



*AI-augmented automation supporting modelling, coding,
testing, monitoring and continuous development in
Cyber-Physical Systems*

D 4.3 - AIDOaRt AI-Augmented tool set - Final Version

This project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 101007350. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Sweden, Austria, Czech Republic, Finland, France, Italy, Spain. This document reflects only the author's view and that the Commission is not responsible for any use that may be made of the information it contains.



Contract number:	101007350
Project acronym:	AIDOaRt
Project title:	AI-augmented automation supporting modelling, coding, testing, monitoring and continuous development in Cyber-Physical Systems
Delivery Date:	April 30, 2024
Authors:	Overall document structure and content: Dario Guidotti (UNISS), Laura Pandolfo (UNISS), Romina Eramo (UNITE), Bilal Said (SOFT)
Contributing Partners:	ABI, ABO, ACO, AIT, AND, AVL, BT, CAMEA, CSY, DT, HIB, IMTA, INT, ITI, JKU, MDU, PRO, QEN, RISE, ROTECH, SOFT, TEK, TUG, UCAN, UNISS, UNIVAQ, UNITE, UOC, VCE, WMO
Date:	April 30, 2024
Version	V1
Revision:	08
Abstract:	This deliverable provides the final version of the AI-augmented tool set as developed in the context of Work Package 4. It builds upon the foundational work established in previous deliverables, notably D4.1 and D4.2, and aligns with analogue outputs from other work packages, such as D2.3 and D3.4, delivered by M28. The primary goal of D4.3 is to enhance the capabilities and interconnections between use cases and solutions, offering a current perspective and pragmatic vision of the AIDOaRt AI-augmented tool set.
Status:	Type: Other, Dissemination Level: PU

DOCUMENT REVISION LOG

VERSION	REVISION	DATE	DESCRIPTION	AUTHOR
V1	01	21/11/2023	First Doc Generation	Dario Guidotti (UNISS), Laura Pandolfo (UNISS)
	02	12/12/2023	Structure Update	Dario Guidotti (UNISS), Laura Pandolfo (UNISS), Romina Eramo (UNITE)
	03	09/01/2024	Initial Modelio Doc Generation	Bilal Said (SOFT)
	04	09/01/2024	Content Update / Micro Tasks	AIDoArt Partners
	05	27/02/2024	Content Update / Micro Tasks	AIDoArt Partners
	06	21/03/2024	Hackathon Report integration	Romina Eramo (UNITE), Vittoriano Mutillo (UNIVAQ), Claudio Di Sipio (UNIVAQ), Riccardo Rubei (UNIVAQ)
	07	21/03/2024	Integrate feedback from internal reviewers	AIDoArt Partners
	08	18/04/2024	Final version for EU submission	Dario Guidotti (UNISS), Laura Pandolfo (UNISS)

Executive Summary

The AIDOaRt approach and its global solution heavily rely on the use of Model Driven Engineering (MDE), combined with Artificial Intelligence (AI) techniques, at different levels of the continuous development (i.e., DevOps) process of large and complex Cyber-Physical Systems (CPSs). The main goal of Work Package (WP) 4 is to enhance the DevOps tool chain by employing AI and Machine Learning (ML) techniques in multiple aspects of the system development process (such as requirements, monitoring, modelling, coding, and testing). According to the AIOps methodology, the AI-augmented tool set supports the monitoring of runtime data (such as logs, events and metrics), software data and traceability (i.e., observation phase), the analysis of both: historical and real time data (i.e., analysis phase) and the automation of development and operation activities (i.e., automation phase).

To realise this in practice, the AIDOaRt AI-augmented tool set has been designed in the first part of the project, cf. the corresponding deliverable [AIDOART-D4.1]. During the second phase of the project, this tool set has been developed, deployed and experimented within the context of the project use cases, cf. the corresponding deliverable [AIDOART-4.2]. This tool set comprises several solutions/tools, mostly coming from our project partners (and possibly supplemented by a few others according to further needs), that are being integrated to achieve the above-mentioned WP4 objectives. To make the glue between WP2 and WP3, this AI-augmented tool set is being developed in good synchronisation with its siblings: AIDOaRt Data Engineering tool set (WP2) and AIDOaRt Core tool set (WP3).

The AIDOaRt AI-augmented tool set provides different and complementary capabilities (according to the AIDOaRt general architecture and functional interfaces in D1.4) that are implemented by different tools and properly integrated with each other. This deliverable D4.3 provides an update to D4.2 and presents the final details on the latest development status regarding the different solutions supporting the AI-augmented tool set main components (i.e., achieved capabilities, adopted AI/ML techniques, etc.). It also describes how they have already been implemented in the context of some use case challenges and related CPSs. Furthermore, this deliverable reports the process and the activities carried out during the AIDOaRt Internal Hackathon editions. It provides insights into the Hackathon organisation process, including inputs and outputs, the template for challenge calls, and a comprehensive analysis of key findings from the Hackathons, encompassing both quantitative data and qualitative insights. Finally, in the Appendix, this deliverable provides the latest version of the mapping of the generic components defined in the tool set to: (1) specific solutions and capabilities (provided by Solution Providers), and (2) use case requirements (provided by Use Case Providers) that can be satisfied through those components.

This present deliverable concludes the work conducted in the context of WP4. However, the solution providers will continue working on their respective solutions and collaborating with the use case providers on their various scenarios until the end of the project. The consolidation and practical applications of the different solutions from the AIDOaRt AI-augmented tool set will be continued in the context of WP5 for the integration of these solutions in the overall AIDOaRt framework and in the different use cases until the end of the project, and likely afterwards. The results of these ongoing progresses and developments will be detailed in forthcoming deliverables (D4.4, D5.8 and D5.9).

Table of Contents

DOCUMENT REVISION LOG	3
Executive Summary	4
Partners Names Acronyms	11
1 Introduction	12
2 List Solutions related to the AIDOrt AI-Augmented tool set	13
3 Progress status of the AIDOrt AI-Augmented tool set	18
3.1 Solution - STGEM (ABO)	18
3.1.1 Overview	18
3.1.2 Capabilities Implementation Status	20
3.1.3 Useful Resources	21
3.2 Solution - ESDE (ACO) - AI/DL based traces analysis	22
3.2.1 Overview	22
3.2.2 Capabilities Implementation Status	26
3.2.3 Useful Resources	27
3.3 Solution - Position Monitoring for Industrial Environment (ACO)	27
3.3.1 Overview	27
3.3.2 Capabilities Implementation Status	31
3.3.3 Useful Resources	32
3.4 Solution - DTsynth (AIT)	32
3.4.1 Overview	32
3.4.2 Capabilities Implementation Status	33
3.5 Solution - Mimir(AND)	33
3.5.1 Overview	33
3.5.2 Capabilities Implementation Status	35
3.6 Solution - Active DoE (AVL)	36
3.6.1 Overview	36
3.6.2 Capabilities Implementation Status	39
3.6.3 Useful Resources	39
3.7 Solution - Keptn (DT).....	40
3.7.1 Overview	40
3.7.2 Capabilities Implementation Status	43
3.7.3 Useful Resources	45
3.8 Solution - HIB_logAnalyzer (HIB)	45
3.8.1 Overview	45
3.8.2 Capabilities Implementation Status	48
3.8.3 Useful Resources	48
3.9 Solution - EMF Views (IMTA)	48
3.9.1 Overview	48
3.9.2 Capabilities Implementation Status	51
3.9.3 Useful Resources	52

3.10	Solution - INT-DET (INT)	53
3.10.1	Overview	53
3.10.2	Capabilities Implementation Status	55
3.11	Solution - INT-DEPTH (INT).....	56
3.11.1	Overview	56
3.11.2	Capabilities Implementation Status	57
3.12	Solution - a2k-runman (ITI)	57
3.12.1	Overview	57
3.12.2	Capabilities Implementation Status	61
3.12.3	Useful Resources	61
3.13	Solution - a2k-depman (ITI)	61
3.13.1	Overview	61
3.13.2	Capabilities Implementation Status	64
3.14	Solution - MOMOT (JKU).....	64
3.14.1	Overview	64
3.14.2	Capabilities Implementation Status	67
3.14.3	Useful Resources	68
3.15	Solution - GAN-Based Instance Model Generator (JKU).....	68
3.15.1	Overview	68
3.15.2	Capabilities Implementation Status	70
3.15.3	Useful Resources	70
3.16	Solution - Requirements Ambiguity Checker (MDU)	71
3.16.1	Overview	71
3.16.2	Capabilities Implementation Status	73
3.16.3	Useful Resources	73
3.17	Solution - TATAT (PRO)	73
3.17.1	Overview	73
3.17.2	Capabilities Implementation Status	74
3.18	Solution - CRT (QEN)	75
3.18.1	Overview	75
3.18.2	Capabilities Implementation Status	76
3.18.3	Useful Resources	77
3.19	Solution - CRTQI (QEN).....	77
3.19.1	Overview	77
3.19.2	Capabilities Implementation Status	79
3.19.3	Useful Resources	79
3.20	Solution - QEDITOR (QEN).....	79
3.20.1	Overview	79
3.20.2	Capabilities Implementation Status	81
3.20.3	Useful Resources	82
3.21	Solution - LogGrouper (RISE)	82
3.21.1	Overview	82



3.21.2	Capabilities Implementation Status	84
3.21.3	Useful Resources	84
3.22	Solution - BugIdentifier (RISE).....	84
3.22.1	Overview	84
3.22.2	Capabilities Implementation Status	86
3.23	Solution - DataAggregator (ROTECH).....	86
3.23.1	Overview	86
3.23.2	Capabilities Implementation Status	87
3.24	Solution - Modelio (SOFT).....	88
3.24.1	Overview	88
3.24.2	Capabilities Implementation Status	90
3.24.3	Useful Resources	90
3.25	Solution - AALpy (TUG)	91
3.25.1	Overview	91
3.25.2	Capabilities Implementation Status	93
3.25.3	Useful Resources	93
3.26	Solution - S3D (UCAN).....	94
3.26.1	Overview	94
3.26.2	Capabilities Implementation Status	95
3.26.3	Useful Resources	95
3.27	Solution - SoSIM (UCAN).....	95
3.27.1	Overview	95
3.27.2	Capabilities Implementation Status	96
3.27.3	Useful Resources	96
3.28	Solution - UNISS_SOL_01 (UNISS) - ReqH	97
3.28.1	Overview	97
3.28.2	Capabilities Implementation Status	99
3.28.3	Useful Resources	99
3.29	Solution - UNISS_SOL_02 (UNISS) - NNVer	99
3.29.1	Overview	99
3.29.2	Capabilities Implementation Status	101
3.29.3	Useful Resources	102
3.30	Solution - UNISS_SOL_03 (UNISS) - ReqT.....	102
3.30.1	Overview	102
3.30.2	Capabilities Implementation Status	104
3.30.3	Useful Resources	105
3.31	Solution - UNISS_SOL_04 (UNISS) - OSPCheck.....	105
3.31.1	Overview	105
3.31.2	Capabilities Implementation Status	107
3.31.3	Useful Resources	107
3.32	Solution - HEPYCODE (UNIVAQ).....	108
3.32.1	Overview	108



3.32.2	Capabilities Implementation Status	111
3.32.3	Useful Resources	112
3.33	Solution - MORGAN (UNIVAQ).....	112
3.33.1	Overview	112
3.33.2	Capabilities Implementation Status	115
3.33.3	Useful Resources	115
3.34	Solution - TWIMO (UNITE)	116
3.34.1	Overview	116
3.34.2	Capabilities Implementation Status	120
4	AIDOaRt AI-Augmented tool set Solutions in Use Cases	121
4.1	Application in ABI Challenge – Formal Verification of NNs: A “Step Zero” Approach for Vehicle Detection – UNISS (UNISS_SOL_01 - NNVer)	121
4.2	Application in ABI Challenge – Adoption of AI and ML techniques for image processing in the automotive context – INT (INT-DET, INT-DEPTH)	123
4.3	Application in AVL Challenge – Real-Driver Emission (AVL_RDE) – TUG (AALpy)	124
4.4	Application in AVL Challenge – Model-based Testing (AVL_MBT, Testing ADAS functions) – TUG (AALpy)	125
4.5	Application in AVL Challenge – Security Testing (AVL_SEC, Learning-based fuzzing of AGL) – TUG (AALpy).....	127
4.6	Application in AVL Challenge - Test Case Validation (AVL_TCV, ADAS Test Space Reduction) - AIT (DTSynth), ABO (STGEM).....	128
4.7	Application in VCE Challenge - MOMoT for FMU-based simulation optimization	130
4.8	Application in VCE Challenge - Keptn for FMI.....	131
4.9	Application in ALSTOM/BT Challenge – Automate Parameterization of Thermal Model	132
4.10	Application in WMO Challenge - Enhanced Test Results Exploration with Natural Language Processing - RISE LogGrouper and BIC-Tool	135
4.11	Application in WMO Challenge – Performance Data Exploration – Copado CRT	137
4.12	Application in WMO Challenge – Online Test Case Selection and Prioritization – ABO (STGEM)	138
4.13	Application in TEKNE Challenge – Design choice exploration/verification – UNIVAQ (HEPSYCODE), UCAN (S3D).....	140
4.14	Application in AVL Challenge – Real-Driver Emission (AVL_RDE) – UNITE (TWIMO), UNIVAQ, MDU	142
4.15	Application in TEKNE challenge – Operating Life – ROTECH	144
4.16	Application in CAMEA Challenge – Power-Aware Radar Configuration – ABO (STGEM) ...	145
4.17	Application in PRO Challenge – Automatic Resizing of Smart Port Computing Resources – PRO & ITI	147
4.18	Application in PRO Challenge – ACORDE	149
5	AIDOaRt Internal Hackathons Report	151
5.1	Introduction	151



5.2	Hackathon Organization	151
5.3	Call for Teams Challenges	152
5.4	Hackathon report	152
5.5	Hackathon survey	157
5.5.1	Participant Role and Domain Areas	158
5.5.2	Experience in the AIDoArT Internal Hackathons	160
5.5.3	Participation in the AIDoArT Internal Hackathons.....	160
5.5.4	Hackathon continuation.....	162
5.5.5	Other information	164
6	Conclusions	165
7	References	166
Appendix	169
A.	Mapping Solutions to AIDoArT AI-Augmented tool set Components.....	169
i.	Mapping to Ingestion & Handling	169
ii.	Mapping to Engagement & Analysis	170
iii.	Mapping to Automation.....	174
iv.	Mapping to AI for Requirements Engineering	176
v.	Mapping to AI for Modeling	178
vi.	Mapping to AI for Code	181
vii.	Mapping to AI for Testing.....	182
viii.	Mapping to AI for Monitoring	186
B.	Mapping Use Case Requirements to AIDoArT AI-Augmented tool set Components ..	189
ix.	Mapping to Ingestion & Handling	189
x.	Mapping to Engagement & Analysis	193
xi.	Mapping to Automation.....	203
xii.	Mapping to AI for Requirements Engineering	207
xiii.	Mapping to AI for Modeling	210
xiv.	Mapping to AI for Code	218
xv.	Mapping to AI for Testing.....	219
xvi.	Mapping to AI for Monitoring	224

Key Terminology Abbreviations

Abbreviations	Terminology
AI	Artificial Intelligence
AIOps	AI Operations
CPS	Cyber-Physical System
CPSoS	Cyber-Physical Systems of Systems
DevOps	Development Operations
LLM	Large Language Model
KPI	Key Performance Indicators
MBE	Model-Based Engineering
MBRE	Model-based Requirements Engineering
MDE	Model-Driven Engineering
ML	Machine Learning
SE	Systems and Software Engineering
SLR	Systematic Literature Review
SMS	Systematic Mapping Study
SysML	Systems Modelling Language
TRL	Technology Readiness Level
WP	Work Package

Partners Names Acronyms

In the following table, we list the partners' acronyms and full names. The short acronyms are used throughout the deliverable text to identify the partner providing a particular case study requirement or data requirement, or providing a given solution.

