

# MODAPTO [101091996]: Modular Manufacturing and Distributed Control via Interoperable Digital Twins



## D6.1 - Specifications and Architecture v1.0

<b>Project Reference No</b>	MODAPTO - 101091996
<b>Deliverable</b>	D6.1 - Specifications and Architecture v1.0
<b>Work package</b>	WP6 - Deployment & Integration
<b>Type</b>	R - Document, report
<b>Dissemination Level</b>	PU - Public (fully open)
<b>Submission Date</b>	29/02/2024
<b>Editor(s)</b>	Spiros Fotis Jr. (AEGIS) Evangelos Loutsetis (AEGIS) Stella Markopoulou (AEGIS) Maria Zoidi (AEGIS)
<b>Contributor(s)</b>	Sakis Dalianis (ATC), George Triantafyllou (ATC), Sotiria Antaranian (ATC), Michael Jacoby (Fraunhofer), Damiano Falcioni (BOC), Alexandre Voisin (UL), Miriam Schleipen (EKS), Thorsten Schmitz (EKS), Stathis Plitsos (UPRC), Pavlos Eirinakis (UPRC), Timoleon Farmakis(AUEB)
<b>Reviewer(s)</b>	Stavros Lounis (AUEB) Miriam Schleipen (EKS)
<b>Version</b>	1.0

## Disclaimer

The MODAPTO project is funded by the European Union under grant agreement ID 101091996. The information and views set out in this publication are those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Executive Agency (REA). Neither the European Union nor the granting authority can be held responsible for them.

## Document Revision History

Version	Date	Modifications Introduced	
		Modification Reason	Modified by
0.1	27/04/2023	Draft ToC	AEGIS
0.2	24/08/2023	Revised Draft ToC	AEGIS
0.3	22/11/2023	Revised Draft ToC	AEGIS
0.4	06/12/2023	Revised Draft ToC	AEGIS
0.5	20/12/2023	ToC with allocation for Input	AEGIS
0.6	02/01/2024	Input EKS	EKS
0.7	11/01/2024	Input from Fraunhofer, BOC and AEGIS	Fraunhofer, BOC and AEGIS
0.8	17/01/2024	Input from ATC, Fraunhofer, BOC, UPRC, UL, EKS and AEGIS	ATC, Fraunhofer, BOC, UPRC, UL, EKS and AEGIS
0.9	24/01/2024	Input from ATC, BOC, Fraunhofer and AEGIS – 2 <sup>nd</sup> Draft Version	ATC, BOC, Fraunhofer and AEGIS
0.91	31/01/2024	Input from ATC, BOC, AEGIS and UL – 3 <sup>rd</sup> Draft	ATC, BOC, AEGIS, UL
0.92	08/02/2024	Input from UPRC, ATC, AEGIS	UPRC, AEGIS, ATC
0.93	13/02/2024	Input from AEGIS, Fraunhofer, ATC, BOC	AEGIS, Fraunhofer, ATC, BOC
0.94	19/2/2024	Input AEGIS, UL, EKS	AEGIS, UL, EKS
0.95	21/2/2024	Input BOC	AEGIS, BOC
0.96	28/2/2024	Internal review from AUEB	AUEB
0.97	28/2/2024	Internal review from EKS	EKS
1.0	29/2/2024	Final revision from AEGIS	AEGIS



## Executive Summary

The MODAPTO project is an innovative project that seeks to revolutionize modular manufacturing by deploying interoperable digital twins. This ambitious initiative aims to foster a dynamic and adaptable manufacturing ecosystem responsive to evolving market demands and technological progressions. Deliverable D6.1 is a pivotal component of this project, laying the groundwork for the envisioned framework. It meticulously outlines the technical specifications and the architectural blueprint, ensuring the system's design is robust, scalable, and aligned with industry standards.

Central to D6.1 is the architectural framework integrating novel and innovative digital twin technology into modular manufacturing processes. This approach enhances operational efficiency and allows for greater flexibility in responding to changes. The document details the structural elements of the system, focusing on interoperability, scalability, and adaptability. It explains how these elements collectively contribute to a more efficient and agile manufacturing process, aligning with MODAPTO's core objective of creating a transformative manufacturing ecosystem.

Looking ahead, D6.1 establishes a solid foundation for subsequent phases of the MODAPTO project. It paves the way for seamless integration with other work packages, setting the stage for continuous development and implementation of advanced manufacturing solutions. This deliverable is a blueprint for the current phase and a strategic guide for future enhancements, ensuring that the MODAPTO project remains at the forefront of manufacturing innovation.



# Table of Contents

<b>1</b>	<b>INTRODUCTION.....</b>	<b>9</b>
1.1	PURPOSE AND SCOPE.....	9
1.2	RELATION TO OTHER WORK PACKAGES, TASKS AND DELIVERABLES.....	9
1.3	STRUCTURE OF THE DELIVERABLE.....	10
<b>2</b>	<b>TECHNICAL REQUIREMENTS .....</b>	<b>11</b>
2.1	PRIORITIZATION OF USER REQUIREMENTS .....	11
2.2	TECHNICAL REQUIREMENTS.....	13
2.2.1	<i>Group 1: Authentication and Authorization.....</i>	<i>14</i>
2.2.2	<i>Group 2: User Interface.....</i>	<i>14</i>
2.2.3	<i>Group 3: Production and Digital Twin Management.....</i>	<i>15</i>
2.2.4	<i>Group 4: MODAPTO System Database and Digital Twins.....</i>	<i>15</i>
2.2.5	<i>Group 5: Development (and implementation) of Smart Services .....</i>	<i>16</i>
2.2.6	<i>Group 6: Security and Infrastructure.....</i>	<i>17</i>
2.2.7	<i>Group 7: System Design, Integration, Monitoring &amp; Compatibility .....</i>	<i>17</i>
2.2.8	<i>Group 8: Reporting, Alerts &amp; User Support.....</i>	<i>17</i>
2.2.9	<i>Group 9: APIs and Data Exchange .....</i>	<i>18</i>
<b>3</b>	<b>MODAPTO ARCHITECTURE .....</b>	<b>19</b>
3.1	BACKGROUND KNOWLEDGE .....	19
3.2	CONCEPTUAL ARCHITECTURE .....	20
3.2.1	<i>Access Control Service .....</i>	<i>23</i>
3.2.2	<i>API.....</i>	<i>23</i>
3.2.3	<i>Catalogue Service .....</i>	<i>23</i>
3.2.4	<i>Data Access Interface .....</i>	<i>23</i>
3.2.5	<i>Data Access Service .....</i>	<i>23</i>
3.2.6	<i>Digital Twin Service .....</i>	<i>23</i>
3.2.7	<i>MODAPTO Module API.....</i>	<i>25</i>
3.2.8	<i>Modularity Services.....</i>	<i>25</i>
3.2.9	<i>Module &amp; DT Management Service.....</i>	<i>25</i>
3.2.10	<i>Non-AAS DT Onboarding Service.....</i>	<i>25</i>
3.2.11	<i>Notification Service .....</i>	<i>26</i>
3.2.12	<i>Service Composition Service .....</i>	<i>26</i>
3.2.13	<i>UI.....</i>	<i>26</i>
3.3	FROM CONCEPTUAL ARCHITECTURE TO HIGH LEVEL TECHNICAL ARCHITECTURE .....	26
3.4	MODAPTO HIGH LEVEL TECHNICAL ARCHITECTURE .....	27
3.4.1	<i>Access Control.....</i>	<i>31</i>
3.4.2	<i>Analytics.....</i>	<i>32</i>
3.4.3	<i>API Gateway .....</i>	<i>32</i>
3.4.4	<i>Data Access Interface .....</i>	<i>32</i>
3.4.5	<i>Digital Twin.....</i>	<i>33</i>
3.4.6	<i>DT Management.....</i>	<i>33</i>
3.4.7	<i>Message Bus.....</i>	<i>34</i>
3.4.8	<i>Modular Production Toolkit .....</i>	<i>34</i>
3.4.9	<i>Notification Center .....</i>	<i>36</i>
3.4.10	<i>Optimization / Co-Optimization .....</i>	<i>36</i>
3.4.11	<i>Orchestrator.....</i>	<i>37</i>
3.4.12	<i>Predictive Maintenance.....</i>	<i>37</i>
3.4.13	<i>Repository .....</i>	<i>38</i>
3.4.14	<i>Self-Awareness.....</i>	<i>39</i>
3.4.15	<i>Service Catalog.....</i>	<i>39</i>
3.4.16	<i>Simulation / Co-Simulation .....</i>	<i>39</i>
3.4.17	<i>Sustainability Analytics .....</i>	<i>40</i>



3.4.18	<i>User Interface</i> .....	41
<b>4</b>	<b>PILOT WORKFLOWS</b> .....	<b>42</b>
4.1	BASIC SYSTEM WORKFLOWS .....	42
4.1.1	<i>US0.1: User Management &amp; Access Control</i> .....	42
4.1.2	<i>US0.2: Production Module &amp; Schema (Re-)Configuration</i> .....	43
4.1.3	<i>US0.3: Event Configuration</i> .....	46
4.1.4	<i>US0.4: User Account Management</i> .....	46
4.2	FFT PILOT .....	49
4.2.1	<i>UC1.2: Examine Different Robot Design Setups</i> .....	49
4.2.2	<i>US1.4: Optimize Robot Movements</i> .....	50
4.3	SEW PILOT .....	52
4.3.1	<i>SEW Workflows</i> .....	52
4.4	CRF/ILTAR PILOT.....	54
4.4.1	<i>CRF/ILTAR Workflows</i> .....	55
<b>5</b>	<b>FUTURE WORK AND NEXT STEPS</b> .....	<b>57</b>
5.1	EXPECTED REVISIONS AND UPDATES .....	57
5.2	PRIORITIZATION IN DEVELOPMENT OF MODAPTO COMPONENTS & TECHNOLOGIES.....	57
5.3	IMPLEMENTATION TIMELINE AND MILESTONES.....	58
<b>6</b>	<b>CONCLUSIONS</b> .....	<b>60</b>
<b>7</b>	<b>REFERENCES</b> .....	<b>61</b>
<b>8</b>	<b>ANNEXES</b> .....	<b>62</b>
ANNEX 1	CONCEPTUAL ARCHITECTURE PILOT WORKFLOWS.....	62
<i>Basic System Workflows</i>	.....	62
<i>FFT Workflows</i>	.....	69
<i>SEW Workflows</i>	.....	71
<i>CRF &amp; ILTAR Workflows</i>	.....	73
ANNEX 2	MAPPING OF TECHNICAL REQUIREMENTS AND USER STORIES.....	75
<i>Group 1: Authentication and Authorization</i>	.....	75
<i>Group 2: User Interface</i> .....	75	
<i>Group 3: Production and Digital Twin Management</i> .....	76	
<i>Group 4: MODAPTO System Database and Digital Twins</i> .....	77	
<i>Group 5: Development (and implementation) of Smart Services</i> .....	78	
<i>Group 6: Security and Infrastructure</i> .....	79	
<i>Group 7: System Design, Integration, Monitoring &amp; Compatibility</i> .....	79	
<i>Group 8: Reporting, Alerts &amp; User Support</i> .....	80	
<i>Group 9: APIs and Data Exchange</i> .....	81	
ANNEX 3	MAPPING PRIORITIZATION OF TECHNICAL REQUIREMENTS IN RELEVANCE TO THE USER REQUIREMENTS. ....	82

## List of Figures

Figure 1: Functional view of the DT reference architecture for manufacturing (from ISO 23247-2 [ISO 23247-2, 2021]). .....	19
--	----



Figure 2: Conceptual architecture of MODAPTO..... 21

Figure 3: MODAPTO Technical Architecture (1st version) ..... 28

Figure 4. Sequence Diagram for US0.1: User Management & Access Control ..... 43

Figure 5. Sequence Diagram for US0.2: Production Module & Schema (Re)Configuration .... 45

Figure 6. Sequence Diagram for UC0.3: Event Configuration..... 46

Figure 7. Sequence Diagram for UC0.4: User Account Management. .... 48

Figure 8. Sequence Diagram for UC1.2: Examine Different Robot Design Setups ..... 50

Figure 9. Sequence Diagram for US1.4: Optimize Robot Movements ..... 51

Figure 10. Sequence Diagram for US2.4 Manage Predictive Maintenance Notifications and  
Prioritize Maintenance Actions ..... 53

Figure 11. Sequence Diagram for US2.6: Re-Optimize Production Schedule..... 54

Figure 12. Sequence Diagram for UC3.2: Receive KH Self-Awareness Notifications ..... 55

Figure 13. Sequence Diagram for UC3.9: Receive KH Quality Notifications ..... 56

Figure 14. Roadmap to Submission of 2nd Version of D6.1 ..... 59

Figure 15 . User Management & Access Control ..... 63

Figure 16. Production Module & Schema Reconfiguration ..... 65

Figure 17. Event Configuration..... 66

Figure 18: User Account Management ..... 68

Figure 19. Examine Different Robot Design Setups ..... 69

Figure 20. Optimize Robot Movements..... 70

Figure 21. Manage Predictive Maintenance Notifications and Prioritize Maintenance Actions  
..... 71

Figure 22. Re-Optimize Production Schedule ..... 72

Figure 23: Receive KH Self-Awareness Notifications ..... 73

Figure 24: Receive KH Quality Notifications ..... 74

## List of Tables

Table 1: Pilot 1 (FFT) User Requirements ..... 11

Table 2: Pilot 2 (SEW) User Requirements ..... 11

Table 3: Pilot 3 (ILTAR) User Requirements ..... 12

Table 4: Pilot 3 (CRF) User Requirements..... 13

Table 5: Group 1 of Technical Requirements ..... 14



Table 6: Group 2 of Technical Requirements ..... 14  
 Table 7: Group 3 of Technical Requirements ..... 15  
 Table 8: Group 4 of Technical Requirements ..... 15  
 Table 9: Group 5 of Technical Requirements ..... 16  
 Table 10: Group 6 of Technical Requirements ..... 17  
 Table 11: Group 7 of Technical Requirements ..... 17  
 Table 12: Group 8 of Technical Requirements ..... 17  
 Table 13: Group 9 of Technical Requirements ..... 18  
 Table 14: MODAPTO Conceptual Architecture Functionalities ..... 22  
 Table 15: MODAPTO Technical Architecture Components ..... 29  
 Table 16. MODAPTO Components prioritization Mapping..... 58

## List of Terms and Abbreviations

Abbreviation	Definition
AAS	Asset Administration Shell
API	Application Programming Interface
CRUD	Create, Read, Update, Delete
Dx.y	Deliverable x.y
DT	Digital Twin



<b>FR/ NFR</b>	Functional Requirement / Non-functional Requirement
<b>HTTPS</b>	Hypertext Transfer Protocol Secure
<b>IoT</b>	Internet of Things
<b>JSON</b>	JavaScript Object Notation
<b>KH</b>	Kit-Holders
<b>MQTT</b>	Message Queuing Telemetry Transport
<b>Mx</b>	Month x
<b>OPC UA</b>	Open Platform Communications Unified Architecture
<b>PLC</b>	Programmable Logic Controller
<b>RBAC</b>	Role-Based Access Control
<b>R/D</b>	Re-use/ Develop
<b>RESTful</b>	Representational State Transfer
<b>Tx.y</b>	Task x.y
<b>UCx.y</b>	Use Case x.y
<b>UI</b>	User Interface
<b>USx.y</b>	User Story x.y
<b>WPx</b>	Work Package x
<b>XML</b>	Extensible Markup Language

# 1 Introduction

MODAPTO aims to revolutionize modular manufacturing using interoperable digital twins that align with industrial standards. This approach allows for the seamless integration of production modules within a unified framework, promoting effective design reconfiguration, decision support, and collective intelligence. The vision of MODAPTO is to create a dynamic, flexible, and efficient manufacturing ecosystem capable of responding swiftly to evolving market demands and technological advancements. D6.1 significantly contributes to the project's key objectives, including developing a toolkit for modular production, advancing digital twin interoperability, and supporting sustainable manufacturing practices. It provides a technical framework for module optimization, predictive maintenance, and DT simulation tools, and establishes the foundation for integrating sustainability capabilities into manufacturing processes.

Ultimately, D6.1 is essential for showcasing the efficacy and sustainability of MODAPTO's manufacturing processes and engaging stakeholders in sustainable manufacturing practices. This document is the first version of the MODAPTO Specifications and Architecture, serving as a foundational blueprint by providing the specifications of the MODAPTO framework and the descriptions of the functional components, aligning with the user requirements and business needs identified in D3.1 (D3.1 - Requirements and KPIs.).

The deliverable essentially outlines the technical underpinnings and architectural design to drive MODAPTO's vision to fruition. The information presented in this deliverable is not final. It is a living document that will be continuously updated, following the technical and business achievements of the project.

## 1.1 Purpose and Scope

Deliverable D6.1 - Specifications and Architecture presents a technical framework based on the inputs and user requirements outlined in D3.1. The focus of this document is to present the system architecture, which results from extensive efforts to identify and define the essential functionalities of the MODAPTO system. The architecture is grounded in the technical requirements and expectations of the system's users.

The document also delves into the components of the MODAPTO Architecture, giving a clear and comprehensive view of how each part contributes to the system's overall functionality. One of the key aspects of this document includes sequence diagrams of indicative use cases from pilot projects, which illustrate the user workflow within the MODAPTO system. These practical examples are essential in understanding how architecture facilitates real-world applications.

D6.1 aims to revise and specify the technical descriptions provided initially in the Description of Action (DoA) as the first version reporting on MODAPTO's architecture. This establishes a solid architectural foundation for the development of the MODAPTO system. The deliverable's outcomes serve as a baseline for the entire development lifecycle of MODAPTO, guiding future enhancements and implementations.

## 1.2 Relation to other Work Packages, Tasks and Deliverables

D6.1 reports on work carried out within the context of MODAPTO's Work Package 6 (WP6) and, more specifically, Task T6.1, which aims to set the technological foundations of the project. Although AEGIS leads this task, all partners have contributed to the definition of the components and the design of the different architectural views, while the use case partners leader elaborated extensively with the analysis of their requirements and the alignment of the use cases to the MODAPTO Framework architecture.



Primarily, D6.1 builds upon the foundational work established in D3.1, which deals with user requirements and key performance indicators. The technical specifications and architectural framework presented in D6.1 directly respond to these requirements, ensuring the project's development aligns with user needs and expectations.

Furthermore, this deliverable feeds into subsequent WPs and Tasks, setting the stage for the development and implementation phases of the MODAPTO project. Its outcomes will provide a concrete architectural basis upon which subsequent developments and integrations will be built under WP4, WP5 and Tasks T6.2, T6.3 and T6.4.

Additionally, the deliverable serves as a reference point for future updates and iterations within the project, establishing a baseline for continuous improvement and adaptation. It ensures that all subsequent deliverables maintain alignment with the project's technical and architectural vision, thereby contributing to the cohesive progression of MODAPTO.

### 1.3 Structure of the Deliverable

The document is structured among the following main sections:

- Section 2, Technical Requirements, discusses the prioritization of user requirements and technical specifications across various groups, from authentication and authorization to system design integration and APIs.
- Section 3 MODAPTO Architecture covers background knowledge, conceptual architecture with a top-down approach, detailed methodologies, and comprehensive architecture with component descriptions.
- Section 4, Pilot Workflows, presents workflows for different use cases and outlines basic system workflows.
- Section 5, Future Work and Next Steps, discusses anticipated revisions, updates, and implementation timelines for future development.
- Section 6 concludes the document by providing summarising remarks
- Section 7 collects the references used throughout the document.

## 2 Technical Requirements

This chapter outlines the prioritization of user requirements as outcome of the D3.1 and the technical requirements and specifications that have been derived from the user requirements and business needs.

### 2.1 Prioritization of User Requirements

In this section we provide the prioritization of the pilot-specific Use Cases (UCs) as presented in D3.1, i.e., “User Requirements and KPIs” (Sections 3.5, 4.5 and 5.5). We use the same UC identification system with D3.1, to avoid any confusion. The priority is defined by the industrial partners for each UC considering whether it is “Nice to Have”, “Should Have” or “Must Have” signifying their perceived importance.

Regarding the System Configuration and Access UCs presented in D3.1 (Section 6.4), they are all prerequisites of the pilot-specific UCs and enable fundamental functionalities of the MODAPTO System, hence they are considered as “Must Have”.

**Table 1: Pilot 1 (FFT) User Requirements**

Pilot 1 (FFT) Use Cases		Priority
UC1.1.1	Define Sustainability and Operational Analytics	Must Have
UC1.1.2	Update Sustainability and Operational Analytics	Nice to Have
UC1.2	Examine Different Robot Design Setups	Nice to Have
UC1.3	Assess Performance of a Robot Design Setup	Must Have
UC1.4	Optimize Robot Movements	Nice to Have
UC1.5	Assess the Impact of a Robot Design Setup in a Production Process via Virtual Commissioning	Should Have
UC1.6	Test Operation of Physical Robot	Should Have

**Table 2: Pilot 2 (SEW) User Requirements**

Pilot 2 (SEW) Use Cases		Priority
UC2.1.1	Monitor All Cells	Must have
UC2.1.2	Monitor Single Cell	Must have
UC2.2	Compare the Status of Identical Cells	Must have
UC2.3.1	Receive Self-Awareness Notifications	Must have
UC2.3.2	Assign Maintenance Action to Production Technician	Nice to Have



UC2.3.3	Receive Maintenance Action Notifications	Should Have
UC2.3.4	Resolve Maintenance Action	Should Have
UC2.4.1	Receive Predictive Maintenance Notifications	Must have
UC2.4.2	Inspect Predictive Maintenance Notification and Set Up Maintenance Action Priority	Should Have
UC2.5.1	Review Current Production Schedule / Routing	Must have
UC2.5.2	Optimize Production Schedule	Must have
UC2.5.3	Optimize Production Routing	Must have
UC2.5.4	Incorporate Predictive Maintenance Actions	Should Have
UC2.6.1	Re-Optimize Production Schedule	Must have
UC2.6.2	Re-Optimize Production Routing	Must have
UC2.7.1	Confirm Optimized Production Schedule	Must have
UC2.7.2	Receive Notification for Production Schedule Update	Should Have
UC2.8	Simulate Performance	Must have
UC2.9	Declare Production Cell	Must have
UC2.10.1	Declare Process Drift	Must have
UC2.10.2	Receive Process Drift Notification	Must have

**Table 3: Pilot 3 (ILTAR) User Requirements**

Pilot 3 (ILTAR) Use Cases		Priority
UC3.1.1	Monitor All Production Machines	Nice to have
UC3.1.2	Monitor Single Production Machine	Should have
UC3.2	Receive Kit-Holder (KH) Self-Awareness Notifications	Must have
UC3.3	Re-Optimize Production Schedule	Should have
UC3.4	Compare and Confirm Optimization Results	Should have
UC3.5	Track Outbound Logistics Operations	Must have
UC3.6.1	Re-Design Plant Setup	Must have
UC3.6.2	Simulate Plant Setup	Must have
UC3.7	Review Kit-Holder's (KH) Usage Data	Should have



**Table 4: Pilot 3 (CRF) User Requirements**

Pilot 3 (CRF) Use Cases		Priority
UC3.8.1	Create Potential Content Information of Kit-Holder (KH)	Must Have
UC3.8.2	Create Potential Content Information of a Kit-Holder (KH) block	Must Have
UC3.8.3	Update Potential Content Information of Kit-Holder (KH)	Must Have
UC3.8.4	Update Potential Content Information of a Kit-Holder (KH) block	Must Have
UC3.9	Receive Kit-Holder (KH) Quality Notifications	Nice to Have
UC3.10.1	Select Kit-Holder (KH) Preparation Process	Must Have
UC3.10.2	Monitor Selected Kit-Holder (KH) Preparation Process	Must Have
UC3.10.3	Receive Delivery Delays Notifications	Should Have
UC3.11	Read Kit-Holder (KH) Information	Must Have
UC3.12	(Re-)Optimize Picking Sequence	Must Have
UC3.13	Compare and Confirm Optimization Results	Must Have

## 2.2 Technical Requirements

The Technical Requirements as derived from all the above sections are analyzed in this section. Having presented the prioritization in the previous section a number of technical requirements derive as presented in the following

Section 2.2 serves as a foundational pillar in developing the MODAPTO system. It systematically catalogues and groups technical requirements derived from the detailed analysis of user needs and interactions documented in D3.1. This section is meticulously structured to cover various system dimensions, from user interface design to data exchange and security protocols.

In the context of academic and practical landscapes, the MODAPTO project aligns with and extends beyond the current advancements in modular manufacturing systems. The integration of RMS (Resource Management Systems) and concepts from ERP (Enterprise Resource Planning) and MES (Manufacturing Execution Systems) reflects existing best practices. However, MODAPTO introduces novel elements that push the boundaries of current systems. For instance, the implementation of a digital twin framework in the modular housing industry, as illustrated by Fischer et al. (2022), offers a model for optimizing production through a Digital Twin (DT) framework, which can be a valuable instrument for understanding different process interdependencies in MODAPTO (Fischer et al., 2022).

Moreover, the focus on modularity in MODAPTO is not just about physical components but also encompasses the flexibility and adaptability of digital twins within the manufacturing process. This novel approach, integrating real-time data and machine learning for predictive analytics, marks a significant leap from traditional RMS implementations, much like the MES-integrated digital twin frameworks discussed by Negri et al. (2020), which highlight the role of digital twins in decision-making processes within Industry 4.0-based manufacturing systems (Negri et al., 2020)

Furthermore, the emphasis on interoperable and scalable APIs in MODAPTO signifies an advancement beyond typical closed and rigid systems often found in current manufacturing contexts. This openness and adaptability align MODAPTO with the principles of Industry 4.0, enabling more dynamic and responsive manufacturing ecosystems, much like the Automation

Pyramid in manufacturing systems highlighted by Martinez et al. (2021), which facilitates the integration of physical and digital worlds in manufacturing (Martinez et al., 2021) Thus, the technical requirements detailed in this section adhere to the existing state-of-the-art and propose innovative solutions that are expected to set new benchmarks in modular manufacturing.

## 2.2.1 Group 1: Authentication and Authorization

Table 5 below presents Group 1 that covers technical requirements related to user verification and access control security protocols. It addresses how users are authenticated and what authorisation levels are necessary for different roles within the MODAPTO system.

**Table 5: Group 1 of Technical Requirements**

TR ID	Title	FR/NFR	R/D	Priority
TR1.1	Integrate an authentication service supporting authentication standards.	FR	R	Must Have
TR1.2	Develop a role-based access control (RBAC) system with predefined roles and permissions.	FR	R	Must Have
TR1.3	Implement an admin dashboard for super-users to manage user roles and permissions.	FR	R	Should Have

## 2.2.2 Group 2: User Interface

Group 2, presented in Table 6 below, focuses on TRs associated with the user interface design, ensuring usability, accessibility and user experience are optimised for effective interaction with the MODAPTO system.

**Table 6: Group 2 of Technical Requirements**

TR ID	Title	FR/NFR	R/D	Priority
TR2.1	Develop a visualization system allowing users to interact with MODAPTO and incorporate system flows.	FR	R	Must Have
TR2.2	Develop a dynamic dashboard creation module with customization options.	FR	R/D	Nice to Have
TR2.3	Enable localization and multi-language support for User Interface.	FR	R/D	Nice to Have
TR2.4	Design responsive user interfaces for mobile device compatibility.	NFR	D	Should Have
TR2.5	Enable dashboards to visualize multiple production cells in the same dashboard (comparison mode).	FR	D	Should Have
TR2.6	Use a micro-fronted approach to design modular and reusable Use Interface components.	NFR	D	Should Have



### 2.2.3 Group 3: Production and Digital Twin Management

In Table 7 below, TRs for managing production processes and integrating digital twins are outlined. This group includes requirements for real-time monitoring and control of manufacturing operations.

**Table 7: Group 3 of Technical Requirements**

TR ID	Title	FR/NFR	R/D	Priority
TR3.1	Develop tool for managing digital twins of production modules (MODAPTO Modules).	FR	D	Must Have
TR3.2	Provide an environment to easily configure a MODAPTO Module.	FR	D	Must Have
TR3.3	Provide an environment to easily configure a production schema composing MODAPTO Modules.	FR	D	Must Have
TR3.4	Develop a User Interface for configuration time.	FR	D	Should Have
TR3.5	Integrate near-real-time data streaming for production monitoring.	FR	D	Must Have
TR3.6	Incorporate Intelligent services for decision support.	FR	R/D	Must Have

### 2.2.4 Group 4: MODAPTO System Database and Digital Twins

Table 8 below enlists technical requirements for database management and the implementation of digital twins, emphasising data integrity, storage, and retrieval efficiency.

