the integration of the component with the IMOCO4.E framework and the rest components of Pilot 1 SW and HW components. Details about the tests that verify and validate these requirements will be reported in the final deliverables of WP6. In particular SW-097 has addressed the following requirements: R028-D5.1-B6, R031-D5.1-B6, R082-D5.1-P1-12, R086-D5.1-P1-16

7.7.5 Capabilities & Limitations (including USP, strengths & weaknesses)

RUL prediction is critical for effective condition monitoring. Achieved effectiveness of relevant methodologies can prove extremely beneficial to the industry domain. However, it is usually difficult to acquire rich datasets produced by machinery in real environments. In real operation machines are not allowed to operate in unhealthy states to collect relevant state data, while their degradation cycles are usually slow. At the same time, datasets from monitoring a multitude of similar machines are difficult to acquire, especially for rare and specific-purpose machinery like the Tissector. Nevertheless, for c multi-factor degradation schemes, as is the case for equipment comprising moving parts that are affected by several types of friction parameters that vary over time, model-based approaches might be difficult to accurately produce feedforward compensation schemes to efficiently and accurately over time counteract friction. In these cases, properly designed ML approaches that can be trained on an arbitrary number of observed parameters have the potential of presenting a viable solution.

7.7.6 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

Python libraries: scikit-learn, catboost, lightgbm: development of the RUL prediction LM component.

MLFlow: ML model management, versioning, and ML pipelines.

Bitbucket: Used for source code version control.

Docker: Containerization of SW-097 and deployment.

7.8 AI for predictive maintenance of a backend semiconductor manufacturing assembly line (ITEC)

7.8.1 Tech Overview

ITEC provides the pilot on semiconductor production in the IMOCO4.E project. The readers are referred to the deliverable (D5.7) and the deliverable (D7.4) for an introduction and details into the technical aspects of predictive maintenance considered in this section. The objective is to maximize production line throughput by minimizing maintenance impact on the production line. We make use of a digital twin to simulate and mimic a production line. This digital twin then functions as an environment to model different policies.

7.8.2 Implementation aspects

Many different policies have been analysed; RL agents have been trained. Based on real production data, an expected improvement of > 1 % throughput increase can be expected. The maintenance policy for one specific activity, the cleaning of the mould, the last machine in the production line, will be implemented in the Nexperia factory in China. Together with a method to send maintenance alerts when product-based maintenance is nearing its end life, resulting in a decrease in operator/technician wait times.

7.8.3 Results

The goal of overall OEE improvement at TRL 5-6, in the scope of the IMOCO4.E project, has already been achieved. Higher TRL implementation in a Nexperia backend factory is still in progress, with an expected first pilot delivery in 3 months.

7.8.4 IMOCO4.E requirements

Req ID	Requirement Description	Verify	Test ID	Result	Comments & Rationales
R003-D2.3-*	AI-components, control algorithms, and digital twin models may use additional data or sensory input interfaces to train the model. However, after completion, it shall only make use of existing data and interfaces of the brownfield system.	I	-	PASS	
R007-D2.3-*	AI-components, (control) algorithms, and digital twin models shall be documented (input, output, parameter interface and user guidance).	I	-	IN PROGRESS	The documentation is evolving and in progress.
R008-D2.3-*	AI-components, (control) algorithms, and digital twin models shall be testable in simulation (e.g. by means of digital twins) and deployable on the physical target.	I	-	PASS	
R020-D2.3-*	Algorithms, AI-components and digital twin models that are intended for real-time deployment shall not adversely affect the responsiveness of the system to user requests.	T	TEST-011	IN PROGRESS	
R021-D2.3-*	Algorithms, AI-components and digital twin models that are intended for relieving the operator from complex tasks shall be safe and predictable.	T	TEST-011	IN PROGRESS	
R026-D2.3	Compliance with data privacy regulations (e.g. GDPR, PIPL) and related cybersecurity regulations or frameworks (e.g. https://www.iotsecurityfoundation.org/iotsf-issues-update-to-popular-iot-security-compliance-framework/)	I	-	PASS	The factories' infrastructure is a closed system.

R038-D2.3-*	All smart control algorithms, AI-components and digital twin models shall be testable in a simulation / virtual environment.	I	-	PASS	
R047-D2.3-*	Any smart control algorithms, AI-components and digital twin models shall not adversely affect the safety of the system.	T -> D	-	IN PROGRESS	
R052-D2.3-*	Sensors monitoring the condition of an asset should be able to push measurements to the IMOCO4.E big-data infrastructure.	T	TEST-011	IN PROGRESS	
R140-D2.3	High-frequent data-collection and logging of sensors / actuators signals upon failures in the system should be present.	I	-	PASS	'Scope' is a tool used to obtain this information.
R146-D2.3	Standardized interfaces shall be defined to guarantee replaceability of the components, if available.	I	-	PASS	Machines communicate to the cloud through standard interfaces (TCP/IP, SECS/GEM)
R148-D2.3-B6-sw	Predictive maintenance software components should create real-time notifications about anticipated malfunctions of monitored assets.	A	TEST-011	PASS	Real-time notification is shown in the GUI. The component's deployment in the factory is not in the IMOCO project's scope.
R183-D2.3	Interfaces to deploy learned networks are present. Note: The main targets are BB1, BB2, BB5, and BB6.	I	-	PASS	The component's deployment in the factory is not in this project's scope. This is inspected on a proof-of-concept deployment.
R190-D2.3	BB8 shall offer AI components including one or more forms of verifiability, for example: Providing a human-interpretable view of the algorithm Providing a	D	TEST-011	PASS	A simulation and digital twin framework with

	framework to assess reliability in a simulation/digital twin environment.				GUI is developed for verifiability.
R182-D2.3	To support integration across all layers, BB8 shall offer industry-standard interfaces to each of the IMOCO4.E layers to exchange data	I	-	PASS	
R183-D2.3	Interfaces to deploy learned networks are present. Note: The main targets are BB1, BB2, BB5, and BB6.	I	-	PASS	The developed network can be used with the main targets with appropriate adaptations.
R190-D2.3	BB8 shall offer AI components including one or more forms of verifiability, for example, providing a human-interpretable view of the algorithm and providing a framework to assess reliability in a simulation/digital twin environment.	D	TEST-011	PASS	
R191-D2.3	Only authorized users have access to systems and data	I	-	PASS	
R193-D2.3	BB8 shall support a computing continuum in the sense that BB8 can operate in all layers, i.e., from the instrumentation layer up to the cloud layer.	D	TEST-011	PASS	The developed network can operate in all layers with appropriate adaptations.

7.8.5 Capabilities & Limitations (including USP, strengths & weaknesses)

The implementation relies on operator adherence to the way of working. If the operator/technician decides to deviate from the advice given by the maintenance planning system, the optimal throughput improvement cannot be guaranteed.

7.8.6 Customizations & Adaptations (including possible modifications and extensions)

Buffer levels are not yet taken into account in the first step of the implementation, this could be an added interesting feature to add once the first implementation is done.