Partner Name Acronym	Full Partner Name
ABI	Abinsula SRL
ABO	Åbo Akademi
ACO	ACORDE Technologies S.A.
AND	Anders Innovations Oy
AIT	AIT Austrian Institute of Technology GmbH
AVL	AVL List GmbH
BT	Bombardier Transportation
CAMEA	CAMEA, spol. s r.o.
CSY	CLEARSY SAS
DT	Dynatrace Austria GmbH
HIB	HI Iberia Ingeniería y Proyectos S.L.
IMTA	Institut Mines-Telecom Atlantique Bretagne-Pays de la Loire
INT	Intecs Solutions S.p.A.
ITI	Instituto Tecnológico de Informática
JKU	Johannes Kepler University Linz
MDH	Maelardalens Hoegskola (Coordinator)
PIO	Previsionio
PRO	Prodevelop SL
QEN	Qentinel Oy
RISE	RISE Research Institutes of Sweden
ROTECH	Ro Technology srl
SOFT	Softteam
TEK	Tekne SRL
TUG	Technische Universitaet Graz
UCAN	Universidad de Cantabria
UNISS	Università degli Studi di Sassari
UNITE	Università degli Studi di Teramo
UNIVAQ	Università degli Studi dell'Aquila
UOC	Fundació per a la Universitat Oberta de Catalunya
VCE	Volvo Construction Equipment AB
WESTMO	Westermo Network Technologies AB

1 Introduction

As outlined in the AIDOaRt project proposal and previous AIDOaRt deliverables, AIDOaRt aims at merging principles and techniques from Model-Driven Engineering (MDE) and Artificial Intelligence (AI) to enhance the facilitation of continuous development, also known as the Development Operations (DevOps) process, for large and intricate Cyber-Physical Systems (CPS).

Work Package (WP) 4 represents a key work package that leverages the capabilities provided by WP2 and WP3 (please, refer to D2.3 "Data collection and representation" and D3.4 "AIDOaRt Core Infrastructure and Framework") to effectively utilise diverse data artefacts, primarily data models, alongside other available software and system engineering models. Furthermore, WP4 delivers AI-augmented tools to support various phases of the DevOps process.

Specifically, the primary objective of WP4 is to facilitate the development of the AI-augmented Tool Set, which extends the AIDOaRt framework established in WP3, catering to the diverse requirements of the various types of CPSs being developed (i.e., the use case requirements defined in WP1).

Additional functionalities concerning various CPS development tasks will encompass the utilisation of AI for requirements analysis, monitoring, modelling, coding, and testing, thereby integrating AI Operations (AIOps) and MDE environments. Following the AIOps methodology, the tool set will enable the ingestion of data, events, and metrics from multiple sources, their analysis through machine learning (ML) and AI techniques, and the automation of development-related operations. This tool set will be deployed in use case challenges and will further be utilised within the context of various project use cases, as outlined in WP1 and WP5.

Considering that D4.3 is the final deliverable in this series, serving as the culmination of WP4 efforts, it represents the ultimate snapshot of the WP4 solutions. Building upon the groundwork laid out in previously released deliverables, particularly D4.1 and D4.2, and in parallel with similar deliverables from other work packages such as D2.3 and D3.4 (already delivered by M28), D4.3 aims to refine the capabilities and interrelationships between use cases and solutions. The primary objective of D4.3 is to present the current perspective and realistic vision of the AIDOaRt AI-augmented tool set.

The document is structured as follows: Section 2 provides an overview of the AIDOaRt AI-Augmented solutions. Section 3 offers the progress status of the AIDOaRt AI-Augmented tool set. Section 4 presents an extensive overview of the current application of the solutions from the AIDOaRt AI-Augmented tool set within specific use case challenges. Section 5 details the outcomes and achievements of the AIDOaRt Internal Hackathons. Concluding remarks are presented in Section 7. Finally, the Appendix contains the final version of the mapping between the solution components fulfilling the functional specifications of the AI-Augmented tool set, along with the mapping of use case requirements and data requirements to the AI-Augmented tool set components

2 List Solutions related to the AIDoArt AI-Augmented tool set

We provide the list of solutions related to the AI-Augmented tool set, their features/capabilities that are delivered/made available at the baseline, intermediate or final stage of the project's roadmap. In the following, we provide a recall of the original milestones' roadmap:

MS1 (M6) to MS3 (M16) = Baseline

MS4 (M20) to MS5 (M31) = Intermediate

MS6 (M28) to MS8 (M32) = Final

Please, note that due to the project extension, some milestones have been rescheduled. Specifically, MS5, originally scheduled for M24, is now planned for M31. Similarly, MS7, initially planned for M32, has been shifted to M37, and MS8, previously set for M36, is now targeted for M42. In the following sections, we will be referring to the current updated milestones. In the previous deliverable D4.2, [AIDoART-D4.2], we included all solutions having features already available as baseline (M0) and/or released from MS1 until MS5, included. In this deliverable, we incorporate solutions having features planned for delivery in the final phase of the project, i.e., with a delivery date between MS6 and MS8. Furthermore, we have included some tools that were originally excluded from this deliverable due to their expected delivery date falling at the intermediate stage of the project. Nevertheless, considering the project's extension, as these tools have undergone ongoing enhancements, their presentation has become necessary. These tools are the following: ESDE (ACO), Position Monitoring for Industrial Environment (ACO), HIB_logAnalyzer (HIB), TATAT (PRO), CRTQI (QEN) and QEDITOR (QEN). Table 1 provides comprehensive details for each solution, outlining its planned features for delivery at the baseline, intermediate phase, or final phase of the project. Moreover, it specifies whether the solution has been included in D4.2 and/or D4.3.

Solution Name	Baseline	Intermediate	Final	Inc in D4.2 ?	Inc in D4.3 ?
STGEM (ABO)			Test generation and prioritization (MS7 (M37))		Yes
ESDE (ACO)		Multi-level functional and performance logs and traces (MS4 (M20)), Automated (or semiautomated) bug detection/prediction (MS5 (M31))		Yes	Yes

Solution Name	Baseline	Intermediate	Final	Inc in D4.2 ?	Inc in D4.3 ?
Position Monitoring for Industrial Environment (ACO)		Monitoring System Design and Development (MS4 (M20)) , AI/ML based analysis of monitored data (MS5 (M31)) , Provide Resilient and Accurate Position (MS4 (M20))		Yes	Yes
DTsynth (AIT)			Digital Twin Learning (MS6 (M28)) , Digital Twin Learning-Data Derivation (MS6 (M28))		Yes
Mimir (AND)			Mimir (MS7 (M37))		Yes
Cloud expertise (AND)			Cloud expertise (MS7 (M37))		Yes
Infrastructure as Code (IaC) expertise (AND)			IaC expertise (MS7 (M37))		Yes
Active DoE (AVL)			Active DoE (MS7 (M37))		Yes
Keptn (DT)			Keptn CloudEvents (MS6 (M28)) , Keptn Control-Plane (MS6 (M28)) , Keptn Quality Gates (MS6 (M28))		Yes
HIB_logAnalyzer (HIB)	HIB-LA (MS2 (M12))			Yes	Yes
EMF Views (IMTA)		Model Viewpoint and View specification (MS4 (M20)) , Model Viewpoint and View navigation and query (MS5 (M31))	Model Viewpoint and View computation (MS6 (M28)) , Model Viewpoint and View update (MS7 (M37))	Yes	Yes
INT-DET (INT)			Real-Time Object Detection (MS6 (M28))		Yes
INT-DEPTH (INT)			Depth Estimation (MS7 (M37))		Yes

Solution Name	Baseline	Intermediate	Final	Inc in D4.2 ?	Inc in D4.3 ?
a2k-runman (ITI)			a2k/tunning (MS7 (M37))		Yes
a2k-depman (ITI)			a2k/optimiser (MS7 (M37))		Yes
MOMOT (JKU)	MOMOT (MS1 (M6))		AI-augmented MOMOT (MS7 (M37))	Yes	Yes
GAN-Based Instance Model Generator (JKU)			Instance_Generator (MS7 (M37))		Yes
Requirements Ambiguity Checker (MDU)			checkRequirementsAmbiguity (MS8 (M42))		Yes
TATAT (PRO)		TestAutomationlink (MS4 (M20))		Yes	Yes
CRT (QEN)			Cloud test execution platform (MS7 (M37))		Yes
CRTQI (QEN)		QI for DevOps (MS4 (M20))		Yes	Yes
QEDITOR (QEN)		QEditor - AI-assisted test authoring (MS5 (M31))		Yes	Yes
VARA (RISE)	Requirements-based reuse analysis and feature reuse recommendation (MS1 (M6))			Yes	
LogGrouper (RISE)			GroupLogs (MS8 (M42))		Yes
BugIdentifier (RISE)			BugInducingCommit (MS8 (M42))		Yes
DataAggregator (ROTECH)			Data aggregation (MS6 (M28))		Yes
Modelio (SOFT)	Modelling (MS1 (M6)), Meta-Modeling (MS1 (M6)), Model Exchange	Automated Requirements Identification, Extraction & Classification (MS5 (M31))	CPS Simulation (MS6 (M28)), Process Execution (MS6 (M28)), DevOps Connector (MS6 (M28)), AI-Assisted Modelling (MS6 (M28)), AI-Optimized Model	Yes	Yes

Solution Name	Baseline	Intermediate	Final	Inc in D4.2 ?	Inc in D4.3 ?
	(Export / Import) (MS1 (M6)) , Model Transformation (MS1 (M6)) , Code Generation (MS1 (M6)) , Reverse engineering (MS1 (M6)) , Documentation Generation (MS1 (M6)) , Model Update from Edited Documentation (MS1 (M6)) , Traceability (MS1 (M6)) , Model Consistency Checking (MS1 (M6))		Checking (MS6 (M28)) , AI-Enhanced Test Generation (MS6 (M28))		
Constellation (SOFT)	Model Distributivity (MS1 (M6)) , Version Control (MS1 (M6)) , Collaboration (access control, parallel editing, synchronization...) (MS1 (M6))	Collaboration Workflow (MS5 (M31))		Yes	
AALpy (TUG)			Automata learning of deterministic systems		Yes

Solution Name	Baseline	Intermediate	Final	Inc in D4.2 ?	Inc in D4.3 ?
			(MS7 (M37)), Automata learning of non-deterministic systems (MS7 (M37)), Automata learning of stochastic systems (MS7 (M37))		
S3D (UCAN)			S3D (MS6 (M28))		Yes
SoSIM (UCAN)			SoSIM (MS6 (M28))		Yes
UNISS_SOL_01 (UNISS)			Requirements consistency verification (MS7 (M37))		Yes
UNISS_SOL_02 (UNISS)			NN verification (MS7 (M37))		Yes
UNISS_SOL_03 (UNISS)			Test generation (MS7 (M37))		Yes
UNISS_SOL_04 (UNISS)			Property verification (MS7 (M37))		Yes
HEPSYCODE (UNIVAQ)			Performance Simulation and Predictions (MS7 (M37)), Design Space Alternatives Exploration (MS7 (M37)), Model-Driven DevOps approach for HW/SW Co-Design (MS7 (M37))		Yes
MORGAN (UNIVAQ)			Recommender of model and metamodel elements (MS7 (M37))		Yes
TWIMO (UNITE)			Domain-specific modeling (MS7 (M37)), MLAnalysis (MS7 (M37))		Yes

Table 1 List of Solutions related to the AIDoArt AI-Augmented tool set

3 Progress status of the AIDOaRt AI-Augmented tool set

This Section presents an overview of the implementation achievements and development status of the AIDOaRt AI-Augmented tool set solutions with regards to the Intermediate milestone, as described in the previous deliverables [AIDOART-D4.1] and [AIDOART-D4.2]. The past and current achievements of each solution are presented and commented on in this section, in order to make explicit the concrete realisations in terms of capabilities development while we are formally concluding the work in the context of WP4.