**Table 8: Group 4 of Technical Requirements**

TR ID	Title	FR/NFR	R/D	Priority
TR4.1	Provide a storage mechanism for MODAPTO services data (i.e., configuration & results).	FR	D	Must Have
TR4.2	Provide CREUD (Create, Read, Execute, Update, Delete) operations for digital twins.	FR	R/D	Must Have
TR4.3	Enable monitoring service to collect real-time data from a production module.	FR	R/D	Must Have
TR4.4	Extend a DT service to be able to run intelligent services.	FR	R/D	Must Have
TR4.5	Extend a DT service to enable on-demand interaction with interconnected DTs.	FR	R/D	Should Have
TR4.6	Develop a service for onboarding non-AAS standard compliant DTs.	FR	D	Must Have
TR4.7	Provide a data retrieval mechanism for efficient data retrieval from the knowledge base.	FR	R/D	Should Have
TR4.8	Enable the definition of DTs following AAS standard.	FR	R/D	Must Have
TR4.9	Enable the definition of DTs as composition of other sub-DTs.	FR	R/D	Must Have

## 2.2.5 Group 5: Development (and implementation) of Smart Services

Group 5 that is presented in Table 9 below, covers TRs for developing smart, automated services within the MODAPTO framework, focusing on enhancing operational efficiency and predictive maintenance.

**Table 9: Group 5 of Technical Requirements**

TR ID	Title	FR/NFR	R/D	Priority
TR5.1	Analytics service supports batch data and real-time data analytics. Batch data is recorded time-serious data of the production modules which is analyzed afterwards. Real-time data is the time-serious data of the production modules during runtime of the production module which is analyzed during data gathering continuously.	FR	R/D	Must Have
TR5.2	The Optimization functionality supports multiple input parameters (e.g., orders and production schedules, quality specifications, shifts planning horizon, materials, carbon footprints, energy consumption, robot movements).	FR	D	Must Have
TR5.3	The Optimization functionality produces suggestions for optimal courses of action (e.g., production schedules, maintenance schedules, optimal robot movement configurations and actions).	FR	D	Must Have
TR5.4	The Simulation functionality supports multiple input parameters (e.g., sustainability parameters, materials, manufacturing methods, robot movements, orders and production schedules, plant setups).	FR	D	Must Have
TR5.5	The Simulation functionality produces sustainability and operational analytics to assess the performance of a MODAPTO module (e.g., specific robot movements, production schedules, plant setups).	FR	D	Must Have
TR5.6	The Self-awareness module should involve a combination of data acquisition, processing, analysis, and implementation of machine learning models for failure prediction, to enable effective self-awareness functionality for assessing degradation and remaining useful life.	FR	R/D	Must Have
TR5.7	The predictive maintenance module should involve a combination of acquisition, processing, analysis, implementation of machine learning models for the generation of maintenance recommendations for the action to be performed.	FR	R/D	Must Have
TR5.8	Capability to execute Smart Services (algorithms) via Digital Twins.	FR	R/D	Must Have
TR5.9	Provide an Orchestrator to combine and reuse smart services for the specific pilots	FR	D	Must Have



## 2.2.6 Group 6: Security and Infrastructure

Group 6 presented in Table 10 below, includes TRs for overall system security and infrastructure, ensuring robust protection against cyber threats and a solid foundation for system operations.

**Table 10: Group 6 of Technical Requirements**

TR ID	Title	FR/NFR	R/D	Priority
TR6.1A	Enable End-to-End Encryption for Data Transmission (examples: HTTPS, MQTTS, OPCUA).	NFR	R	Must Have
TR6.1B	Implement Encryption for Sensitive User Data.	NFR	R	Must Have
TR6.2	Ensure MODAPTO Compatibility with Standard Backup and Recovery Processes.	NFR	n/a	Should Have
TR6.3	Ensure MODAPTO Compatibility with Scalable Cloud-Based Infrastructures.	NFR	n/a	Nice to have

## 2.2.7 Group 7: System Design, Integration, Monitoring & Compatibility

Group 7 presented in Table 11 below, focuses on TRs for system design, integration with existing systems, continuous monitoring, and compatibility with various manufacturing environments.

**Table 11: Group 7 of Technical Requirements**

TR ID	Title	FR/NFR	R/D	Priority
TR7.1	Connect existing Data Sources to MODAPTO.	FR	D	Must Have
TR7.2	Provide means to integrate with existing standards/tools for monitoring.	FR	TBD	Should Have
TR7.3	Implement a comprehensive integration testing strategy that includes periodic testing of components' functionalities.	NFR	TBD	Should Have
TR7.4	Implement a continuous integration/deployment pipeline with version control. (CI/CD)	NFR	TBD	Should Have
TR7.5	Adopt a service-oriented / microservice based architecture for modularity and scalability.	NFR	TBD	Should Have

## 2.2.8 Group 8: Reporting, Alerts & User Support

Table 12 below, presents the TRs of Group 8 that are related to system reporting, alert generation, and user support mechanisms, aiming for proactive system management and user assistance.

**Table 12: Group 8 of Technical Requirements**

TR ID	Title	FR/NFR	R/D	Priority
-------	-------	--------	-----	----------



TR8.1	Provide a dashboard that presents real-time and historical aggregated data.	FR	D	Should Have
TR8.2	Provide a notification system for notifications like alerting etc., displayed via the User Interface.	FR	D	Must Have
TR8.3	Provide User Feedback capabilities.	FR	D	Should Have
TR8.4	Create user manuals, online documentation, and video tutorials.	NFR	D	Should-Have
TR8.5	Support creation/saving/exchange of process drifts among relevant user roles.	FR	D	Must Have
TR8.6	The notification system receives notifications from any component and pushes them to the GUI in real-time.	FR	D	Must Have

### 2.2.9 Group 9: APIs and Data Exchange

Table 13 below, presents the final group that deals with TRs for APIs and data exchange protocols, ensuring seamless and efficient communication between system components and external entities.

**Table 13: Group 9 of Technical Requirements**

TR ID	Title	FR/NFR	R/D	Priority
TR9.1	Provide (RESTful) APIs with endpoints for external integrations.	NFR	D	Must Have
TR9.2	Adopt Standard Data Interchange Formats like JSON, XML, etc.	NFR	R	Must Have
TR9.3	Provide comprehensive API documentation with sample requests and responses.	NFR	R/D	Should Have
TR9.4	Provide Data Management Mechanisms that comply with defined data retention policies.	NFR	D	Must-Have
TR9.5	Implement (User) data validation checks at input points. (Need to further define the system data. Split this TR into two based on the origin and nature of data).	FR	R/D	Must Have

## 3 MODAPTO Architecture

Road to MODAPTO technical architecture

### 3.1 Background knowledge

Industry 4.0 is revolutionizing industrial production by bridging the physical and the virtual world and further improving digitalization. The concept of Digital Twin is one of the cornerstones of Industry 4.0. Digital Twins (DTs) are the virtual representations of resources organizing and managing information and being tightly integrated with artificial intelligence, machine learning and cognitive services to further automate and optimize production.

ISO has published a series of standards (ISO 23247) that describe a blueprint for creating and working with DTs in the context of manufacturing. Following a bottom-up approach, it introduces key terms and concepts of the DT domain and defines an abstract multi-level framework, or better a reference architecture, for DT systems.

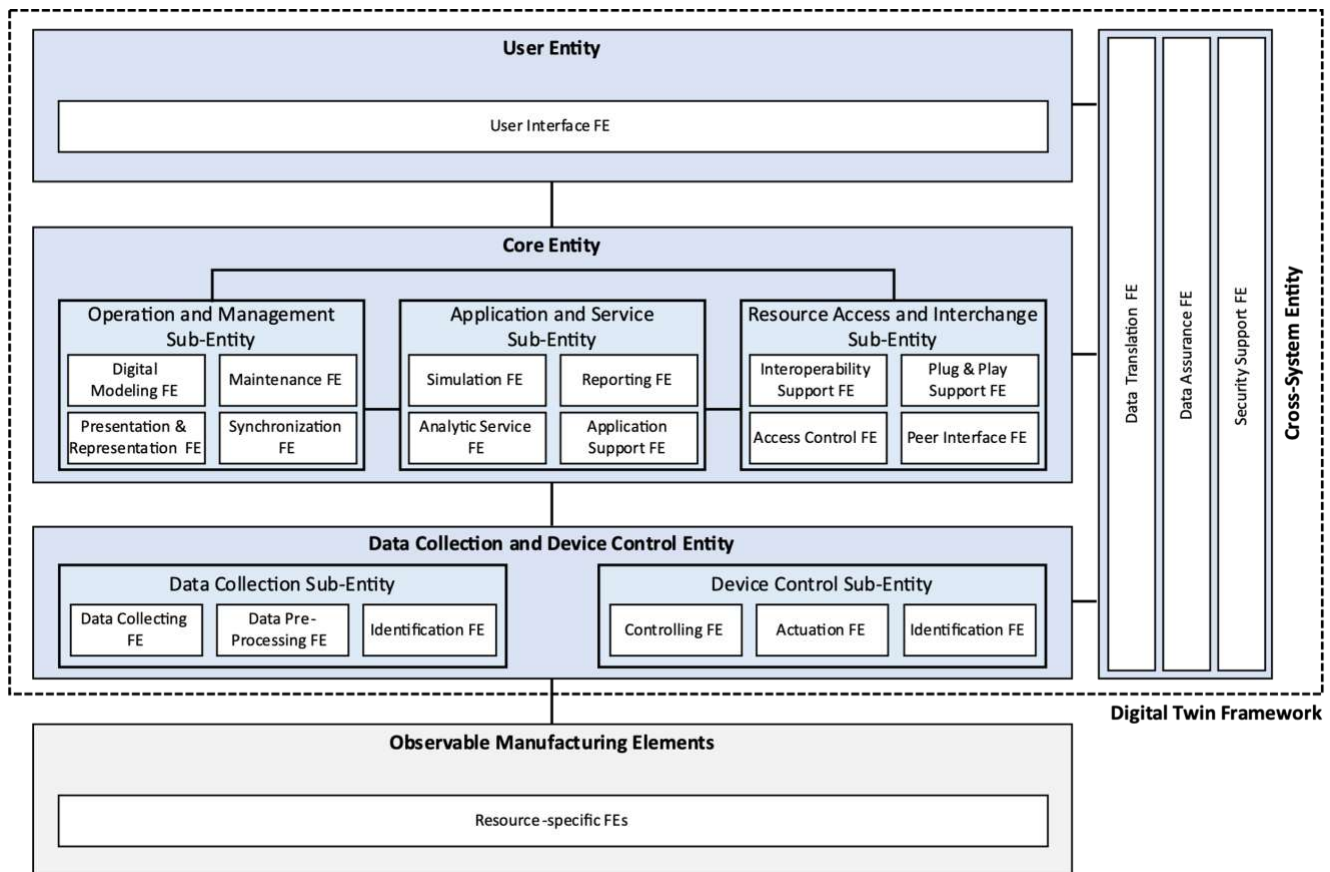


Figure 1: Functional view of the DT reference architecture for manufacturing (from ISO 23247-2 [ISO 23247-2, 2021]).

While ISO 23247 provides a generic framework for creating DT-related systems, the Asset Administration Shell (AAS) specification<sup>1</sup> establishes a concrete and hands-on implementation of DTs with primary focus on industrial production.

<sup>1</sup> <https://industrialdigitaltwin.org/en/content-hub/aasspecifications>



The AAS specification was initially created by the Plattform Industrie 4.0, a network of companies, associations, trade unions, science, and politics in Germany but is now maintained and developed by the Industrial Digital Twin Association (IDTA). It is also being published as international standard IEC 63278. The AAS specification defines a meta-model for modelling DTs, protocol-agnostic APIs to discover and interact with DTs as well as a concrete mapping of these APIs to HTTP/REST.

This high-level functional view of the MODAPTO architecture has been defined over parallel discussions and is fully aligned with the design and development efforts of all the technical partners of MODAPTO. It is a valuable asset that is tightly bound with the technical background of all the experienced partners and the tools and components they bring to this project. The road towards a final technical architecture, that will realize the interoperable Digital Twins and any low-level component interaction, was paved during this process, since behind any high-level asset there were also preparations to align the functional view with a detailed technical view. For this process, the background knowledge over design strategies, technologies, tools, and communications mechanisms of all the technology providers has been fully utilized and taken into consideration.

## 3.2 Conceptual Architecture

For designing a complex software system such as MODAPTO, we decided to start with creating a conceptual architecture. The main purpose of the conceptual architecture is to create a better understanding of the basic or core structure and functionalities of a system without going into technical details. This enables the creation of a visual description of the system that is easy to understand for externals, e.g., potential future users of MODAPTO, by creating enough “order” at system level while leaving room for the organic emergence of (technical) details at a later stage of the system design.

We followed a formal process to create the conceptual architecture based on the user requirements presented in deliverable D3.1. A conceptual architecture typically consists of boxes and arrows. In contrary to a technical architecture, boxes in the conceptual architecture do not represent software components but rather a set of related domain-level responsibilities or functionalities. They are not required to have a direct counterpart in the final software. Arrows in a conceptual architecture typically describe the data flow between the boxes. However, we decided to not include arrows in our conceptual architecture as this heavily impaired the readability of the diagram when we tried defeating the intended purpose to provide an easy-to-understand high-level view of MODAPTO.

To create the boxes of the conceptual architecture from the technical requirements, the first step is to identify the key concepts contained within the requirements. These are typically the nouns used but to some degree also can be verbs, e.g. when describing functionality. In a second step, each key concept is categorized into one of the following categories: data, function, stakeholder, system, or abstract concept. Depending on the category, the key concepts will be treated differently and might become part of the conceptual architecture or be dismissed. Key concepts that are categorized as data might not be represented as a box but potentially should appear on one of the arrows as a part of the information flow. Key concepts categorized as functions are the most relevant. In the next step, they will be grouped into sets of related functionalities where each of these sets constitutes a box in the conceptual architecture. Key concepts categorized as stakeholders are ignored and may never become part of the conceptual architecture. Key concepts categorized as systems or abstract concepts might become part of the conceptual architecture depending on the case. For example, when categorized as a system, it often makes sense to introduce a box representing the interface functionality to that system.

Obviously, this is not a deterministic approach as identification of key concepts, their categorization, grouping of functional key concepts, etc. are partially subjective decisions. The

same applies to the way the final boxes are drawn and aligned to each other. Although there is no explicit semantic to the location of boxes, i.e. there is no meaning of one box being above another box in the final diagram, software developers tend to arrange such diagrams according to a layered architecture. However, this is not an entirely bad thing to do, as it not only aligns with ISO 23247 as shown in Figure 1 but also improves readability.

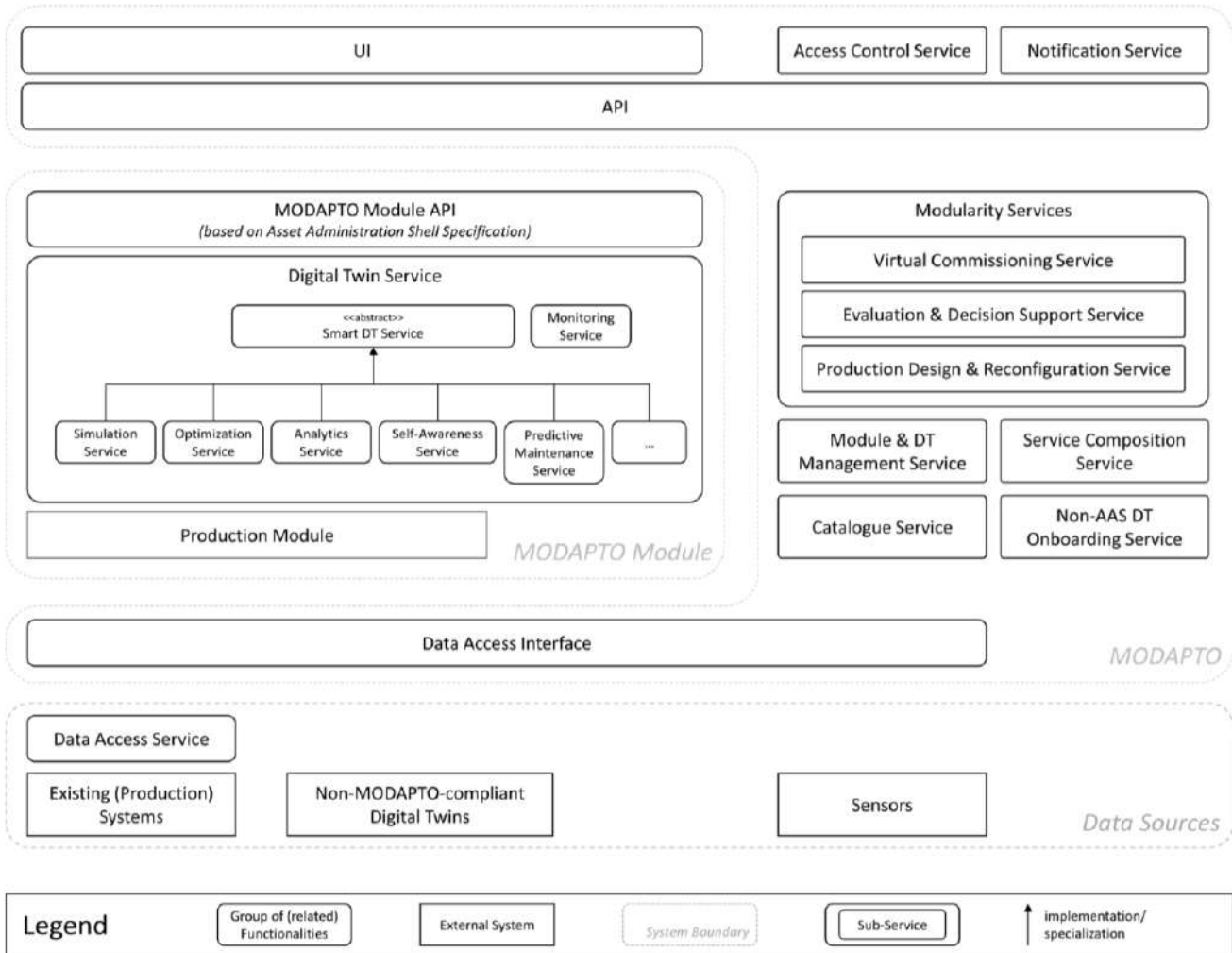


Figure 2: Conceptual architecture of MODAPTO.

The functionalities presented in the conceptual architecture on Figure 2 are grouped in three main categories that reflect the different systems boundaries: MODAPTO, MODAPTO Module, and Data Sources.

- **MODAPTO:** represents the MODAPTO system as a whole and contains generic and cross-module functionalities.
- **MODAPTO Module:** groups the functionalities of a MODAPTO Module, that is an integrated unit within a reconfigurable manufacturing system that combines the principles of a production module and its corresponding Digital Twin.

*The production module* is an asset designed to perform specific manufacturing tasks or processes can be seamlessly integrated or reorganized with other production modules, fostering the creation of a flexible, modular, and reconfigurable production environment. This environment can operate in a manual, semi-manual, or automated mode. The asset itself may take the form of a physical entity (such as a machine or robotic system) or a logical entity (such as a group of machines or a production line).

The MODAPTO module enriches the production modules with their digital counterparts, enabling seamless adaptation, modularity, and interoperability. By leveraging Digital Twin technology, the MODAPTO Module facilitates real-time monitoring, optimization, and decision-making, enhancing the flexibility and efficiency of manufacturing systems. A detailed description of the MODAPTO Module concepts is available in the MODAPTO deliverable D3.1.

- **Data Source:** represents use case specific functionalities provided by pilot partners.

In Table 14 and in the following subsections all the MODAPTO and MODAPTO Module functionalities will be introduced in detail.

**Table 14: MODAPTO Conceptual Architecture Functionalities**

Name in diagram	Summary of provided functionalities
<b>Access Control Service</b>	Responsible for the registration, authentication and authorization of the system users
<b>Analytics Service</b>	The analytics service employs real-time data analysis to interpret signals emanating from a production process, providing actionable insights for optimizing efficiency and performance.
<b>API</b>	Facilitates smooth communication and coordination among various modules and systems
<b>Catalogue Service</b>	Oversees component services metadata, facilitating service discovery and communication among project components
<b>Data Access Interface</b>	Facilitates the management and exchange of data within the system
<b>Data Access Service</b>	Works as a counterpart of Data Access Interface, other components will have to implement it to have access to data
<b>Digital Twin Service</b>	Represents the DT part of the MODAPTO module
<b>Evaluation &amp; Decision Support Service</b>	Enable the reporting and visualization of KPIs with custom dashboards
<b>MODAPTO Module API</b>	AAS-based REST API to communication with the DT
<b>Modularity Services</b>	Collect services for the final user, that relies on the MODAPTO Module concept
<b>Module &amp; DT Management Service</b>	<ul style="list-style-type: none"> <li>• CRUD MODAPTO module metadata</li> <li>• Create &amp; manage DTs</li> <li>• add/assign Smart DT Service to MODAPTO modules</li> </ul>
<b>Monitoring Service</b>	Enables monitoring the state of a DT (i.e. its MODAPTO module)
<b>Non-AAS DT Onboarding Service</b>	Enables onboarding of existing non-AAS compliant DTs
<b>Notification Service</b>	Triggers and routes real time notifications and system alerts to the User Interface
<b>Optimization Service</b>	Provides suggestions for the (close-to-) optimal course of action depending on the MODAPTO module that utilizes it and the objective and constraints of the corresponding problem.
<b>Predictive Maintenance Service</b>	Provides recommendation for maintenance actions to be performed. The recommendation could be a periodization, grouping of maintenance action or scheduling of the actions.

<b>Production Design &amp; Reconfiguration Service</b>	Allows the definition and composition of MODAPTO modules
<b>Self-Awareness Service</b>	Evaluates the health of the assets of a module in terms of pending degradation and the prognostics of the Remaining Useful Life.
<b>Service Composition Service</b>	Coordinates various components to execute complex workflows to exchange information and implement the specified processes
<b>Simulation Service</b>	Simulates the execution and evaluates the performance of one or more MODAPTO module(s) based on the corresponding operational and sustainability criteria.
<b>Smart DT Service</b>	Abstract representation of logic or an algorithm a MODAPTO module might offer
<b>UI</b>	A tailored and user-friendly interface for efficiently managing production modules and their Digital Twins
<b>Virtual Commissioning Service</b>	Support customer decision by enabling the life cycle assessment analysis of a new MODAPTO module.

### 3.2.1 Access Control Service

It will be responsible for the registration, authentication and authorization of the system users. It will provide a UI to allow an administrator to manage the users of the system and perform operations like creating, updating or deleting users in the system.

### 3.2.2 API

The API will be designed to enable seamless communication and coordination between different modules and systems within MODAPTO. This could involve endpoints for retrieving and updating information related to Digital Twins, production module status, and data necessary for collective intelligence functionalities.

### 3.2.3 Catalogue Service

Manages records of component services metadata within the MODAPTO project, in order to assist in the service discovery and communication between the MODAPTO components to be implemented in the course of the project.

### 3.2.4 Data Access Interface

It defines a set of methods, operations, or protocols that components within the system can use to interact with the data storage mechanisms, managing and exchanging data within the system. This allows the MODAPTO components to receive data from external systems like ERP or MES systems following standardized techniques, in order to perform their operations.

### 3.2.5 Data Access Service

The Data Access Service implements the functionality specified by the Data Access Interface. It serves as the concrete implementation of the data access logic, providing the actual mechanisms for retrieving, storing, updating, and deleting data within the system. Components interact with the Data Access Service to perform operations on the underlying data sources.

### 3.2.6 Digital Twin Service

The Digital Twin Service represents the DT part of the MODAPTO module.



### ***3.2.6.1 Smart DT Service***

In the conceptual architecture, the Smart DT Service is an abstract representation of logic or an algorithm a MODAPTO module might offer. In AAS terms, this would be an operation that can be invoked on the DT. MODAPTO will offer different kinds of Smart DT Services like, but not limited to, simulation, optimization, analytics, self-awareness, and predictive maintenance.

### ***3.2.6.2 Monitoring Service***

The Monitoring service reflects the need to monitor the state of a MODAPTO module (or rather its associated DT) and subscribe to changes in state. This functionality is implicitly provided by the AAS API. However, it is explicitly depicted as a separate box as boxes in this diagram represent required sets of functionalities rather than software components, thus it is expected that this box will have no explicit correspondence in a more technical architecture diagram or code. It is also not considered a Smart DT Service as monitoring does not require any logic or algorithm.

### ***3.2.6.3 Simulation Service***

This service enables the (co-)simulation of the MODAPTO modules' operation. It may refer to a single production asset, a single production process or a collection of assets and processes. It may require input from a single or multiple modules of the manufacturing process as well as data stemming from enterprise and manufacturing information systems and other sources. Based on this input, the Simulation Service calculates and returns the corresponding metrics on the operation of the respective MODAPTO module(s).

### ***3.2.6.4 Optimization Service***

This service enables the optimization of the MODAPTO modules' operation. It may refer to a single production asset, a single production process or a collection of assets and processes. It may require input from a single or multiple modules of the manufacturing process as well as data stemming from enterprise and manufacturing information systems and other sources. Based on this input, the Optimization Service identifies and returns the optimal course of action based on specified operational and sustainability constraints and objectives as well as specified metrics with respect to the proposed solution.

### ***3.2.6.5 Analytics Service***

The analytics service employs real-time (or batch) data analysis to interpret signals emanating from a production process, providing actionable insights for optimizing efficiency and performance. The analysis of energy consumption of a production module is an example for an analysis of such process signals. If necessary, the analytics service might also include statistics to discover trends or compare results.

### ***3.2.6.6 Self-Awareness Service***

The self-awareness service will provide information about the current status (health) of the physical part of the production module. It might provide information about anomaly detection, fault diagnostics and prognostics. It requires monitoring data from the physical part of the production module as input.

### ***3.2.6.7 Predictive Maintenance Service***

The predictive maintenance service will provide information about the maintenance actions that have to be performed. This could be, depending on the use case, prioritization of the pending action to be performed and/or scheduling of these action. The inputs of the service are the pending action to be performed, information from the self-awareness service of the considered physical parts, information from the Computerized Maintenance Management System.



### 3.2.7 MODAPTO Module API

The MODAPTO Module API represents the API used to interact with any MODAPTO module. As each MODAPTO module is represented by a DT, it will be based on the HTTP/REST API of the AAS specification. The API will allow the interaction with the MODAPTO module via its DT, e.g. by reading and writing values or invoking operations.

Existing AAS sub model templates will be reused if they prove suitable for the task. AAS sub model templates can be considered domain-specific extensions of the AAS specification providing some semantic interoperability within a given domain. A potential candidate sub model template to be reused as part of the MODAPTO module API is the sub model template “Provision of Simulation Models”<sup>2</sup>.

### 3.2.8 Modularity Services

This component group functionalities relies on one or more MODAPTO modules.

#### 3.2.8.1 Virtual Commissioning Service

The service offers functionalities that enable virtual commissioning and Life-Cycle Assessment (LCA) of new production modules to support decisions about their integration in the industrial production processes. This helps designers and planners in decision making and does not require the physical presence of the production modules.

#### 3.2.8.2 Evaluation & Decision Support Service

This service enables the evaluation of production process performance by providing comprehensive dashboards and reporting functionalities. Users can leverage these tools to gain insights into the efficiency and effectiveness of their production processes.

#### 3.2.8.3 Production Design & Reconfiguration Service

This service allows the definition and (re-)configuration of MODAPTO modules with attributes and dependencies in form of formal and, when possible, standard-based models. This formal representation of MODAPTO modules and their combination in form of process flow will then be used by the different smart services like the (co-)simulation.

### 3.2.9 Module & DT Management Service

The Module & DT Management Service is responsible for creating and managing MODAPTO modules and their associated DTs within the MODAPTO ecosystem. This includes CRUD (create, read, update, delete) functionality for MODAPTO module metadata, creating and managing DTs for the MODAPTO modules, adding/assigning Smart DT Services to MODAPTO modules as well as registering existing MODAPTO modules.