7.8.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

Lessons learnt from the first phase of the pilot is to try and aim for a solution which is as lightweight as possible, trying to omit the use of multiple IT infrastructure layers, and keeping the SW solution as close to the machine as possible.

7.9 VR application for driving the iGo neo forklift within a virtual environment (NURO)

7.9.1 Tech Overview

Nuro's VR application (SW-098) for driving the iGo neo forklift from STILL represents a ground-breaking advancement in the realm of warehouse operations and autonomous vehicle technology. At its core, this application offers a highly immersive virtual reality environment where operators can simulate driving the iGo neo forklift within a meticulously crafted 3D warehouse setting. Nuro has painstakingly designed this virtual environment to accurately reflect real-world warehouse conditions, complete with dynamic elements such as moving humans and other autonomous forklifts. By providing operators with a realistic training platform, this VR application allows them to gain valuable hands-on experience in navigating complex warehouse environments without the risk of accidents or damage to property.

Integral to the functionality of Nuro's VR application is the seamless integration of AI algorithms for path planning of the iGo neo autonomous forklift. Leveraging state-of-the-art artificial intelligence technology, these algorithms enable the iGo neo to autonomously navigate through the virtual warehouse environment with precision and efficiency. By analysing real-time data on obstacles, traffic patterns, and environmental conditions, the AI path planning system dynamically adjusts the forklift's route to optimize safety and productivity. This sophisticated integration of AI-driven path planning algorithms not only enhances the realism of the VR simulation but also lays the groundwork for future implementation of autonomous driving capabilities in real-world warehouse settings.

Moreover, Nuro's VR application serves a critical role in facilitating perception understanding for operators of the iGo neo autonomous forklift. As the forklift manoeuvres through the virtual warehouse environment, operators are presented with a rich array of visual and auditory cues that mimic real-world sensory inputs. By experiencing first-hand the challenges and nuances of navigating crowded warehouse spaces, operators can develop a deeper understanding of how to effectively interact with their surroundings and make informed decisions in dynamic operational environments. Ultimately, Nuro's VR application represents a game-changing tool for training and enhancing the skills of forklift operators, paving the way for safer, more efficient warehouse operations in the era of autonomous vehicles.

7.9.2 Implementation aspects

Several implementation aspects have been carefully addressed in the development of Nuro's VR application for driving the iGo neo forklift within a virtual environment:

1. Virtual Environment Creation: Extensive efforts have been made to create a highly realistic 3D virtual warehouse environment. This involves meticulous attention to detail in modelling various elements such as aisles, shelves, pallets, and obstacles to accurately simulate real-world warehouse conditions.
2. User Interface Design: The user interface of the VR application has been meticulously designed to be intuitive and user-friendly. This includes menu navigation, control schemes, and interactive elements to ensure a seamless user experience.
3. Forklift Model Integration: The iGo neo forklift model from STILL has been integrated into the VR environment, ensuring accurate representation of the vehicle's size, dimensions, and

functionality. This involves importing 3D models, textures, and animations to create a lifelike simulation of the forklift.

4. AI Path Planning Integration: Sophisticated AI algorithms for path planning have been seamlessly integrated into the VR application. These algorithms enable the iGo neo forklift to navigate autonomously within the virtual environment, avoiding obstacles and adhering to predefined paths.
5. Sensor Simulation: The VR application simulates sensor data such as proximity sensors and cameras onboard the iGo neo forklift. This allows users to experience realistic feedback and perception understanding while operating the vehicle within the virtual environment.
6. Performance Optimization: Optimization techniques have been employed to ensure smooth performance and responsiveness of the VR application, even on lower-end hardware configurations. This includes efficient rendering techniques, level-of-detail optimization, and resource management strategies.
7. Training Scenarios Design: A variety of training scenarios have been designed to cater to different skill levels and learning objectives. These scenarios cover a range of tasks and challenges commonly encountered in real-world warehouse operations, providing users with a comprehensive training experience.

By addressing these implementation aspects, Nuro has ensured that its VR application for driving the iGo neo forklift offers a high-quality, immersive, and effective training experience for operators in warehouse environments.

7.9.3 Results

For the result of this development please refer Section 6.9.

7.9.4 IMOCO4.E requirements

Please refer to Section 6.9.12.

7.9.5 Capabilities & Limitations (including USP, strengths & weaknesses)

Nuro's VR application for driving the iGo neo forklift within a virtual environment offers several key capabilities and strengths, including a highly immersive and realistic simulation of warehouse operations, seamless integration of AI-driven path planning algorithms for autonomous navigation, and comprehensive training scenarios tailored to various skill levels. Its unique selling proposition (USP) lies in its ability to provide operators with hands-on experience in a safe and controlled environment, facilitating skill development and proficiency in operating the iGo neo forklift. However, the application may have limitations such as hardware requirements for VR setup, potential learning curve for users unfamiliar with virtual reality technology, and the inability to fully replicate the nuances of real-world operational challenges. Despite these limitations, Nuro's VR application represents a cutting-edge solution for enhancing training effectiveness and improving safety in warehouse environments.

7.9.6 Customizations & Adaptations (including possible modifications and extensions)

Nuro's VR application for driving the iGo neo forklift within a virtual environment offers ample opportunities for customizations and adaptations to suit the specific needs and preferences of users and organizations. One possible modification could involve the customization of training scenarios to simulate unique warehouse layouts, operational challenges, or specialized tasks relevant to a particular industry or company. Additionally, users may have the option to adapt the VR environment to mirror their own warehouse facilities, allowing for more realistic training experiences tailored to their specific operational requirements. Furthermore, the application could be extended to support additional forklift models or other types of warehouse equipment, providing users with a broader range of training opportunities and enhancing the versatility of the platform.

Moreover, Nuro's VR application can be adapted to integrate with existing training programs or learning management systems (LMS), allowing for seamless integration into organizational training workflows. This could involve features such as data synchronization with employee training records, progress tracking, and performance analytics to monitor skill development and proficiency levels over time. Additionally, the application could be customized to support multi-user collaboration, enabling team-based training exercises or virtual competitions to foster a collaborative learning environment. By offering customizable features and adaptable functionalities, Nuro's VR application empowers users to tailor their training experiences to their unique needs and objectives, ultimately enhancing the effectiveness and efficiency of their training programs.

7.9.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

In developing Nuro's VR application for driving the iGo neo forklift within a virtual environment, a robust methodology and toolchain were employed to ensure efficiency and effectiveness throughout the development process. The methodology involved a combination of agile development practices and iterative design cycles, allowing for continuous feedback and refinement of features based on user testing and stakeholder input. Furthermore, the integration of various tools and technologies played a crucial role in the development process, including 3D modelling software for creating the virtual environment, AI algorithms for path planning, and VR development platforms for building and deploying the application. However, it's important to acknowledge the limitations of these tools, such as the complexity of integrating AI algorithms and the learning curve associated with VR development platforms, which may have impacted development timelines and resource allocation.