For each of the concerned solutions, information is provided by the corresponding solution provider according to the following structure:

1. In the first subsection, an overview of the solution is provided, consisting of a synopsis table that offers a summary of key information, along with a descriptive textual section explaining the development of the solution.
2. In the second subsection, a 'Capabilities and implementation status' table is presented, detailing for each capability its description, implementation level, delivery date, licence, deviations, and any accompanying notes.
3. In the third subsection, all the relevant resources associated with the solution, including online materials, scientific publications, videos, and other pertinent sources are reported.

It should be noted that this deliverable describes 34 tools which are related to the AIDOaRt AI-Augmented tool set with implementation progress delivered in the final phase of the project, i.e., with a delivery date between MS6 (M28) and MS8 (M42).

3.1 Solution - STGEM (ABO)

3.1.1 Overview

Solution: STGEM		Partner: ABO
Current TRL: [TRL 4]	License: MIT	Contacts: iporres@abo.fi , japeltom@abo.fi , aashraf@abo.fi
Online Resource: https://gitlab.abo.fi/stc/stgem	Collaborators: Alstom, AVL, CAMEA, Westermo	
Description: STGEM (System Testing using Generative Models) is an academic tool for automatic test generation and prioritisation for software-intensive CPSs. It uses AI/ML techniques to automate and improve the performance of several activities in the testing process including test generation, test selection and prioritisation, and test scheduling activities.		
Improvement: STGEM has steadily improved as more features have been added (support for multiple output requirements) and bugs have been fixed.		
Intended Users: Researchers (interested in software test generation and prioritisation) and industry (companies like Alstom, AVL, CAMEA, Westermo which are interested in using AI/ML techniques to automate and improve test generation, test selection and prioritisation, and test scheduling).		

Satisfied Requirement: <ul style="list-style-type: none"> GR Mod 02: Use formal models, automated reasoning and/or ML techniques for test generation GR Mod 06: Use AI-based methods for easy configuration GR Test 01: AI/ML techniques for test case generation from high level models GR Test 02: Automatic execution of test cases GR Test 04: Evaluation of testing results GR Test 06: Integrate and analyse testing phase into the AIDOaRt DevOps pipeline 	
AI-Augmented tool set components implemented: <ul style="list-style-type: none"> AI for Testing: STGEM uses AI/ML techniques to automate and improve test generation, test selection, and test scheduling and execution components 	Task: T4.1, T4.2, T4.3, T4.4
Use within AIDOaRt Use Cases	Already planned use: AVL_TCV_UCS1: SCENIUS Test Case Selection Validator AVL_TCV_UCS2: SCENIUS parameter recommender BT_UCS2: Automated model parametrization CAM_UCS3: Enable low-power device configuration W_USC_1: AI-Augmented DevOps development process
	Potential foreseen links: AVL_MBT_UCS1: Model-Based Testing of Functional Scenarios
Use within AIDOaRt challenges: <ul style="list-style-type: none"> 5th Hackathon: Parameter space reduction, Power-aware radar configuration tuning 4th Hackathon: Automatic parametrization of PS controllers 3th Hackathon: Automatic parametrization of PS controllers, Power-aware radar configuration tuning 	

Table 2 Synopsis table of the STGEM Solution

STGEM is a software pipeline intended for automatic falsification of black-box systems such as CPS. The main goal is to provide for researchers a modular and easy-to-use pipeline for the development and evaluation of black-box falsification algorithms. In addition to providing falsification capabilities to anyone, STGEM includes a reference implementation of the ML-based CPS falsification algorithms (OGAN, WOGAN) developed in ABO within the AIDOaRt project. It also provides means to reproduce and validate our research experiments independently. STGEM is developed in Python using standard ML libraries, and it allows the usage of extensions written in other programming languages.

As shown in Figure 1, the pipeline consists of three main components: the system under test, requirement monitor, and falsification algorithm. Each of the components is designed modularly to allow an efficient DevOps development of STGEM. The component for the system under test allows any black-box system with signal inputs and outputs to be integrated into STGEM. Any black-box system can be integrated, but we provide a specific interface to integrate MATLAB models and simulators which are often used in CPS development and testing. We allow the user to write a custom requirement monitor for monitoring the system behaviour, and we support out-of-the-box requirement monitoring of requirements expressed formally in signal temporal logic which is used in many standard CPS benchmarks. We support arbitrary falsification algorithms, and we currently implement the OGAN and WOGAN algorithms which are based on our research. These are novel falsification algorithms using novel ML techniques based on Generative Adversarial Networks.

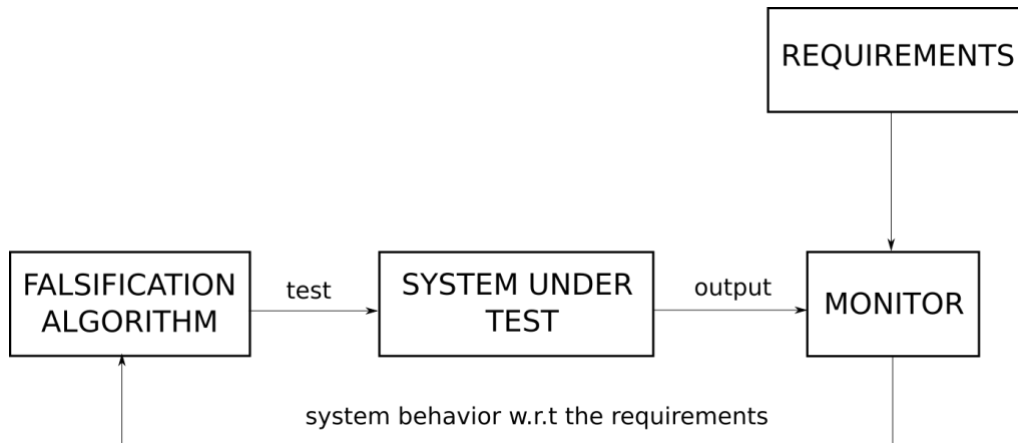


Figure 1 Three main components in the STGEM pipeline

3.1.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
Modular design, API design	Implementation Level: fully implemented Estimated Delivery Date: MS3 (M16) License: MIT	--
Support for MATLAB models, support for requirements in signal temporal logic	Implementation Level: fully implemented Estimated Delivery Date: MS3 (M16) License: MIT	--
Support for ML-based falsification algorithms	Implementation Level: fully implemented Estimated Delivery Date: MS7 (M37) License: MIT	OGAN and WOGAN algorithms fully implemented. Research into new falsification algorithms finished.
Evaluation capabilities, documentation, and tutorials	Implementation Level: partially implemented	Preliminary evaluation support and documentation exists. Example usage reported as part of research papers. Paper tool with improved documentation in progress.

<p>Test generation and prioritisation: An experimental system for automatic test generation for software intensive systems. It uses ML to automate and improve the performance of many components used in test generation including test input selection, test scheduling and oracle synthesis. ML models are created online during the testing process using adaptive learning algorithms and also offline, between system integration builds, using supervised learning algorithms.</p>	<p>Implementation Level: Fully implemented Estimated Delivery Date: MS7 (M37) License: MIT Deviation:</p>	<p>--</p>
--	---	-----------

Table 3 Capabilities Implementation Status of the STGEM Solution

3.1.3 Useful Resources

Main website, source code repository, installation instructions and documentation:

<https://gitlab.abo.fi/stc/stgem>

Related publications:

- Jarkko Peltomäki and Ivan Porres. Falsification of Multiple Requirements for Cyber-Physical Systems Using Online Generative Adversarial Networks and Multi-Armed Bandits. The 6th. Intl. Workshop on Testing Extra-Functional Properties and Quality Characteristics of Software Systems, ITEQS 2022.
- Jarkko Peltomäki, Frankie Spencer and Ivan Porres. Wasserstein Generative Adversarial Networks for Online Test Generation for Cyber Physical Systems. The 15th Intl. Workshop on Search-Based Software Testing, SBST 2022.
- Jarkko Peltomäki, Frankie Spencer and Ivan Porres. WOGAN in the SBST 2022 CPS Tool Competition. The 15th Intl. Workshop on Search-Based Software Testing, SBST 2022.
- Jesper Winsten, Ivan Porres. WOGAN at the SBFT 2023 Tool Competition - Cyber-Physical Systems Track. The 16th Intl. Workshop on Search-Based and Fuzz Testing, SBFT 2023.
- Claudio Menghi, Paolo Arcaini, Walstan Baptista, Gidon Ernst, Georgios Fainekos, Federico Formica, Sauvik Gon, Tanmay Khandait, Atanu Kundu, Giulia Pedrielli, Jarkko Peltomäki, Ivan Porres, Rajarshi Ray, Masaki Waga and Zhenya Zhang. ARCH-COMP 2023 Category Report: Falsification, Proceedings of 10th International Workshop on Applied Verification of Continuous and Hybrid Systems (ARCH23), vol 96, pages 151-169.
- Jesper Winsten, Valentin Soloviev, Jarkko Peltomäki, Ivan Porres, Adaptive test generation for unmanned aerial vehicles using WOGAN-UAV. Preprint available at <https://gitlab.abo.fi/aidoart/stgem/-/blob/main/www/articles/wogan-uav-sbft2024-competition-preprint.pdf>

- Jarkko Peltomäki, Jesper Winsten, Maxime Methais, Ivan Porres, Testing Cyber-Physical Systems with explicit output coverage. Preprint available at <https://gitlab.abo.fi/aidoart/stgem/-/blob/main/www/articles/EOC-ITEQS2024-preprint.pdf>
- Jarkko Peltomäki, Ivan Porres, Requirement falsification for Cyber-Physical Systems using generative models, arXiv 2023. Preprint available at <https://arxiv.org/abs/2310.20493>

Talks:

- <https://youtu.be/Vwxu6TtzBYs?t=8349>
- <https://www.youtube.com/watch?v=EF13eiidhA0>

3.2 Solution - ESDE (ACO) - AI/DL based traces analysis

3.2.1 Overview

Solution: ESDE - AI/DL based traces analysis		Partner: ACORDE
Current TRL: 3	License: Proprietary	Contacts: fernando.herrera@acorde.com
Online Resource: https://sistemas.acorde.com/ acorde@acorde.com		Collaborators: None
<p>Description: ESDE is an Electronic System Level embedded (ESL) Software Development Environment developed by ACORDE to improve embedded software (eSW) design productivity. The basic architecture of this tooling was introduced in section 3.1.1 of D2.2. This environment aims integration of several ESL concepts for a more productive development and validation of embedded software. Some of those concepts conjugated by ESDE are the high-level specification of eSW (in SystemC); the automated generation of the eSW implementation (e.g., “.elf” or “.bin” production); and the validation of that eSW implementation on a Virtual Platform (VP) model. The main research and advance achieved in AIDoArt is the exploitation of the VP model, i.e. to enable from the execution of real eSW implementation the production of traces that, in turn, can be exploited by means of AI/ML technologies for validation purposes, e.g. bug detection, performance anomalies. A main advantage of VP technology here is that instrumenting the VP for trace production does not involve instrumentation-related perturbation of performance estimates.</p>		
<p>Improvement: ESDE tool had initial TRL4. However, the AI/DL based traces analysis feature researched in AIDoArt had initial TRL 2. By the end of Y2, in D4.2, a first version of the capability to extract key traces from the virtual platform model was ready. Later (M28) a more defined, modular architecture of the traces generation and representation capabilities, with a first example on application-level traces was reported in D3.4. Currently, there is a first Proof-of-Concept (PoC), where an AI/DL based anomalies analysis has been performed.</p>		
<p>Intended Users: Current target is generating a modular framework and methodology, suited to different VP environments, for different processor targets and tool chains for internal use, i.e. aimed to improve productivity of ACORDE embedded developers.</p>		

Satisfied Requirement: <ul style="list-style-type: none"> GR Mon 1.2 - Monitoring in AIDOaRt is able to access on-line data GR Mon 2.2 - Monitoring in AIDOaRt could be done with the purpose of identifying deviations, anomalies or security events 	
Satisfied Use Case Requirements: <ul style="list-style-type: none"> PRO_R05 - Detect automatically anomalies in the solution during the execution based on AI 	
AI-Augmented tool set components implemented: <ul style="list-style-type: none"> <u>AI DevOps Engineering - AI for Monitoring- AI/ML for Anomaly Detection</u> - Monitoring or probing of specific metrics on embedded virtual system used to detect anomalous execution patterns (relation to testing too). 	Task: T4.1 and T4.2.
Use within AIDOaRt Use Cases	Already planned use: Early and automated validation of embedded software, e.g. positioning software, implementation, detecting potential bugs and/or performance anomalies. Potential foreseen links: Practically, any other use case in and out of AIDOaRt with embedded software development.
Use within AIDOaRt challenges: This work has not been exposed to AIDOaRt challenges.	

Table 4 Synopsis table of the ESDE Solution

Figure 2 reproduces the AIDOaRt extensions of the ESDE architecture (advanced in D2.3, and D3.4 for the data and infrastructure contexts), this time highlighting the extension directly related to WP4, i.e. the integration of the Generic Anomalies Analysis (GAA) in the flow.