### 3.2.10 Non-AAS DT Onboarding Service

The Non-AAS DT Onboarding Service, as its name suggests, enables the integration and onboarding of existing DTs that are modelled using other standards than the AAS, for example the Digital Twin Definition Language (DTDL) used by Microsoft Azure.

---

<sup>2</sup><https://github.com/admin-shell-io/submodel-templates/tree/main/published/Provision%20of%20Simulation%20Models/1/0>

### 3.2.11 Notification Service

The notification service is designed to keep the end-user informed about a range of events and updates through the user interface. Whether it's critical information or calls to action, this service ensures that users stay well-informed and engaged with timely alerts.

### 3.2.12 Service Composition Service

Coordinates and manages multiple components, stringing together multiple tasks in order to execute a larger workflow or process based on the user requirements. Interacts with components like Optimization, Simulation, Service catalog, etc. to exchange the necessary information and implement these workflows.

### 3.2.13 UI

The UI will be designed to provide a customized and user-friendly experience for managing the production modules and their Digital Twins and interacting with them. This might include visualization of various dashboards about the production modules, results of the services, e.g. analytics service, and available interaction elements for the Digital Twin Management Services.

## 3.3 From Conceptual Architecture to High Level Technical Architecture

Once the Conceptual Architecture for MODAPTO is defined, the next step is to bridge the theoretical concepts with practical implementation. The progression from a conceptual architecture which highlights the core functionalities of MODAPTO, to a detailed technical architecture is critical for translating abstract ideas into tangible, workable solutions. This transition involves a deep understanding of the project's aims and the integration of these aims with technological possibilities.

The conceptual architecture, initially outlined, served as a blueprint, providing an overarching view of MODAPTO's objectives and capabilities. This phase focused on defining core functionalities, such as modularity, interoperability, and scalability, laying the groundwork for the system's goal. This stage was crucial in establishing a clear vision and scope for the project, ensuring that all subsequent technical developments align with the project's overarching goals. Moving to the technical architecture, we delve into system design specifics, including the selection of technologies, components, and their interactions. This phase marks a shift from 'what' the system is intended to achieve to 'how' these objectives will be met. The technical architecture encapsulates a more granular view, detailing the software, hardware, and network components, their configurations, and the rationale behind these choices. This transition is pivotal in transforming the MODAPTO concept into a viable, functional system ready for implementation and testing.

Conceptual architecture is an asset that drives us to envision and design the technical components that will realize MODAPTO's vision. The first version of the high-level technical architecture that is presented on this deliverable contains more details regarding sub-components and interactions that can implement all MODAPTO features and functionalities. Such components that are easily noticeable are, among others, the Message Bus that is a necessity for real-time communications and notifications, the Orchestrator which will manage a very big part of the system's logic and interconnections, and a more detailed layout of the data bases and management services needed.

In the following section we present the initial version of the technical architecture of MODAPTO. This version is a high-level approach from the technical scope enlisting core MODAPTO components and technologies that are meant to align with the functionalities defined by the

Conceptual Architecture. Initial steps towards the completion of this high-level technical architecture are described in detail in Section 3.4 below

### 3.4 MODAPTO High Level Technical Architecture

The methodology employed to shape the MODAPTO Architecture involved consolidating valuable insights from diverse sources, including D3.1 and tasks associated with T6.1. This comprehensive approach encompassed various critical elements:

1. *Analysis of End-User Requirements*: A meticulous examination of end-user requirements, as extensively detailed in the user stories described in D3.1 (also referenced in Section 2.1).
2. *Functional and Non-Functional Requirements*: Evaluation of both functional and non-functional requirements.
3. *Technical Requirements*: Exploration of technical requirements derived from user stories, further discussed in Section 2.2.
4. *Conceptual Architecture and Core Functionalities*: Exploration of the conceptual architecture, emphasizing core functionalities (detailed in Section 3.2).
5. *Description of Action (DoA)*: Detailed insights extracted from the Description of Action (DoA).
6. *Component Descriptions*: Detailed descriptions of components (Sections 3.4.3 to 3.4.18).

Collaboratively, through joint and ad-hoc sessions involving all consortium partners and technology providers engaged in T6.1 activities, this wealth of information laid the foundation for defining the MODAPTO Technical Architecture (Figure 3). The interconnected core components or groups of components were mapped based on the current knowledge and agreements reached among the participating partners.

The consortium actively maintains a dynamic version of the MODAPTO Architecture, continually updating it in sync with the development of the MODAPTO project. This ensures that the latest architectural diagram accompanies each integrated release of MODAPTO, effectively portraying the ongoing evolution of the system.

The definition of the first version of the MODAPTO Technical Architecture was derived from the comprehensive information presented in this document. MODAPTO focuses on two technological pillars.

Pillar 1: Distributed Intelligence & Control via Interoperable Digital Twins

Pillar 2: A Modular Production Framework & Toolkit for design, reconfiguration and decision support

Each colored box in the Architectural diagram (Figure 3) represents a component, with relevant sub-components in some cases, and MODAPTO module represents groups of components for Pillars 1 and 2, respectively. The displayed interconnections illustrate relationships between components, signifying either control & data flow (solid line) or data flow (dotted line). The 'Data Sources' entity at the bottom of the diagram denotes assets provided by stakeholders. It is not an integral part of the MODAPTO Architecture but rather signifies the connection between the MODAPTO system and the stakeholders' existing system infrastructure.

As already mentioned, the final state of the MODAPTO Technical Architecture will be presented in every new release of the MODAPTO system, and the definitive version will be detailed in a future deliverable.

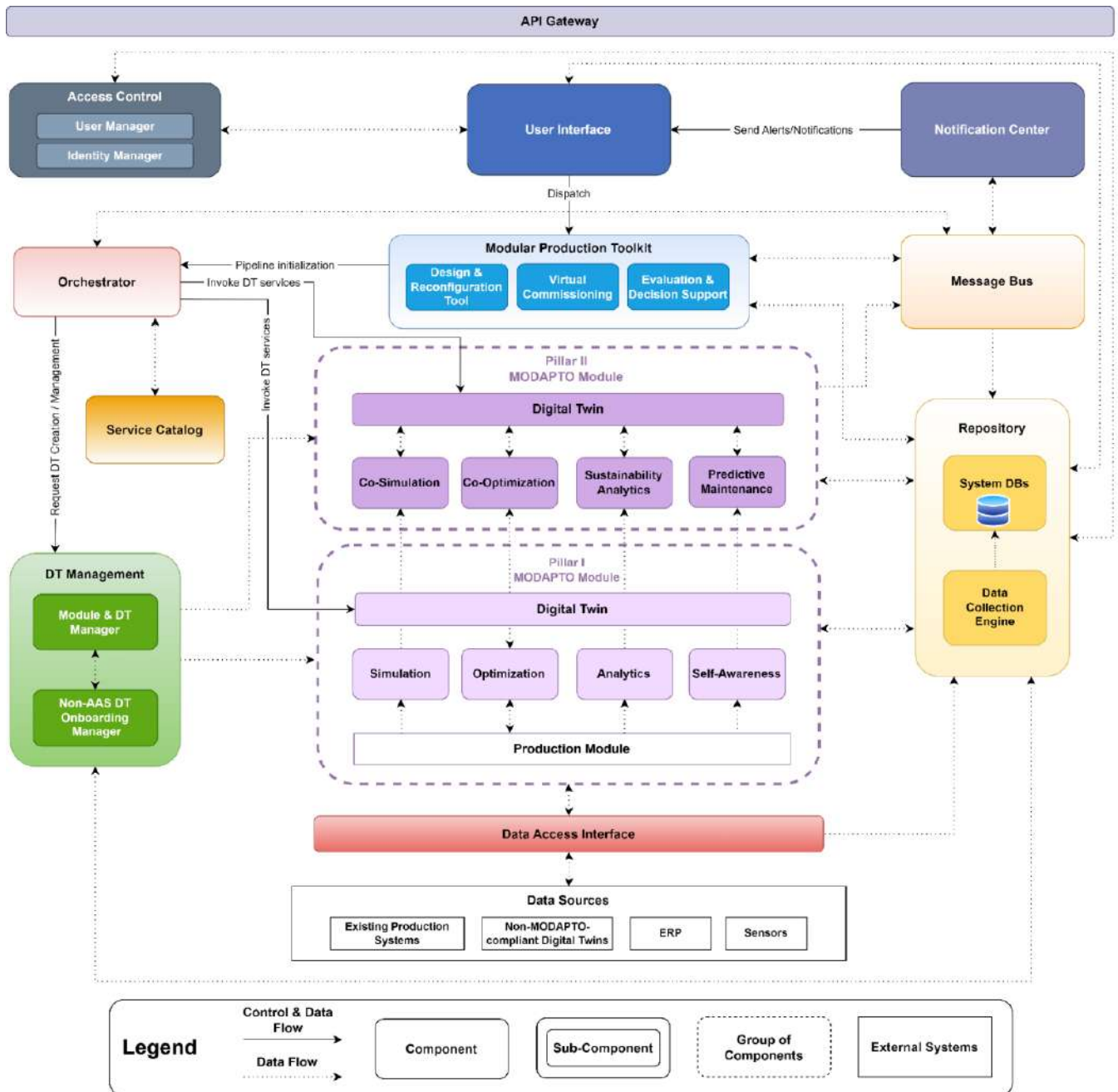


Figure 3: MODAPTO Technical Architecture (1st version)

Table 15 below enlists the MODAPTO Technical Architecture components that are described in detail in the following subsections.



**Table 15: MODAPTO Technical Architecture Components**

Name in diagram	Summary of provided functionalities	Related TR(s)	Related Project Task
<b>Access Control</b>	Responsible for delivering user management for the registration, authentication and authorization processes for system users		
<b>Analytics</b>	This component serves for the analysis of recorded data, e.g. energy consumption, from the production modules.	TR3.5, TR4.3, TR5.1	T4.5, T5.3
<b>API Gateway</b>	Access management to and from the internal APIs of the installed physical node.	TR9.1, TR9.2, TR9.3, TR9.4 TR6.1A, TR6.1B, TR6.3	T6.3
<b>Data Access Interface</b>	Comprises a collection of services designed to facilitate the management and exchange of data between the system and Data Sources.		
<b>Design &amp; Reconfiguration tool</b>	Allow the definition and (re-)configuration of MODAPTO modules and their combination in the production schedules	TR2.1, TR2.2, TR2.4, TR2.6, TR3.1, TR3.2	T5.3
<b>Digital Twin</b>	Each MODAPTO Module will be represented by a Digital Twin. The DT will be able to connect to existing data sources and execute operations, e.g. analytics, simulation or optimization.	TR04.2, TR04.3, TR04.4, TR04.5, TR04.8, TR04.9, TR06.1A, TR07.1, TR09.1, TR09.2, TR09.3	T4.2
<b>DT Management</b>	This component is responsible for creating and managing DTs of MODAPTO modules.	TR03.1, TR03.2, TR04.2, TR04.6, TR06.3, TR09.2	T6.2 & T6.3
<b>Evaluation &amp; Decision Support</b>	Will handle diverse data, events. It will Offer intuitive visualizations, multi-purpose dashboards, timeline inspection, and preconfigured views for effective evaluation and decision-making.	TR2.1, TR2.2, TR6.1A, TR9.1	T5.3 & T5.1
<b>Identity Manager</b>	Responsible for delivering user management, authentication, and authorization services.	TR01.1, TR01.2, TR06.1A, TR06.1B	T6.3
<b>Message Bus</b>	Supporting real-time and asynchronous communication, for the seamless exchange of information, commands, and updates among various components of the system.	TR04.3, TR03.5, TR07.5, TR08.1, TR08.6, TR09.1	T5.1
<b>Modular Production Toolkit</b>	This component exposes the functionalities of the Design and Reconfiguration tool, Virtual Commissioning, and Evaluation and Decision Support tool	TR2.1, TR2.2, TR2.4, TR2.6, TR3.1, TR3.2, TR6.1A, TR07.4,	T5.3



		TR07.5, TR9.1, TR09.4	
<b>Module &amp; DT Manager</b>	This component is responsible for creating and managing MODAPTO modules and their associated DTs	TR03.1, TR03.2, TR04.2, TR06.3	T6.2 & T6.3
<b>Non-AAS DT Onboarding Manager</b>	Enables the integration and onboarding of existing DTs that are modelled using other standards than the AAS, for example the Digital Twin Definition Language (DTDLE) used by Microsoft Azure	TR04.6, TR09.2	T6.2
<b>Notification Center</b>	Responsible for initiating and directing real-time notifications and system alerts.		
<b>Optimization / Co-optimization</b>	Service to optimize the operation of a single or multiple MODAPTO modules	TR5.1, TR5.2, TR5.3, TR5.9	T4.3, T5.4
<b>Orchestrator</b>	Coordination and management of multiple components to execute larger workflows or processes	TR05.9, TR09.1, TR09.3, TR07.4	T5.1
<b>Predictive maintenance</b>	Provides recommendation for maintenance actions	TR05.1, TR05.7, TR09.1, TR09.3	T5.5
<b>Repository / System DBs / Data Collection Engine</b>	The central database of the system, storing diverse data and providing real-time updates to all system components. It also offers analytical tools for insights into KPIs and triggers alerting services for real-time notifications.	TR04.1, TR04.7, TR09.1, TR09.4	T5.3
<b>Self-awareness</b>	Assess the health of the assets a module and prognosticate the Remaining Useful Life.	TR05.1, TR05.7, TR09.1, TR09.3	T4.5, T5.5
<b>Service Catalog</b>	Responsible for generating and overseeing records of smart services.	TR09.1, TR09.2, TR09.3, TR09.4	T5.1
<b>Simulation / Co-Simulation</b>	Service to simulate the operation of a single MODAPTO Module or co-simulate the operation of multiple MODAPTO Modules.	TR02.1, TR02.2, TR07.5	T4.3, T5.4
<b>Sustainability Analytics</b>	Service to calculate sustainability properties (e.g. energy consumption, carbon emissions) based on monitoring of production modules and Analytics Service of production modules.	TR3.5, TR4.3, TR4.8, TR5.1, TR7.1, TR7.2	T4.5, T5.3
<b>User Interface</b>	This component serves as the main entry point for the user, providing reusable User Interface		



	widgets for input forms and specific functionality.		
<b>User Manager</b>	Responsible for the registration, authentication and authorization processes for system users	TR01.1, TR01.2, TR01.3, TR06.1A, TR06.1B, TR06.2, TR06.3, TR02.6, TR09.1, TR09.4	T6.3
<b>Virtual Commissioning</b>	This component serves for testing, evaluating, analyzing and optimizing the production modules during the commissioning within the virtual world (before the real commissioning or in parallel to the operation of the production modules).	TR5.1, TR5.2, TR5.4, TR5.5, TR7.1, TR7.2	T5.3

### 3.4.1 Access Control

#### 3.4.1.1 User Manager

<b>Objectives / Brief description</b>	The User Manager will be responsible for the registration, authentication and authorization of the system users. This component works as an abstract layer in front of the Identity Manager utilizing the relevant functionalities of this tool.
<b>Requirements &amp; Development</b>	(Min) Memory: 2 GB RAM, CPU: 4 cores, Disk storage size: 10 GB. It will be provided as a set of docker images that will expose APIs and UIs. So, Docker and Docker compose or Kubernetes (depending on final set up) is needed. The suggested operating system is Linux OS e.g. Ubuntu.
<b>Interactions</b>	The UI of the User Manager will be part of the container UI that will share the user token with the other services or UIs. Additionally, a component will be able to access the manager's services through its APIs. It interacts with the Identity Manager for authentication and authorization acting this way as an abstraction layer in front of it.

#### 3.4.1.2 Identity Manager

<b>Objectives / Brief description</b>	The Identity Manager will be responsible for providing user management, authentication and authorization services.
<b>Requirements &amp; Development</b>	(Minimum) Memory: 1 GB RAM, CPU: 2 cores, Disk storage size: 1 GB. It will be provided as a set of docker images that will expose APIs and UIs. So, Docker and Docker compose or Kubernetes (depending on final set up) is needed. The suggested operating system is Linux OS e.g. Ubuntu. The proposed tool to be based on is Keycloak <sup>3</sup> .

<sup>3</sup> <https://www.keycloak.org/>



<b>Interactions</b>	The component will be used by the User Manager, to authenticate, authorize and register users.
---------------------	--

### 3.4.2 Analytics

<b>Objectives / Brief description</b>	This component offers the means to analyze the operation of MODAPTO module(s). This component can be applied either to the production module-level or across production modules.
<b>Requirements &amp; Development</b>	Analytics serve as pre-requisite for the optimization and require access to time-serious process data of production modules for the training phase. The optimization might then happen as an application phase. Analytics may require combining recorded data with other parameters in mathematical formulas. The analytics requires data gathering from the production modules including data access to the time-series data of the production modules. Analytics requires for some parts the graphical representation of information to be analyzed to detect correlations.
<b>Interactions</b>	<p>The Analytics component should divide into two phases:</p> <ul style="list-style-type: none"> <li>- Phase 1: Training (based on recordings of the production modules to find out factors to contribute to e.g. specific sustainability parameters)</li> <li>- Phase 2: Application (with pre-collected information of training phase, based on pre-calculated simulation models or parameters)</li> </ul>

### 3.4.3 API Gateway

<b>Objectives / Brief description</b>	The API Gateway will be responsible for controlling the access to and from the internal APIs of the physical node it is installed.
<b>Requirements &amp; Development</b>	<p>(Minimum) Memory: 2 GB RAM, CPU: 4 cores, Disk storage size: 1 GB.</p> <p>It will be provided as a docker image that will expose APIs and UIs. So, Docker and Docker compose or Kubernetes (depending on final set up) is needed. The suggested operating system is Linux OS e.g. Ubuntu and the proposed solution is Traefik<sup>4</sup>.</p>
<b>Interactions</b>	The component will be used by all components indirectly as in the docker compose or yaml files there has to be a configuration involving the API Gateway.

### 3.4.4 Data Access Interface

<b>Objectives / Brief description</b>	Provides a set of functionalities that enable the MODAPTO system to interact with external Data Sources, like ERPs, sensors etc.
---------------------------------------	--

<sup>4</sup> <https://traefik.io/>



<b>Requirements &amp; Development</b>	TBD
<b>Interactions</b>	Interacts with Data sources, the Repository and the MODAPTO Module.

### 3.4.5 Digital Twin

<b>Objectives / Brief description</b>	This represents the DT of the MODAPTO modules. It enables monitoring of properties as well as executing operations such as simulation or optimization and will be realized based on the AAS specification.
<b>Requirements &amp; Development</b>	<p>The DTs can be implemented using existing open-source software such as FA<sup>3</sup>ST Service<sup>5</sup> (developed by the project partner Fraunhofer IOSB) or Eclipse BaSyx<sup>6</sup>.</p> <p>This type of software can typically run on multiple platforms and operation systems from single-board computers such as Raspberry Pi to regular Windows/Linux/Mac workstations and servers or even containerized in the cloud.</p>
<b>Interactions</b>	<p>Creating a DT happens via the standardized AAS HTTP/REST API and requires providing the AAS model of the DT.</p> <p>Once running, users can interact with the DT using HTTP and/or OPC UA (depending on the configuration) via a standardized API.</p> <p>Synchronization between a DT and its underlying asset(s) is not standardized, i.e. is implementation specific. However, there is ongoing work on creating two AAS sub model templates (Asset Interface Description, Asset Interfaces Mapping Configuration) which aims to provide a standardized way to configure such asset synchronization. Both sub model templates are currently in review and will probably be released in Q1/2024 or early Q2/2024 which might be in time to be used with MODAPTO to define asset synchronization of a DT in an implementation-agnostic way.</p>

### 3.4.6 DT Management

#### 3.4.6.1 Module & DT Manager

<b>Objectives / Brief description</b>	The Module & DT Manager is responsible for creating and managing MODAPTO modules and their associated DTs within the MODAPTO ecosystem.
<b>Requirements &amp; Development</b>	DTs in MODAPTO will be based on the AAS specification. Creating DTs means we need to create an AAS-compliant model file from code based on the user input. This can be done using existing open-source software such as Eclipse AAS4J <sup>7</sup> which allows to create AAS models from code and serialize them in a standard-compliant file format (which can be JSON, XML or AASX).
<b>Interactions</b>	<ul style="list-style-type: none"> <li>• CRUD MODAPTO module information</li> </ul>

<sup>5</sup> <https://github.com/FraunhoferIOSB/FAAAS-Service>

<sup>6</sup> <https://github.com/eclipse-basyx>

<sup>7</sup> <https://github.com/eclipse-aas4j/aas4j>



	<ul style="list-style-type: none"> <li>• Create &amp; manage DTs for MODAPTO modules</li> <li>• Add/Assign Smart DT Service to a MODAPTO module</li> <li>• Register existing MODAPTO module</li> </ul>
--	--

### 3.4.6.2 Non-AAS DT Onboarding Manager

<b>Objectives / Brief description</b>	The Non-AAS DT Onboarding Service, as its name suggests, enables integration and onboarding of existing DTs that are modelled using other standards than the AAS, for example the Digital Twin Definition Language (DTDL) used by Microsoft Azure.
<b>Requirements &amp; Development</b>	TBD
<b>Interactions</b>	<ul style="list-style-type: none"> <li>• Onboard/Import existing standardized non-AAS DT</li> </ul>

## 3.4.7 Message Bus

<b>Objectives / Brief description</b>	The Message Bus will facilitate the exchange of information, commands, and updates between different elements of the MODAPTO system in an asynchronous manner. The Message Bus can support real-time communication, allowing Digital Twins and production modules to share data seamlessly and include information about the status of modules, reconfiguration requests, and other relevant data for decision support and collective intelligence.
<b>Requirements &amp; Development</b>	(Min) Memory: 6GB RAM, CPU: 4 cores, Disk storage size: 100 GB. It will be provided as a set of docker images that will expose APIs and UIs. So, Docker and Docker compose or Kubernetes (depending on final set up) is needed. The suggested operating system is Linux OS e.g. Ubuntu. Potential solutions are Kafka or MQTT.
<b>Interactions</b>	The Orchestrator and potentially any component that needs to interact asynchronously.

## 3.4.8 Modular Production Toolkit

### 3.4.8.1 Design & Reconfiguration Tool

<b>Objectives / Brief description</b>	This component will allow the formal definition of a MODAPTO Module and Production Schedule involving multiple modules, in the form of models (standard based when possible). The abstraction of the MODAPTO Module concepts in form of models will simplify user experience while maintaining a formally defined semantic. This module definition will be used to automatically create the MODAPTO Module Digital Twin instance from the Module & DT Manager, and by any other services that need structured information on the MODAPTO Module or Production Schedule like the Simulation and Optimization.
---------------------------------------	--



<p><b>Requirements &amp; Development</b></p>	<p>This component will be potentially based on ADOxx, a platform that (through definition of the schema/meta-model) allow the creation of custom modelling environment for domain specific concepts (in case of MODAPTO, the modules) with their relations and attributes.</p> <p>ADOxx, thanks of the community contribution and the Adonis commercial product, already contains the meta-model definition of standards like BPMN and other non-standard model types like KPIs, company map, etc. including model specific features like simulation.</p> <p>In MODAPTO we can define a model type for the modules concept, allowing to represent the relevant information of a module, its interaction with other modules, and its behaviors/functionalities (through binding with executable models like BPMN or Workflows), also in the optic to support simulation, optimization and the other MODAPTO relevant functionalities.</p> <p>Integration with other MODAPTO functionalities and pilot specific customizations can be possible thanks to the OLIVE part of the ADOxx platform. This allows to have on the frontend side, the main UI of the Design &amp; Reconfiguration tool as the result of the composition of different “widgets” showing the available features (like the modelling of MODAPTO modules, or their simulation, etc.). On the backend side the OLIVE platform allows to define and orchestrate the needed microservices for the specific widget feature.</p>
<p><b>Interactions</b></p>	<p>It will potentially interact with:</p> <ul style="list-style-type: none"> <li>• User Interface</li> <li>• Orchestrator</li> <li>• Repository</li> <li>• All other components of Modular Production Toolkit</li> <li>• All Pillar 2 MODAPTO Module services in general</li> </ul> <p><b>Inputs:</b> modules info from user design or other components via REST Api; format to define.</p> <p><b>Output:</b> modules info; format to define.</p>

### 3.4.8.2 Evaluation & Decision Support

<p><b>Objectives / Brief description</b></p>	<p>This component offers a solution that can handle many heterogeneous data and events and provide intuitive visualizations for IT and non-IT experts that reveal hidden relationships and insights. Multi-purpose dashboards, timeline inspection of data and preconfigured views are the key innovative features of the tool.</p>
<p><b>Requirements &amp; Development</b></p>	<p>Evaluation &amp; Decision Support component will be based on the Advanced Visualization Toolkit (AVT)<sup>8</sup></p> <p>APIs are required to get the indicator values and any generated events. A JavaScript based web application offers the frontend of AVT. End-users only need a browser to access AVT. A dockerized version of the toolkit is also available</p>
<p><b>Interactions</b></p>	<p>AVT can interact with any component providing data to be visualized for the end users.</p> <p><b>Expected input:</b> Data from knowledge base</p>

<sup>8</sup> <https://avt.aegisresearch.eu/>



**Expected output:** User-friendly web interface

### 3.4.8.3 Virtual Commissioning

<b>Objectives / Brief description</b>	The virtual commissioning component aims at the setup and initial programming of complex automation systems based on digital models and methods. It integrates Production Modules in a virtual environment including 3D geometry and behavior simulation. With the help of virtual commissioning, the PLC and robot programs are pre-programmed and tested by software-technical mapping of the real system.
<b>Requirements &amp; Development</b>	Digital Twin of production module.
<b>Interactions</b>	Set up the 3D geometry and behavior simulation project. Input: Robot backups (zip), PLC programs (ap15), simulation cells (psz), recorded signal data (binary or csv) Output: YAMS/ViPer/SCOUT project (xml and binary)

### 3.4.9 Notification Center

<b>Objectives / Brief description</b>	Manages all the notifications and system alerts generated by the MODAPTO components for the end user.
<b>Requirements &amp; Development</b>	(Min) Memory: 2GB RAM, CPU: 4 cores, Disk storage size: 50 GB. It will be provided as a docker image that will expose APIs. So, Docker and Docker compose or Kubernetes (depending on final set up) is needed. The suggested operating system is Linux OS e.g. Ubuntu.
<b>Interactions</b>	Interacts with Message Bus and User Interface.

### 3.4.10 Optimization / Co-Optimization

<b>Objectives / Brief description</b>	This component offers the means to optimize the operation of MODAPTO module(s) taking into account operational and sustainability constraints and targeting towards specific objectives. This component can be applied either to the production module-level or across production modules.
<b>Requirements &amp; Development</b>	There are 2 types of optimization algorithms that can be developed. The choice of each type depends on the application and the problem size. Each algorithm type has different dependencies. At this point, we can only describe the standard requirements of each type. <ul style="list-style-type: none"> <li>Exact algorithm: provides optimal solutions. Most frequently, it depends on a commercial optimization solver, such as IBM ILOG CPLEX or GUROBI.</li> <li>Heuristic algorithm: Provides close-to-optimal solutions. Usually, it is not dependent on a commercial solver such as above, however there are exceptions. Highly recommended for large-scale optimization problems.</li> </ul>



	<p>Both types of algorithms should return a result within 1 hour (maximum). This stems from the application environment (manufacturing area) and the user requirements (time limitations to reach a decision).</p> <p>At maximum, the hardware limitations of both approaches should not exceed 8 CPU cores and 64GB RAM, while the use of a GPU seems currently unnecessary.</p>
<b>Interactions</b>	<p>The Optimization component should be triggered either by the user or by the system any time a specific event takes place. The required data could involve:</p> <ul style="list-style-type: none"> <li>• Pilot 1: robot type, robot movements, energy consumption, movement duration/time/prerequisites</li> <li>• Pilot 2: planning horizon (hours, days, weeks), production orders, cell capacities, shifts schedule</li> <li>• Pilot 3:             <ul style="list-style-type: none"> <li>- ILTAR: planning horizon (hours, days, weeks), production orders, shifts schedule</li> <li>- Stellantis: set of KH available for picking, production orders, robot movements, movement duration/time/prerequisites</li> </ul> </li> </ul> <p>This data can be either provided by the user or by the system. In both cases, the corresponding optimization algorithms work with batches of data, not streams, incorporating a single data batch per execution.</p>

### 3.4.11 Orchestrator

<b>Objectives / Brief description</b>	The Orchestrator coordinates and manages multiple components, stringing together multiple tasks in order to execute a larger workflow or process based on the user requirements.
<b>Requirements &amp; Development</b>	<p>(Min) Memory: 4 GB RAM, CPU: 4 cores, Disk storage size: 20 GB.</p> <p>It will be provided as a set of docker images that will expose APIs and UIs. So, Docker and Docker compose or Kubernetes (depending on final set up) is needed. The suggested operating system is Linux OS e.g. Ubuntu.</p>
<b>Interactions</b>	Probably with the “high-level” versions of components like Optimization, Simulation the Service catalog etc.