Despite the challenges posed by tool limitations, the generic usability of Nuro's VR application was prioritized to ensure accessibility and ease of use for users with varying levels of technical expertise. This involved extensive user testing and interface design optimizations to streamline the user experience and minimize learning curves. Additionally, lessons learned throughout the development process were invaluable in informing future iterations of the application and identifying areas for improvement. By leveraging an iterative approach and incorporating user feedback, Nuro's VR application demonstrates a commitment to continuous improvement and innovation in the realm of virtual training solutions for warehouse operations.

7.10 Modelling and predictive maintenance of medical robot manipulator (PEN-PMS)

7.10.1 Tech Overview

Pilot 4 deals with a medical robotic manipulator that is used for minimally invasive image guided therapy. The device generates and logs messages about its operation, as well as possible error messages and warnings when the operation is not according to the specifications. Additionally, time series data about the positions, velocities, accelerations, motor currents, as well as their setpoint values in the motion control system are logged.

As part of Task 5.2 the data that is logged by the devices is regularly entered into tables in a database, where it can be accessed in a structured way. This facilitates continuous monitoring and development of machine learning models for predictive maintenance in Task 5.7.

7.10.2 Implementation aspects

The modelling toolchain uses SQL queries to the database to access the data, and python for data processing and model development. This gives flexibility in terms of the type of model that is used. Trained models are run at regular intervals on new data from the database.

7.10.3 Results

As part of the ETL development, the factory data and the data from the field for the same system are linked through the factory order number and the system code and serial number.

One type of model that PEN and PMS have developed in Pilot 4 is a machine learning model that relates to movement issues of the system along a specific axis. Such issues will require servicing the machine. For systems in the field, the customers' requests for service and the subsequent service actions are logged. This data is used to generate a label for each day and for each system. The period starting 30 days before a service request and ending at the completion of the service action is labelled 'bad', the other days are labelled 'good'.

Several message identifiers are chosen that correspond to movement along the specific axis, as well as other axes along which the system may move at the same time. For each day and each system, the number of occurrences of these message identifiers in the message logs is counted. These numbers, as well as the numbers for the week before, are chosen as the model features.

An extreme gradient boosting classifier (XGBoost) was trained on the features and labels of 137 systems; the data of 59 system was used as the test set to evaluate the model performance. At a false positive rate of 1 %, the true positive rate is 10.8 %. With these settings, the model raises an alert in the 30 days before a service request in 67 of the 128 cases for the 59 test systems.

A second type of model that was developed makes use of the time series data. As the software update that will enable the logging of this type of data has not been rolled out yet to the field, only the time series data of a handful of systems at PMS is currently available.

The model traces the difference between the actual position and the motion control system setpoint position for a specific axis and gives an alert when the error becomes larger than what is normal for a well-behaved system, but well before the error becomes problematic and system itself generates an error. Further development of this model and other model that make use of time series data will occur when more data becomes available.

7.10.4 IMOCO4.E requirements

Requirement		/L/BB/P-D-UC	Verification type (I/D/T/A)
ID	Description		
R134-D5.1-P4-1	Smart control algorithms, AI-components and digital twin models may use additional data or	P4	I

	sensory input interfaces to train the model, however after completion it shall only make use of existing data and interfaces of the brown field system.		
R136-D5.1-P4-3	All collected data from different sources (e.g., factory, field) but the same system shall contain a common unique identifier to enable linking of the data sources.	P4	I

In addition, the work addresses the following requirements from deliverable (D7.1)

PILOT 4	
ID	Requirement
Req-P4-001	Smart control algorithms, AI-components and digital twin models may use additional data or sensory input interfaces to train the model, however after completion it shall only make use of existing data and interfaces of the brown field system.
Req-P4-002	All smart control algorithms, AI-components and digital twin models shall be testable in both simulation (e.g., by means of digital twins) and deployed on the physical target.
Req-P4-003	Real-time (digital twin) models or algorithms require at maximum a data sample rate of 500 Hz.
Req-P4-004	Results from algorithms for condition monitoring and predictive maintenance shall have <5 % false positive detections.
Req-P4-005	All smart control algorithms, AI-components and digital twin models that are intended for real-time deployment shall be compatible with code generation from MATLAB / Simulink.
Req-P4-006	Any smart control algorithms, AI-components and digital twin models shall not adversely affect the safety of the system.

7.10.5 Capabilities & Limitations (including USP, strengths & weaknesses)

The availability of event logs and motion trace data from a worldwide install base makes it possible to develop predictive maintenance models, such as the models described above. Currently the number of systems that send motion trace data is limited, because the software update that will enable this for systems in the field is delayed. Nevertheless, a model that only uses daily error event counts can predict about half the number of service requests. Once the motion trace data becomes available at a larger scale, we can build models to predict the errors based on irregularities in the motion traces.

7.10.6 Customizations & Adaptations (including possible modifications and extensions)

The framework allows for flexibility in terms of the model development. For example, replacing the XGBoost model with another type, such as a logistic regression or a support vector machine, is a matter of importing and using the right model.

7.10.7 Methodology & Toolchains (including tool integration aspects, tool limitations, generic usability & lessons learnt)

Model development is done in python, and currently makes extensive use of the packages pandas, numpy, the machine learning package scikit-learn and the vertica-python package for connecting to and querying the database. Model performance depends on the quality of the labelling of the training data. In our event-based model, we used an implicit labelling of days based on the dates of the service requests. Human annotation might give better results but can be very labour intensive.

7.11 Framework for Digital Twin set-up (REDEN)

7.11.1 Tech Overview

A digital twin is in general used to enhance the performance of a product or system. For example, the system can become more accurate but also the increase of reliability can be seen an enhancement in performance. Equal to other aspects of a product or system, the digital twin needs to be developed/engineered. As a developer you want to test and optimize the digital twin well in advance any hardware is created. While for example the strength of a product can be analysed using finite element models, there is no specific tooling available for testing, optimizing, and developing a digital twin. We have created a framework for the development phase of a Digital Twin.

The developed framework is related to SW component SW-51 "Multi-body model parameters estimation using Bayesian filters".

7.11.2 Implementation aspects

The developed framework for a digital twin is focused on creating an environment to test and optimize the digital twin in the design phase. It is not intended to be used as part of the physical object. Therefore, implementation aspects are not considered.

7.11.3 Results

Introduction

Digital Twin is a highly relevant topic that has diverse applications. By predicting the future behaviour of a product, it can contribute to making well-informed decisions. Digital twins can be utilized for various purposes across different stages of a product's life cycle. This leads to a tailored approach that is specific to each individual case, rather than a generic one.

The idea behind the digital twin framework is that it can be used for multiple products, without requiring significant effort to set up. The main benefits of a general digital twin framework are accessibility and timesaving. The framework makes setting up a digital twin more accessible, as it does not require an in-depth understanding of digital twins to build one. Even individuals with limited knowledge of digital twins can utilize this framework.

Another critical advantage of the framework is that it allows users to focus solely on designing the technical aspects of the digital twin. Since the framework is already available, users only need to fill in the appropriate building blocks. This saves a considerable amount of time and ensures that the focus is on the essential aspects of the project.