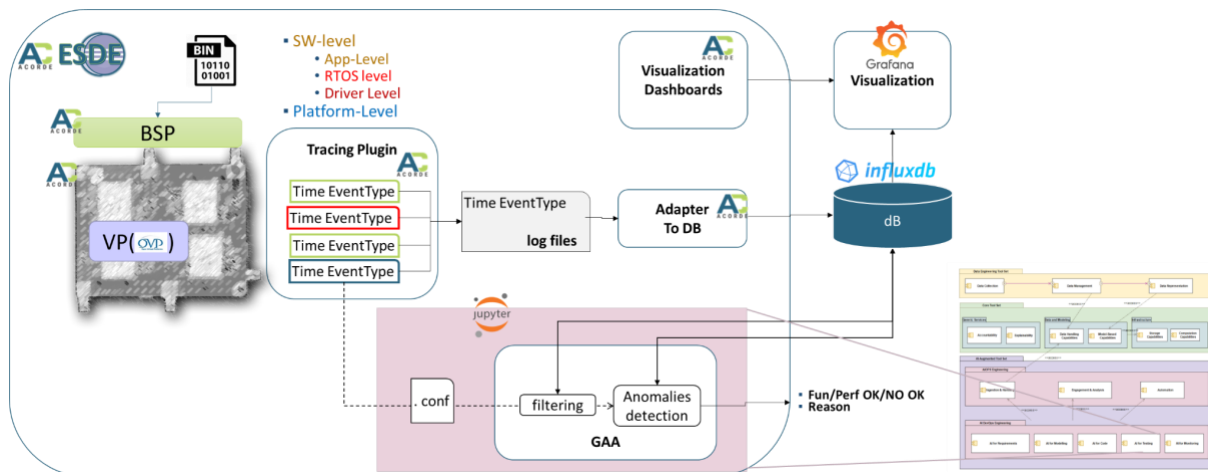


Figure 2 ESDE extended with Generic Anomalies Analysis service (AIDOaRt AI for monitoring)

In D4.2 the first version of the multi-level tracing capability (start/end of functions, executed instructions and access to memory regions), a mandatory step for the analysis, was reported. This tracing capability is implemented as C/C++ code, integrated on the specific Open Virtual Platform (OVP) simulation control application, and invoking monitoring services of the OVP simulation engine. For a more exploitable result, much of this code (represented as “Tracing Plugin” in the figure), has been separated from the OVP simulation control wrapping and from the monitoring services.

Regarding the GAA part, for a former PoC a Jupyter notebook implementation has been developed. That notebook includes a basic configuration, i.e., which input traces are used for training and inference, and the specific type of neural network. The notebook applies the training with the traces of a running application on the virtual platform, and then the anomaly detection on a newer test set on traces on the same application and virtual platform, but with different input data.

For the time being, a more complex and more multi-level application example than the one reported in D3.4 has been developed, which contains application code, relying on a RTOS (FreeRTOS), over a single ARM core-based platform model. The example receives a stream of data and continuously processes fixed chunks of data. Each data chunk arrived is a vector of integer data randomly ordered. The processing done is the ordering of the data chunk. The ordering is performed by two (FreeRTOS) application tasks. Each task takes over the ordering of a half of the input data chunk.

A multi-level trace was produced for this example, including the entries/exits of the two ordering tasks (application), several RTOS events (tick timer, context switch), and hardware platform level traces, i.e. instructions and data memory accesses. A former set of experiments were done, with a configuration which selected traces on memory accesses. The experiment was specifically designed to generate a training set with a normal data input (data chunks randomly ordered) and a test set which includes an interval of anomalous data (data chunks strictly ordered, in the opposite sense required by the algorithm).

The research questions were if a) the anomalous input data involved some difference or stress in the platform, i.e. some low-level pattern or anomaly pattern on memory traces, and in that case, b) if the AI/DL analysis on memory traces enables the automated detection of such anomaly.

Figure 3 shows a zoom in on the fetch memory trace in an interval transitioning between the normal input to the anomalous input. This trace was enabled by the features developed in previous steps of the project. Tracing information also enabled colouring the traces for representing access to different functionalities (enabling distinction between application and RTOS fetches).

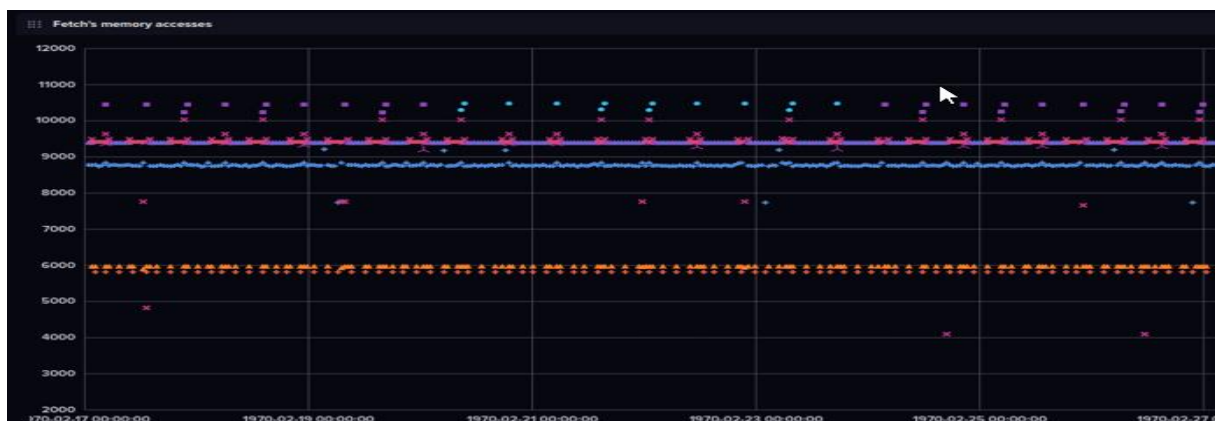


Figure 3 Fetch memory trace from the VP tracing extension developed in AIDOaRt

In this case, this allows observing, right above address 9000, a red-purple alternation pattern associated with the alternation of the two ordering tasks. A deeper look reveals that the former (“normal”) part of the alternation shows a similar time latency for each task. However, the later part

of the zoom in, once the anomalous input data are fed, reflects different, more unbalanced tasks durations, as the corner input data case enforces one of the tasks to work more. Therefore, the anomaly has a lower-level reflection that answers the first research question.

The next step was to find if/how AI/DL can help to automate the finding of that pattern anomaly. Configurations with convolutional autoencoders were tested so far. These configurations were able to detect the anomaly interval. Figure 4 shows the original (blue) vs the reconstructed (red) memory access traces for the training and test datasets (delimited by the vertical black bar).

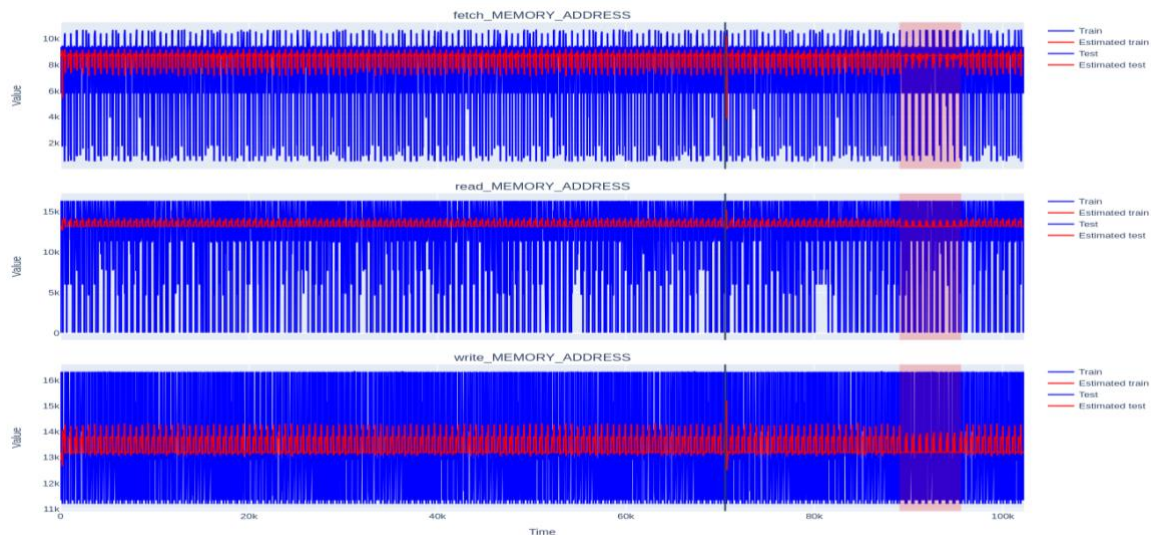


Figure 4 Memory traces for the training and test set (until/from black vertical bar)

The wide light red vertical stripe delimits the interval where the anomalous input data were injected. The following figure shows the MAE reconstruction error. Horizontal black line reflects the configured anomaly threshold. Figure 5 shows how a configuration enables the automatic detection of the anomaly interval by detecting an excessive error on the reconstruction of memory fetch patterns.

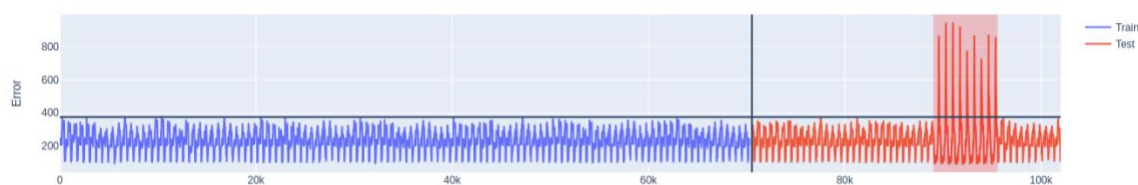


Figure 5 Input data anomaly detected through reconstruction error on memory fetches

While the second research question was answered, further research has shown that this is a research area where further effort is required for a mature, optimal solution. For instance, another successful configuration showed better reconstruction error for the normal scenario (shown in Figure 6), but less clear detections, which was apparently counter intuitive because it used more neurons (640 vs 312). However, Figure 5 configuration uses a different quantity of layers and neurons' distribution that led to a more effective and efficient detection.

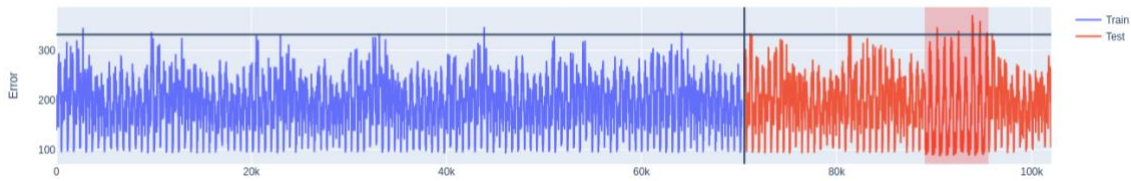


Figure 6 A configuration with less reconstruction error, but less effective detection

The exploration of other configurations and neural network types, as well as the exploration of other type of high-level (e.g. programming error, like deadlock) or lower level (e.g., RTOS bug, platform failure) issues are of interest, and so aimed for future steps.

3.2.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
<p>Multi-level functional and performance logs and traces: Framework able to provide time series of performance metrics at different levels of abstraction and at different implementation levels (platform HW, platform SW, application SW)</p>	<p>Implementation Level: Implemented Estimated Delivery Date: MS4 (M20) License: Proprietary – ACO (*). Deviation: No deviation</p>	<p>Implemented. Traces generation mechanism, supporting the traces types mentioned in the previous section has been tested in more examples, showing some stability to consider this feature complete.</p>
<p>Automated (or semi automated) bug detection/prediction: Ability to learn and detect or predict possible functional or performance bugs based on performance traces. Two possibilities envisioned. First one, based on offline trace analysis, can handle non-causal analysis for better anomalies/errors detection. This feature will support product fixes and development smoothly integrated in a DevOps environment. A second possibility is to simulate the system equipped with monitoring probes and a ML engine capable of using the collected metrics for an on-the-fly (real-time, or at least causal) computation for detecting/predicting anomalies.</p>	<p>Implementation Level: Implemented Estimated Delivery Date: MS5 (M31) License: Proprietary – ACO (*). Deviation: No Deviations. From the former stages of the project, ACORDE has prioritised the exploration of AI/DL techniques for anomalies on time traces on its application the Position Monitoring for Industrial Environment Solution, as was assessed to enable a closer cooperation with other partners and an exploitation with potential for new business opportunities. Anyhow, ACORDE committed a best-effort to explore the AI/DL techniques at a different scenario, i.e. VP-based validation, with the main</p>	<p>Current AI/DL analysis is implemented as a Jupyter notebook, able to read from traces generated by the tracing plugin, via an intermediate database.</p>

	initial objective of ACORDE productivity improvement.	
--	---	--

Table 5 Capabilities Implementation Status of the ESDE Solution

(*) It refers to specific extensions performed on ESDE to obtain functional and performance logs and traces developed by ACO and considered key for protecting competitive advantages achieved thanks to AIDOaRt. It excludes any open-source and/or third-party source used, and any open-source extension that needs to be published. Eventually, ACO might release as open-source fixes and extensions which do not compromise ACO competitive advantages granted by AIDOaRt.

3.2.3 Useful Resources

Related publications:

- <https://towardsdatascience.com/lstm-autoencoder-for-anomaly-detection-e1f4f2ee7ccf>

Other relevant resources: <https://c4d.lias-lab.fr/index.php/WP6-20> (ESDE status after COMP4DRONES project)

3.3 Solution - Position Monitoring for Industrial Environment (ACO)

3.3.1 Overview

Solution: Position Monitoring for Industrial Environment		Partner: ACORDE
Current TRL: 3/4	License: Proprietary	Contacts: fernando.herrera@acorde.com
Online Resource: https://sistemas.acorde.com/ acorde@acorde.com		Collaborators: Prodevelop, ITI
<p>Description: An industrial monitoring solution consisting of an edge infrastructure that can be smoothly connected to cloud resources, capable to execute several services on the edge (prominently database service and AI-based anomaly analysis). At the edge side, the developed infrastructure consists of a gateway able to run the data gathering and pre-processing services, which greatly releases the cloud from storing local irrelevant data and from filtering computation, more suitable for the edge. The gateway also supports a deep learning-based service that can detect anomalous behaviours of the monitored plant. At the IoT sensing side, a specific Positioning IoT solution, to provide more cost-effective, resilient and accurate geo-referencing of the monitored assets has been developed.</p>		
<p>Improvement: From the initial technological concept (TRL 2) discussed with Prodevelop, ACORDE was able to run by Y2 an initial demonstrator of Generic Anomalies Analysis (GAA), able to run the anomaly detection service at the gateway platform. Since then, ACORDE has achieved a significant advance through the development of several aspects/features:</p> <ul style="list-style-type: none"> - A specific Location Anomalies Analysis (LAA) deep-learning method has been developed, focused on the detection of anomalies on the location traces of the monitored assets. - Anomaly interval analysis (via anomalies times clustering) and LAA region analysis - xAI capabilities for Root cause (responsible) parameter and device analysis. 		