### 3.4.12 Predictive Maintenance

<b>Objectives / Brief description</b>	The predictive maintenance service will provide maintenance strategies at “line level”, by developing dynamic maintenance optimization algorithms for prioritizing and/or scheduling maintenance activities (pillar 2). The objective could be cost minimization and/or resource optimization taking into account the constraints of the use cases (such as the number of people available for maintenance interventions, dependencies between components of the production modules).
<b>Requirements &amp; Development</b>	TBD
<b>Interactions</b>	Users may interact with the algorithm to adjust optimization objectives, set priorities or constraints, or make decisions based on their specific needs.



	<p>The module depends on the data acquisition systems and self-awareness component, which will allow the data collection needed for the training and test of the algorithms.</p> <p>Expected input: equipment data (like list of equipment, cost of components) Maintenance information from Computerized Maintenance Management System (like, maintenance order and reports, historic of failures, duration of maintenance operation, cost of labor), operational data (like production schedule), information related to the resources (like maintenance manpower).</p> <p>Expected output: optimized maintenance strategies at line level, that will be dynamically adapted based on the operative conditions and the optimization goal (e.g., minimization of costs, optimization of resource allocation).</p>
--	--

### 3.4.13 Repository

Objectives / Brief description	Repository stands at the core of the architecture, functioning as a central repository. It seamlessly interacts with components from both Pillar 1 and Pillar 2. This Database Management System (DBMS) offers a RESTful Application Programming Interface (API) connected to an underlying physical data store. In the project context, the Repository is the database for all data required by other MODAPTO components. It facilitates seamless sharing through a RESTful API, tailored to the specific needs of our technical partners.
Requirements & Development	TBD
Interactions	<p><b>Expected input:</b> Any data streams or information from other components that must be stored.</p> <p><b>Expected output:</b> Data Objects. (JSON)</p>

#### 3.4.13.1 System DBs

Objectives / Brief description	A Database that stores diverse data, including information generated by MODAPTO modules/components and preconfigured datasets.
Requirements & Development	TBD
Interactions	<p><b>Expected input:</b> Any data streams or information from other components that must be stored.</p> <p><b>Expected output:</b> Data Objects. (JSON)</p>

#### 3.4.13.2 Data Collection Engine

Objectives / Brief description	A service that collects data through various sources in real-time (e.g. message bus) and aggregates data from external sources of the system, harmonizing them based on the defined data model.
Requirements & Development	TBD



<b>Interactions</b>	<p><b>Expected input:</b> Messages coming to message bus and data from external sources.</p> <p><b>Expected output:</b> Data Objects. (JSON)</p>
---------------------	--

### 3.4.14 Self-Awareness

<b>Objectives / Brief description</b>	<p>This module includes the development and use of algorithm(s) for low level maintenance decisions (as state detection, health monitoring, diagnostics and prognostics). The functionality of the algorithm will be tested at “single-module level” for predictive maintenance decisions. The module will leverage Deep Learning (DL) models. The training of the DL models will be performed off-line thanks to historical databases. Once trained, the algorithm will be made available in MODAPTO to be included in DT.</p>
<b>Requirements &amp; Development</b>	TBD
<b>Interactions</b>	<p>The module depends on the data acquisition systems, which will feed the algorithms once embedded in DT. Furthermore, the output of the algorithms should be made available to be used by other services.</p> <p>Expected input: operational parameters monitored through sensors on the production modules or/and historical data provided by the company or available datasets provided in the literature. Computerized Maintenance Management System maintenance order and reports.</p> <p>Expected output: prevision of the process deviations from normal behavior and diagnostics of the causes, Remaining Useful Life.</p>

### 3.4.15 Service Catalog

<b>Objectives / Brief description</b>	<p>The Service Catalog creates and manages records of smart services within the MODAPTO project</p>
<b>Requirements &amp; Development</b>	<p>(Min) Memory: 4 GB RAM, CPU: 4 cores, Disk storage size: 10 GB.</p> <p>It will be provided as a set of docker images that will expose APIs and UIs. So, Docker and Docker compose or Kubernetes (depending on final set up) is needed. The suggested operating system is Linux OS e.g. Ubuntu.</p>
<b>Interactions</b>	<p>The Orchestrator and any other component that will need service discovery data.</p> <p>Expected input: in general, the components will provide metadata about the service they provide e.g. type of service, inputs, outputs etc. and / or the message topics and messages they use.</p> <p>Expected output: the list of available services and relevant metadata.</p>

### 3.4.16 Simulation / Co-Simulation



<b>Objectives / Brief description</b>	This component offers a means to simulate the execution of MODAPTO module(s) to calculate and return corresponding metrics on their operation. This component can be applied either to the production module-level or across production modules involved in a production decision.
<b>Requirements &amp; Development</b>	<p>It may simulate the operation of a single component or require the coordination of multiple components being simulated simultaneously (co-simulation).</p> <p>Both types of components (co- or simulation) should return a result within 1 hour (maximum). This stems from the application environment (manufacturing area) and the user requirements (time limitations to reach a decision).</p> <p>At maximum, the hardware limitations of both approaches should not exceed 8 CPU cores and 64GB RAM, while the use of a GPU seems currently unnecessary.</p>
<b>Interactions</b>	<p>The Simulation component should be triggered either by the user or by the system any time a specific event takes place. The required data could involve:</p> <ul style="list-style-type: none"> <li>• Pilot 1: robot type, robot movements, energy consumption, movement duration/time/prerequisites</li> <li>• Pilot 2: planning horizon (hours, days, weeks), production orders, cell capacities, shifts schedule.</li> <li>• Pilot 3:             <ul style="list-style-type: none"> <li>- ILTAR: planning horizon (hours, days, weeks), production orders, shifts schedule.</li> <li>- Stellantis: set of KH available for picking, production orders, robot movements, movement duration/time/prerequisites</li> </ul> </li> </ul> <p>This data can be either provided by the user or by the system. In both cases, the corresponding simulation methods work with batches of data, not streams, incorporating a single data batch per execution.</p>

### 3.4.17 Sustainability Analytics

<b>Objectives / Brief description</b>	This component offers the means to analyze the sustainability properties (e.g. energy consumption or carbon emission) of MODAPTO module(s) based on general analytics component considering operational and sustainability constraints (e.g. energy mix) and targeting towards specific objectives. This component can be applied either to the production module-level or across production modules.
<b>Requirements &amp; Development</b>	Analytics serve as pre-requisite for the optimization and require access to time-series process data of production modules for the training phase. The optimization might then happen as application phase. Analytics may require combining recorded data with other parameters in mathematical formulas, e.g. the calculation of carbon emissions out of energy consumption and the energy types used. The analytics requires data gathering from the production modules including data access to the time-series data of the production modules. Analytics requires for some parts the graphical representation of information to be analyzed to detect correlations.
<b>Interactions</b>	<p>The Analytics component should divide into two phases:</p> <ul style="list-style-type: none"> <li>• Phase 1: Training (based on recordings of the production modules to find out factors to contribute to e.g. specific sustainability parameters)</li> </ul>



	<ul style="list-style-type: none"> <li>Phase 2: Application (with pre-collected information of training phase, based on pre-calculated simulation models or parameters)</li> </ul> <p>For Pilot 1, the relevant information includes robot type, robot movements incl. Speed, acceleration, etc., energy consumption, movement duration/time/prerequisites</p>
--	--

### 3.4.18 User Interface

<b>Objectives / Brief description</b>	This component is the main entry point for the user. Potentially, it will be composed of reusable User Interface widgets, some generic like input forms, and other more specifics provided also by other MODAPTO components, allowing to expose and interact with the specific component functionality.
<b>Requirements &amp; Development</b>	TBD
<b>Interactions</b>	<ul style="list-style-type: none"> <li>Access Control</li> <li>Notification Centre</li> <li>Orchestrator</li> <li>Repository</li> <li>Message Bus</li> <li>All components of Modular Production Toolkit</li> </ul>

## 4 Pilot Workflows

This section provides an overview of the workflows and sequence diagrams for indicative use cases, as documented in D3.1. The sequence diagrams map user flows onto the MODAPTO-identified functionalities and components, providing insights into user experience and system usability. They also guide the development team in prioritizing functionalities ensuring user-centric development. Analyzing these workflows helps identify potential issues, allowing timely modifications. This section highlights our commitment to delivering an efficient and effective system, forming the backbone of our approach towards implementing the first MODAPTO prototype. As the project progresses with the development and deployment of additional functionalities to the MODAPTO system, these workflows will be revisited at a more detailed technical level.

### 4.1 Basic System Workflows

Rooted in the insights of Section 6 of D3.1, this section covers the foundational workflows of the MODAPTO system. It focuses on universally applicable features such as resource management, user roles, access rights, and event configuration. Subsections 4.1.1 through 4.1.4 address key aspects of system interaction in detail. It highlights the roles and responsibilities of different user types and the process of configuring and maintaining the system. This section showcases our commitment to developing a flexible and efficient system that caters to the diverse needs of the manufacturing industry.

#### 4.1.1 US0.1: User Management & Access Control

In this user story, a super user of the system e.g. a super administrator or a pilot admin performs a user management action for the users of the MODAPTO system. This action can be to create a new user, update a user's information or delete a user from the system. For simplicity, in the following diagram we focus more on the creation of a new user.

After a successful login, the super user enters the system and navigates to the user management web page. There the super user decides to create a new user and provides the User Interface with the necessary information e.g. username, name, role etc. and submits the data to the MODAPTO system. The data are validated by the User Interface and upon successful validation are sent to the Access Control module. The Access Control module executes the request by updating the users database and sends back the proper outcome message and code, e.g. if it was completed correctly, it will send back the new userID and the HTTP code 200, while if there was an error it will send back HTTP code 500. In case of a success, the Access Control module also sends an email to the new user, in order to notify them that they are now able to access the system. The User Interface, based on the response, presents to the super user the appropriate information message.

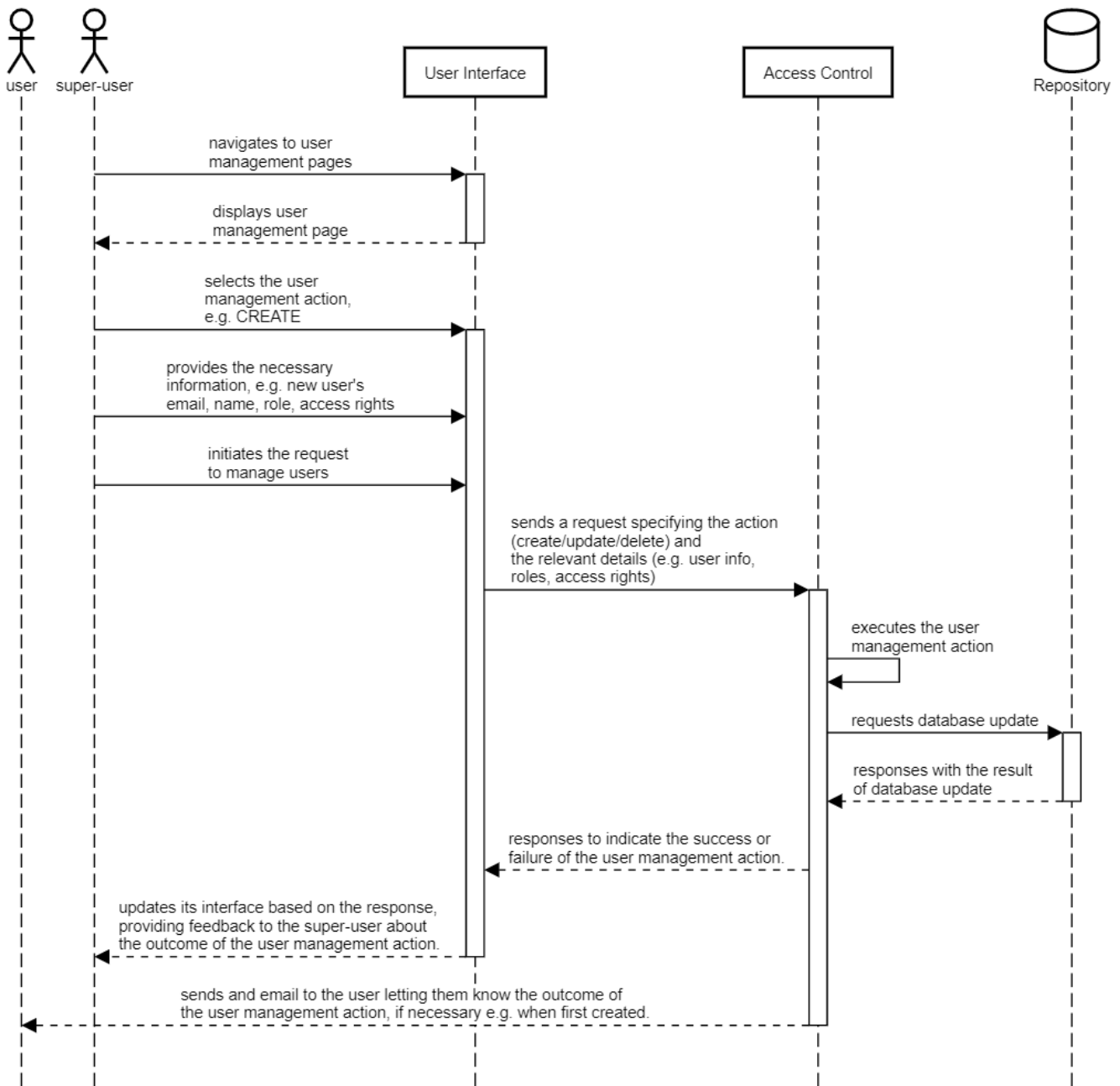


Figure 4. Sequence Diagram for US0.1: User Management & Access Control

### 4.1.2 US0.2: Production Module & Schema (Re-)Configuration

In this user story, shown in Figure 5, the MODAPTO System supports the Super User in the (re-)configuration of a MODAPTO Module or production schema. The MODAPTO User Interface here just exposes the functionality of the Design & Reconfiguration Tool, where the user will interact to define, in form of models, the configuration of a new MODAPTO Module (as a Module Object in a DT Model type) or of a Production schema (as dependencies between referenced Module Objects in a Production Model type).



The user initially can visualize the list of available DT Models and decide to create a new or open an existing one. The Design & Reconfiguration Tool will present a canvas to the user where different Module Objects can be created, and their attributes edited.

As soon as the user finalizes the module configuration and saves the model, the DT Model is stored in the MODAPTO Repository and the model is exported to the Module & DT Manager, responsible to transform it in an AAS standard template, with the help (in case of needs) of the Non-AAS DT Onboarding DT Manager that will support the mapping of the Module Object attributes in case these are not following an AAS standard.

The resulting AAS Template is then sent to the Digital Twin that will initialize new (or update existing) capabilities of MODAPTO Module.

When all the MODAPTO Modules have been configured, a Production Model can be created, referencing the existing Module Objects in the DT Models and adding dependency relations between them.

The stored Production Model will then be available to all the Smart services that require it, like co-optimization or co-simulation.

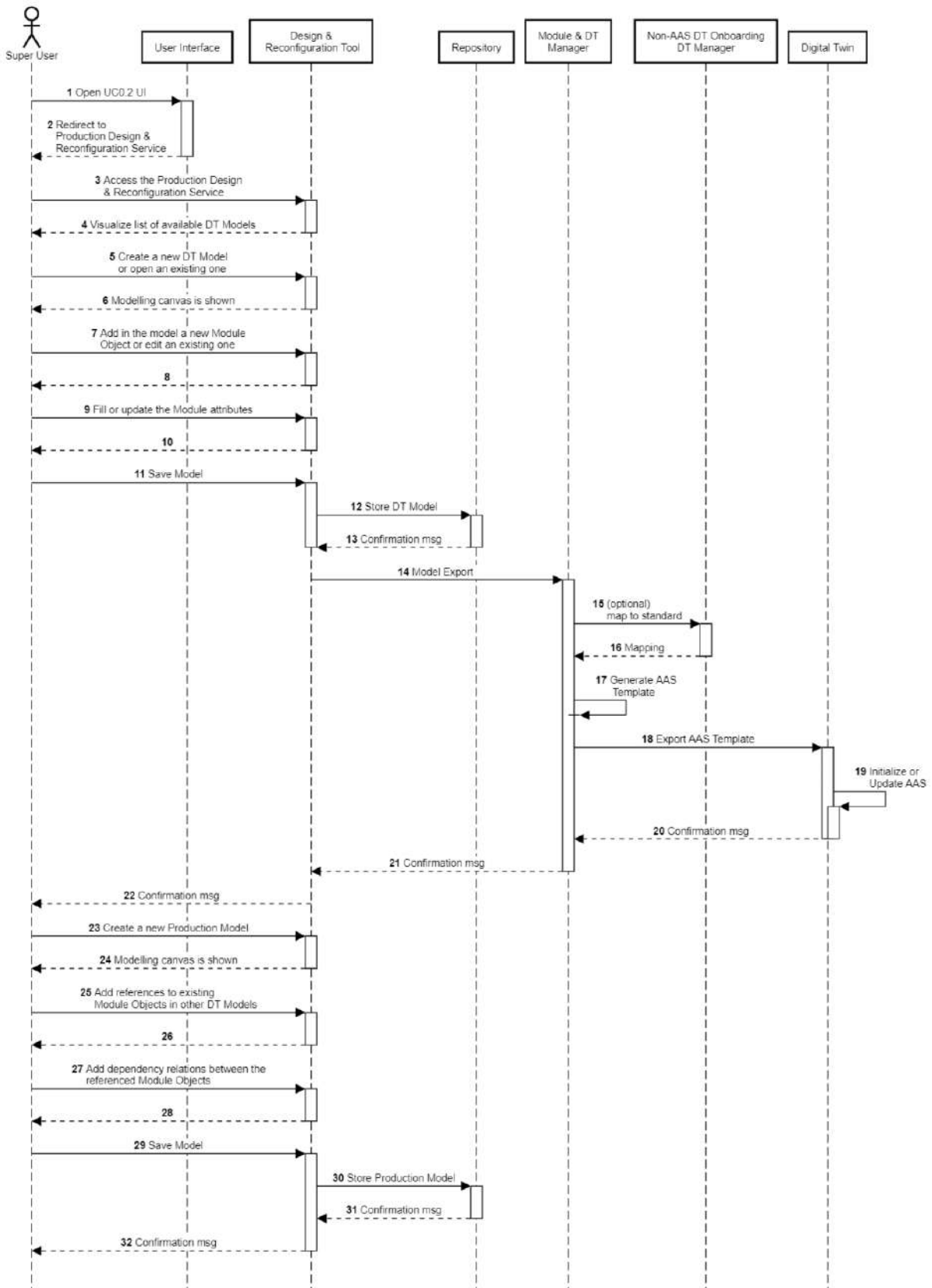


Figure 5. Sequence Diagram for US0.2: Production Module & Schema (Re)Configuration

### 4.1.3 US0.3: Event Configuration

User Story 0.3 involves a super-user configuring or updating custom events in the production environment. This event configuration is a crucial mechanism that will allow the user to receive notifications whenever the specified configured events occur. The primary use case involves composing custom events, which consists of filling out forms with event-related information: event name, type, thresholds, related modules, and a severity level. This story highlights the importance of flexible event management in a dynamic production setting.

Use Case 0.3, "Configure Custom Event," involves a Super-User setting up and defining the parameters of a custom event within the MODAPTO system. This process is essential for tailoring the system's response to specific operational scenarios, enhancing its adaptability to unique production environments. The Super-User defines various aspects of a custom event, such as its type, the conditions that trigger it, the relevant production modules it affects, and the user roles that should be notified and a severity level that will impact the presentation and persistence of the relevant notification incident when this event occurs. The super user-user will have easy access to an interface with all the history of the active or idle past configurations with the ability to create a new configuration or edit an existing one using simple but solid forms. This customization allows the system to react dynamically to specific conditions in the production process, ensuring that relevant personnel are promptly alerted and can take appropriate actions.

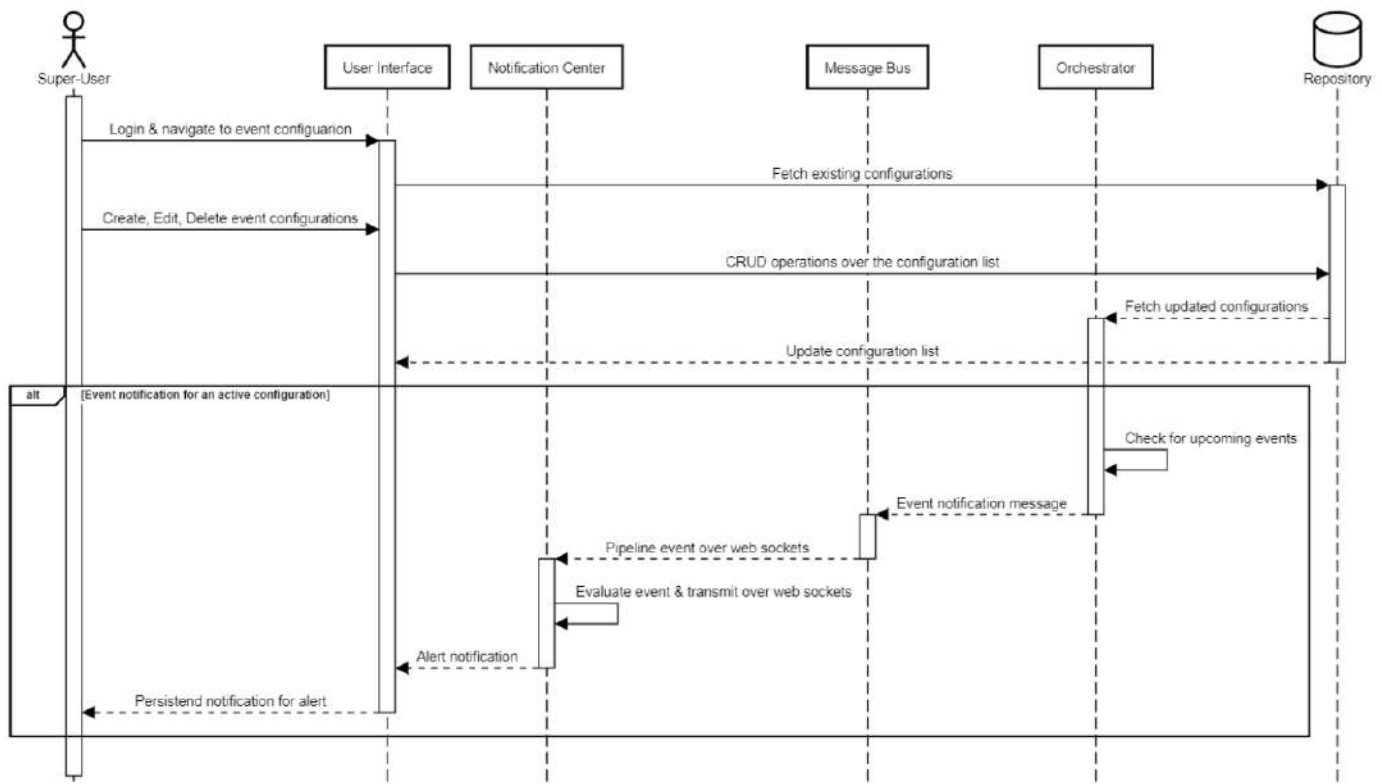


Figure 6. Sequence Diagram for UC0.3: Event Configuration

### 4.1.4 US0.4: User Account Management

US0.4 is about user account management and consists of four use-cases

- UC0.4.1: Active Account
- UC0.4.2: Login

- UC0.4.3: Edit Account Information
- UC0.4.4: Logout

The corresponding workflow is depicted in Figure 7 below.

As a precondition, a new account must be created, and an activation link sent to the user via email. To activate the account (UC0.4.1), the user clicks on the navigation link and provides a new password to be set to the UI. The User Interface instructs the Access Control to update the user password which forwards this request to the Persistence. When done, the result is provided to the user.

Once the account is activated, the user can login (UC0.4.2). For that, the user accesses the User Interface and provides his/her credentials which the User Interface validates via the Access Control. To validate the credentials the Access Control needs to fetch the credentials from the Repository before validating them. Once the credentials have been validated successfully the user is redirected to the landing page.

After login, the user might want to update his/her account information (UC0.4.3). For that that user navigates to the user settings page where he/she can view and update his/her account information. Once sent, the User Interface forwards the updated account information to the Access Control which in turns updates the user data in the Repository.

Finally, once the user logs out, the end of the current session is announced to the Access Control and the user is redirected to the login screen (UC0.4.4).

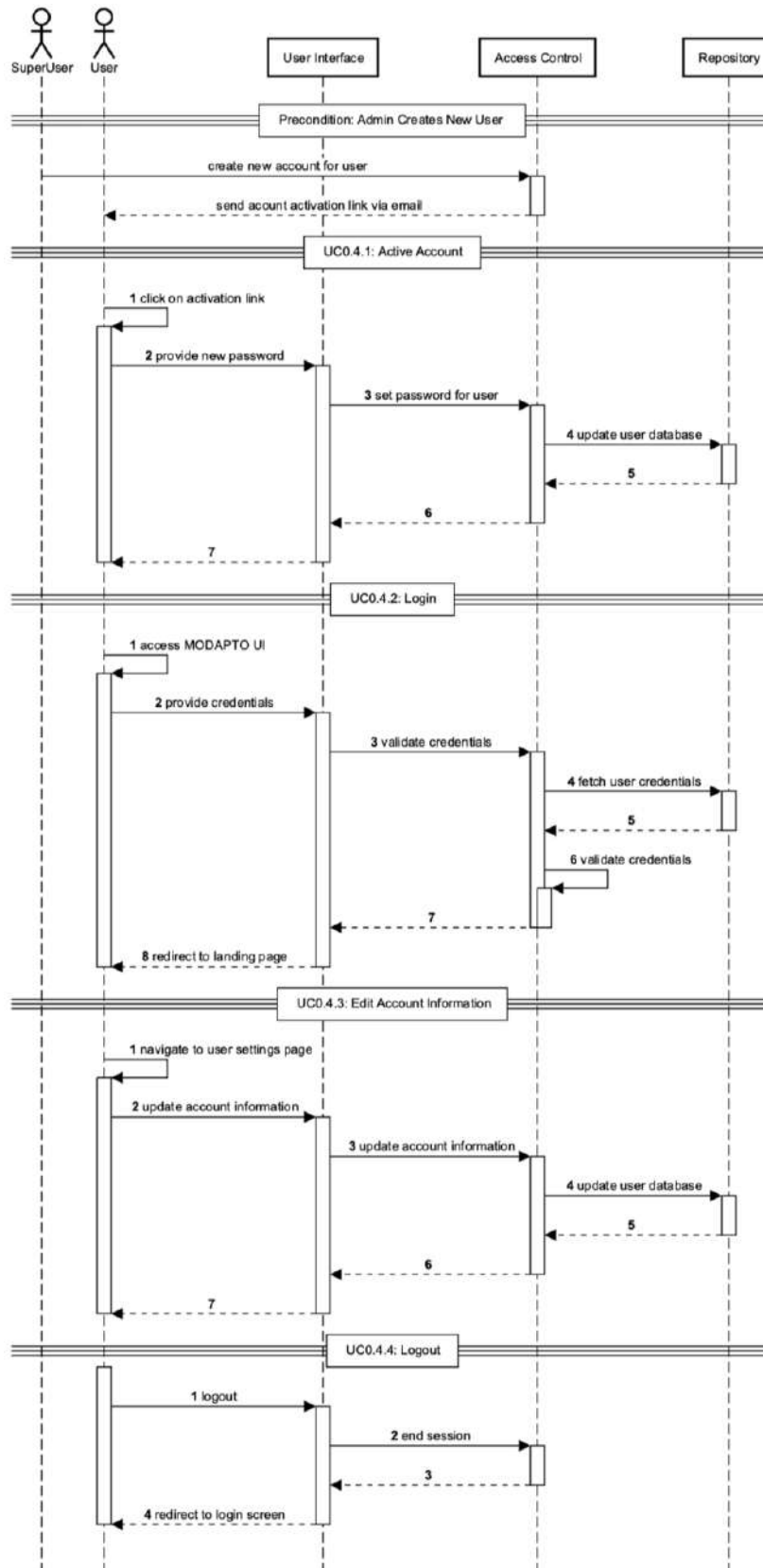


Figure 7. Sequence Diagram for UC0.4: User Account Management.