Technical Solution

The starting point on developing the first concept of a digital twin framework is the 5D-DT concept, five-dimensional digital twin concept. This concept describes five main building blocks for a digital twin framework. This concept consists of services, data object, physical object, virtual object, and the connections. The idea is that the user can add services, i.e., simulation services or visualization services, create a data object and add the necessary initialization data, add a physical object, i.e., a .csv file with data, add a virtual object, i.e., prediction function and the connections which contains the communication and data transfer between blocks.

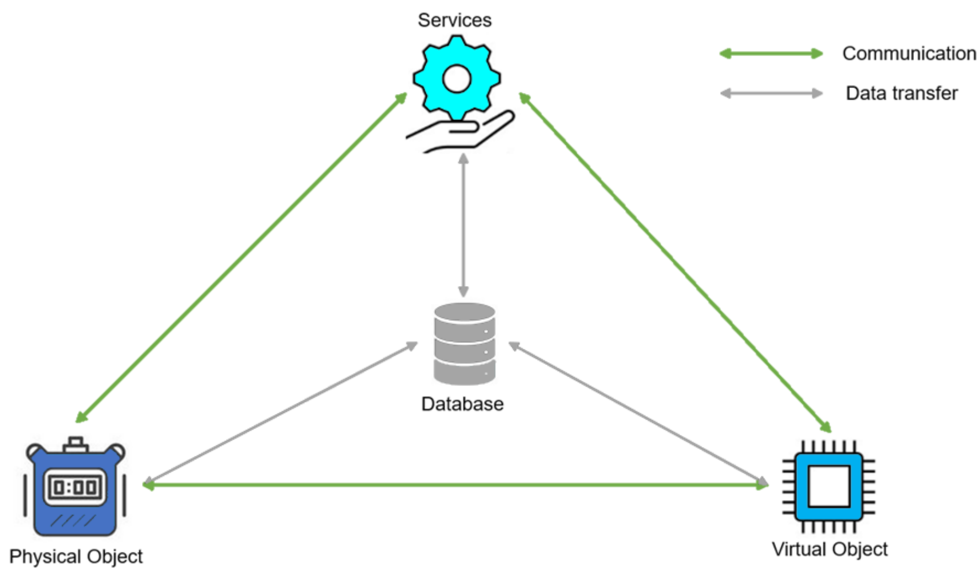


Figure 145: 5D-DT concept

Once the 5D-DT concept was established, we began putting it into practice. We started by incorporating Bayesian filters, specifically Particle filters and Kalman filters, as foundational services. Our aim was to devise an adaptable approach that seamlessly integrates these filters into the framework without causing disruptions to other components, such as physical objects, virtual objects, and initialization data.

It turned out that all Bayesian filters follow a similar operational pattern, with common steps like predict, update, and estimate, utilizing data such as state variables and measurement variables. This breakthrough allowed us to expand the framework by introducing additional services like visualization and data storage. Developers can easily add these services to the framework, providing users with pre-built functionalities without the need to start from scratch.

The outcome is a Python-compatible framework capable of handling multiple services. For instance, virtual objects require prediction and measurement functions, while physical objects can accept input from either a .csv file or a designated function. The framework takes care of managing connections and data flow, ensuring a user-friendly experience for both developers and end-users.

Experimental Analysis

The framework serves as a tool for creating digital twins for various applications, particularly for predicting factors like remaining useful lifetime. Currently, it is configured to operate on existing data and requires user initiation, lacking real-time data updates. However, the framework is designed to easily adapt to real-time functionality.

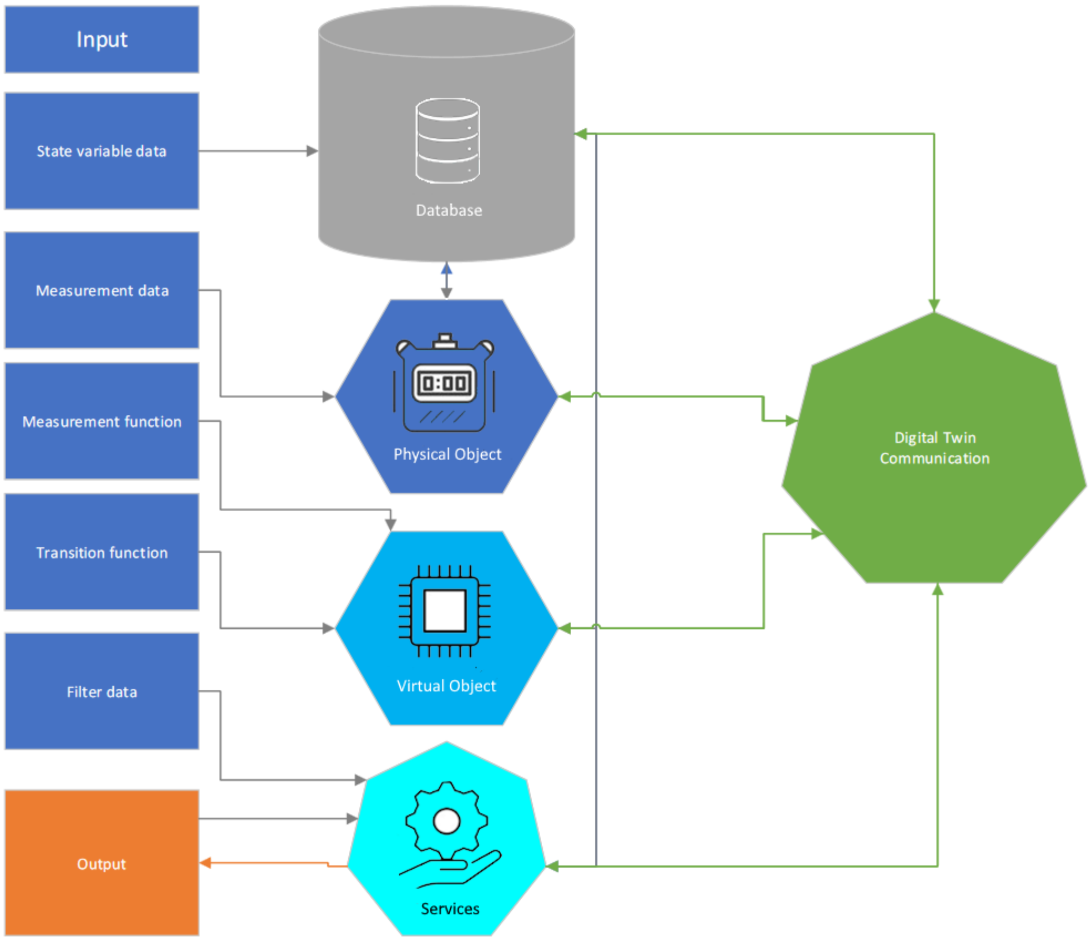


Figure 146: Overview of digital twin framework

To establish a digital twin using the framework, users must incorporate the blue blocks outlined in Figure 146. The naming conventions align with the data requirements of the filters. In Python, the framework library can be loaded and utilized to configure the digital twin. Once configured, the digital twin can be executed, generating estimates of states and measurements based on input data.