- All GW analysis services (data collection and formatting for storage in a time-base data base and LAA), are running in a full host docker based demo (4-th version of the demo)
- The design of the devices of the Positioning Monitoring Solution (Positioning IoT or PloT and Base Station or BS) is completed. Its implementation is almost complete. Final integration steps are ongoing for field testing.
- A second version of the gateway platform has been already designed and implemented, which will bring enhanced features, e.g. in manageability and storage capability.
- An initial instrumentation of the gateway platform for monitoring the computational, memory and communication resources has been set up. The aim is to extract performance data that can serve to dimension the platform or to extract performance models by partners (collaboration with ITI).

This means a status pretty close to TRL 4. It is considered that TRL 4 will be reached once the dockerized services are moved to the gateway platform (version 5 of demo). Other planned steps for the final phase of the project are the test of PloT and BE in a relevant environment to reach TRL 5.

Intended Users: Providers of Port monitoring & management solutions like Prodevelop. It could be likely suited or easily adapted for monitoring on other industrial scenarios, which can include different types of sensors, and also moving assets on outdoors scenarios.

Satisfied Requirement:

- GR Mon 1.2 - Monitoring in AIDoArT is able to access on-line data
- GR Mon 2.2 - Monitoring in AIDoArT could be done with the purpose of identifying deviations, anomalies or security events
- GR Mon 2.3 - Monitoring in AIDoArT could be done with the purpose of identifying clusters of some type of anomalies (spatial, temporal, etc)
- GR Mon 2.6 - Monitoring in AIDoArT could be done with the purpose of identifying root-causes to anomalies/deviations.

Satisfied Use Case Requirements:

- PRO_R05 - Detect automatically anomalies in the solution during the execution based on AI

AI-Augmented tool set components implemented:

- AIOPS Engineering- Engagement & Analysis- AI/ML for Anomaly Detection - DL techniques used for detecting anomalous patterns on monitored data

Task: Mostly T4.2 and T4.4, but also activity related to T4.1 and T4.3.

Use within AIDoArT Use Cases

Already planned use: Used in PRO (Smart Port Monitoring Platform or SMPM) use case, mostly on the UCS3 and UCS5

Potential foreseen links: No foreseen links

Use within AIDoArT challenges:

The ACORDE Position Monitoring Solution tackles the anomalies detection challenge of the Use Case Scenario 3 (UCS3) for the detection of anomalies in the PRO use case. The solution has been progressively developed across all the hold hackathon challenges. The solution is also related to the Use Case Scenario 5 (UCS5) for the challenge of the PRO use case, to proper dimensioning of SPMP resources (that start to be explicitly addressed from the 4-th hackathon).

Table 6 Synopsis table of the Positioning Monitoring for Industrial Environment Solution

Figure 7 provides an update of the overall solution for Position Monitoring in an industrial environment, in the context of the smart port monitoring context (referenced to the status in D4.2).



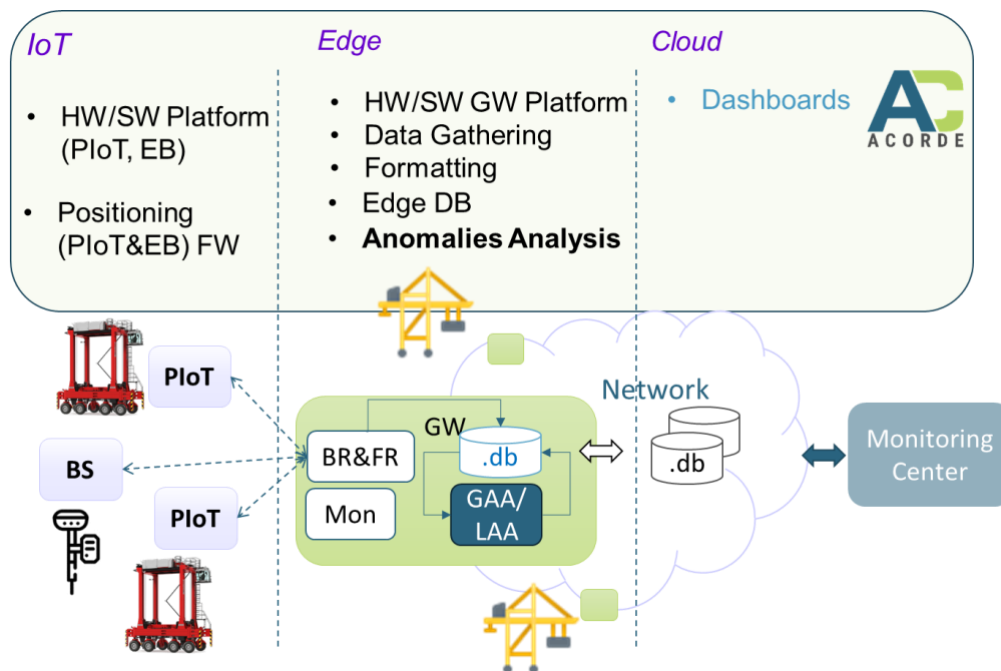


Figure 7 HW/SW architecture of Industrial Positioning Monitoring on the Computing Continuum

On the IoT side, the positioning solution developed by ACORDE is composed of a local Base Station (BS), able to provide corrections to the Positioning IoTs (PloTs) set up on the dynamic assets of the plant (straddle carriers in the port). Location information is sent to the Gateway (GW) via any of the available communication interfaces.

At the GW side, ACORDE solution has already set up the data broker and formatting (BR&FR) services to gather sensed location data and store them into the time series database. In addition, a first version of a set of services for monitoring the performance of the GW (and which enables further collaboration with ITI has been developed. It is represented as the “Mon” box. Finally, in the most direct relation to WP4 activities, a service able to provide generic and specific (location) Anomaly Analysis (GAA/LAA) has been developed. This service is based on the Generic Anomaly Analysis (GAA) already mentioned in D4.2 and aimed for a generic set of variables, but now targeted to the analysis of a set of 2D positions. For that, specific functionality for data preparation (filtering, reformatting) has been defined, e.g., defining the usage of north and east components in a NED coordinates system, and setting a predefined order of these components to source the LSTM network for training, that must be strictly respected at inference time. The same protocol is used to prepare the data in operation time, which are presumed to be fed in geo-referenced coordinates (Latitude, Longitude, height) and must be converted into NED and ordered according to the configuration used for training before inference.

The capabilities of new Location Anomaly Analysis (LAA) analysis were illustrated through an example introduced at the 5-th Hackathon held at Linz in Dec. 2023. This example is based on synthetic data. While synthetic, many aspects were considered in this data generation in order to reflect a realistic scenario. In particular, such data generation considered feasible tracks and directions of the straddle carriers on the port layout; aspects of the port operation; functionality of the areas in the layout (straddle carrier parking areas, truck docking areas, container and transit areas), container orientation,

width of transit zones, physical constraints to direction changes and velocity limits of the straddle and expected positioning data error. The left-hand side of Figure 8 overlays 800 trajectories (different colours are used per trajectory) corresponding to 800 load/unload operations performed by 2 straddle carriers (SCs). They represent the normal operation and form the training set that was used to train the DL model, the core of the LAA service. On the right-hand side, a test set, consisting of 80 operations of 2 straddle carriers, most of them representing the “normal trajectories”, according to the ones observed in the train set. Moreover, this test set was specifically designed to present at least 2 separated anomalies, i.e. on different SCs, and of different types.



Figure 8 Representation of training and test sets over a map

The implemented LAA service was configured with a detection threshold which represents a percentile on the distribution of the reconstruction error in the training set. This provides the flexibility to quantitatively define normality, allowing some outliers in the train set. The LAA service was able to find the two types of anomalies, as represented in Figure 9.

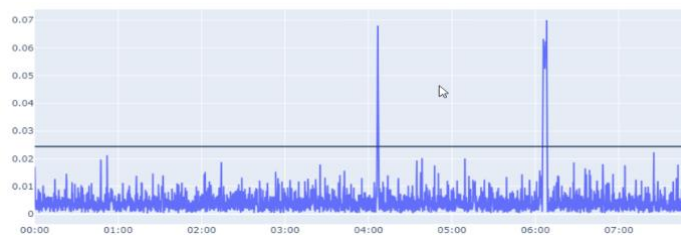


Figure 9 Two anomaly intervals detected when a reconstruction threshold is surpassed

The regions corresponding to the detected anomalies are marked with circles in the b) “Test Set” figure. One of them is somewhat apparent in the 2D map, as it reflects an unusual trajectory, bypassing the normal path to get back to the SC parking area from the container’s area. However, the other anomaly, which takes place somewhere in the middle of the container’s area, is not apparent in a 2D geo-representation. The reason for that anomaly being unnoticeable in the 2D geo-reference representation is that it has to do with the time spent by that straddle carrier stopped doing some load/unload operation at the containers area. Both the training set and the test set contain operations where the straddle carriers transit different parts of the container’s area, and stop at different points of that container’s area, spending a certain amount of time (with some predefined bounds). A great

benefit of the LSTM technology employed is that this type of neural network can learn the patterns that mean normality of different transit patterns on the container's area, both in terms of location and time, so an anomaly is detected not only because of an innovative pater on location related variables, but also because of the innovation of they over the time variable.

3.3.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
Monitoring System Design and Development: Integration of monitoring infrastructure based on docker containers and open software solutions (Grafana, Prometheus, InfluxDB, Zabbix...)	Implementation Level: Implemented Estimated Delivery Date: MS4 (M20) Licence: Proprietary – ACO (*). Deviation: No deviation	All the services (data broker, data formatting, database, anomaly analysis, performance monitoring) at the gateway side have been implemented and dockerized.
AI/ML based analysis of monitored data: ACO aims at enabling an automated AI/ML based processing of monitored data from the Monitoring System which enables detection (and eventually prediction) of anomalies	Implementation Level: Implemented Estimated Delivery Date: MS5 (M31) Licence: Proprietary – ACO (*). Deviation: No deviation	GAA and LAA demonstrators have been implemented. Jupyter notebook versions and python implementations integrated in docker containers available.
Provide Resilient and Accurate Position: ACO has experience in the design, development, and validation of positioning systems. That includes both HW and SW design. ACO can provide a customised positioning IoT that abides the costs, robustness, accuracy, connectivity and monitoring capabilities required by the port use case.	Implementation Level: Implemented Estimated Delivery Date: MS4 (M20) Licence: Proprietary – ACO (*). Deviation: No deviation	Base Station and PloT electronics are completely integrated. Ready for mounting on industrial casing required for pilot and for former in-house in-field tests.

Table 7 Capabilities Implementation Status of the Position Monitoring for Industrial Environment Solution

(*) It refers to specific AI/ML analysis methods applied on the data retrieved from the monitoring solution on the Cloud-Edge-IoT architecture developed by ACO and considered key for protecting competitive advantages achieved thanks to AIDOaRt. It excludes any open-source and/or third-party source used, and any open-source extension that needs to be published. Eventually, ACO might release as open-source fixes and extensions which do not compromise ACO competitive advantages granted by AIDOaRt.

3.3.3 Useful Resources

A hierarchical set of README files in markdown format document the configuration and setup of the gateway services has been developed. They are not publicly available, and are in principle aimed just for ACORDE usage to set up this system.

The documentation of the PIoT and BS for the positioning solution is ongoing and will be completed once the PIoT and BE development is completed (TRL4). This documentation is intended for the setup of the final user.

3.4 Solution - DTSynth (AIT)

3.4.1 Overview

Solution: DTSynth		Partner: AIT
Current TRL: 4	License: Proprietary	Contacts: benjamin.rainer@ait.ac.at
Online Resource: n/a		Collaborators: AVL, TUG
<p>Description: DTSynth consists of a collection of methods that focus on the generation of test scenarios for CPSs and digital twins. Test cases are selected based on a measure of criticality. The input is a scenario description with appropriate parameters and possible parameter spaces, the definition of a “criticality” measure and number of test cases to be generated. The output is a set of test scenarios fulfilling the criticality measure.</p> <p>DTSynth uses standard AI/ML techniques and tools to learn a mapping from a latent space to the space of critical test cases for the System Under Test according to the given criticality measure. The generated test cases are evaluated according to the following three criteria:</p> <ol style="list-style-type: none"> 1. Validity: the test cases must fall within the predefined search space. 2. Criticality: the test cases must fulfil the predefined criticality measure. 3. Diversity: the test cases must be as different as possible from each other, ideally uniformly distributed over the space of critical test cases. 		
<p>Improvement: We started initially at TRL 2 with a concept in mind and a set use case. The version from Y2 has been significantly improved as we transitioned from analytical methods to generative AI methods, drastically increasing the number of parameters being considered in the individual scenarios.</p>		
<p>Intended Users: DTSynth can be used in scenario-based testing where identifying relevant test cases is challenging because they represent just a small fraction of the whole search space.</p>		
<p>Satisfied Requirement:</p> <ul style="list-style-type: none"> ● AVL_TCV_R01 ● AVL_TCV_R02 ● AVL_TCV_R03 		
<p>AI-Augmented tool set components implemented:</p> <ul style="list-style-type: none"> ● Analytical test scenario generator ● Generative AI test scenario generator 		<p>Task: T4.2, T4.3</p>
<p>Use within AIDOaRt Use Cases</p>	<p>Already planned use: AVL_TCV_UCS1, AVL_TCV_UCS2</p>	
	<p>Potential foreseen links: AVL_TCV_UCS1, AVL_TCV_UCS2</p>	

Use within AIDOaRt challenges:

- 4th Hackathon: Parameter space reduction
- 5th Hackathon: Parameter space reduction

Table 8 Synopsis table of the DTsynth Solution

The DTsynth solution focuses on the synthesis of a Digital Twin and recommendation of test scenario parameters. It has two core capabilities: 1) Digital Twin Learning: Learning a digital twin with one or more learning approaches. Depending on the data available, the learning is passive (from existing data) or active (from interaction with artefacts). In order to increase understandability and trust, explainable AI approaches may be combined with automata learning methods [from TUG]. 2) Digital Twin Learning-Data Derivation: Based on the digital twin stimuli to the system are derived to generate more detailed data. With this data, the digital twin may be further improved. Model-based testing techniques may be applied for that sake.