## 4.2 FFT Pilot

### 4.2.1 UC1.2: Examine Different Robot Design Setups

UC1.2 describes how a Robot Simulation User examines different robot design setups and selects the one that fits best given the operational and sustainability needs and specifications of the customer. The corresponding workflow is depicted in Figure 8.

Preconditions are that the user is logged in, the different robot design setups already have been created (see UC0.2 in Section 4.1.2), and the sustainability analytics have been defined for the different robot design setups (see UC1.1 in deliverable D3.1).

First, the user accesses the “Examine Different Robot Design Setup” page and provides/uploads the task specification. This task specification might include operational aspects such as reachability of the robot, operational time constraints, or existing infrastructure in the manufacturing site, as well as sustainability aspects such as energy consumption constraints. Whether these task specifications are created via MODAPTO or external tools is to be decided in the future.

Once the user submits the task specifications, the User Interface forwards the request to find all robot design setups that can perform the given task to the Evaluation & Decision Support. The Evaluation & Decision Support now first gets a list of all existing robot designs from the Module & DT Manager. As this list only contains limited metadata such as ID and URL of the DTs, the Evaluation & Decision Support fetches the detailed information from each MODAPTO Module Digital Twin directly. Based on this detailed information it can filter them depending on if they can fulfil the task. Once done, the Evaluation & Decision Support Service returns the result to the User Interface which displays it to the user. Now the user can select the preferred robot design setup from the list.

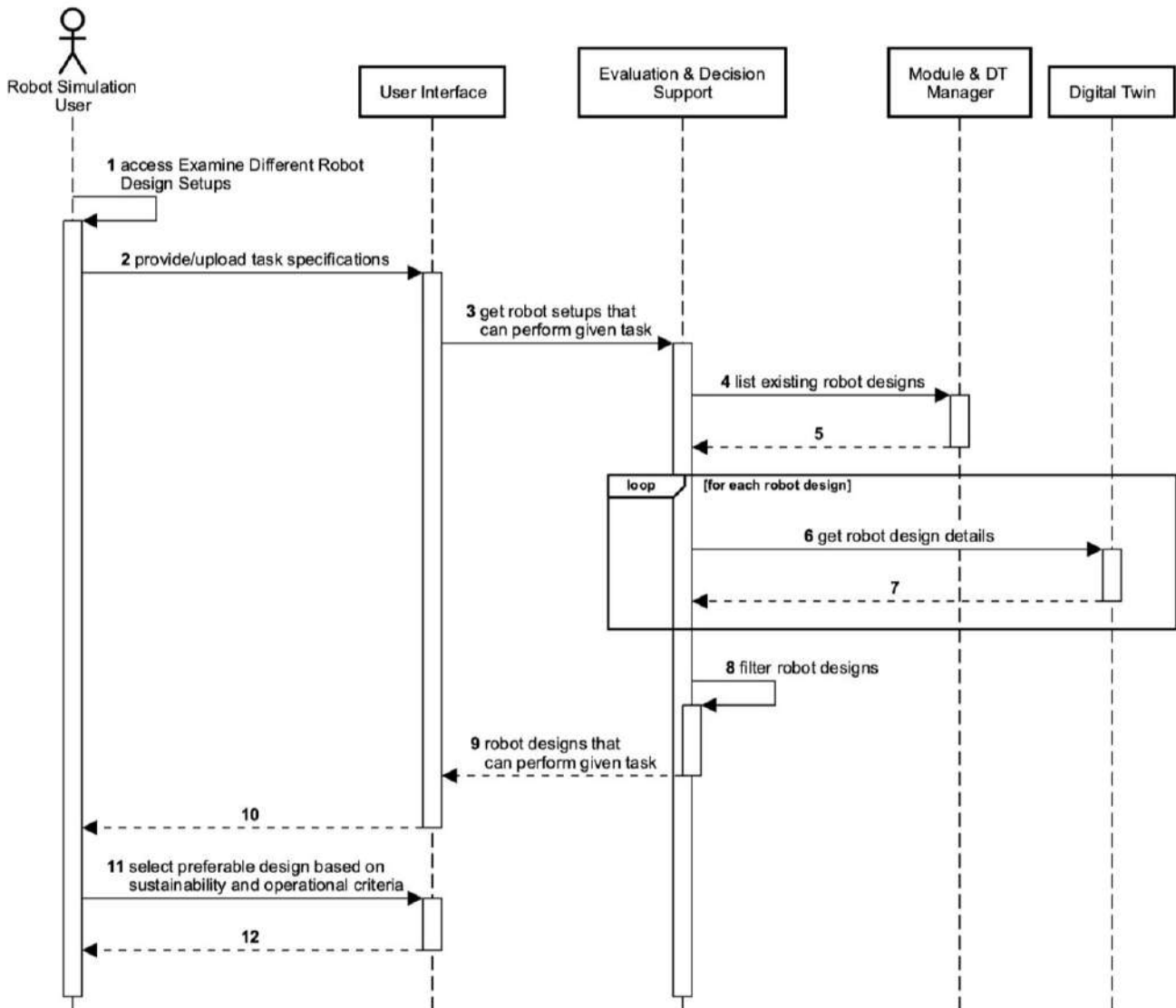


Figure 8. Sequence Diagram for UC1.2: Examine Different Robot Design Setups

## 4.2.2 US1.4: Optimize Robot Movements

In this user story, shown in Figure 9, the MODAPTO System supports the FFT Robot Programmer User in the optimization of a robot (defined as a MODAPTO Module) movement in accordance with provided parameters like carbon emissions and energy consumption.

The MODAPTO User Interface here interacts first with the Design & Reconfiguration Tool to retrieve all the available robot designs in form of MODAPTO Modules and show the list to the user. Then, the robot programmer user selects the desired MODAPTO Module to optimize and will provide the optimization parameters (like sustainability and operational criteria) and constraints (like time/energy limitation).

When all the parameters are provided, the Orchestrator is triggered to first retrieve, from the MODAPTO Repository, the details of the Module to optimize, and then trigger the Optimization of the MODAPTO Module. The optimization job will start in background and as soon as is completed the resulting report will be stored in the MODAPTO Repository, and returned to the



user that could review it and decide to update the MODAPTO Module accordingly in the Design & Reconfiguration Tool. At the end, the user will trigger the execution of the next FFT UC, relative to the Virtual Commissioning, by notifying the Virtual Commissioning User through a message sent via the Message Bus to the Notification Center.

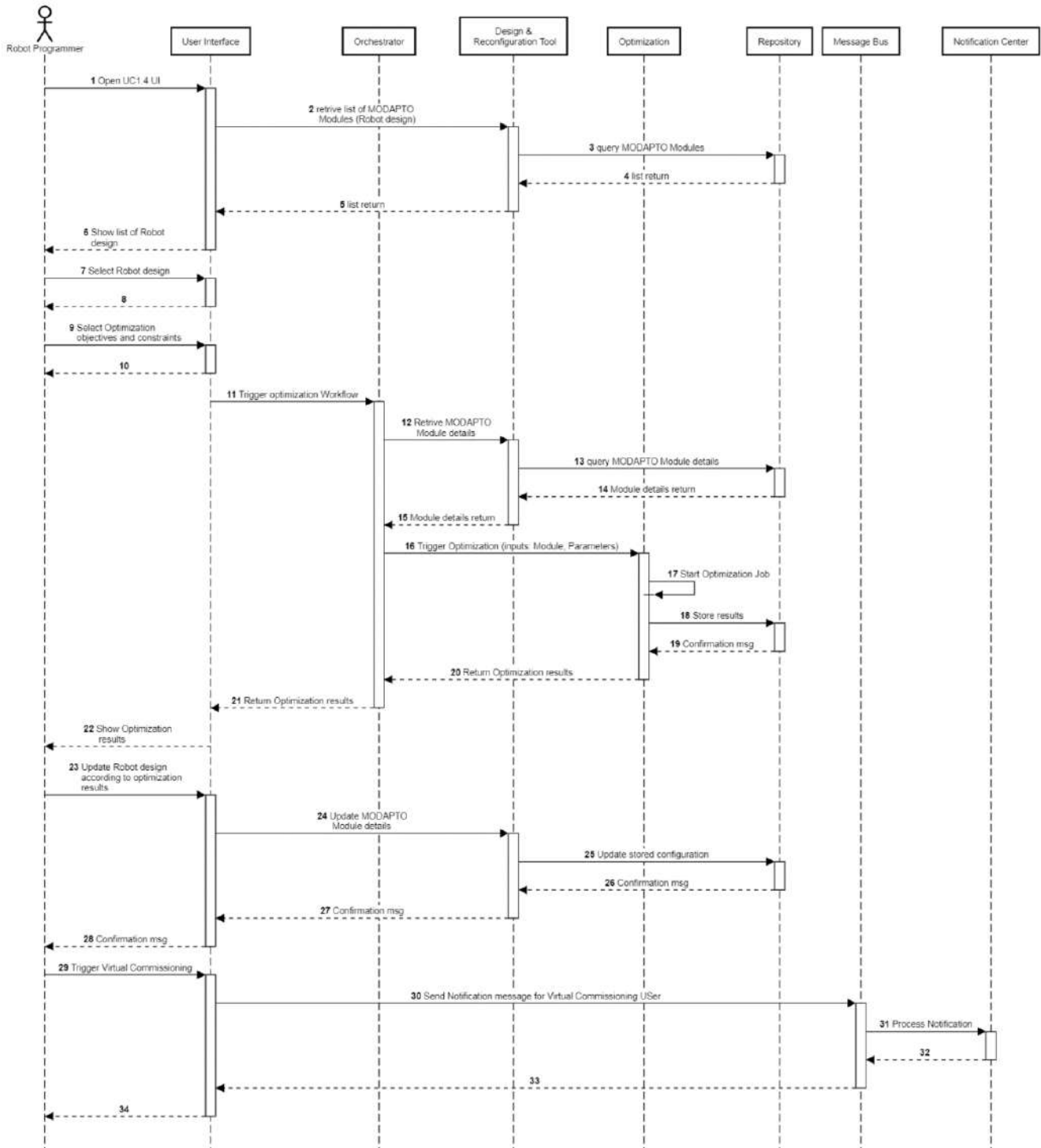


Figure 9. Sequence Diagram for US1.4: Optimize Robot Movements

## 4.3 SEW Pilot

This pilot of MODAPTO involves collaboration with SEW USOCOME, a significant player in drive systems and automation solutions since 1959. The focus is on SEW's assembly plant in Brumath, France, with a significant daily production of motors and gearboxes. SEW's production process, highlighting modularity, enables diverse product variants. Despite an emphasis on rapid delivery, the specific category of gearmotors the pilot is focused on currently follows a monolithic production line. Challenges such as bottlenecks and disruptions persist, despite simulation tools for production line planning. Currently, production planning, scheduling, and managing unexpected events are manually handled by engineering teams. MODAPTO will support SEW's transition from monolithic to modular and reconfigurable production. Focusing on the "small factory" responsible for final product assembly, the project aims to enhance efficiency and adaptability in SEW's manufacturing processes within a facility housing multiple dedicated "small factories."

### 4.3.1 SEW Workflows

#### 4.3.1.1 US2.4: *Manage Predictive Maintenance Notifications and Prioritize Maintenance Actions*

In this user story, the Shop-Floor Manager, Maintenance Engineer, Production Scheduling Team Member, or Innovation Engineer wishes to receive predictive maintenance suggestions for production modules. The focus of the sequence diagram will be on the generation and communication of predictive maintenance suggestions. It is presumed that the user has already logged in to the system.

The Predictive Maintenance Service examines for upcoming maintenance notifications and generates a message of the optimal maintenance suggestions for the designated production modules. These suggestions may include recommended priority for the maintenance work order and associated parameters indicating potential failures. The Message Bus then forwards this message to the Notification Center and stores the related information.

Once the Notification Center receives a notification regarding a suggested maintenance, it sends the notification to the User Interface with use of web sockets. Subsequently, the User Interface presents the received predictive maintenance suggestions to the user in a comprehensible format, for example with a toast notification and by increasing the notification count on the bell icon.

When the user clicks on the notification, the UI displays all the related information for this suggested maintenance. The user can decide on whether they accept the recommended priority for the maintenance work order or not. Their decision is passed to the Notification Center which notifies the Message Bus, leading to the update of relevant stored information.

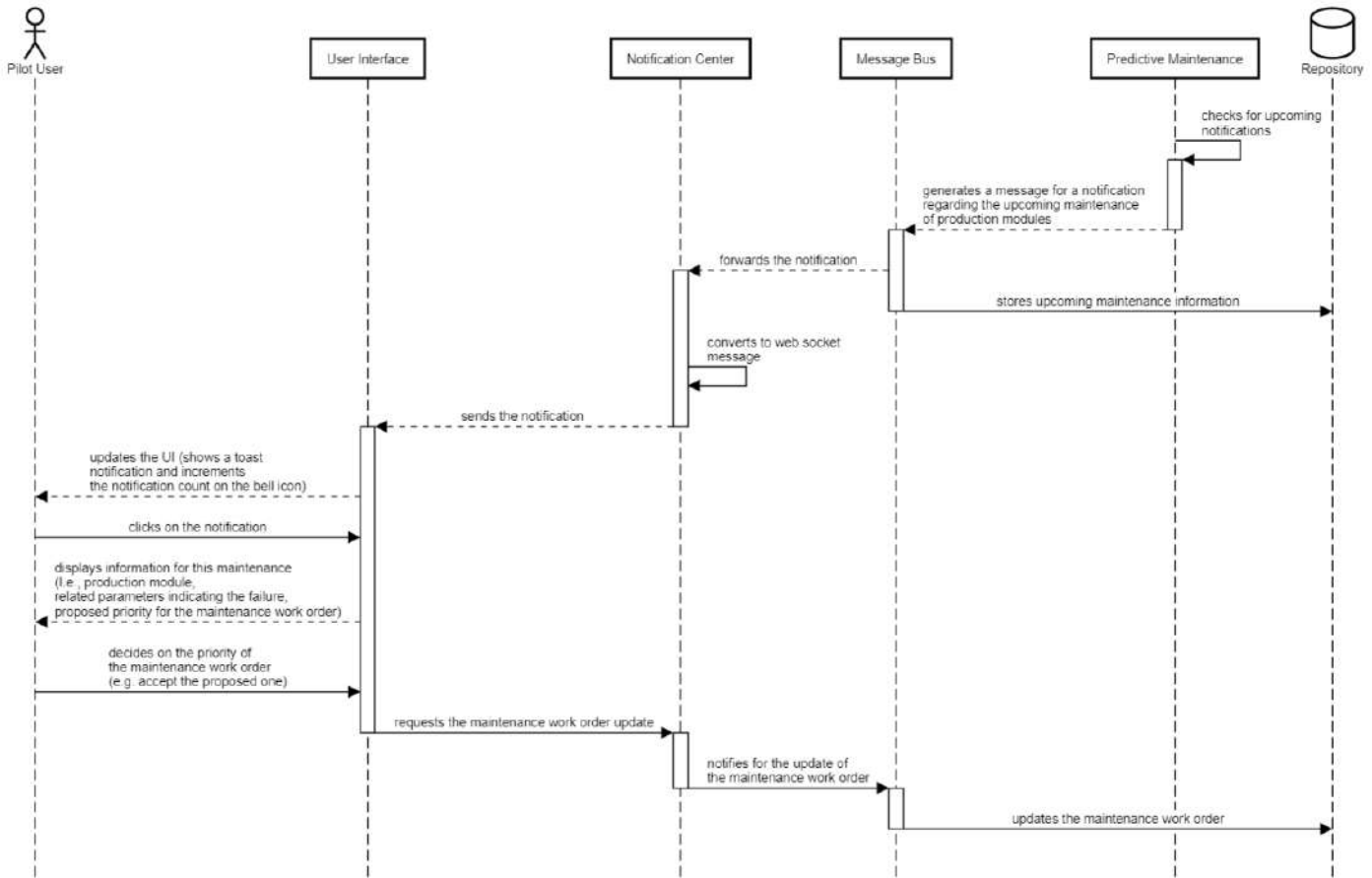


Figure 10. Sequence Diagram for US2.4 Manage Predictive Maintenance Notifications and Prioritize Maintenance Actions

#### 4.3.1.2 US2.6: Re-Optimize Production Schedule

In this user story, the Production Scheduling Team, Shop-Floor Management, or Innovation Engineer initiates the process to re-optimize the production schedule or re-route production based on custom configuration, constraints, and objectives. It is presumed that the user has already logged in to the system.

The user navigates through the User Interface to the section related to the production schedule. The User Interface requests the production schedule, orders, and resources from the Design & Reconfiguration Tool, which retrieves them from the database, and displays the information to the user.

At this point, the user can set the production configuration, constraints, and objectives through the UI. For the purpose of this sequence diagram, it is assumed that the user wishes to re-optimize the production schedule. The Design & Reconfiguration Tool stores re-optimization inputs in the database and sends a message through the Message Bus to signify the re-optimization request. The user is informed about the initialization of the re-optimization process.

The Message Bus triggers the Orchestrator to initiate the Optimization, through the Digital Twin, instructing the Optimization Service to perform re-optimization calculations. Following the calculations, the Optimization Service stores the results in the database and sends the results upon completion to the Digital Twin in order to be forwarded to the Orchestrator. The Orchestrator sends the message through the Message Bus, which creates a notification of the optimization results for the User Interface.

Finally, the User Interface receives the notification and displays it to the user. The user can open the notification, leading to the User Interface displaying the optimization results as a Gantt chart.

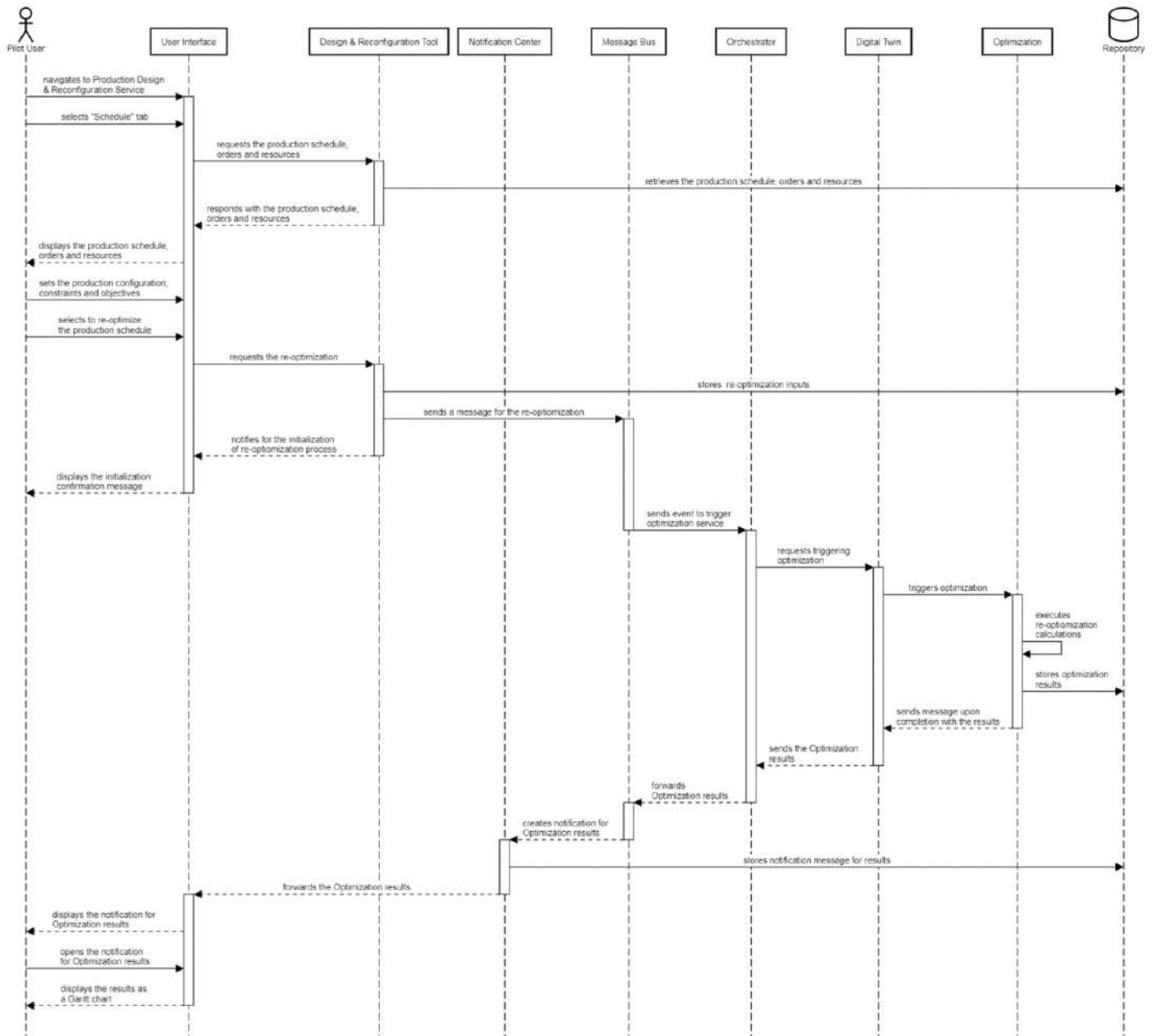


Figure 11. Sequence Diagram for US2.6: Re-Optimize Production Schedule

#### 4.4 CRF/ILTAR Pilot

This section provides an overview of the CRF+ILTAR pilot project, as detailed in Section 5.1 of D3.1. This pilot involves a collaboration between ILTAR (an SME in the polymer material sector) and CRF (an automotive manufacturer subsidiary of Stellantis Group). The pilot focuses on developing and utilizing a modular and intelligent Kit-Holder (KH) designed to enhance warehousing and production operations. The KH, developed by ILTAR, features self-awareness capabilities for real-time monitoring, which aids in optimizing design and production processes. Integrating this advanced KH into CRF's operations aims to automate and streamline warehousing and production workflows. In the following subsections, sequence

diagrams will be presented to depict the User Workflow, illustrating specific Use Cases of this pilot as described in D3.1. This approach will provide a clear visualization of the interactions and processes involved in the pilot, demonstrating the practical application of the MODAPTO system in a real-world manufacturing environment.

## 4.4.1 CRF/ILTAR Workflows

### 4.4.1.1 UC3.2: Receive KH Self-Awareness Notifications

User Story 3.2 in the CRF/ILTAR pilot involves the role of a Production Scheduler at ILTAR. This user story centers on receiving notifications regarding Kit-Holders (KHs) that do not meet the required standards, which are essential for planning new production orders. The story is based on the use case of receiving KH self-awareness notifications, highlighting the need for real-time updates on the condition and compliance of the KHs with the set requirements. This enables proactive planning and scheduling for manufacturing, ensuring the quality and efficiency of the production process.

Use Case 3.2, titled "Receive KH Self-Awareness Notifications," is centered around the Production Scheduler receiving automated alerts concerning the status of kit-holders (KHs) in the production environment. This use case is critical for predictive maintenance and operational continuity, as it allows for immediate response to any KHs not meeting operational standards due to wear or other factors. It's an integral component of the system's self-awareness capabilities, designed to ensure that all equipment functions within the specified requirements, thereby minimizing downtime and maintaining productivity.

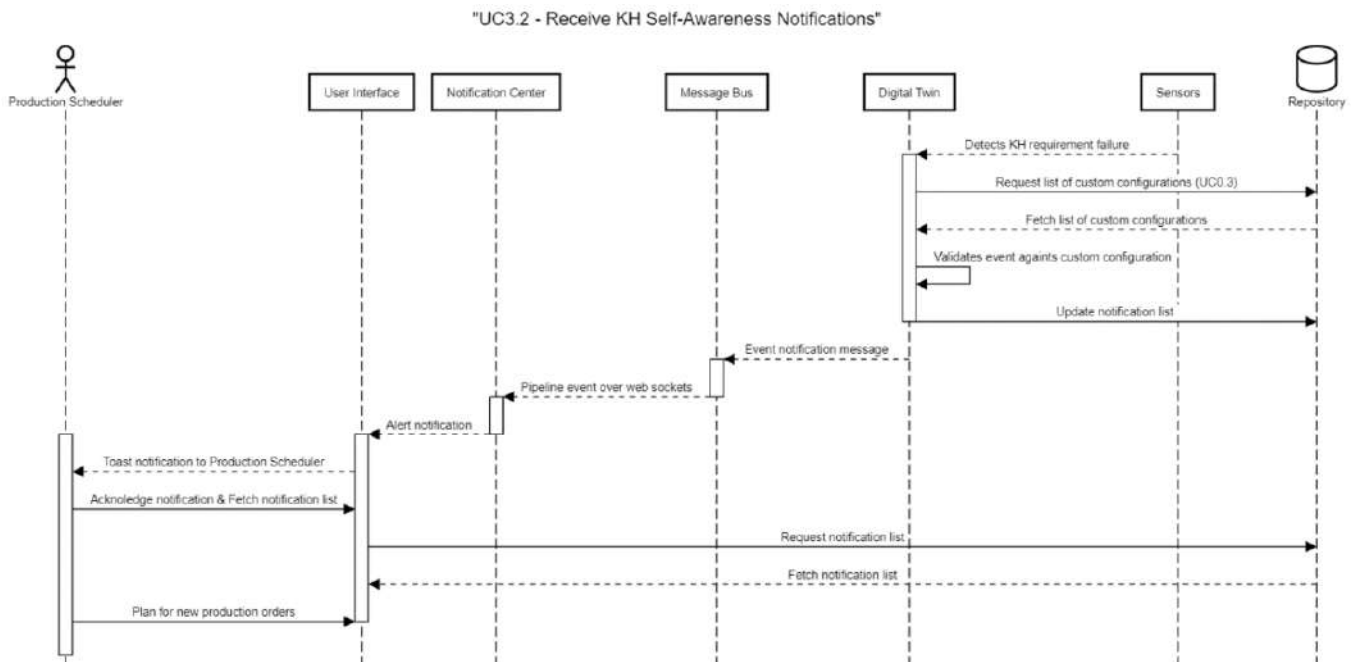


Figure 12. Sequence Diagram for UC3.2: Receive KH Self-Awareness Notifications

### 4.4.1.2 UC3.9: Receive KH Quality Notifications

User Story 3.9 in the CRF/ILTAR pilot involves a Production Manager who needs to receive notifications about Kit-Holders (KHs) not meeting quality requirements. This requirement is crucial for identifying and excluding KHs that are unsuitable for production. The key use case described involves the Production Manager receiving and reviewing these quality-related notifications, which are generated based on the self-awareness capability of each KH. This

process ensures that the production maintains high-quality standards by promptly identifying and addressing any KHS that do not meet the required criteria.

Use Case 3.9, "Receive KH Quality Notifications," is focused on the procedure through which a Production Manager in the MODAPTO system is notified of and responds to quality issues with kit-holders (KHS). This use case is vital for maintaining high-quality standards in the production environment, as it allows for prompt identification and response to any issues with the KHS. The self-awareness capabilities of the KHS enable them to autonomously detect quality problems, ensuring that the Production Manager is quickly informed and can take necessary corrective actions.