Future plans

The existing model displays considerable potential, yet it is in the early stages of development. The foundation is in place and functioning effectively. Prospective enhancements involve the creation of additional services, such as a prediction service, improved visualization services, and a service providing an overview of the digital twin's processes. Exploring the integration of new filters, as well as non-filter services like neural networks and machine learning models, is also on the agenda. A crucial consideration is whether the current framework's naming conventions and setup are applicable to such services.

To broaden accessibility, there is merit in exploring the development of a graphical user interface (GUI) for the digital twin framework. Additionally, a beneficial step forward would be to establish a direct link between the real physical object, sensors, and the physical object within the framework setup. This entails integrating real-time measurement data into the digital twin, with the virtual object performing real-time calculations using a simulation model, such as one from Abaqus.

7.11.4 Capabilities & Limitations (including USP, strengths & weaknesses)

Although the prediction aspect is not yet a built-in service and requires user implementation, several advantages and drawbacks are worth noting.

Pros:

- Simple setup for using the framework in Python.
- Facilitates the creation of a graphical user interface (GUI) based on the framework.
- Adaptable to handle real-time data with easy updates.

Cons:

- Currently exclusive to Python usage.
- Naming and setup may be overly focused on Bayesian filter principles.
- Real-time data handling is not supported now.
- Lacks a generalized prediction service.
- Limited possibilities currently; expansion is necessary.

In general, the framework helps the developer of the digital twin by providing a well-defined structure which is indispensable in the development of complex digital twins. All connections between the different dimensions of the digital twin are predefined and a set of building blocks is available.

7.11.5 Customizations & Adaptations (including possible modifications and extensions)

The framework is built in such a way that it is very generic. The modular structure gives the user the possibility to easily add features and extend the functionality. Now the framework can be called within Python but calls from other software languages is also possible.

8. Conclusion

This document describes the achievements within work package 5 of IMOCO4.E project – Digital twins and their interaction with the cloud. This WP aimed to prepare the background in terms of developed components, tools and approaches for wider utilization of digital twins in the industry. The presented results often take advantage of AI approaches as they can handle highly complex tasks.

WP5 is tightly related to the preparation of building blocks BB6 and BB9. BB6 contains algorithms for condition monitoring, predictive maintenance, and self-commissioning of industrial motion control systems. Its components are described in chapters 3, 4 and 7. BB9 is about cyber-security tools and trustworthy data management. This topic is dealt with in Chapter 2, which covers real-time data communication, storage, processing, and visualization. It also deals with cyber-security protection. BB8 targets the integration of AI on different hardware platforms, which was primarily realized in WP3. This document targets more AI algorithms rather than their implementation. Therefore, BB8 is addressed here rather marginally. Chapter 5 is dedicated to modeling aspects of digital twins. Dealing with complex multidomain models with adaptive features running in real-time is a real challenge, as demonstrated in several examples. Chapter 6 describes mainly the utilization of augmented and virtual reality concepts as human-machine interfaces. It tightly relates to the previous section about proper modelling and, of course, the depreciation of the latencies to make the digital twin look and feel like it is alive.

This document was preceded by the set of confidential deliverables (D5.3), (D5.4), (D5.5), (D5.6) and (D5.7) where the technical details were presented per individual tasks of WP5.

This document is closing the activities in WP5. Developed components, tools, and approaches will be further used in WP6, where they are integrated and tested, and in WP7, where the main aim is to prove their functionality when integrated into pilot and demonstrator applications.

9. References

(Achouch et al., 2022) Achouch, M., Dimitrova, M., Ziane, K., Sattarpanah Karganroudi, S., Dhouib, R., Ibrahim, H., & Adda, M. (2022). On Predictive Maintenance in Industry 4.0: Overview, Models, and Challenges. Applied Sciences, 12(16), 8081. https://doi.org/10.3390/app12168081

(Al-amri et al., 2021) Al-amri, R., Murugesan, R. K., Man, M., Abdulateef, A. F., Al-Sharafi, M. A., & Alkahtani, A. A. (2021). A Review of Machine Learning and Deep Learning Techniques for Anomaly Detection in IoT Data. Applied Sciences, 11(12), 5320. https://doi.org/10.3390/app11125320

(Algolia, 2024) Algolia website [Online]. Available: https://www.algolia.com/ [Accessed 02/2024]

(Altimore et al., 2023) Altimore, P., w-azure, Gremban, K., Meadows, P., Arya, H., Oumghar, K., Street, C., Shahan, R. , Vaes, K. and Depickere, G. (February, 2023). How an IoT Edge device can be used as a gateway, Available: https://learn.microsoft.com/en-us/azure/iot-edge/iot-edge-as-gateway?view=iotedge-1.4#Transparent%20gateways [Accessed 02/2024]

(Amazon CloudSearch, 2024) Amazon CloudSearch website [Online]. Available: https://aws.amazon.com/cloudsearch/ [Accessed 02/2024]

(Amazon Kinesis Data Streams, 2024) What is Amazon Kinesis Data Stream? https://docs.aws.amazon.com/streams/latest/dev/introduction.html [Accessed 02/2024]

(Apache Kafka clients, 2024) Apache Kafka clients [Online]. Available: https://cwiki.apache.org/confluence/display/KAFKA/Clients [Accessed 02/2024]

(Apache Kafka documentation, 2024) Apache Kafka documentation [Online]. Available: https://kafka.apache.org/documentation/ [Accessed 02/2024] (Amazon Kinesis Data Streams, 2024) Amazon Kinesis Data Streams website [Online]. Available: https://aws.amazon.com/kinesis/data-streams/ [Accessed 02/2024]

(Apache Kafka Performance) Apache Kafka Performance: Benchmark testing and results for Apache Kafka's performance on the latest hardware in the cloud [Online]. Available: https://developer.confluent.io/learn/kafka-performance/ [Accessed 02/2024]

(Apache Kafka website, 2024) Apache Kafka website [Online]. Available: https://kafka.apache.org/ [Accessed 02/2024]

(Apache Pulsar, 2024) Apache Pulsar website [Online]. Available: https://pulsar.apache.org/ [Accessed 02/2024]

(Apache Solr, 2024) Apache Solr website [Online]. Available: https://solr.apache.org/ [Accessed 02/2024]

(Atlassian Bitbucket, 2024) Atlassian Bitbucket website [Online]. Available: https://bitbucket.org/product [Accessed 02 2024].

(Atlassian Jira, 2024) Atlassian Jira website [Online]. Available: https://www.atlassian.com/software/jira [Accessed 02 2024].

(Auth0, 2024) Auth0 website [Online]. Available: https://auth0.com/ [Accessed 02/2024]

(AWS IAM, 2024) AWS Identity and Access Management website [Online]. Available: https://aws.amazon.com/iam/ [Accessed 02/2024]

(Azure Cognitive Search, 2024) Azure Cognitive Search website [Online]. Available: https://azure.microsoft.com/en-us/products/ai-services/ai-search/ [Accessed 02/2024]

(Basora et al., 2019) Basora, L., Olive, X., & Dubot, T. (2019). Recent Advances in Anomaly Detection Methods Applied to Aviation. Aerospace, 6(11), 117. https://doi.org/10.3390/aerospace6110117

(Bergmann et al., 2019) Bergmann, P., Fauser, M., D. and Steger, N. "A Comprehensive Real-World Dataset for Unsupervised Anomaly Detection", IEEE Conference on Computer Vision and Pattern Recognition, 2019.