3.4.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
Digital Twin Learning: Derivation of a Digital Twin using passive and/or active learning, with a focus on understandable and trustable twins (probably by means of explainable AI and combination with automata learning approaches from TUG)	Implementation Level: Implemented Estimated Delivery Date: MS6 (M28) License: Proprietary Deviation: none.	This has been achieved by a collaboration with TUG. However, this feature has already been implemented and tested in a lab environment (research prototype).
Digital Twin Learning-Data Derivation: Derivation of stimuli to the original system to get more detailed data to improve an already existing, preliminary Digital Twin. Possibly applying model-based testing techniques.	Implementation Level: Implemented Estimated Delivery Date: MS6 (M28) License: Proprietary Deviation: none.	We implemented two approaches: One that uses analytical means to narrow down generated test cases to “interesting ones” (defined by a measure) for Digital Twins and the other approach is developed using an generative AI approach.

Table 9 Capabilities Implementation Status of the DTsynth Solution

3.5 Solution - Mimir(AND)

3.5.1 Overview

Solution: Mimir		Partner: AND
Current TRL: 4	License: Proprietary	Contacts: paul@anders.com
Online Resource: NA	Collaborators: ABO	
Description: Mimir is an AI assistant for project management. It uses two step AI/ML analysis to estimate impact of the ticket to the project and offer improvements to the ticket content. Aim is to		

<p>analyse Scrum project’s performance and provide advisory feedback for example based on user stories content. Mimir provides suggestions for better user stories, better estimations or give feedback about missing details in the issues based on the ML model. It also suggests changes to scrum project based on sprint retrospectives, velocity etc. Mimir can be used in all AIDOaRt projects using the Scrum project framework. Models can be trained to each project separately to provide more context specific help for project managers and developers. Model can give a refined task description or implementation suggestions.</p>	
<p>Improvement: Mimir is a new product developed during the AIDOaRt project. Development is constantly ongoing and features are added and tested.</p>	
<p>Intended Users: Project Managers and Product Owners in software development utilising Agile methods.</p>	
<p>Satisfied Requirement:</p> <ul style="list-style-type: none"> ● GR_COD_01 ● GR_COD_02 	
<p>AI-Augmented tool set components implemented:</p> <p>Automation</p>	<p>Task: T4.1, T4.3</p>
<p>Use within AIDOaRt Use Cases</p>	<p>Already planned use: NA</p>
	<p>Potential foreseen links:</p> <p>Any use case in and out of AIDOaRt that uses Scrum project framework for project management</p>
<p>Use within AIDOaRt challenges:</p> <ul style="list-style-type: none"> ● #2: Initial planning and proposal of new product (Mimir) ● #3: Architecture planning ● #4: Review of work done and feature lock ● #5: Planning for a tool to help evaluating model performance 	

Table 10 Synopsis table of the Mimir Solution

The Mimir ML model is an advanced machine learning system designed to enhance the efficiency and effectiveness of software development. It achieves this by leveraging data sourced from Version Control systems and Project Management tools. This model stands out in its ability to analyse a given task by comparing it to similar tasks, focusing on their performance in developer commits. It has been observed that tasks which lack sufficient content, have missing information, or possess confusing descriptions tend to result in more commits, an increased number of pull requests (PRs), and generally longer code. To address the diverse needs of different projects, Mimir ML doesn't rely on a universal model. Instead, it employs a variety of context-specific models that are separately trained, ensuring a tailored approach to various development scenarios.

Mimir architecture

- **Data Exporter**
 - Pulls data from Version Control and Project Management tools.
 - Creates databases for ML training
- **Mimir Plugin**
 - Shows changes suggested via backend in PM tool.
- **Mimir Backend**
 - Provides REST API for the plugin
 - Gets analysis from ML model
 - Maps values to common cases that will act as a prefix for the LLM
 - Gets suggested changes from LLM
 - Provides change/correction suggestion for the plugin.

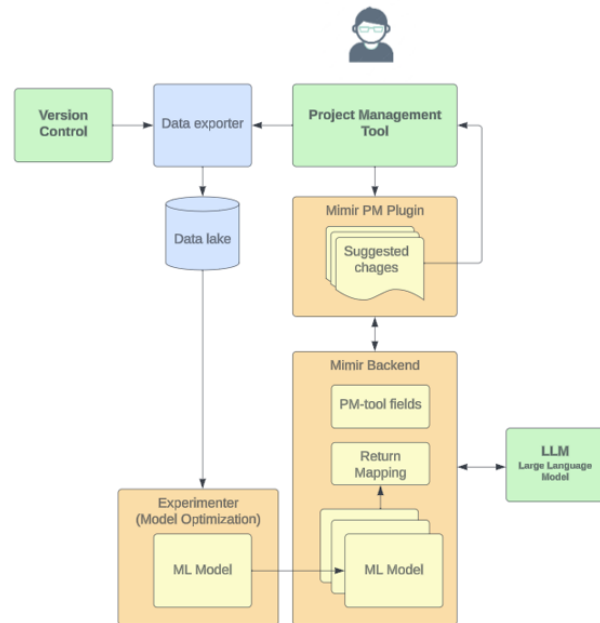


Figure 10 Mimir Solution Architecture

3.5.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
<p>Mimir: Mimir is an AI assistant for project management tools. It uses two step AI/ML analysis to estimate impact of the ticket to the project and offer improvements to the ticket content. It has been specifically trained to work in software development projects.</p> <p>Mimir provides suggestions for better user stories, better estimations or give feedback about missing details in the issues based on the ML model. It also suggests changes to scrum projects based on sprint retrospectives, velocity etc.</p> <p>AI-model is trained from project management tickets and version control history.</p>	<p>Implementation Level: Partially implemented</p> <p>Estimated Delivery Date: Product is not likely to be ready for public release during the project.</p> <p>License: Proprietary</p> <p>Deviation: N/A</p>	<p>Product is in MVP phase and new features are being evaluated for public release.</p>

Table 11 Capabilities Implementation Status of the Mimir Solution

3.6 Solution - Active DoE (AVL)

3.6.1 Overview

Solution: Active DoE		Partner: AVL
Current TRL: 5	Licence: proprietary AVL	Contacts: nico.didcock@avl.com gerald.stieglbauer@avl.com
Online Resource: https://www.avl.com/en/testing-solutions/all-testing-products-and-software/connected-development-software-tools/avl-cameo-5		Collaborators: ALSTOM
<p>Description: Two solution approaches were investigated</p> <ol style="list-style-type: none"> AVL CAMEO Active DoE: A Simulink/FMU model is prepared by the user, the model parameters and targets (e.g. difference model vs data) are exposed to CAMEO via an interface. An intelligent testing approach determines the optimal model parameters. Symbolic Regression Models: A physical formula is entered by the user - either for the temperature or its derivative as a function of the parameters. An optimization algorithm determines the unknown formula coefficients. <p><u>Solution architecture:</u> This use case requires a solution implementation that supports remote parameter identification. Since real driving cycles shall be used for model identification and model identification typically requires computationally intensive algorithms, an approach is required that splits the workload between a data-generating on-board unit, as well as a computationally high performing remote unit, e.g. a server HPC or a cloud environment. For this reason, the solution was implemented as a service, which can be used by the on-board data-collecting unit as a client. The <i>inputs to this service</i> component are:</p> <ul style="list-style-type: none"> - The model structure including parameter specifications and constraints - Cycle data collected from the UUT <p>The <i>outputs of the service</i> are either:</p> <ul style="list-style-type: none"> - Identified model parameters trained on the cycle data 		
<p>Improvement:</p> <ul style="list-style-type: none"> - Symbolic Regression Models: <ul style="list-style-type: none"> - Initial TRL 4 - Current TRL 5-6 - Advancements: For the given UC the enhancement via the use of symbolic regression models advances related to learning speed and accuracy have been concluded. 		
<p>Intended Users: The solutions could be used in the case environment of requirements engineering within Alstom (BT), to reduce the test hours for the motor thermal test. The intended users of the solution are application engineers responsible for the calibration of electric motors. Since understanding the thermal behaviour of the component w.r.t. the inputs (current) and the states (motor speed, etc.) is crucial for optimal control, improving the model accuracy can result in safer and more efficient operation.</p>		

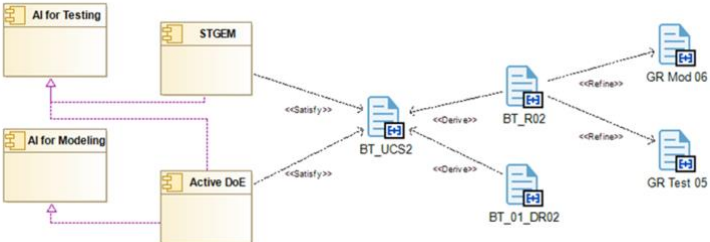
Satisfied Requirement: <ul style="list-style-type: none"> ● Generic Requirements <ul style="list-style-type: none"> ○ GR Mod 06 - Use AI-based methods for easy configuration ● Use Case Requirement <ul style="list-style-type: none"> ○ BT_R02 - ML aided control model parameterization during propulsion system testing (-> refines GR Mod 06) 	
AI-Augmented tool set components implemented: <ul style="list-style-type: none"> ● AI for Modeling ● AI for Testing 	Task: T4.2: AIDOaRt solutions for Engagement & Analysis [ITI] (M7-M32) T4.3: AIDOaRt solutions for Automation [UNIVAQ] (M7-M32) T4.4: AIDOaRt Case studies support and feedback analysis [ABO] (M7-M36)
 <pre> graph TD AI_Testing[AI for Testing] AI_Modeling[AI for Modeling] STGEM[STGEM] Active_DoE[Active DoE] BT_UCS2[BT_UCS2] BT_R02[BT_R02] BT_01_DR02[BT_01_DR02] GR_Mod_06[GR Mod 06] GR_Test_05[GR Test 05] AI_Testing -.-> <<Satisfy>> BT_UCS2 AI_Modeling -.-> <<Satisfy>> BT_UCS2 STGEM -.-> <<Satisfy>> BT_UCS2 Active_DoE -.-> <<Satisfy>> BT_UCS2 BT_UCS2 -.-> <<Derive>> BT_R02 BT_UCS2 -.-> <<Derive>> BT_01_DR02 BT_R02 -.-> <<Refine>> GR_Mod_06 BT_R02 -.-> <<Refine>> GR_Test_05 </pre>	
Use within AIDOaRt Use Cases	Already planned use: ALSTOM, in the context of BT_UCS_02- “Automated Model Parameterization”
	Potential foreseen links: Future collaboration with AVL, to use the demo version of AVL CAMEO Active DoE tool in the test lab for getting the right thermal parameters.
Use within AIDOaRt challenges: The challenge is thrown during the Hackathon #2, and subsequently during the #3 and #4 focusing on finding the suitable Machine learning algorithms to tackle the challenge. The Hackathon #5 is focused on possible implementation of the solution in the test bench setup for the validation.	

Table 11 Synopsis table of the Active DoE Solution

Architecture Overview (Figure 11): The target of the use case is to identify model parameters for rotor/stator temperature models based on real cycle data. Ideally, the data is processed in real time during the execution of the cycle. This means that the models which are created from the unit under test can be validated while the cycle is still ongoing, enabling possibilities to detect errors during data collection or model identification as early as possible without the necessity to execute dedicated model verification tests. The solution architecture comprises on-board data collection, model identification as well as model verification on the data collection unit. Since model identification procedures tend to be computationally intense, the proposed solution uses a server-client setup where computationally intensive tasks are performed on a server (e.g. a cloud environment) and data collection/model verification are performed on the on-board unit, which acts as a client to the modelling server.

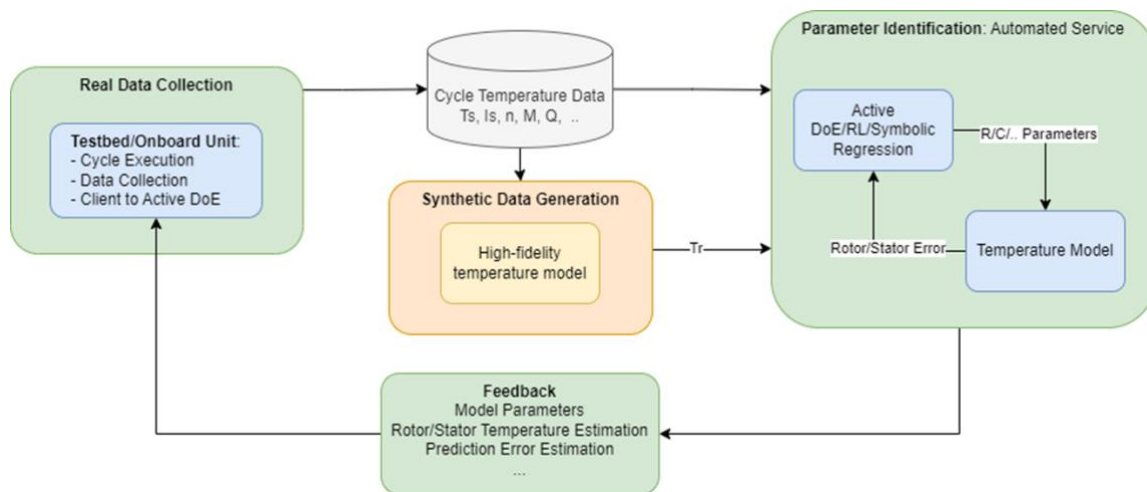


Figure 11 Solution architecture overview: a data-generating client and a modelling server

Modelling Algorithms: Two types of model identification algorithms were investigated:

1. **Black-Box Parameter Optimization Algorithms:** These methods are characterised by an optimization strategy (e.g. a reinforcement learning strategy) that loss associated with a given set of parameters. Based on the learned loss function the algorithm iteratively improves the parameters until the optimal loss is discovered. While these RL-type methods are theoretically universally applicable, the demerit is that genericity comes at the cost of computational efficiency, as learning the loss associated with multi-dimensional parameter maps in general requires a large number of simulations to be performed.
2. **Semi-Physical Modelling:** Some machine learning techniques specifically target the identification of parameter-functions within physical formulas. If the user is able to provide a physical formula as a structural base, some machine learning techniques can be applied in order to learn the unknown parts - e.g. some parameter functions - of this formula. Since the machine learning algorithm is provided with some physical knowledge, the learning of the loss function - and consequently its optimization - can be performed much faster than when using black-box techniques alone. A regression method called 'Symbolic Regression' has been implemented in AVL CAMEO 5R1 that allows the identification of parameter functions within physical formulas based on measurement data.