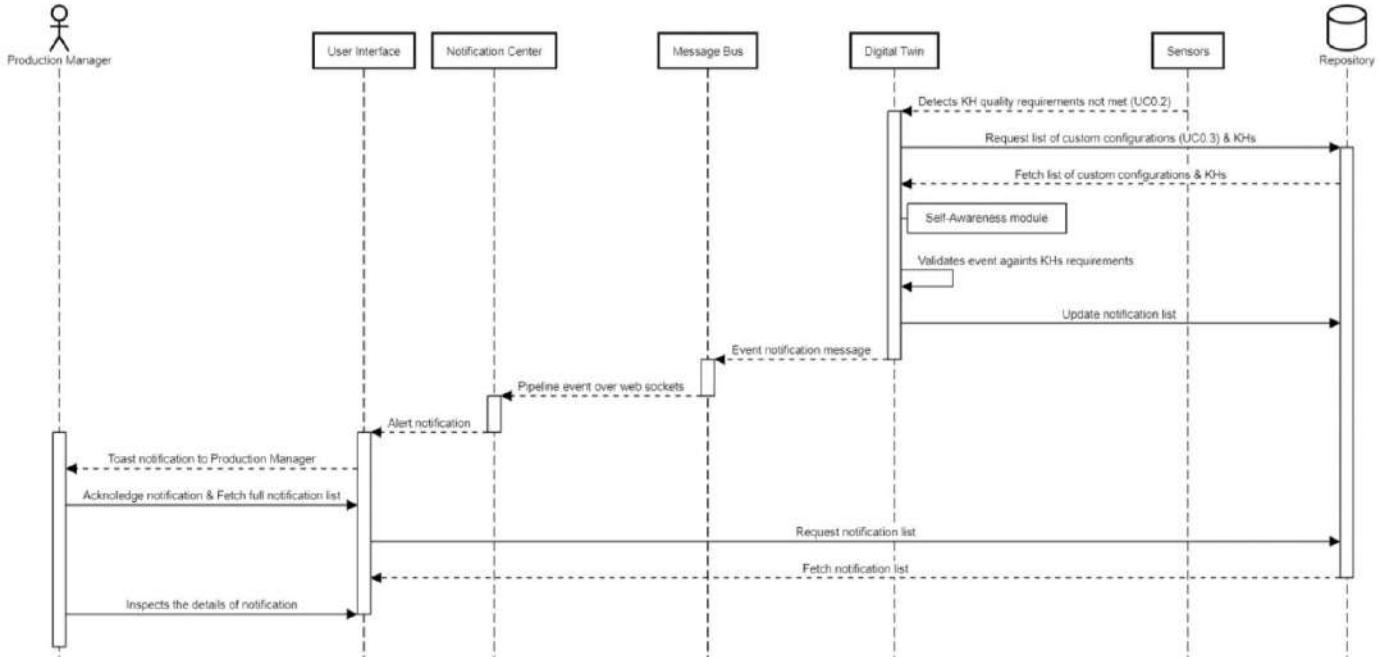


Figure 13. Sequence Diagram for UC3.9: Receive KH Quality Notifications

## 5 Future Work and Next Steps

The MODAPTO project's advancement is marked by a systematic and iterative approach, encapsulating the essence of agile development methodologies. As we envisage the maturation of this deliverable by Month 22, our objective is to enhance the technical granularity and integrate the insights gleaned from our pilot evaluations seamlessly. This process solidifies our commitment to meeting contemporary needs and equips the architecture with the flexibility to address emergent technological trends and market demands.

The forthcoming version of the MODAPTO deliverable will reflect a concerted effort to refine our technical framework and ensure it is robust enough to meet current and future requirements. This refinement will be rooted in a comprehensive analysis of the pilot studies, providing a pivotal feedback mechanism for the project's advancement. We are committed to a blueprint that not only addresses the functionality and interoperability of the system components but also encapsulates a forward-thinking perspective that anticipates the trajectory of technological evolution. Our dynamic approach strives for a resilient and adaptable architecture, facilitating a seamless transition through the project's various phases and beyond.

### 5.1 Expected Revisions and Updates

The upcoming revisions of this deliverable will involve a thorough technical explanation influenced by constructive feedback from the early use case evaluations. In anticipation of this, specific improvements will be directed towards refining the high-level architecture, focusing on aligning it more closely with the nuanced demands of each pilot. The process has been designed to be iterative, ensuring a robust and agile architectural framework.

The forthcoming enhancements to D6.1 will incorporate more technical details rigorously informed by the initial feedback from use case executions. This process will be characterized by a meticulous examination and refinement of the architecture, emphasizing addressing each pilot study's distinct and evolving requirements. Developing detailed sequence diagrams will be pivotal, offering clarity on user workflows and facilitating both interoperability and modularity reinforcement. These enhancements are integral to the project's ethos of agility and will serve as the cornerstone for a robust and adaptive architectural framework for MODAPTO.

The improvements outlined for the second iteration of D6.1 will involve a thorough technical deepening shaped by constructive feedback from the initial feedback cycles. The process will ensure that the architecture meets the specific requirements identified from the pilot studies and is robust enough to support the dynamic nature of the modular production systems. The envisioned architectural refinement will enable MODAPTO to stand at the forefront of distributed control and modular manufacturing while embodying the principles of agile and responsive development.

### 5.2 Prioritization in Development of MODAPTO Components & Technologies

In the Annex section, Annex 3 presents a detailed mapping of the “Must Have” TRs to the relevant “Must Have” User Requirements (User Stories (US) and Use Cases (UCs)). This mapping aided in the prioritization of MODAPTO components development, as presented in Table 16 below. The components listed in this table are aligned with the “Must Have” technical requirements (TRs) either directly or through their critical support of functionalities associated with those requirements:



**Table 16. MODAPTO Components prioritization Mapping**

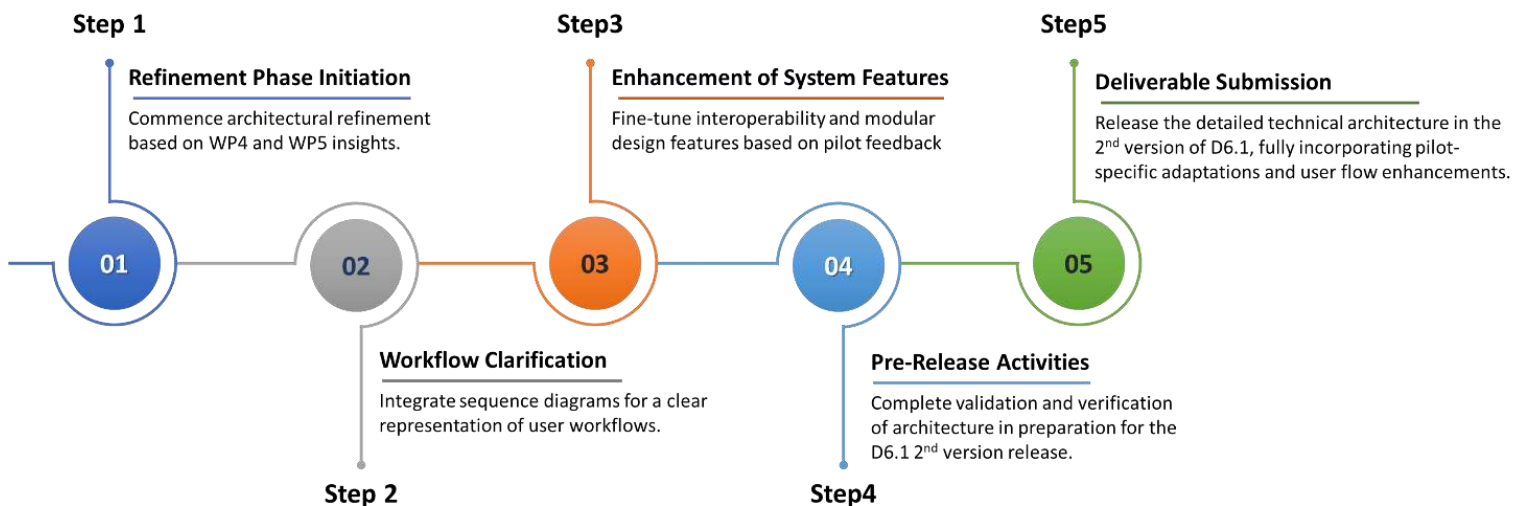
Component	Key Related TR(s)	Importance
Identity Manager	TR1.1, TR1.2	Essential for securing the system through user authentication and authorization, foundational for all user interactions within the system.
Digital Twin	TR3.1, TR4.2, TR4.8	Central to the digital twin technology of MODAPTO, enabling management and representation of production modules.
Module & DT Manager	TR3.1, TR3.2, TR4.2	Facilitates the configuration and management of digital twins and MODAPTO modules, crucial for the flexibility of the production environment.
Message Bus	TR3.5, TR4.3	Supports real-time communication within the system, essential for monitoring and operational responsiveness.
Repository	TR4.1	Acts as the central storage for system data, ensuring data availability and supporting data-driven functionalities.
Design & Reconfiguration tool	TR3.1, TR3.2	Allows users to define and configure MODAPTO modules, directly affecting the system's adaptability to changing production needs.
Evaluation & Decision Support	TR3.6	Provides the necessary tools for data analysis and decision-making, vital for real-time evaluation and operational insights.
Non-AAS DT Onboarding Manager	TR4.6	Enables the system to integrate digital twins that are not compliant with the AAS standard, ensuring broader interoperability.
Optimization / Co-optimization	TR5.2, TR5.3	Critical for the system's optimization capabilities, providing suggestions for the most efficient production courses of action.
Orchestrator	TR5.9	Manages and coordinates complex workflows across the system, integrating various components for seamless system operations.
Predictive maintenance Service	TR5.7	Predicts maintenance needs to improve system reliability and reduce downtime, which is crucial for continuous production.
Self-awareness module	TR5.6	Monitors the health of production assets, providing data that's essential for maintenance and system reliability.
Service Catalog	None directly, supports TR5.9	While not directly associated with a "Must Have" TR, it supports the Orchestrator, which is associated with a "Must Have" TR, by managing smart service metadata.
Simulation / Co-Simulation	TR5.4	Enables testing and validation of production strategies through simulation, which is key for operational planning and risk mitigation.
API Gateway	TR6.1A, TR6.1B	Secures and manages access to the system's APIs, ensuring secure data transmission and supporting the system's integration capabilities.
Virtual Commissioning	None directly, supports TR7.4, TR7.5	Important for testing and validating production setups virtually, but not directly associated with a "Must Have" TR. It's included due to its support for system configuration and deployment, which are linked to "Must Have" TRs in the broader sense.

This targeted approach ensures that development efforts will be concentrated on the most critical aspects of the MODAPTO system, thereby optimizing resource allocation and development timelines.

### 5.3 Implementation Timeline and Milestones

Building on the robust foundation laid out in the current iteration of D6.1; the next phase will involve a deep dive into the technical architecture, informed by the nuanced feedback received from the execution and evaluation of the use cases. This phase is presented in Figure 14 below in a series of steps that will be followed:

- As a 1<sup>st</sup> step moving forward, we will synthesize the feedback from WP4 and WP5 to refine the architectural details.
- Step 2 will see the integration of detailed sequence diagrams to elucidate the user workflows within the system.
- Following this, in step 3 we will fine-tune the interoperability features and modular design, ensuring the architecture is robust and fully aligned with the specific needs identified in the pilot studies.
- The final preparations for the 2<sup>nd</sup> version of D6.1, including enhancements and architecture validation integration, will be conducted as part of the 4<sup>th</sup> Step.
- 



**Figure 14. Roadmap to Submission of 2nd Version of D6.1**

As we approach the final step (step 5), the 2<sup>nd</sup> version of deliverable D6.1 will be refined by Month 22, incorporating a detailed technical architecture responsive to pilot-specific demands. The sequence diagrams will be integrated to clarify user interactions and enhance interoperability features. The timeline leading up to this milestone includes the integration of feedback, technical detailing, and validation activities, all adhering to the agile ethos of MODAPTO, thus ensuring the deliverable is robust, responsive, and aligned with the objectives outlined in the Grant Agreement and the work plan structure.

## 6 Conclusions

In this initial version of the MODAPTO deliverable, we have laid a comprehensive groundwork detailing the conceptual framework, technical specifications, and architectural designs central to the project. The deliverable meticulously addresses integrating modular manufacturing concepts with advanced digital twin technologies. This integration is pivotal for enhancing real-time monitoring and optimization of manufacturing processes and setting a precedent in the industry for flexibility and efficiency. The architecture, developed with a forward-looking vision, ensures that MODAPTO aligns with the ambitious goals of Industry 4.0, fostering a new standard in manufacturing processes.

The document also sets a clear trajectory for the project's technical progression. Recognizing the dynamic nature of technological advancement, this initial version is a foundational stone for future iterations. Scheduled for submission in Month 22, the second version of the deliverable will reflect the evolutionary journey of MODAPTO, incorporating new insights, technological advancements, and feedback received from this version. This iterative approach ensures continual improvement and relevance of the project. It exemplifies a commitment to adaptive and responsive project management in rapidly evolving industrial needs and challenges.

In conclusion, this initial version of deliverable D6.1 of MODAPTO represents a significant milestone in the project's trajectory. It sets high standards for future advancements and serves as a blueprint for future progress.



## 7 References

Fischer, A., Llorens, J., Cai, Z., Wilke, M., Kessler, S., & Fottner, J. (2022). Implementation of a Digital Twin Framework in the Modular Housing Industry. 2022 IEEE 28th International Conference on Engineering, Technology and Innovation (ICE/ITMC) & 31st International Association For Management of Technology (IAMOT) Joint Conference, 1-9. <https://doi.org/10.1109/ICE/ITMC-IAMOT55089.2022.10033306>.

ISO 23247-2, 2021. Automation systems and integration, Digital twin framework for manufacturing, Part 2: Reference architecture. <https://www.iso.org/standard/78743.html>

Martinez, E., Ponce, P., Macias, I., & Molina, A. (2021). Automation Pyramid as Constructor for a Complete Digital Twin, Case Study: A Didactic Manufacturing System. Sensors (Basel, Switzerland), 21. <https://doi.org/10.3390/s21144656>.

Negri, E., Berardi, S., Fumagalli, L., & Macchi, M. (2020). MES-integrated digital twin frameworks. Journal of Manufacturing Systems, 56, 58-71. <https://doi.org/10.1016/j.jmsy.2020.05.007>.

Project Reference No MODAPTO - 101091996. (2023). D3.1 - Requirements and KPIs.



## 8 Annexes

### Annex 1 Conceptual Architecture Pilot Workflows

#### Basic System Workflows

##### *US0.1: User Management & Access Control*

In this user story, a super user interacts with the MODAPTO system to manage users. Actions include creating, updating, or deleting users. The focus here will be on creating a new user in the diagram.

After logging in, the super user navigates to the user management page, creates a new user by providing necessary details (e.g., username, name, role), and submits the data. The User Interface validates the data and forwards it to the Access Control Service module. Upon successful validation, the module processes the request and returns an outcome message. If successful, the module also sends an email to the new user to notify them of their system access. The User Interface displays appropriate information based on the response.

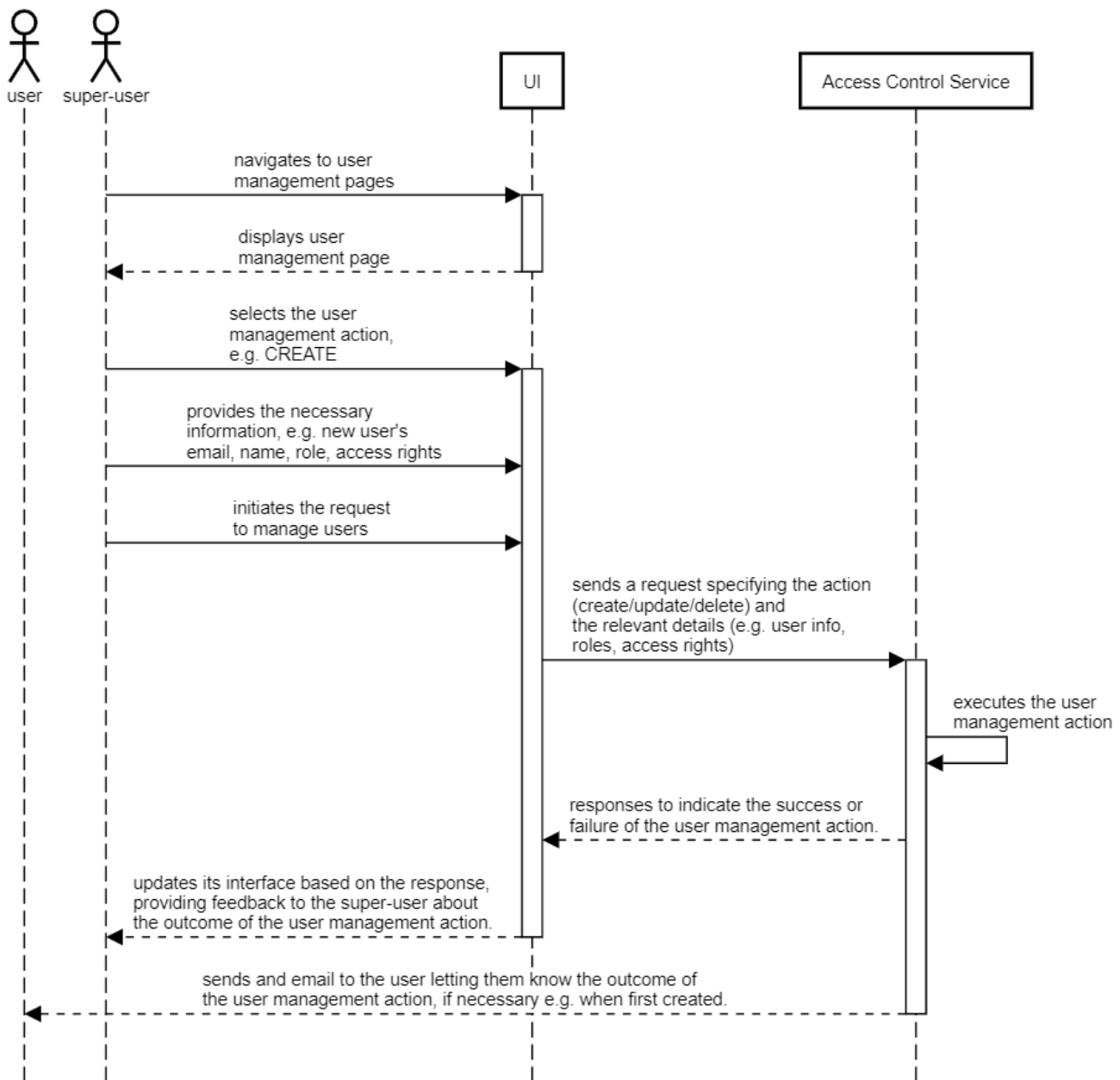


Figure 15 . User Management & Access Control

### UC0.2: Production Module & Schema Reconfiguration

In this use case, the MODAPTO System supports the Super User in the (re-)configuration of a MODAPTO Module or production schema.

The UI here just expose the functionality of the Production Design & Reconfiguration Service, where the user will interact to define, in form of models, the configuration of a new MODAPTO Module (as a Module Object in a DT Model type) or of a Production schema (as dependencies between referenced Module Objects in a Production Model type).

The user initially can visualize the list of available DT Models and decide to create a new or open an existing one. The MODAPTO System will present a canvas to the user where different Module Objects can be created, and their attribute edited.

As soon as the user finalize the module configuration and save the model, the MODAPTO System will store the DT Model in its own repository and export the model to the Module & DT Management Service, responsible to transform it in an AAS standard template, with the help



(in case of needs) of the Non-AAS DT Onboarding Service that will support the mapping of the Module Object attributes in case these are not following an AAS standard.

The resulting AAS Template is then sent to MODAPTO Module API that will initialize new (or update existing) capabilities of MODAPTO Module.

When all the MODAPTO Modules have been configured, a Production Model can be created, referencing the existing Module Objects in the DT Models and adding dependency relations between them.

The stored Production Model will then be available to all the Smart services that require it, like optimization or co-simulation.

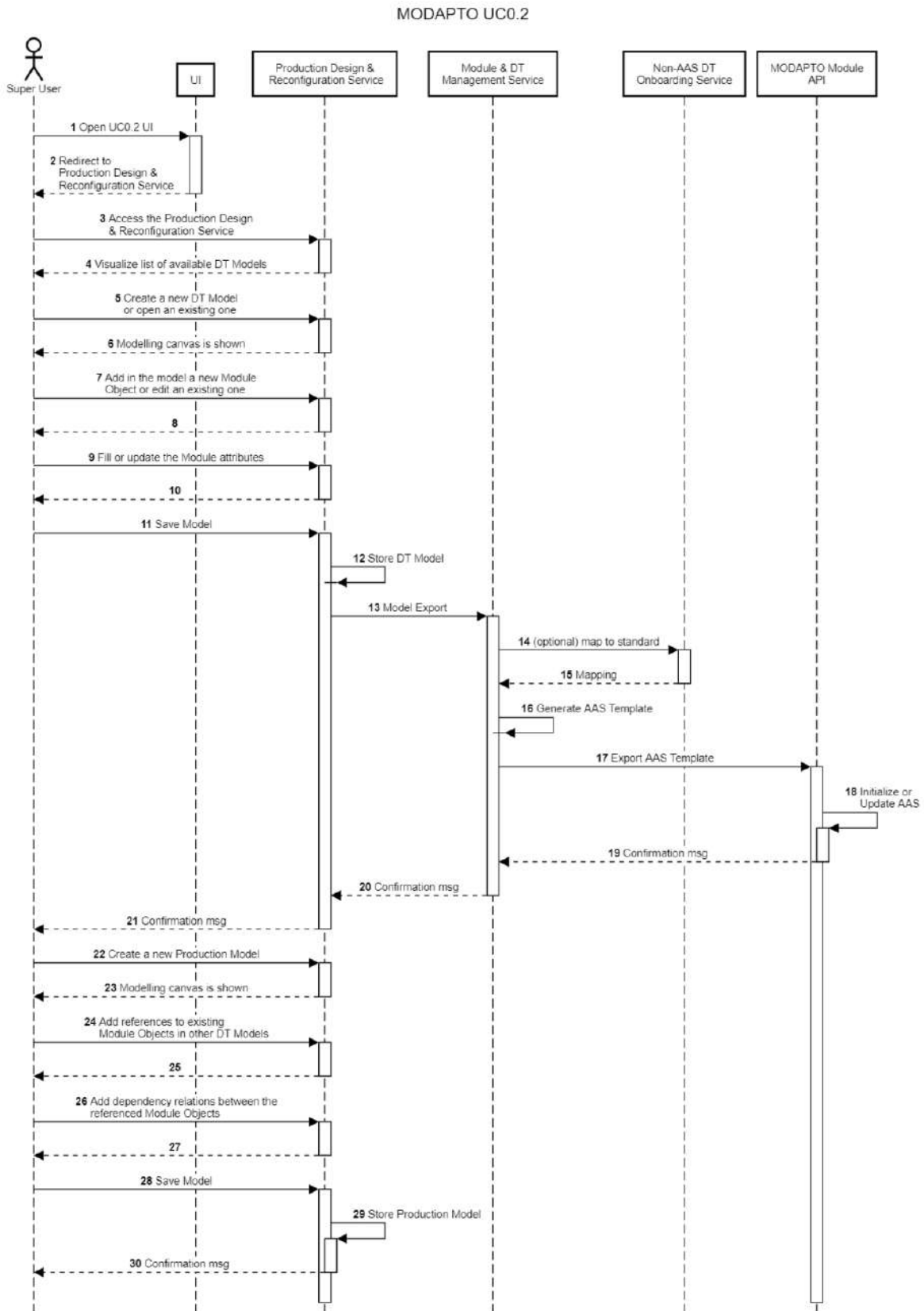


Figure 16. Production Module & Schema Reconfiguration

### UC0.3: Event Configuration

User Story 0.3 in the MODAPTO project involves a super-user configuring or updating custom events in the production environment. The focus is on ensuring that notifications are triggered whenever these events occur. The primary use case involves composing custom events, which consists of filling out a form with necessary information like event name, type, triggers, and related modules. This story highlights the importance of flexible event management in a dynamic production setting.

Use Case 0.3, "Configure Custom Event," involves a Super-User setting up and defining the parameters of a custom event within the MODAPTO system. This process is essential for tailoring the system's response to specific operational scenarios, enhancing its adaptability to unique production environments. The Super-User defines various aspects of a custom event, such as its type, the conditions that trigger it, the relevant production modules it affects, and the user roles that should be notified. This customization allows the system to react dynamically to specific conditions in the production process, ensuring that relevant personnel are promptly alerted and can take appropriate actions.

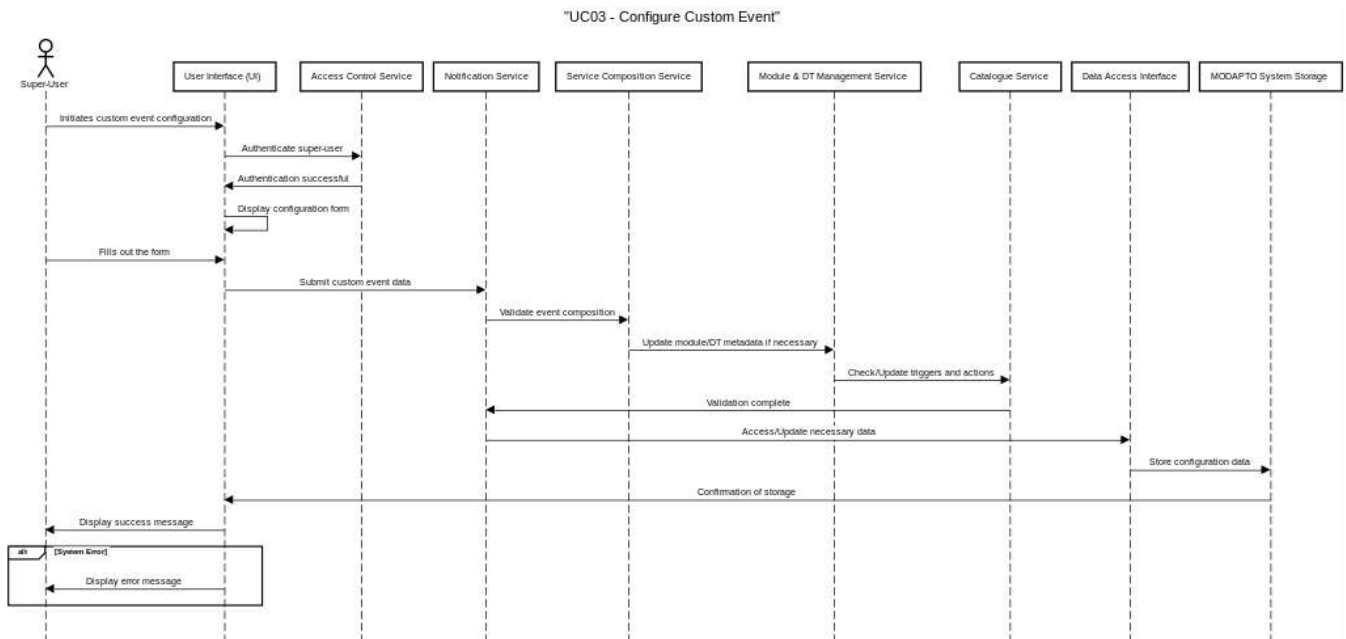


Figure 17. Event Configuration

The sequence diagram for UC0.3 presented in Figure 17 outlines these steps: Initially, the super-user interacts with the User Interface (UI) to initiate the configuration process. The UI, in turn, requests authentication from the Access Control Service, which confirms the super-user's credentials. Upon successful authentication, the User Interface presents a form to the super-user, who fills out the necessary details for the custom event and submits it back to the User Interface. The User Interface then forwards this data to the Notification Service, which engages the Service Composition Service to validate the custom event's composition. Assuming validation is successful, the Service Composition Service may instruct the Module & DT Management Service to update relevant metadata, while also interacting with the Catalogue Service to register event triggers and actions. Concurrently, the Data Access Interface may be invoked to handle any required data transactions. Once all information is processed and validated, the MODAPTO System Storage persists the custom event configuration. Finally, the User Interface confirms the successful storage of the configuration to the super-user and

displays an appropriate success message. An alternative flow is considered for system errors, where the User Interface informs the super-user of any issues encountered during the process.