(Berman et al., 2019) Berman, D., Buczak, A., Chavis, J., & Corbett, C. (2019). A Survey of Deep Learning Methods for Cyber Security. Information, 10(4), 122. https://doi.org/10.3390/info10040122

(Çorak et al., 2018) Çorak, Burak & Okay, Feyza & Güzel, Metehan & Murt, Sahin & Ozdemir, Suat. (2018). Comparative Analysis of IoT Communication Protocols. 1-6. 10.1109/ISNCC.2018.8530963

(D2.3) D2.3 (2022) Overall requirements on IMOCO4.E reference framework. Public deliverable of IMOCO4.E project.

(D3.2) D3.2 (2022) Perception and instrumentation Layer requirements and specifications (final iteration). Public deliverable of IMOCO4.E project.

(D3.3) D3.3 (2023) Novel low/self powered real-time sensors (BB3). Confidential demonstrator of IMOCO4.E project.

(D3.4) D3.4 (2023) New SoC+FPGA and multi-many core platforms for AI and smart data processing (BB1). Confidential demonstrator of IMOCO4.E project.

(D3.5) D3.5 (2023) AI based high speed perception and vision (BB2, BB8). Confidential demonstrator of IMOCO4.E project.

(D3.7) D3.7 (2024) Final design report on Perception & Instrumentation Layer. Public deliverable of IMOCO4.E project.

(D4.8) D4.8 (2024) Final design report on advance control layer development final report. Public deliverable of IMOCO4.E project.

(D5.1) D5.1 (2022) Integral (system level) requirements for valuable twinning methods (first iteration). Public deliverable of IMOCO4.E project.

(D5.2) D5.2 (2023) Integral (system level) requirements for valuable twinning methods (second iteration). Public deliverable of IMOCO4.E project.

(D5.3) D5.3 (2023) Trustworthy and Secure Dataset management, storage and processing tools. Confidential demonstrator of IMOCO4.E project.

(D5.4) D5.4 (2023) Algorithms for condition monitoring, predictive maintenance and self-commissioning of industrial motion control systems. Confidential demonstrator of IMOCO4.E project.

(D5.5) D5.5 (2023) Modelling and simulation of complex multi-axis systems. Confidential demonstrator of IMOCO4.E project.

(D5.6) D5.6 (2023) Augmented and virtual reality through digital twins. Confidential report of IMOCO4.E project.

(D5.7) D5.7 (2023) AI methods for monitoring and predictive maintenance at higher IMOCO4.E layers. Confidential demonstrator of IMOCO4.E project.

(D6.1) D6.1 (2022) Guideline of IMOCO4.E methodology and toolchains. Public deliverable of IMOCO4.E project.

(D6.4) D6.4 (2023) Validation reports (first iteration). Confidential deliverable of IMOCO4.E project.

(D6.6) D6.6 (2024) Confidential deliverable of IMOCO4.E project.

(D7.1) D7.1 (2022) Definition of the pilots. Confidential deliverable of IMOCO4.E project.

(D7.4) D7.4 (2023). Pilot 2: integration and validation report (initial version). Confidential deliverable of IMOCO4.E project.

(D7.10) D7.10 (2024). Pilot 3: integration and validation report (final version). Confidential demonstrator of IMOCO4.E project.

(Datadog, 2024) Datadog website [Online]. Available https://www.datadoghq.com/ [Accessed 02/2024]

(Deb et al., 2002) Deb, K., Pratap, A., Agarwal, S. and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation, 6(2), pp.182–197.

(Dobbelaere & Sheykh, 2017) Dobbelaere, P. & Sheykh E. K. (2017). Kafka versus RabbitMQ: A comparative study of two industry reference publish/subscribe implementations: Industry Paper. 227-238. 10.1145/3093742.3093908.

(Docker website, 2024) Docker website [Online]. Available: https://www.docker.com/ [Accessed 02 2024].

(D'silva et al., 2017) D'silva, G. M. A., Khan, G. and Bari, S. "Real-time processing of IoT events with historic data using Apache Kafka and Apache Spark with dashing framework," 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), 2017, pp. 1804-1809, doi: 10.1109/RTEICT.2017.8256910.

(Eclipse Mosquitto, 2024) Eclipse Mosquitto website [Online]. Available: https://mosquitto.org/ [Accessed 02/2024]

(ElasticSearch Clients, 2024) ElasticSearch Clients [Online]. Available: https://www.elastic.co/guide/en/elasticsearch/client/index.html [Accessed 02 2024]

(Elastic website, 2024) Elastic website [Online]. Available: https://www.elastic.co/ [Accessed 02/2024]

(FreeIPA, 2024) FreeIPA website [Online]. Available: https://www.freeipa.org/ [Accessed 02/2024] (Grafana, 2024) Grafana website [Online]. Available https://grafana.com/ [Accessed 02/2024]

(Gluu, 2024) Gluu website [Online]. Available: https://gluu.org/ [Accessed 02/2024]

(Google Cloud Pub/Sub, 2024) Google Cloud Pub/Sub website [Online]. Available: https://cloud.google.com/pubsub [Accessed 02/2024]

(Goubej et al., 2023) Goubej, M., Tvrz, J., Kubeš, B., Robust and optimal design of fixed structure controllers in collocated motion systems. In 2023 IEEE 28th International Conference on Emerging Technologies and Factory Automation (ETFA),

(Grafana, 2024) Grafana Labs – The open observability platform. https://grafana.com/ [Accessed 02/2024]

(Howey & van der Anker, 2023) Howey, A. & van der Anker, F. (2023). 'User Experience of Automated Guided Vehicles', 'Bericht zum 69. Arbeitswissenschaftlichen Kongress vom 01. – 03. März 2023', GfA-Press, Sankt Augustin

(Hyperopt, 2024) Hyperopt website [Online]. Available: https://hyperopt.github.io/hyperopt/ [Accessed 02 2024].

(Chalapathy et al., 2019) Chalapathy R. and Chawla S., (2019), "Deep Learning for Anomaly Detection: A Survey," arXiv e-prints, p. arXiv:1901.03407.

(InfluxDB, 2024) InfluxDB website [Online]. Available: https://opensearch.org/ [Accessed 02/2024]

(Instana, 2024) Avoid costly downtime by using IBM Instana. https://www.ibm.com/products/instana [Accessed 02 2024].

(IoRT, 2024) Internet of Robotic Things (IoRT) https://ieeexplore.ieee.org/abstract/document/7805273 [Accessed 02/2024]

(IoS, 2024) Internet of Skills (IoS) https://www.sciencedirect.com/science/article/abs/pii/B9780128213438000149#:~:text=The%20Internet%20of%20Skills%20will,acquisition%20and%20performance%20of%20skills. [Accessed 02/2024]

(ISO18738, 2003) ISO 18738:2003: Lifts (elevators) - Measurement of lift ride quality. 1st edition, Switzerland. August 2003.