Status: AVL CAMEO provides a service component called 'Active DoE' which offers the possibility to optimise components based on black box modelling and optimization techniques. Previous development effort has been to develop the Symbolic Regression algorithms and verify performance based on measurement data provided by the use case provider. Current development focuses on implementing the Symbolic Regression models into Active DoE and adapting the service accordingly.

3.6.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
<p>Active DoE: Online Design of Experiments (DoE) component that iteratively updates a test set based on the most recent measurement information. Constraints can be parameterised for system inputs and system outputs. The component uses regression models based on the current measurements in order to satisfy constraints for the test candidates and improve the models in relevant areas. The test set updates are optimal in the sense that they comply with predicted output restrictions and avoid previously measured points as well. For multiple outputs the test set can be distributed on the Pareto frontier of the outputs, i.e. online model-based multi-objective optimisation.</p>	<p>Implementation Level: Partially Implemented Estimated Delivery Date: MS7 (M39) Licence: Proprietary AVL Deviation: Postponing solution integration evaluation due to missing SaaS interface (see comments)</p>	<p>As of writing this report, a prototype solution has been created, however final evaluation and integration with ALSTOM use case has to be enabled by implementing a SaaS interface. Due to the prolongation of the project, this is still aimed and achievable.</p>
<p>Symbolic Regression: Semi-physical machine learning approach combining physical knowledge with machine learning models. A physical formula is entered by the user - either for the temperature or its derivative as a function of the parameters. An optimization algorithm determines the unknown formula coefficients.</p>	<p>Implementation Level: Partially Implemented Estimated Delivery Date: MS7 (M39) Licence: Proprietary AVL</p>	<p>A symbolic regression algorithm has been integrated into AVL CAMEO 5R2 Since the use case provider intends to use the solution on board of the vehicle (in contrast to a classical testbed) the solution still has to be implemented into a SaaS environment.</p>

Figure 12 Capabilities Implementation Status of the Active DoE Solution

3.6.3 Useful Resources

Genetic Programming with Structure Templates:

- Mohammad Zhian Asadzadeh, Hans-Peter Gänsler, Manfred Mücke, Symbolic regression based hybrid semiparametric modelling of processes: An example case of a bending process, Applications in Engineering Science, Volume 6, 2021, 100049, ISSN 2666-4968, <https://doi.org/10.1016/j.apples.2021.100049>.

Symbolic Regression with Shape Constraints:

- G. Kronberger, F. O. de Franca, B. Burlacu, C. Haider, M. Kommenda; Shape-Constrained Symbolic Regression—Improving Extrapolation with Prior Knowledge. *Evol Comput* 2022; 30 (1): 75–98. doi: https://doi.org/10.1162/evco_a_00294.
- D. Piringer, S. Wagner, C. Haider, A. Fohler, S. Silber, M. Affenzeller; Improving the Flexibility of Shape-constrained Symbolic Regression with Extended Constraints, EUROCAST 2022, accepted to be published.

3.7 Solution - Keptn (DT)

3.7.1 Overview

Solution: Keptn		Partner: Dynatrace
Current TRL: 6	License: Apache License 2.0	Contacts: https://cloud-native.slack.com/archives/C017GAX90GM https://cloud-native.slack.com/archives/C046WBJN3PA anna.reale@dynatrace.com andreas.hametner@dynatrace.com
Online Resource: https://keptn.sh/ https://github.com/keptn/		Collaborators: VCE, JKU, AVL
<p>Description: Keptn integrates seamlessly with cloud-native deployment tools such as ArgoCD, Flux, and Gitlab to bring application awareness to your Kubernetes cluster. Keptn supplements the standard deployment tools with features to help you ensure that your deployments are in a healthy state.</p> <p>Deployment Observability Make any Kubernetes workload observable. If you deploy with ArgoCD, Flux, GitLab, kubectl, etc. we provide you:</p> <ul style="list-style-type: none"> • Automated App-Aware DORA metrics (OTel Metrics) • Troubleshoot failed deployments (OTel Traces) • Trace deployments from Git to cloud (traces across stages) <p>Gather metrics from anywhere Standardise access for all Observability Data for K8s. The Keptn Metrics Operator provides the following features:</p> <ul style="list-style-type: none"> • Define Keptn Metrics once for Dynatrace, DataDog, AWS, Azure, GCP, ... • Access all those metrics via Prometheus or K8s Metric API • Eliminate the need of multiple plugins for Argo Rollouts, KEDA, HPA, ... <p>Orchestrate Deployment Checks To reduce complexity of custom checks use Keptn to:</p> <p>Pre-Deploy:</p> <ul style="list-style-type: none"> • Validate external dependencies • Confirm that images are scanned <p>Post-Deploy:</p> <ul style="list-style-type: none"> • Execute tests • Notify stakeholders • Promote to next state • Automatically validate against your SLOs (Service Level Objectives) 		

<p>Improvement: Keptn was originally in beta version, while during the course of the project it has been finalised to stable. The handling of git, the database, the quality gates and the core logic of the tool have significantly evolved to be able to sustain a CSP based usage.</p>	
<p>Intended Users: SREs, Electronic Engineering Teams, Embedded SW Engineering Teams</p>	
<p>Satisfied Requirement: Generic Requirements</p> <ul style="list-style-type: none"> ● GR Mod 04 - Use semi-automatic model synthesis for design- and run-time verification ● GR Mod 08 - Use a retrospective analysis of model quality and prospective application to similar models using AI-augmented methods ● GR Mod 10 - Deploy a solution on the Cloud as a set of corresponding services and follow their execution ● GR Mod 11 - Integrate DevOps workflows and CI/configuration models <p>Use Case Requirement</p> <ul style="list-style-type: none"> ● VCE_R04 - Use of automated tools for compliance verification ● VCE_R06 - Integration of DevOps workflows and continuous integration/ configuration of models and corresponding technical solutions ● PRO_R01 - Use an Infrastructure as Code Language able to deploy the solution in different cloud providers and using different architectures / approaches (Containers & virtual machines) ● PRO_R04 - Automatically test the architecture (infrastructure tests) ● PRO_R08 - Self-healing and self-learning solution to minimise the downtime of the platform by detecting and correcting the problems automatically. Avoiding the manual recovery of the problems 	
<p>AI-Augmented tool set components implemented: Automation</p>	<p>Task: T4.1, T4.2, T4.3 and T4.4</p>
<p>Use within AIDoArt Use Cases</p>	<p>Already planned use: VCE_UCS_02 - Validation and verification of architecture models VCE_UCS_03 - AI augmented DevOps workflow and data analytics</p> <p>Potential foreseen links: PRO_UCS1 - Automatic deployment of the solution based on Infrastructure as code PRO_UCS2 - Automatic validation of the platform provisioning PRO_UCS4 - Detection and correction of service interruptions PRO_UCS5 - Platform dynamic adaptation</p>
<p>Use within AIDoArt challenges:</p> <ul style="list-style-type: none"> ● Architecture optimization + Optimization of Development Processes (3rd Hackathon) ● Continuous delivery of FMU (3rd and 4th Hackathon) ● Heterogeneous model federation and integration (4th Hackathon) ● Combined architecture modelling and analysis in the eclipse tooling framework (5th hackathon) ● Keptn for FMI (5th hackathon) 	

Table 12 Synopsis table of the Keptn Solution

To enable CPS design Keptn can orchestrate containerised Simulink simulation and enforce quality gates to determine whether the simulation passes. This flow assumes a library, or a set of developed Simulink models exists.

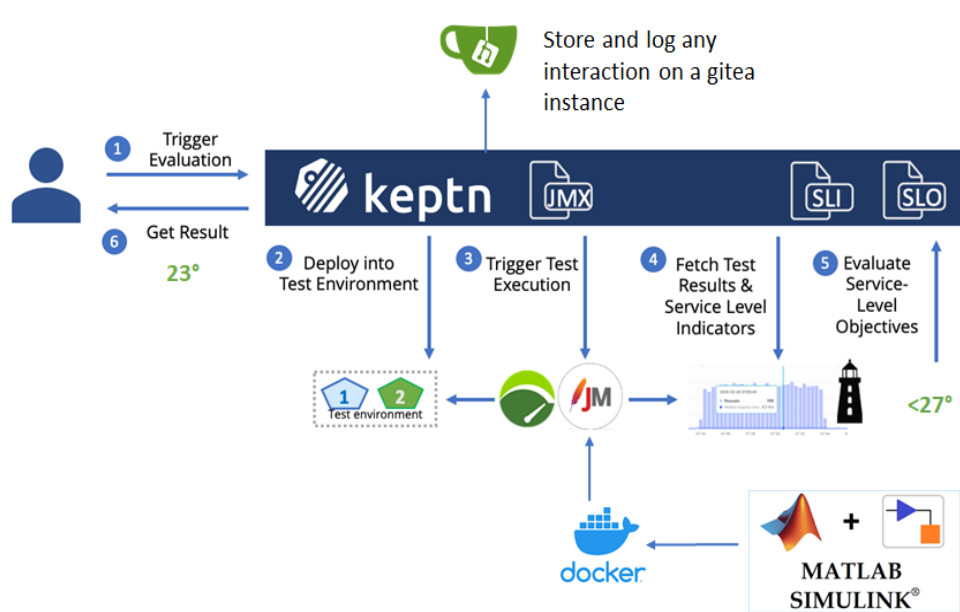


Figure 13 Keptn for CSP simulations

The first effort towards integration of CSP modelling within Keptn revolved around containerization of the MATLAB Compiler: to package the Simulink model into a Docker container means that the FMU can be run as stand-alone independently from the platform. The container includes the model, the necessary Linux operating system, supporting libraries, and a size-optimised MATLAB Runtime. This container can be easily shared and executed on target machines without MATLAB or Simulink installed and can interact with Keptn via a Job executor service.

Given designers and engineers specific criteria for the simulation, we can define quality gates. These criteria could include:

- **Performance Metrics:** Set thresholds for key performance indicators (KPIs) such as simulation time, memory usage, or accuracy.
- **Functional Requirements:** Specify conditions that the simulation must meet (e.g., stability, convergence, system temperature).

Finally, a configuration of the job executor is provided to run the simulation using the specified input parameters, in this way the simulation is easily repeatable and its quality results are available for any engineer.

To enable this scenario many improvements were needed in the Keptn tool that were specifically coming from the CSP domain:

- 1. Resource Service Enhancement:** improve our Git-Based Tracking, Keptn leverages Git repositories to manage internal project status. This ensures efficient tracking of deployment sequences and related information. The complete rework on the service was crucial due to intensive needs of CSP and HW simulations. The original project was not able to sustain larger amounts of shared files and resources.
- 2. Introduction of zero downtime in the control plane:** to reduce losses of events and sequences' information, improve accountability and repeatability of CSP simulations.

3. **Quality Gates Refinement:** we improved performance of our quality gate service to increase risk mitigation: by enforcing quality gates in CPS, Keptn ensures that only reliable and well-tested models are considered after the simulation.
4. **Event and Sequence Preservation:** Critical event and sequence information is retained, vital for maintaining system state and reliability. Some inconsistency in our sequences emerged during the project scope and were addressed.