#### ***UC0.4: User Account Management***

UC0.4 is about user account management and consists of four sub-use-cases

- UC0.4.1: Active Account
- UC0.4.2: Login
- UC0.4.3: Edit Account Information
- UC0.4.4: Logout

The corresponding workflow is depicted in Figure 18.

As a precondition, a new account must be created, and an activation link sent to the user via email. To activate the account (UC0.4.1), the user clicks on the navigation link and provides a new password to be set to the User Interface. The User Interface instructs the Access Control Service to update the user password. Once done, the result is provided to the user. Once the account is activated, the user can login (UC0.4.2). For that, the user accesses the User Interface and provides his/her credentials which the User Interface validates via the Access Control Service. Once the credentials have been validated successfully the user is redirected to the landing page. After login, the user might want to update his/her account information (UC0.4.3). For that that user navigates to the user settings page where he/she can view and update his/her account information. Once sent, the User Interface forwards the updated account information to the Access Control Service.

Finally, once the user logs out, he/she will be redirected to the login screen (UC0.4.4).

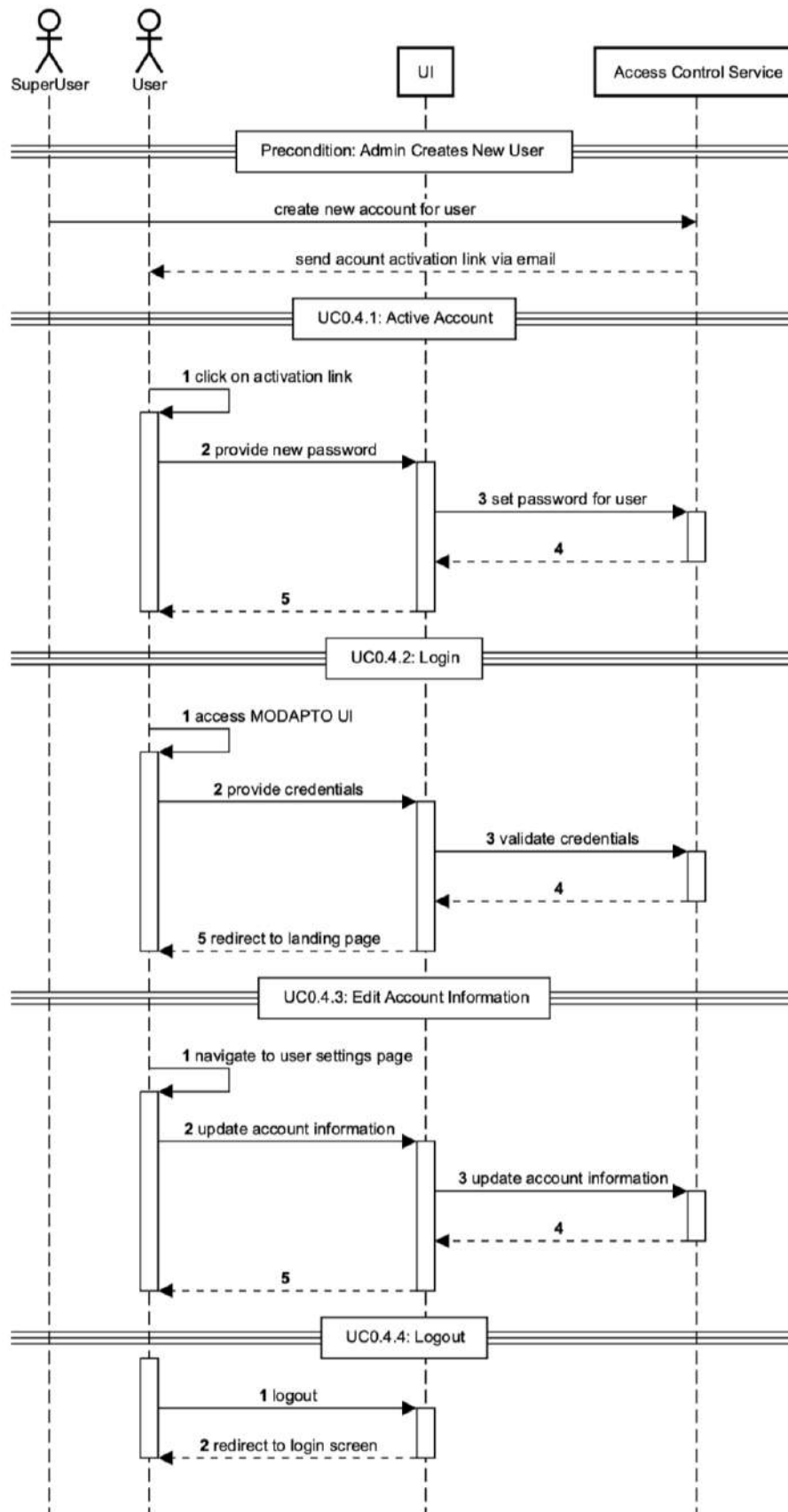


Figure 18: User Account Management

## FFT Workflows

### UC1.2: Examine Different Robot Design Setups

UC1.2 describes how a Robot Simulation User examines different robot design setups and selects the one that fits best given the operational and sustainability needs and specifications of the customer. The corresponding workflow is depicted in Figure 19.

Preconditions are that the user is logged in, the different robot design setups already have been created (see UC0.2 in Section 4.1.2), and the sustainability analytics have been defined for the different robot design setups (see UC1.1).

First, the user accesses the “Examine Different Robot Design Setup” page and provides/uploads the task specification. This task specification might include operational aspects such as reachability of the robot, operational time constraints, or existing infrastructure in the manufacturing site, as well as sustainability aspects such as energy consumption constraints. Whether these task specifications are created via MODAPTO or external tools is to be decided in the future.

Once the user submits the task specifications, the User Interface forwards the request to find all robot design setups that can perform the given task to the Evaluation & Decision Support Service which fulfils the request by fetching all available robot design setups from the Module & DT Management Service and filtering them depending on if they can fulfil the task. Once done, the Evaluation & Decision Support Service returns the result to the UI which displays it to the user. Now the user can select the preferred robot design setup from the list.

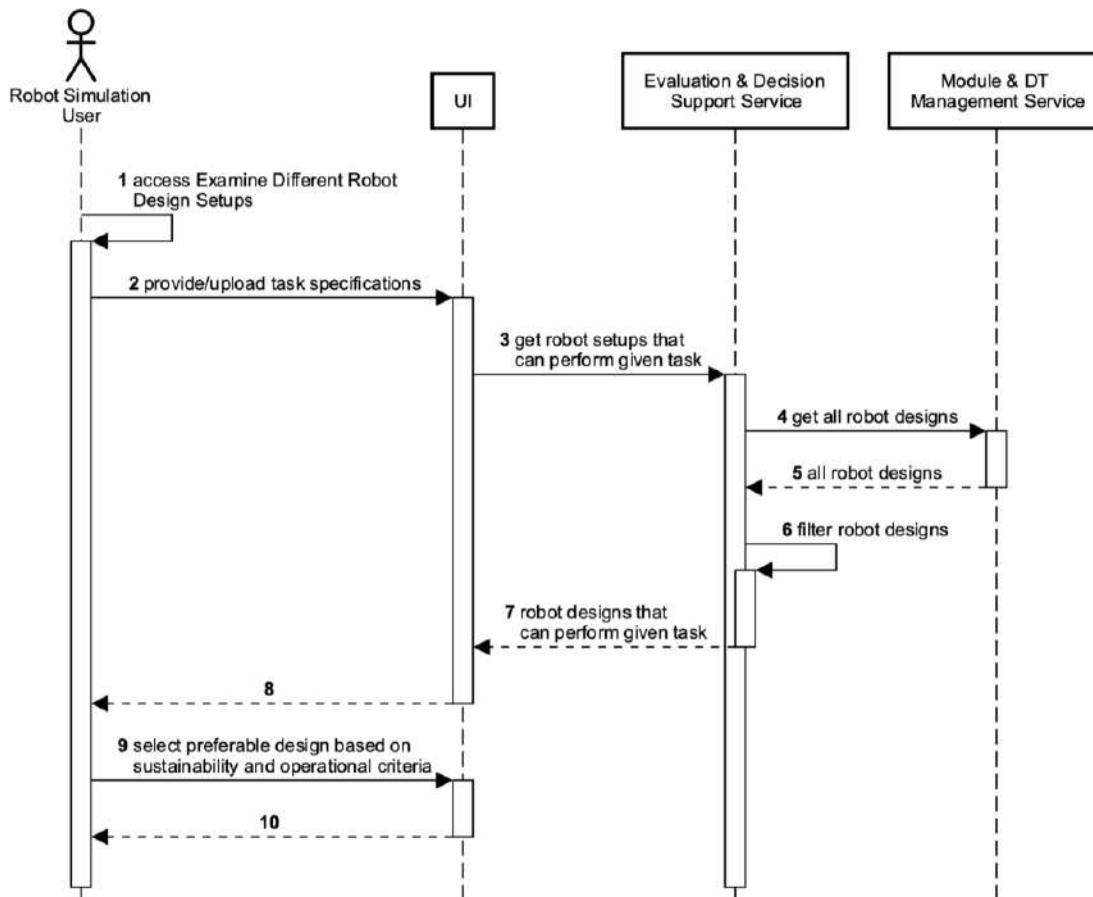


Figure 19. Examine Different Robot Design Setups



### UC1.4: Optimize Robot Movements

In this use case, the MODAPTO System supports the FFT Robot Programmer User in the optimization of a robot (defined as a MODAPTO Module) movement in accordance with provided parameters like carbon emissions and energy consumption.

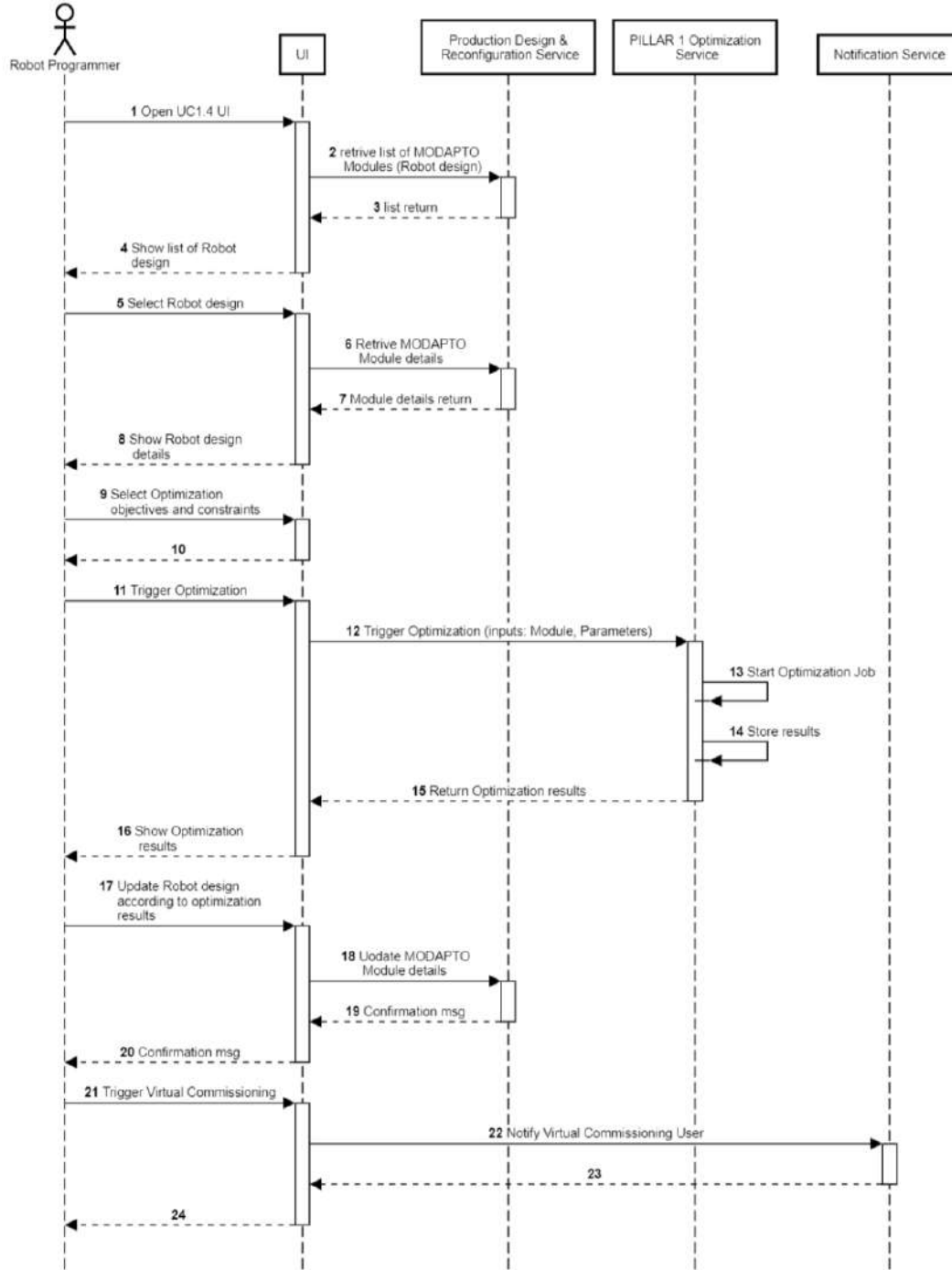


Figure 20. Optimize Robot Movements

The User Interface here interacts first with the Production Design & Reconfiguration Service in order to retrieve all the available robot designs in the form of MODAPTO Modules and show the list to the user. Then, the robot programmer user selects the desired MODAPTO Module to optimize and, after an overview of the Module details, will provide the optimization

parameters (like sustainability and operational criteria) and constraints (like time/energy limitation).

When all the parameters are provided the Optimization Service can be triggered. The optimization job will start in the background and as soon as it is completed the resulting report will be stored and returned to the user that can review it and decide to update the MODAPTO Module accordingly.

At the end, the user will trigger the execution of the next FFT UC, relative to the Virtual Commissioning, by notifying the Virtual Commissioning User through the Notification Service.

## SEW Workflows

### US2.4: Manage Predictive Maintenance Notifications and Prioritize Maintenance Actions

In this user story, the user seeks predictive maintenance advice for production modules. The sequence diagram focuses on generating and communicating these suggestions. Assuming the user is logged in, the Predictive Maintenance Service checks for upcoming maintenance needs and generates suggestions, including priority and potential failure indicators. The Notification Service then delivers these suggestions to the UI, where they are presented to the user, possibly through toast notifications and icon updates. If the user clicks on a notification, the User Interface displays detailed maintenance information. The user can then accept or reject the suggested priority, with their decision relayed to the Notification Service for system update.

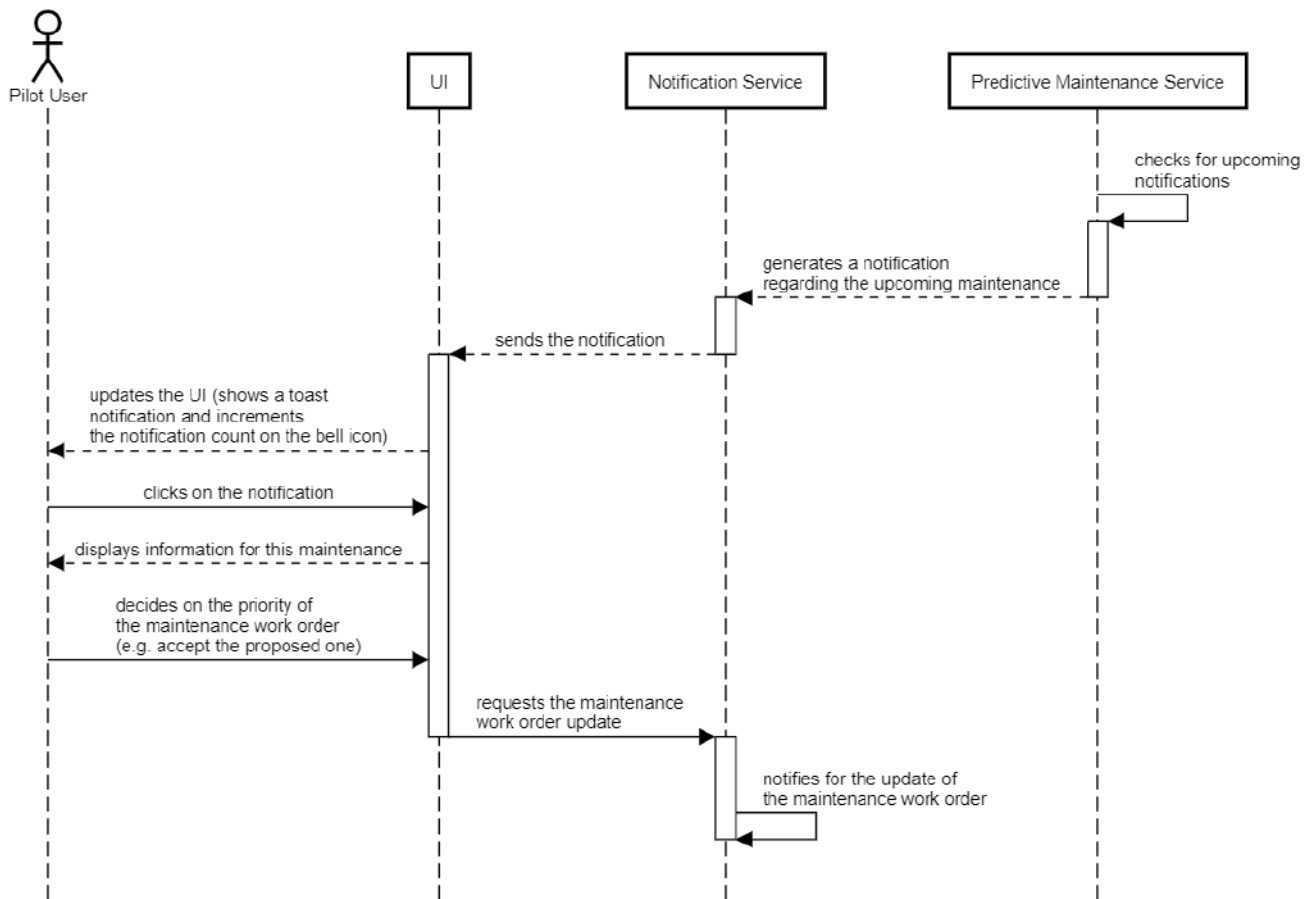


Figure 21. Manage Predictive Maintenance Notifications and Prioritize Maintenance Actions

### US2.6: Re-Optimize Production Schedule

In this user story, the user, logged into the system, begins the process to re-optimize the production schedule or re-route production based on specific criteria. They navigate the User Interface to access production-related information, which is retrieved from the Production Design & Reconfiguration Service. The user then configures production constraints and objectives via the UI. Assuming they opt to re-optimize the schedule, a message is sent to the Notification Service via the Production Design & Reconfiguration Service, signaling the start of the re-optimization process. The user is notified of this initiation. The Notification Service prompts the Service Composition Service to trigger optimization calculations via the Optimization Service. Once completed, the Optimization Service notifies the user of the results through the Notification Service. Finally, the User Interface presents these results as a Gantt chart to the user.

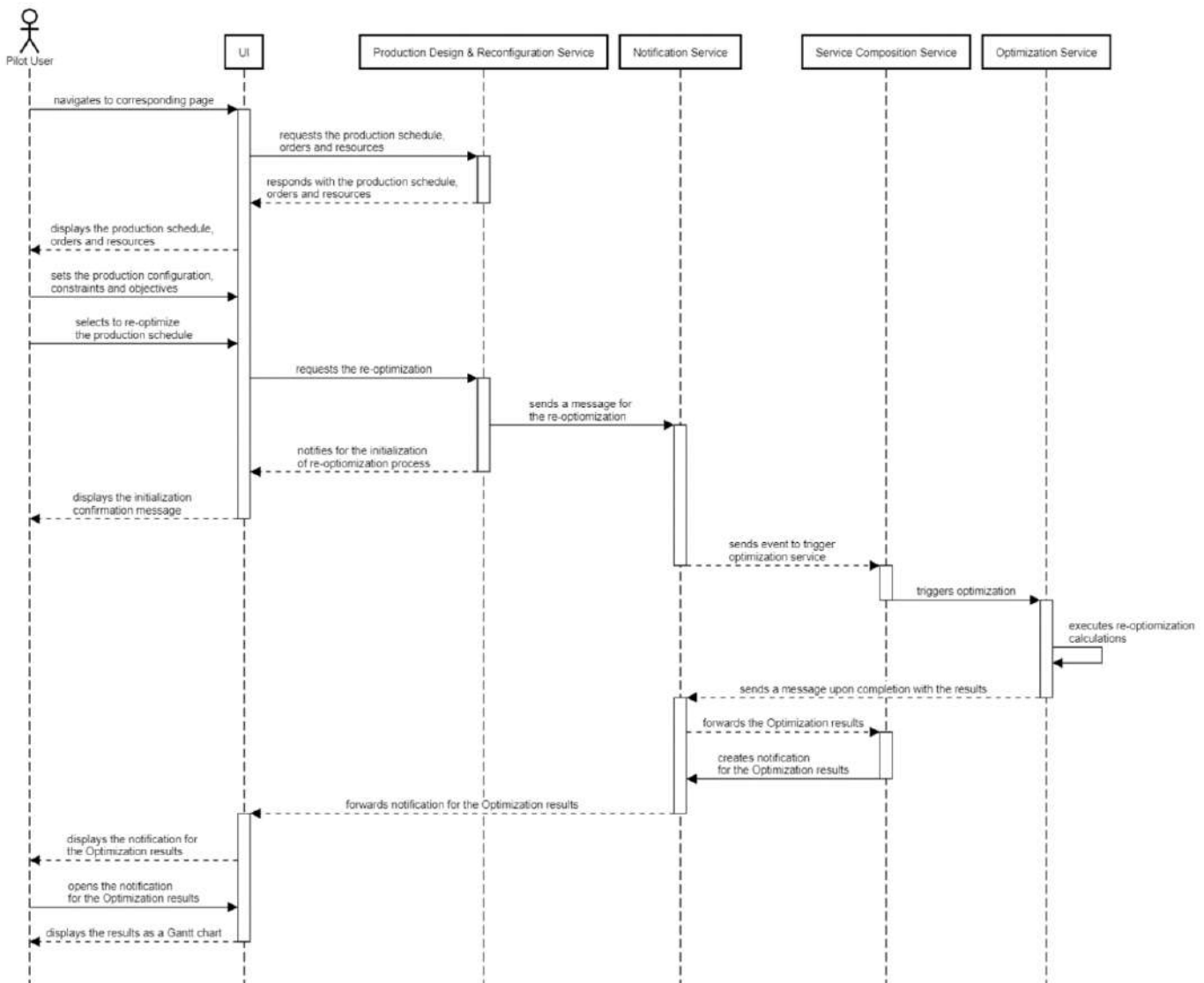


Figure 22. Re-Optimize Production Schedule

## CRF & ILTAR Workflows

### UC3.2: Receive KH Self-Awareness Notifications

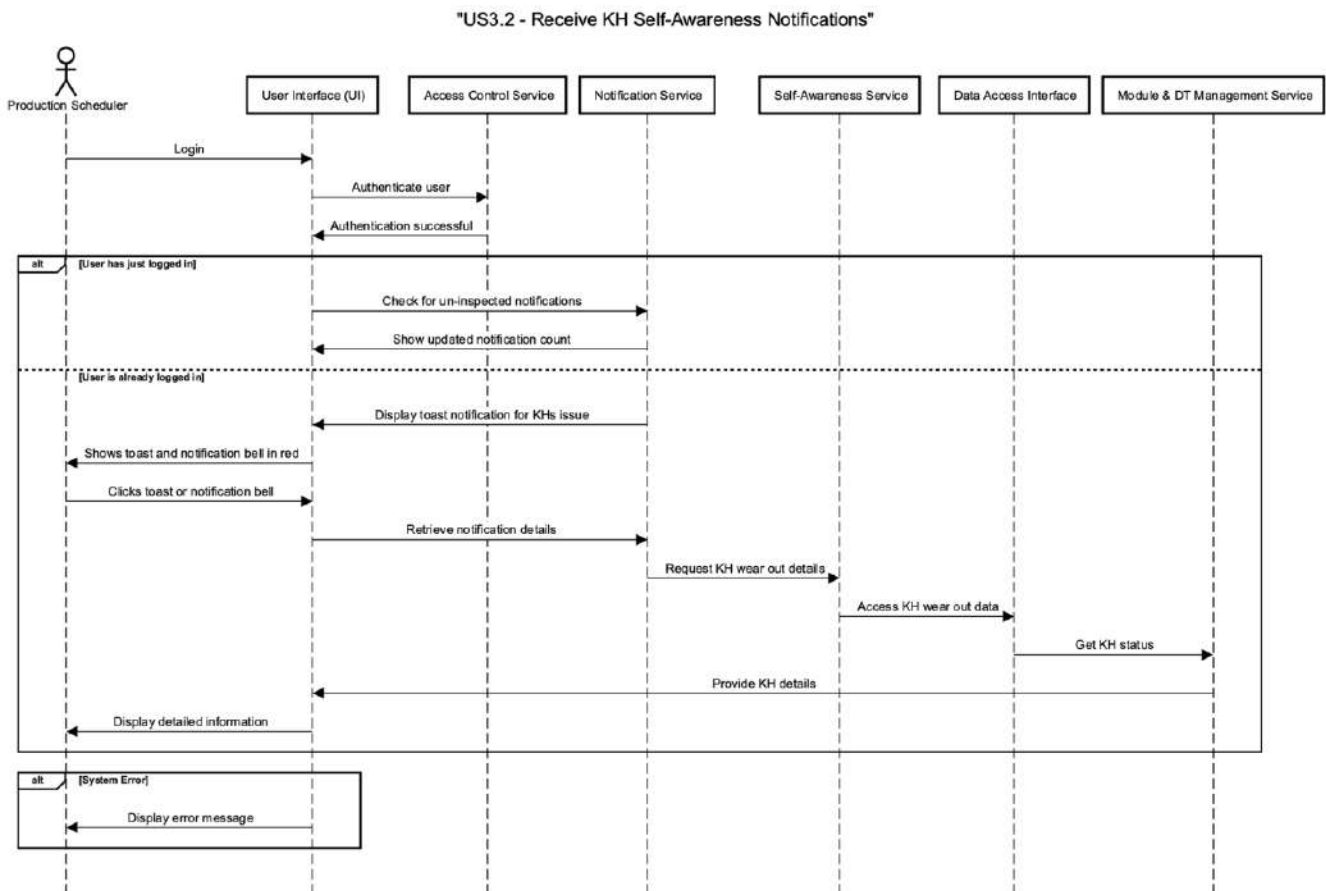


Figure 23: Receive KH Self-Awareness Notifications

The sequence diagram for UC3.2 illustrates the flow of information from the detection of an issue with a KH to the Production Scheduler being notified. In this diagram, the sequence begins with the Production Scheduler logging into the MODAPTO System via the User Interface (UI). The User Interface requests the Access Control Service to authenticate the user, which upon success allows the user to proceed. The Notification Service checks for any un-inspected notifications if the user has just logged in and updates the notification bell count. If the user is already logged in, a toast notification about KHs not meeting requirements is displayed by

the Notification Service through the UI. When the Production Scheduler clicks on the toast or the notification bell, the User Interface requests the details of the notification from the Notification Service, which in turn interacts with the Self-Awareness Service to retrieve detailed information about the wear out of the KHs. The Self-Awareness Service accesses the necessary data through the Data Access Interface, which may involve getting the status of KHs from the Module & DT Management Service. The details are then presented to the Production Scheduler through the UI. In case of a system error at any point in the process, an error message is displayed to the Production Scheduler. This sequence diagram takes into account the preconditions such as user login and configuration of production modules, as well as the post-condition of notifying the user about the status of the KHs. It also reflects the alternative flows for system errors and the scenario where the user has just logged in.