(ISO2631, 1997) ISO 2631-1:1997: Mechanical vibration and shock, Evaluation of human exposure to whole-body vibration. 2nd edition, Switzerland, July 2021.

(ISO8041, 2017) ISO 8041-1: Human response to vibration Measuring instrumentation Part 1: General purpose vibration meters.1st edition, Switzerland. May 2017.

(Jimenez-Cortadi et al., 2019) Jimenez-Cortadi, A., Irigoien, I., Boto, F., Sierra, B., & Rodriguez, G. (2019). Predictive Maintenance on the Machining Process and Machine Tool. Applied Sciences, 10(1), 224. https://doi.org/10.3390/app10010224

(Kang et al., 2021) Zhuangwei Kang, Robert Canady, Abhishek Dubey, Aniruddha Gokhale, Shashank Shekhar, and Matous Sedlacek. 2021. A Study of Publish/Subscribe Middleware Under Different IoT Traffic Conditions. In Proceedings of the International Workshop on Middleware and Applications for the Internet of Things (M4IoT'20). Association for Computing Machinery, New York, NY, USA, 7–12. https://doi.org/10.1145/3429881.3430109

(Kelly, 1997) Kelly, R. (1997). PD Control with Desired Gravity Compensation of Robotic Manipulators. The International Journal of Robotics Research, 16(5), pp.660–672. doi:https://doi.org/10.1177/027836499701600505.

(Kibana, 2024) Kibana website [Online]. Available https://www.elastic.co/kibana/ [Accessed 02/2024]

(Keras, 2024) Keras website [Online]. Available: https://keras.io/ [Accessed 02 2024].

(Lee et al., 2017) Lee, C., Kwak, S., Kwak, J. and Oh, S. (2017). Generalization of Series Elastic Actuator Configurations and Dynamic Behavior Comparison. Actuators, 6(3), p.26. doi:https://doi.org/10.3390/act6030026.

(Lee et al., 2020) Lee, J., Ni, J., Singh, J., Jiang, B., Azamfar, M., and Feng, J. (2020). "Intelligent Maintenance Systems and Predictive Manufacturing." ASME. J. Manuf. Sci. Eng. November 2020; 142(11): 110805. https://doi.org/10.1115/1.4047856

(Lee & Hur, 2017) Lee, S.T. and Hur, J., 2017, May. Simplified equivalent model of PMSM for analyzing influence of inter-turn fault on motor characteristics. In 2017 IEEE International Electric Machines and Drives Conference (IEMDC) (pp. 1-6). IEEE.

(Li et al., 2023) Li, J., Rakin, A. S., Chen, X., Yang, L., He, Z., Fan, D., & Chakrabarti, C. (2023, March 13). Model Extraction Attacks on Split Federated Learning. arXiv.org. Retrieved August 11, 2023, from https://arxiv.org/abs/2303.08581

(Librdkafka, 2024) Librdkafka - the Apache Kafka C/C++ client library. Available: https://github.com/confluentinc/librdkafka [Accessed 02/2024]

(Maktoubian et al., 2021) Maktoubian, J., Taskhiri, M. S., & Turner, P. (2021). Intelligent Predictive Maintenance (IPdM) in Forestry: A Review of Challenges and Opportunities. Forests, 12(11), 1495. https://doi.org/10.3390/f12111495

(Malhotra et al., 2015) Malhotra, P., Vig, L., Shroff, G.M., & Agarwal, P, (2015), "Long Short Term Memory Networks for Anomaly Detection in Time Series", in The European Symposium on Artificial Neural Networks, Computational Intelligence, Bruges, Belgium, 2015

(Malhotra et al., 2016) Malhotra P., Ramakrishnan A., Anand G., Vig L., Agarwal P. and Shroff G., (2016) "LSTM-based Encoder-Decoder for Multi-sensor Anomaly Detection," arXiv e-prints, p. arXiv:1607.00148.

(Maruhn, 2021) Maruhn, P. (2021). ‚VR Pedestrian Simulator Studies at Home: Comparing Google Cardboards to Simulators in the Lab and Reality‘. Frontiers in Virtual Reality, 2:746971. Available at: DOI: 10.3389/frvir.2021.746971.

(Maruhn, 2022) Maruhn, P., Hurst, S. (2022). Effects of Avatars on Street Crossing Tasks in Virtual Reality. In: Black, N.L., Neumann, W.P., Noy, I. (eds) Proceedings of the 21st Congress of the International Ergonomics Association (IEA 2021). IEA 2021. Lecture Notes in Networks and Systems, vol 223. Springer, Cham. https://doi.org/10.1007/978-3-030-74614-8_26

(Microsoft Azure Event Hubs, 2024) Microsoft Azure Event Hubs website [Online]. Available: https://azure.microsoft.com/en-us/products/event-hubs [Accessed 02/2024]
(Microsoft Entra ID, 2024) Microsoft Entra ID website [Online]. Available: https://www.microsoft.com/en-us/security/business/identity-access/microsoft-entra-id [Accessed 02/2024]
(MLflow, 2024) MLflow website [Online]. Available: https://mlflow.org/ [Accessed 02 2024].
(MQTT, 2024) MQTT website [Online]. Available: https://mqtt.org/ [Accessed 02/2024]
(Nayar, 2019) Nayar, N.U., 2019. Model-based fault detection and diagnosis of BLDC motors working at variable speed using wavelet transform (Doctoral dissertation).

(NumPy, 2024) NumPy website [Online]. Available: https://numpy.org/ [Accessed 02 2024].

(NVIDIA, 2024) NVIDIA Jetson Benchmarks available: https://developer.nvidia.com/embedded/jetson-benchmarks [02/2024]

(Okta, 2024) Okta website [Online]. Available: https://www.okta.com/ [Accessed 02/2024]
(Olah, 2015) Olah, C., (2015), "Understanding LSTM Networks," cola's blog (entry posted on August 27, 2015) [Online]. Available: https://colah.github.io/posts/2015-08-Understanding-LSTMs/. [Accessed 04 2023].

(OpenSearch, 2024) OpenSearch website [Online]. Available: https://opensearch.org/ [Accessed 02/2024]
(Palmese et al., 2021) F. Palmese, E. Longo, A. E. C. Redondi and M. Cesana, "CoAP vs. MQTT-SN: Comparison and Performance Evaluation in Publish-Subscribe Environments," 2021 IEEE 7th World Forum on Internet of Things (WF-IoT), New Orleans, LA, USA, 2021, pp. 153-158, doi: 10.1109/WF-IoT51360.2021.9595725.
(pandas, 2024) pandas website [Online]. Available: https://pandas.pydata.org/ [Accessed 02 2024].

(Pang et al., 2021) G. Pang, G., Shen, C., Cao, L. and van den Hengel, A. (2021), "Deep Learning for Anomaly Detection: A Review," ACM Comput. Surv. 54, 2, Article 38 (March 2022), 38 pages. https://doi.org/10.1145/3439950

(Pech et al., 2021) Pech, M., Vrchota, J., & Bednář, J. (2021). Predictive Maintenance and Intelligent Sensors in Smart Factory: Review. Sensors, 21(4), 1470. https://doi.org/10.3390/s21041470

(Phuttharak & Loke, 2023) Phuttharak, J., & Loke, S. W. (2023). An Event-Driven Architectural Model for Integrating Heterogeneous Data and Developing Smart City Applications. Journal of Sensor and Actuator Networks, 12(1), 12. https://doi.org/10.3390/jsan12010012
(PingIdentity, 2024) PingIdentity website [Online]. Available: https://www.pingidentity.com/ [Accessed 02/2024]
(Poór et al., 2019a) Poór, P., Ženíšek, D. & Basl, J. (2019). Historical Overview of Maintenance Management Strategies: Development from Breakdown Maintenance to Predictive Maintenance in Accordance with Four Industrial Revolutions. In: Proceedings of the International Conference on Industrial Engineering and Operations Management. Plzeň: IEOM Society International, 2019. s. 495-504. ISBN 978-1-5323-5949-1 , ISSN 2169-8767.

(Poór et al., 2019b) Poór, P.; Basl, J.; Zenisek, D. (2019) "Predictive Maintenance 4.0 as next evolution step in industrial maintenance development,", International Research Conference on Smart Computing and Systems Engineering (SCSE), Colombo, Sri Lanka, 2019, pp. 245-253, doi: 10.23919/SCSE.2019.8842659.

(PyCaret, 2024) PyCaret website [Online]. Available: https://pycaret.org/ [Accessed 02 2024].

(Quix, 2024) Quix website [Online]. Available: https://quix.io/ [Accessed 02 2024].

(Redpanda, 2024) Redpanda website [Online]. Available: https://redpanda.com/ [Accessed 02/2024]

(Rueckert et al., 2017) Rueckert, E., Moritz Nakatenus, Samuele Tosatto and Peters, J. (2017). Learning inverse dynamics models in O(n) time with LSTM networks. doi:https://doi.org/10.1109/humanoids.2017.8246965.

(NATS, 2024) NATS website [Online]. Available: https://nats.io/ [Accessed 02/2024]

(Ran et al., 2019) Ran, Y.; Zhou, X.; Lin, P.; Wen, Y.; Deng, R. (2019), "A Survey of Predictive Maintenance: Systems, Purposes and Approaches". arXiv 2019, arXiv:1912.07383.

(Relic, 2024) Relic website [Online]. Available https://newrelic.com/ [Accessed 02/2024]

(Rodriguez et al., 2015) Rodriguez, P. , Sahoo, S., Pinto, C. T. and Sulowicz, M. "Field current signature analysis for fault detection in synchronous motors," 2015 IEEE 10th International Symposium on Diagnostics for Electrical Machines, Power Electronics and Drives (SDEMPED), Guarda, Portugal, 2015, pp. 246-252, doi: 10.1109/DEMPED.2015.7303697.

(Schneider & Bengler, 2020) Schneider, S., & Bengler, K. (2020). 'Virtually the same? Analysing pedestrian behaviour by means of virtual reality'. Transportation research part F: traffic psychology and behaviour, 68, 231-256.
DOI: 10.1016/j.trf.2019.11.005

(Schwind et al., 2019) Schwind, V. et al. (2019). 'Using Presence Questionnaires in Virtual Reality'. CHI 2019. Available at: https://doi.org/10.1145/3290605.3300590

(Signoz, 2024) Signoz website [Online]. Available https://signoz.io/ [Accessed 02/2024]

(Shi et al., 2023) Shi, J., Wan, W., Hu, S., Lu, J., & Zhang, L. Y. (2022, October 6). Challenges and Approaches for Mitigating Byzantine Attacks in Federated Learning. arXiv.org. Retrieved August 11, 2023, from https://arxiv.org/abs/2112.14468

(Sphinx Search, 2024) Sphinx Search website [Online]. Available: https://sphinxsearch.com/ [Accessed 02/2024]

(Splunk, 2024) Splunk website [Online]. Available https://www.splunk.com/ [Accessed 02/2024]

(Splunk, 2024) Instana website [Online]. Available https://www.ibm.com/products/instana/ [Accessed 02/2024]

(TensorFlow, 2024) TensorFlow website [Online]. Available: https://www.tensorflow.org/ [Accessed 02 2024].

(Tiddens et al., 2022) Tiddens, W., Braaksma, J. and Tinga, T. (2022), "Exploring predictive maintenance applications in industry", Journal of Quality in Maintenance Engineering, Vol. 28 No. 1, pp. 68-85. https://doi.org/10.1108/JQME-05-2020-0029

(UNSW-NB15, 2024) The UNSW-NB15 Dataset [Online]. Available: https://research.unsw.edu.au/projects/unsw-nb15-dataset [Accessed 02/2024]

(UT1, 2024) Unity-Technologies. (n.d.). GitHub - Unity-Technologies/Unity-Robotics-Hub: Central repository for tools, tutorials, resources, and documentation for robotics simulation in Unity. GitHub. https://github.com/Unity-Technologies/Unity-Robotics-Hub, [Accessed: 25.02.2024].

(UT2, 2024) Unity-Technologies. (n.d.-a). GitHub - Unity-Technologies/ROS-TCP-Connector. GitHub. https://github.com/Unity-Technologies/ROS-TCP-Connector, [Accessed: 25.02.2024].

(Villalba & Carrera, 2019) Villalba, Á., Carrera, D. (2019). Multi-tenant Pub/Sub Processing for Real-Time Data Streams. In: , et al. Euro-Par 2018: Parallel Processing Workshops. Euro-Par 2018. Lecture Notes in Computer Science(), vol 11339. Springer, Cham. https://doi.org/10.1007/978-3-030-10549-5_20

(Yaacoub et al., 2020) Yaacoub, J.-P. A., Salman, O., Noura, H. N., Kaaniche, N., Chehab, A. and Malli, M. "Cyber-physical systems security: Limitations, issues and future trends," Microprocessors and microsystems, vol. 77, p. 103201, 2020.

(VDMA, 2023) VDMA (2023). 'Design of visible and audible signals of driverless industrial trucks'. Available at: http://bit.ly/3vsqIKa

(Wiczorek & Protzak, 2022) Wiczorek, R., Protzak, J. (2022). 'Evaluation of an assistance system supporting older pedestrians' road crossing in virtual reality and in a real-world field test'. Front. Psychol. 13:966096. doi: 10.3389/fpsyg.2022.966096

(WSO2 Identity Server, 2024) WSO2 Identity Server website [Online]. Available: https://wso2.com/identity-and-access-management/ [Accessed 02/2024]

(Zabbix, 2024) Zabbix website [Online]. Available https://www.zabbix.com/ [Accessed 02/2024]

(Zhu et al., 2023) Zhu, Y., Wen, H., Zhao, R., Jiang, Y., Liu, Q., & Zhang, P. (2023). Research on Data Poisoning Attack against Smart Grid Cyber–Physical System Based on Edge Computing. Sensors, 23(9), 4509. MDPI AG. Retrieved from http://dx.doi.org/10.3390/s23094509