### UC3.9: Receive KH Quality Notifications

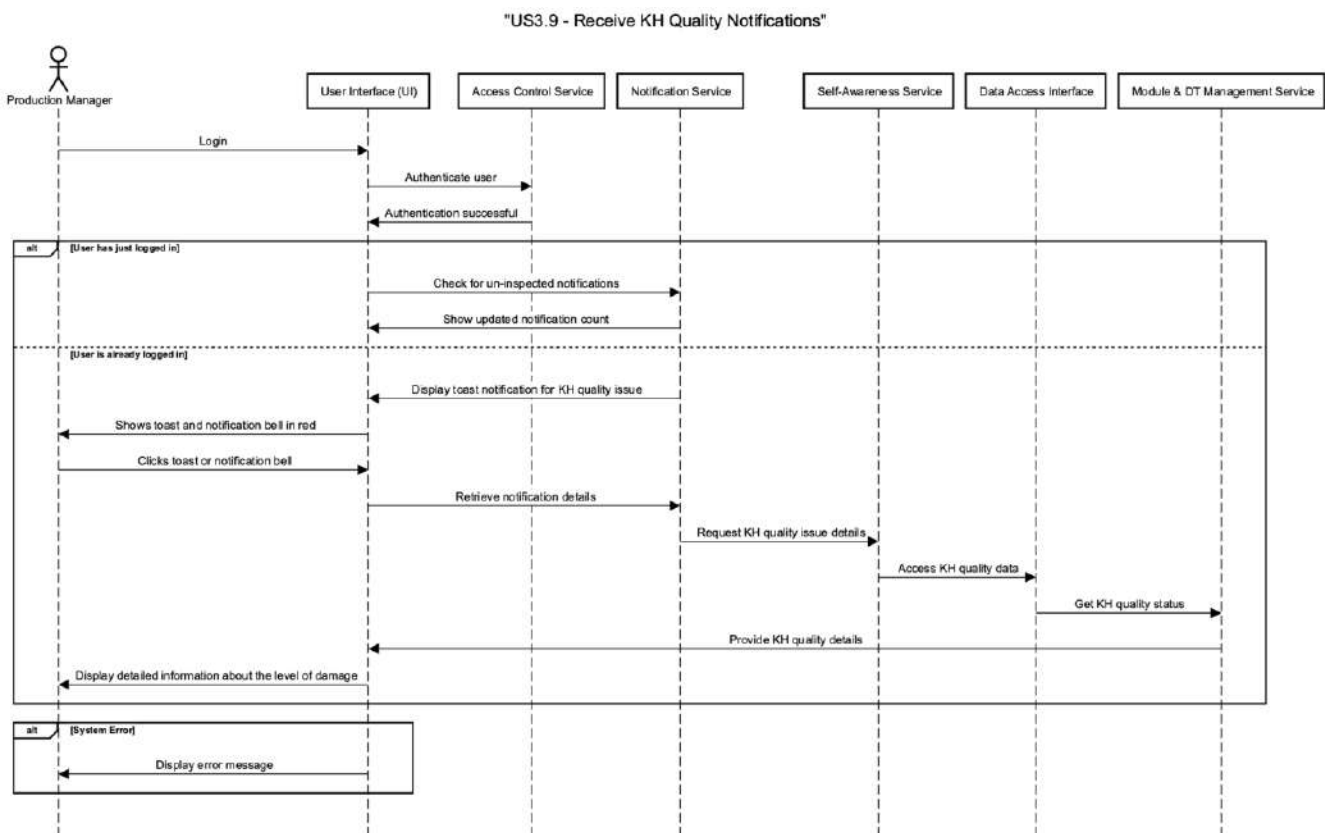


Figure 24: Receive KH Quality Notifications

The sequence diagram for UC3.9 outlines the following interactions: In this sequence, the Production Manager logs into the MODAPTO System and is authenticated by the Access Control Service through the UI. Depending on whether the manager has just logged in or is already logged in, the User Interface interacts with the Notification Service to present the number of un-inspected notifications or to show a toast notification about KHs not meeting quality requirements. Upon clicking the toast or notification bell, the User Interface requests the Notification Service to retrieve the details of the notification. The Notification Service then communicates with the Self-Awareness Service to request detailed information about the quality issue. The Self-Awareness Service utilizes the Data Access Interface to access the relevant quality data, which may involve the Module & DT Management Service to get the current quality status of the KHs. The details are then sent back to the UI, which presents the Production Manager with detailed information regarding the level of damage of the KH

modules. Alternative flows are included to account for system errors and the scenario where the user has just logged in.

## Annex 2 Mapping of Technical Requirements and User Stories

The following subsections include the Technical Requirements (TRs) for the MODAPTO architecture alongside the relevant User Stories from D3.1. This annex aims to strengthen the selection of each Technical Requirement by providing an indicative user story from D3.1, which justifies why the TR was chosen. Each table below provides a mapping between the Technical Requirements of each group and the User Stories that are analyzed in detail in D3.1.

### Group 1: Authentication and Authorization

The TRs of this group focus on authentication, access control, and admin management, which are foundational elements in any system that involves multiple user roles and permissions. These elements are crucial for ensuring that the right users have proper access to the system's functionalities, which aligns with several User Stories across different pilots.

TR ID	Title	Relevant User Story from D3.1
TR1.1	Integrate an authentication service supporting authentication standards	<b>US0.4 (MODAPTO System Configuration):</b> As this user story involves user account access and management, users must be authenticated according to standard security and data integrity protocols.
TR1.2	Develop a role-based access control (RBAC) system with predefined roles and permissions.	<b>US0.1 (MODAPTO System Configuration):</b> This user story directly involves managing different user roles and access rights, which is the essence of an RBAC system. Implementing RBAC ensures that users have access rights appropriate to their roles.
TR1.3	Implement an admin dashboard for super-users to manage user roles and permissions.	<b>US0.1 (MODAPTO System Configuration):</b> The requirement for an admin dashboard to manage roles and permissions aligns with the need expressed in this user story for a super-user to manage user roles and access rights, indicating a direct correlation.

### Group 2: User Interface

Mapping this group of TRs to relevant User Stories from D3.1 requires a focus on visualization, customization, accessibility, and compatibility aspects of the software system. These TRs are centered around enhancing the user interface (UI) experience, making it more adaptable, interactive, and user-friendly.

TR ID	Title	Relevant User Story from D3.1
TR2.1	Develop a visualization system allowing users to interact with MODAPTO and incorporate system flows.	<b>US2.8 (Pilot 2):</b> This user story involves designing and reconfiguring production processes, where a robust visualization system would be crucial for understanding and interacting with complex system flows.
TR2.2	Develop a dynamic dashboard creation module with customization options.	<b>US1.2 (Pilot 1):</b> The need for examining different robot design setups would benefit from a dynamic dashboard that can be customized to display relevant operational and sustainability criteria.



TR2.3	Enable localization and multi-language support for User Interface.	<b>US3.7 (Pilot 3):</b> As a Design Expert reviewing usage data, having a User Interface that supports multiple languages can enhance accessibility and user experience, especially in diverse and international teams.
TR2.4	Design responsive user interfaces for mobile device compatibility.	<b>US3.6 (Pilot 3):</b> A Plant/Automation Manager simulating new systems would benefit from mobile-compatible UIs, allowing for flexibility and accessibility in different work environments
TR2.5	Enable dashboards to visualize multiple production cells in the same dashboard (comparison mode)	<b>US2.2 (Pilot 2):</b> Comparing the performance of production cells directly aligns with the capability to visualize multiple cells in a single dashboard, enhancing analysis and decision-making processes.
TR2.6	Use a micro-fronted approach to design modular and reusable User Interface components.	<b>General Applicability:</b> This requirement, while not directly mapped to a specific user story, underpins the development philosophy for the User Interface components, ensuring modularity and reusability across various parts of the system, aligning with the overall need for a flexible and scalable User Interface as seen in multiple user stories as in <b>US1.3 (Pilot 1)</b> and <b>US2.8 (Pilot 2)</b>

### Group 3: Production and Digital Twin Management

Mapping Group 3 of TRs to relevant User Stories from D3.1 involves focusing on the digital twins of production modules, configuration environments, user interface design for configuration, data streaming for monitoring, and intelligent decision support. These TRs are crucial for managing and optimizing production processes and decision-making.

TR ID	Title	Relevant User Story from D3.1
TR3.1	Develop a tool for managing digital twins of production modules (MODAPTO Modules).	<b>US2.8 (Pilot 2):</b> Designing and reconfiguring production processes using different setups requires managing digital twins to simulate and assess performance, aligning with the need for a tool to handle these digital representations.
TR3.2	Provide an environment to configure a MODAPTO Module easily	<b>US2.8 (Pilot 2):</b> This user story, involving the design and reconfiguration of production steps, directly benefits from an environment that allows easy configuration of MODAPTO Modules.
TR3.3	Provide an environment to easily configure a production schema composing MODAPTO Modules	<b>US2.5 (Pilot 2):</b> Optimizing production schedules and incorporating maintenance suggestions necessitates an environment where production schemas can be easily configured and manipulated.
TR3.4	Develop a User Interface for configuration time.	<b>US0.2 (MODAPTO System Configuration):</b> Managing and configuring production schemas requires a user-friendly User Interface for configuration tasks.
TR3.5	Integrate near-real-time data streaming for production monitoring.	<b>US2.1 (Pilot 2):</b> Monitoring the current capacity of production cells in near-real-time is essential for identifying bottlenecks, making real-time data streaming a vital requirement.
TR3.6	Incorporate Intelligent services for decision support.	<b>US2.4 (Pilot 2) &amp; US3.3 (Pilot 3 - ILTAR):</b> Both these user stories involve making informed decisions based on maintenance suggestions and production schedule optimization, which would benefit from intelligent decision support services.



## Group 4: MODAPTO System Database and Digital Twins

Mapping Group 4 TRs to relevant User Stories from D3.1 involves focusing on data storage, Digital Twins' management, real-time monitoring, intelligent services, and data retrieval mechanisms. These requirements are geared towards enhancing the functionality and interoperability of DTs in the MODAPTO system.

TR ID	Title	Relevant User Story from D3.1
TR4.1	Provide a storage mechanism for MODAPTO services data (i.e., configuration & results).	<b>US0.2 (MODAPTO System Configuration):</b> This user story, involving configuring/updating production schemas, requires robust data storage for configurations and results, aligning with the need for a dedicated storage mechanism
TR4.2	Provide CREUD (Create, Read, Execute, Update, Delete) operations for digital twins.	<b>US2.8 (Pilot 2):</b> The ability to design and reconfigure production processes using digital twins necessitates comprehensive CREUD operations to manage these digital representations effectively.
TR4.3	Enable monitoring service to collect real-time data from a production module.	<b>US2.1 (Pilot 2):</b> Monitoring the current capacity of production cells in real-time for identifying bottlenecks requires a monitoring service that collects real-time data.
TR4.4	Extend a DT service to be able to run intelligent services.	<b>US2.4 (Pilot 2):</b> Receiving predictive maintenance suggestions necessitates digital twins capable of running intelligent services to provide insightful analysis and predictions.
TR4.5	Extend a DT service to enable on-demand interaction with interconnected DTs.	<b>US3.6 (Pilot 3 - ILTAR):</b> Simulating new material handling systems and assessing their impact requires interactive digital twins that can communicate and interact with each other.
TR4.6	Develop a service for onboarding non-AAS standard compliant DTs.	<b>General Applicability:</b> While not directly mapped to a specific user story, this requirement supports including and integrating diverse digital twins, ensuring system flexibility and compatibility with various DT formats. For example, this TR find applicability in <b>US2.2 (Pilot2)</b> and <b>US3.6 (Pilot 3 - ILTAR)</b>
TR4.7	Provide a data retrieval mechanism for efficient data retrieval from the knowledge base.	<b>US3.7 (Pilot 3 - ILTAR):</b> Reviewing KH usage data for improvement purposes requires an efficient data retrieval mechanism to access relevant information from the knowledge base.
TR4.8	Enable the definition of DTs following AAS standard	<b>General Applicability:</b> Adhering to the AAS standard in defining digital twins ensures consistency and interoperability across the system, relevant to various user stories where digital twins are utilized. Indicative applicability of this TR is found in <b>US1.1 (Pilot 1)</b> and <b>US3.4 (Pilot3)</b> .
TR4.9	Enable the definition of DTs as composition of other sub-DTs	<b>US2.8 (Pilot 2):</b> Designing different setups of the production process might involve composing digital twins from sub-DTs, allowing for more granular and detailed simulation and analysis.



## Group 5: Development (and implementation) of Smart Services

Group 5 primarily focuses on analytics, optimization, simulation, and the implementation of smart services within the MODAPTO framework. These requirements are key to enhancing the system's capability in handling data, making informed decisions, and improving overall operational efficiency.

TR ID	Title	Relevant User Story from D3.1
TR5.1	Creation of analytics tools for batch data. Provide Support for batch Data and real-time data Analytics	<b>US1.2 (Pilot 1):</b> Analyzing different robot design setups requires analytics tools capable of processing batch and real-time data to make informed decisions.
TR5.2	The Optimization functionality supports multiple input parameters (e.g., orders and production schedules, quality specifications, shifts, planning horizon, materials, carbon footprints, energy consumption, robot movements)	<b>US2.5 (Pilot 2):</b> Optimizing the production schedule involves considering various parameters like orders, quality specifications, and energy consumption, necessitating comprehensive optimization functionality.
TR5.3	The Optimization functionality produces suggestions for optimal courses of action (e.g., production schedules, maintenance schedules, optimal robot movement configurations and actions).	<b>US2.6 (Pilot 2):</b> Re-optimizing the production schedule based on evolving events requires suggestions for the optimal course of action, aligning with this requirement.
TR5.4	The Simulation functionality supports multiple input parameters (e.g., sustainability parameters, materials, manufacturing methods, robot movements, orders and production schedules, plant setups).	<b>US1.4 (Pilot 1):</b> Optimizing robot movements for a given design setup requires a simulation functionality that can handle diverse input parameters.
TR5.5	The Simulation functionality produces sustainability and operational analytics to assess the performance of a MODAPTO module (e.g., specific robot movements, production schedules, plant setups).	<b>US1.5 (Pilot 1):</b> Assessing the sustainability and operational performance of a production process using different robot designs requires detailed sustainability and operational analytics
TR5.6	Creation of self-awareness service.	<b>General Applicability:</b> This requirement, while not directly mapped to a specific user story, supports the development of a more intuitive and responsive system, beneficial across various user stories such as US2.4 (Pilot2) and US3.6 (Pilot 3)
TR5.7	Creation of predictive maintenance service.	<b>US2.4 (Pilot 2):</b> Receiving predictive maintenance suggestions requires a service designed to align with the user story's need.
TR5.8	Implementation of Smart Services.	<b>US3.3 (Pilot 3 - ILTAR):</b> Re-optimizing the production schedule based on resource configuration necessitates smart services for intelligent and efficient decision-making.
TR5.9	Provide an Orchestrator to combine and reuse smart services for the specific pilots	<b>General Applicability:</b> This requirement ensures efficient management and utilization of smart services, which is beneficial across various pilots and their



		respective user stories, like <b>US2.6 (Pilot 2)</b> and <b>US3.3 (Pilot 3 - ILTAR)</b> .
--	--	---

## Group 6: Security and Infrastructure

Group 6 of TRs focuses on security, data protection, and compatibility with standard IT processes and infrastructures. These are vital for the reliability and safety of the MODAPTO system.

TR ID	Title	Relevant User Story from D3.1
<b>TR6.1A</b>	Enable End-to-End Encryption for Data Transmission (examples: HTTPS, MQTTS, OPCUA).	<b>General Applicability:</b> While not directly tied to a specific user story, this requirement is fundamental to ensuring the security of data transmission across all user interactions within the system. Indicatively, applicability for this TR is met in US2.3 (Pilot 2 and in US3.10 (Pilot 3 -CRF)
<b>TR6.1B</b>	Implement Encryption for Sensitive User Data.	<b>US0.4 (MODAPTO System Configuration):</b> As this user story involves managing user account details, encrypting sensitive user data is essential for privacy and security
<b>TR6.2</b>	Ensure MODAPTO Compatibility with Standard Backup and Recovery Processes.	<b>General Applicability:</b> This requirement, though not specific to a particular user story, is crucial across the system to ensure data integrity and availability, especially in scenarios involving critical production data such as in US1.4 (Pilot 1) and US3.5 (Pilot 3 - ILTAR)
<b>TR6.3</b>	Ensure MODAPTO Compatibility with Scalable Cloud-Based Infrastructures.	<b>US2.1 (Pilot 2):</b> Monitoring the current capacity of production cells in (near) real-time implies a need for scalable and reliable cloud infrastructure to handle fluctuating data loads and ensure system availability.

## Group 7: System Design, Integration, Monitoring & Compatibility

TRs of Group 7 focus on integration with existing systems, testing strategies, continuous deployment, and architectural design for modularity and scalability. These requirements are essential for ensuring that the MODAPTO system is robust, flexible, and integrates seamlessly with existing infrastructures.

TR ID	Title	Relevant User Story from D3.1
<b>TR7.1</b>	Connect existing Data Sources to MODAPTO.	<b>General Applicability:</b> This requirement is fundamental for all user stories that rely on existing data sources for informed decision-making and operational efficiency within the MODAPTO system. Indicatively, this TR finds applicability in <b>US1.2 (Pilot 1)</b> and <b>US3.1 (Pilot 3 - ILTAR)</b>
<b>TR7.2</b>	Provide means to integrate with existing standards/tools for monitoring.	<b>US2.1 (Pilot 2):</b> This user story, focusing on monitoring the capacity of production cells, requires integration with existing monitoring tools and standards for effective real-time analysis.



TR7.3	Implement a comprehensive integration testing strategy that includes periodic testing of components' functionalities.	<b>General Applicability:</b> While not specific to a particular user story, this requirement is crucial for ensuring the reliability and proper functioning of the MODAPTO system across various user scenarios. For example, the importance of a robust integration testing strategy is evident in both <b>US1.3 (Pilot 1)</b> and <b>US2.7 (Pilot 2)</b> .
TR7.4	Implement a continuous integration/deployment pipeline with version control. (CI/CD)	<b>General Applicability:</b> This technical requirement supports all user stories by ensuring the system is always up-to-date, stable, secure, critical for operational continuity and efficiency. For instance, in both <b>US0.2 (MODAPTO System Configuration)</b> and <b>US3.2 (Pilot 3 -ILTAR)</b> , implementing a CI/CD pipeline with version control is crucial. It ensures the system is consistently updated with the latest features and improvements, enhancing the overall user experience and system functionality.
TR7.5	Adopt a service-oriented / microservice based architecture for modularity and scalability.	<b>General Applicability:</b> A microservice-based architecture underpins the system's flexibility and scalability, which benefits various pilots and user stories, especially those requiring system modularity and adaptability. For example, in <b>US1.5 (Pilot 1)</b> and <b>US2.9 (Pilot 2)</b> , adopting a microservice-based architecture enhances the system's ability to handle diverse and dynamic requirements. It enables the development of specific, targeted services that can be independently updated, scaled, and maintained, leading to a more robust, flexible, and efficient system.

## Group 8: Reporting, Alerts & User Support

TRs of Group 8 are focused on user interface enhancements, including dashboards, notification systems, user feedback capabilities, and documentation. These requirements are crucial for enhancing user interaction, providing relevant information, and ensuring ease of use.

TR ID	Title	Relevant User Story from D3.1
TR8.1	Provide a dashboard that presents real-time and historical aggregated data.	<b>US2.1 (Pilot 2):</b> This user story requires a view of the current capacity of production cells, which would benefit from a dashboard displaying both real-time and historical data.
TR8.2	Provide a notification system for notifications like alerting etc., displayed via the UI.	<b>US2.3 (Pilot 2):</b> This user story involves receiving notifications about production modules about to fail, necessitating a robust notification system within the UI.
TR8.3	Provide User Feedback capabilities.	<b>General Applicability:</b> User feedback capabilities are essential across all user stories to allow users to report issues, suggest improvements, and interact with the system effectively. Indicatively, TR08.3 finds applicability in <b>US1.4 (Pilot 1)</b> and <b>US3.7 (Pilot 3 - ILTAR)</b>
TR8.4	Create user manuals, online documentation, and video tutorials.	<b>General Applicability:</b> Providing comprehensive documentation and tutorials supports all user stories by enabling users to understand and use the MODAPTO system effectively. For example, in user stories such as <b>US0.1 (MODAPTO System Configuration - Super-user)</b>



		and <b>US3.12 (Pilot 3 – CRF)</b> , the availability of well-structured and comprehensive documentation and tutorials is key to enabling users to interact with and utilize the MODAPTO system effectively. For super-users and production managers, these resources provide the necessary guidance to navigate complex system functionalities, leading to a smoother, more efficient operational process.
<b>TR8.5</b>	Support creation/saving/exchange of process drifts among relevant user roles	<b>US2.10 (Pilot 2):</b> As operators need to report process drifts in their production cell, the system must support the creation, saving, and exchange of such information among relevant user roles.
<b>TR8.6</b>	The notification system receives notifications from any component and pushes them to the GUI in real-time	<b>US2.3 (Pilot 2) &amp; US3.9 (Pilot 3 - CRF):</b> These user stories involve receiving timely notifications about production issues and quality requirements, necessitating a system that can push notifications from various components to the User Interface in real time.

## Group 9: APIs and Data Exchange

TRs of Group 9 focus on system integration, data interchange, API documentation, data management, and data validation. These requirements are critical for ensuring that the MODAPTO system can effectively communicate with external systems, handle data securely and efficiently, and maintain data integrity.

TR ID	Title	Relevant User Story from D3.1
<b>TR9.1</b>	Design (RESTful) APIs with endpoints for external integrations.	<b>General Applicability:</b> This requirement is crucial for all user stories that involve integrating the MODAPTO system with external systems or tools, facilitating seamless data exchange and functionality expansion. For example, in <b>US2.5 (Pilot 2)</b> and <b>US3.11 (Pilot 3 – CRF)</b> , RESTful APIs for external integrations play a pivotal role. For the Production Scheduling Team Member, it allows for enhanced and more efficient scheduling capabilities by integrating diverse data sources and tools. For the Production Manager, it enables seamless communication between the MODAPTO system and external systems, ensuring efficient and accurate automation of the picking process.
<b>TR9.2</b>	Adopt Standard Data Interchange Formats like JSON, XML, etc.	<b>General Applicability:</b> Standard data interchange formats ensure the system can interact efficiently with various other systems and tools relevant to multiple user stories that depend on external data sources or need to share data. For instance, in <b>US1.1 (Pilot 1)</b> and <b>US3.4 (Pilot 3)</b> , standard data interchange formats are essential. For the Robot Simulation User / Process Engineer, it enables effective collaboration and data sharing with other stakeholders and systems, enhancing the sustainability assessment process. The Production Scheduler ensures that data from different scheduling systems can be easily aggregated and compared, leading to more accurate and efficient scheduling decisions.
<b>TR9.3</b>	Provide comprehensive API documentation with sample requests and responses.	<b>General Applicability:</b> Comprehensive API documentation is essential for developers and users who interact with the MODAPTO system's API, enhancing the system's usability and integration capabilities across various scenarios. For example, in both <b>US0.3 (MODAPTO System Configuration -</b>



		<b>Super-user</b> ) and <b>US3.12 (Pilot 3)</b> , having accessible, detailed API documentation with practical examples is critical to empowering users and developers to leverage the capabilities of the MODAPTO system's APIs fully. For the Super-user, it simplifies the process of custom event configuration, while for the Production Manager, it aids in the practical analysis and comparison of complex data sets.
<b>TR9.4</b>	Provide Data Management Mechanisms that comply with defined data retention policies	<b>General Applicability:</b> Data management mechanisms adhering to retention policies are essential across all user stories to ensure data security, privacy, and compliance, especially where sensitive or critical data is involved, for example, in both <b>US0.4 (MODAPTO System Configuration)</b> and <b>US2.2 (Pilot 2)</b> , having robust data management mechanisms in place that adhere to data retention policies is vital. For individual users managing their accounts, it ensures their data is handled responsibly. For professionals analyzing production cell data, it guarantees that sensitive information is managed securely and complies with organizational and legal standards.
<b>TR9.5</b>	Implement (User) data validation checks at input points. (Need to further define the system data. Split this TR into two based on the origin and nature of data)	<b>General Applicability:</b> Data validation checks at input points are crucial to maintain data integrity and prevent errors or security issues across all user scenarios where data is entered or uploaded into the system. For example, in <b>US1.1 (Pilot 1)</b> and <b>US2.9 (Pilot 2)</b> , data validation at input points is critical in maintaining data accuracy and preventing errors. For the Robot Simulation User / Process Engineer, it ensures the input data for simulations and calculations is correct, enhancing the reliability of the design process. For the Operator, it guarantees that the operational data they input is accurate and helpful for production tracking and analysis. Implementing these validation checks is key to ensuring the MODAPTO system's overall data integrity and operational efficiency.

### Annex 3 Mapping Prioritization of Technical Requirements in relevance to the User Requirements.

Based on the information provided regarding the Technical Requirements (TRs) and the User Requirements (User Stories (US) and Use Cases) (UCs) and their priority level, the following table maps the "Must Have" TRs to the relevant "Must Have" USs or UCs.

TR ID		Relevant "Must Have" User Stories/Use Cases
<b>TR1.1</b>	Integrate an authentication service supporting authentication standards	UC0.4.2 - Login
<b>TR1.2</b>	Develop a role-based access control (RBAC) system with predefined roles and permissions.	UC0.1.1 - Create User Role, UC0.1.2 - Update User Role, UC0.1.3 - Delete User Role
<b>TR3.1</b>	Develop tool for managing digital twins of production modules (MODAPTO Modules).	UC1.1.1 - Define Sustainability and Operational Analytics



TR3.2	Provide an environment to easily configure a MODAPTO Module	UC0.2.1 - Configure Production Module
TR3.3	Provide an environment to easily configure a production schema composing MODAPTO Modules	UC0.2.3 - Configure Production Schema
TR3.5	Integrate near-real-time data streaming for production monitoring.	UC3.2 - Receive KH Self-Awareness Notifications
TR3.6	Incorporate Intelligent services for decision support	UC3.6.1 - Re-Design Plant Setup, UC3.6.2 - Simulate Plant Setup
TR4.1	Provide a storage mechanism for MODAPTO services data (i.e., configuration & results).	UC0.2.4 - View Production Schemas
TR4.2	Provide CREUD (Create, Read, Execute, Update, Delete) operations for digital twins	UC0.3 - Configure Custom Event
TR4.3	Enable monitoring service to collect real-time data from a production module	UC3.5 - Track Outbound Logistics Operations
TR4.4	Extend a DT service to be able to run intelligent services	UC3.8.1 - Create Potential Content Information of KH
TR4.6	Develop a service for onboarding non-AAS standard compliant DTs	General Applicability
TR4.8	Enable the definition of DTs following AAS standard	General Applicability
TR4.9	Enable the definition of DTs as composition of other sub-DTs	UC3.8.2 - Create Potential Content Information of a KH block
TR5.1	Creation of analytics tools for batch data. Provide Support for batch Data and real-time data Analytics	UC3.11 - Read KH Information
TR5.2	The Optimization functionality supports multiple input parameters	UC3.12 - (Re-)Optimize Picking Sequence
TR5.3	The Optimization functionality produces suggestions for optimal course of action	UC3.13 - Compare and Confirm Optimization Results
TR5.4	The Simulation functionality supports multiple input parameters	General Applicability
TR5.5	The Simulation functionality produces sustainability and operational analytics to assess the performance of a MODAPTO module	UC1.3 - Assess Performance of a Robot Design Setup
TR5.6	Creation of self-awareness service	UC3.2 - Receive KH Self-Awareness Notifications
TR5.7	Creation of predictive maintenance service	General Applicability



<b>TR5.8</b>	Implementation of Smart Services	UC3.6.2 - Simulate Plant Setup
<b>TR5.9</b>	Provide an Orchestrator to combine and reuse smart services for the specific pilots	General Applicability
<b>TR6.1A</b>	Enable End-to-End Encryption for Data Transmission (examples: HTTPS, MQTTS, OPCUA)	General Applicability
<b>TR6.1B</b>	Implement Encryption for Sensitive User Data.	General Applicability
<b>TR7.1</b>	Connect existing Data Sources to MODAPTO	General Applicability
<b>TR8.2</b>	Provide a notification system for notifications like alerting etc., displayed via the UI.	UC3.9 - Receive KH Quality Notifications

Mapping these relationships is essential for ensuring that high-priority requirements are aligned with the critical functionalities necessary for the successful operation of the MODAPTO system. This alignment ensures that the development focus is correctly placed on the core features that will provide the most value and are vital for the development of MODAPTO.