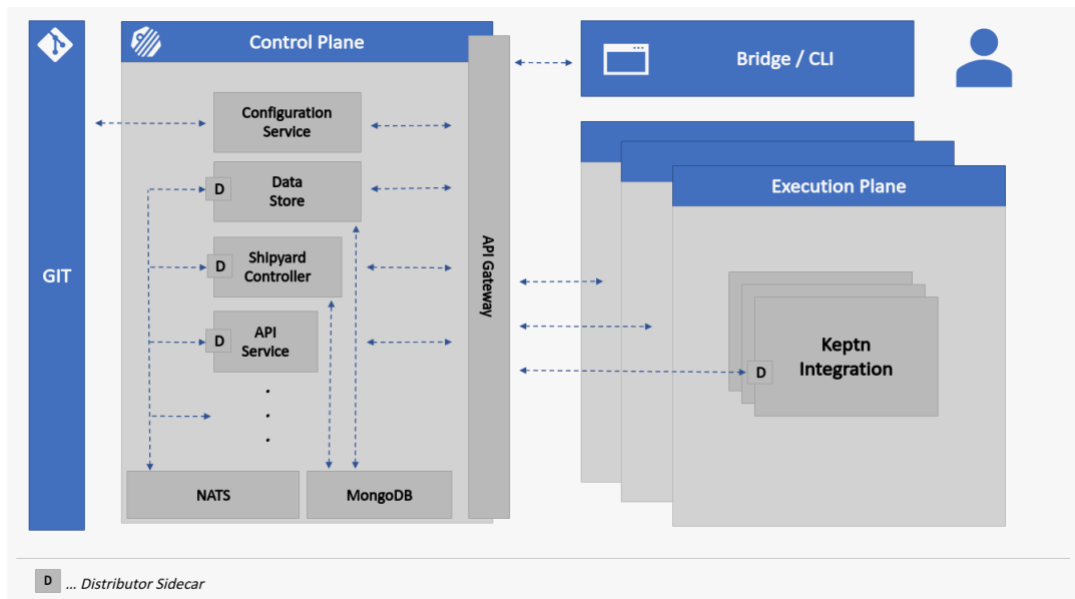


Figure 14 Keptn Current Architecture

3.7.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
Keptn CloudEvents: The event-based approach is built upon a well-defined set of Keptn events; currently in v2.0.	Implementation Level: Fully Implemented Estimated Delivery Date: MS6 (M28) License: Keptn is made available under the terms of the Apache 2.0 license. Any use of this Background by any Participant is subject to and must conform to Apache 2.0 (https://github.com/keptn/keptn/blob/master/LICENSE). Deviation: none	All Keptn events conform to the CloudEvents specification in version v1.0 . The CloudEvents specification is a vendor-neutral specification for defining the format of event data. In the course of the Keptn project, the event data is defined for the use-cases of application delivery and remediation as well as life-cycle orchestration tasks such as deployment, test, evaluation, release, problem, etc. The specification of Keptn CloudEvents is not limited to the mentioned tasks and can be easily extended by following the proposed format. The Keptn project is currently in the progress of aligning the Keptn events

		with the event specification from the Continuous Delivery Foundation (CDF) with the goal of establishing an industry-wide eventing standard for application life-cycle orchestration.
<p>Keptn Control-Plane: Keptn is built for Kubernetes and consists of a couple of Keptn core services that altogether form the Keptn control-plane.</p>	<p>Implementation Level: Fully Implemented Estimated Delivery Date: MS6 (M28) License: Keptn is made available under the terms of the Apache 2.0 license. Any use of this Background by any Participant is subject to and must conform to Apache 2.0 (https://github.com/keptn/keptn/blob/master/LICENSE). Deviation: none</p>	<p>The control-plane is responsible for orchestrating the life-cycle of an application managed by Keptn. Execution-plane service can connect to the control-plane to interact with Keptn via CloudEvents sent through NATS. The CloudEvents are currently stored in a MongoDB that serves as the datastore for all events that are sent via Keptn and allows for full traceability of life-cycle events. The architecture of the Keptn project can be found in the Keptn documentation. Keptn's architecture allows any tool to be integrated into the application life-cycle orchestration managed by Keptn. These execution plane services can run within the same cluster as the Keptn control plane or on different clusters, allowing to orchestrate multi-cluster deployments, tests, evaluations, and operational tasks such as remediation orchestration or ChatOps. please provide detailed explanation.</p>
<p>Keptn Quality Gates: A central component of Keptn are quality gate evaluations based on service-level objectives (SLOs).</p>	<p>Implementation Level: Fully Implemented Estimated Delivery Date: MS6 (M28) License: Keptn is made available under the terms of the Apache 2.0 license. Any use of this Background by any Participant is subject to and must conform to Apache 2.0 (https://github.com/keptn/keptn/blob/master/LICENSE). Deviation: none</p>	<p>Keptn builds upon SRE best practices such as service-level indicators (SLIs) and allows to declaratively define SLOs for them. These SLOs define quality criteria for the applications and act as a gatekeeper during software delivery before promoting any application or microservice from one environment (e.g., hardening) to the next environment (e.g., production).</p>

Table 13 Capabilities Implementation Status of the Keptn Solution

3.7.3 Useful Resources

- <https://keptn.sh/>
- <https://github.com/keptn/>
- <https://v1.keptn.sh/docs/1.0.x/concepts/architecture/>

Related Publications

Cederbladh, J., Reale, A., Bergsten, A., Mikelöv, R., & Cicchetti, A. (2023, October). Barriers for Adopting FMI-Based Co-Simulation in Industrial MBSE Processes. In *2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)* (pp. 510-519). IEEE.

3.8 Solution - HIB_logAnalyzer (HIB)

3.8.1 Overview

Solution: HIB_logAnalyzer (HIBLA)		Partner: HI Iberia
Current TRL: 4	License: Proprietary	Contacts: rsantos@hi-iberia.es
Online Resource: N/A		Collaborators: None
<p>Description: HIBLA is a log analysis system for the TAMUS restaurant management system. It takes as an input the aggregated daily logs of the system (cloud server and POS [point-of-sale] devices) and analyses it in search of patterns of interest such as:</p> <ul style="list-style-type: none"> • Frequency and duration of connectivity losses. • Total system load (CPU, memory, disk use). • User-induced traces (requests, interactions, transactions). 		
<p>Improvement: The system so far has been purely analytical, but we have taken the first steps to transition it into a predictive system, able to anticipate issues in the operation of TAMUS with time in advance to take preventive or mitigation measures. For this, it uses neural networks that take into account the specifications of the underlying hardware (operating system version, hardware characteristics, past errors) and the measured values in the logs.</p>		
<p>Intended Users: System Operators of the TAMUS restaurant application.</p>		
<p>Satisfied Requirement:</p> <ul style="list-style-type: none"> • HIB_R01 - The AIDOaRt AI algorithms must be able to analyse log files (text) from the restaurant application. <ul style="list-style-type: none"> ○ Implemented in HIB_UCS1 		
<p>AI-Augmented tool set components implemented:</p> <ul style="list-style-type: none"> • AI for monitoring. 		<p>Task:</p> <ul style="list-style-type: none"> • T4.1 for the ingestion and pre-formatting of the log data. • T4.4 for the development and connection with the use case needs.
<p>Already planned use: HIB_UCS1</p>		

Use within AIDOaRt Use Cases	Potential foreseen links: Potentially UC10 (Westermo) through collaboration in final months of the project and through the use of the AIDOMarket in the Marseille meeting.
Use within AIDOaRt challenges: Not yet, envisaged for final F2F meeting Marseille.	

Table 14 Synopsis table of the HIB_logAnalyzer Solution

The logs ingestion system loads the logs from the execution of the TAMUS system on a daily basis. The logs (at around 30 MB a day) contain system logs from the execution of the system. This contains traces of the requests from the devices, the responses from the cloud server and so on.

```
ESC[0mPOST /api/v1/getTamusChanges ESC[32m200 ESC[0m2.845 ms - 58ESC[0m
[13-04-2023 03:32:04][app.js:223] No Session - /api/v1/pingAlive
[13-04-2023 03:32:04][v1.js:2342] ECR 1761082761 not found in database. Company: 40, ips: 192.168.1.2
[13-04-2023 03:32:04][v1.js:2342] ECR 151cada44e488d11 not found in database. Company: 40, ips: 192.168.1.2
[13-04-2023 03:32:04][v1.js:2342] ECR 1761082761 not found in database. Company: 40, ips: 192.168.1.2
[13-04-2023 03:32:04][v1.js:2342] ECR 1761082761 not found in database. Company: 40, ips: 192.168.1.2
[13-04-2023 03:32:04][v1.js:308 - pingAlive()] Ping alive from ECR -763949524 to tamus-umami database. [OK]
ESC[0mPOST /api/v1/pingAlive ESC[32m200 ESC[0m19.107 ms - 44ESC[0m
[13-04-2023 03:32:05][app.js:223] No Session - /api/v1/pingAlive
[13-04-2023 03:32:05][v1.js:308 - pingAlive()] Ping alive from ECR 1819530136 to tamus-lateral database. [OK]
```

Figure 15 TAMUS system logs

Critically, the system collects data about the connectivity of the different mobile devices used in the deployment. These are usually smartphones used by waiters for order taking and a device that centralizes access to the cloud server in each of the restaurants (called in Spanish TPV or *Terminal de Punto de Ventas* - Point of Sales Terminal). The connectivity of these devices is critical because the overall functioning of the TAMUS system requires that all transactions are registered for all deployments on a frequent basis.

To measure the health of the connectivity, special traces tagged as PING_ALIVE are sent periodically from the TPV to the cloud server and an ACK signal is generated. By analysing statistically the time required for these signals to traverse the system, we can calculate a metric of the quality of the connection.

This provides TAMUS with good diagnostic tools of the connection details, but in AIDOaRt we planned to use AI in order for the system to predict downtime and warn the operators to take action. These operators, by means of manually inspecting logs of the system and instances of failure, know that there are tell-tale signs of impending failure in the usage of computing resources by the TPV system. Thus, in order to predict such failures, we decided to include timely metrics of the status of the system in each PING_ALIVE message as depicted below:



```

[07-02-2024 09:37:53][v1.js:335 - pingAlive()] [PING_ALIVE] ECR: -236588076 -->
tamus-lateral. STATS: [ CPU Usage: 0% Virtual Memory: 155086848B Free Swap Memory:
3619807232B Total Swap Memory: 8213852160B Free Physical Memory: 1597878272B Total
Physical Memory: 4187320320B] [OK]

[07-02-2024 09:38:55][v1.js:335 - pingAlive()] [PING_ALIVE] ECR: -236588076 -->
tamus-lateral. STATS: [ CPU Usage: 3.7537630462682214% Virtual Memory: 159363072B
Free Swap Memory: 3771744256B Total Swap Memory: 8213852160B Free Physical Memory:
1620299776B Total Physical Memory: 4187320320B] [OK]

[07-02-2024 09:39:57][v1.js:335 - pingAlive()] [PING_ALIVE] ECR: -236588076 -->
tamus-lateral. STATS: [ CPU Usage: 1.093154003657093% Virtual Memory: 168050688B
Free Swap Memory: 3767296000B Total Swap Memory: 8213852160B Free Physical Memory:
1626533888B Total Physical Memory: 4187320320B] [OK]

```

Figure 16 TAMUS PING_ALIVE and system load logs

In addition to these, each system is characterised by its hardware specifications (e.g., make and model of the CPU, quantity of memory and OS version). These are tabulated separately and generally static as the system does not significantly change over time. The HIBLA module then can use this data (dynamic trace time metrics, static hardware specifications and other metrics such as number of transactions per device, etc.) in order to feed a NN model to predict downtime. This is being tested at the time of the writing of this D4.3 and the results will be documented in future deliverables such as D4.4 and WP5 deliverables.

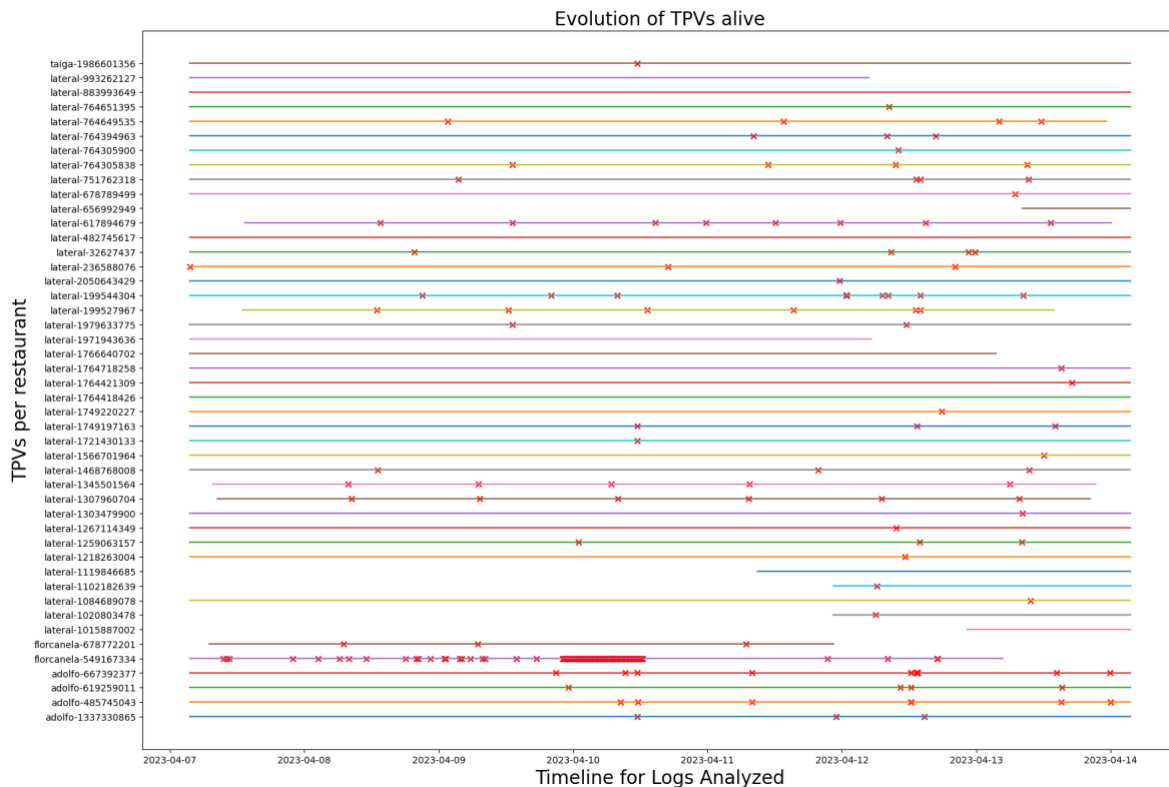


Figure 17 TAMUS PING_ALIVE time series analysis and failure detection

The intended final result of the tool is a report that summarises the logs for a given period, typically one week as mentioned. In the report, several metrics are calculated (from system uptime to number of transactions handled, total and simultaneous) and a range of alerts are delivered, based on occurrences of certain signals in the logs (e.g., system storage running low and prediction of exhaustion, unusually high CPU/memory usage, issues of connectivity between the software and the hardware devices such as the waiter smartphones). The report will also contain predictive analysis of the TAMUS deployment, pointing out potential future risks of failure.

This tool substitutes then manual analysis of these logs, which typically take up to 2 hours per week to a dedicated member of the team.

3.8.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
HIB-LA: AI service using text analytics and NLP tools to make sense of long and verbose logfiles produced by the use cases. The service uses a mix of statistical analysis and Spanish and English language NLP to provide an alert system based on the contents of the logs which are supervised semi- automatically.	Implementation Level: Partially Implemented Estimated Delivery Date: M57 (M37) License: Proprietary HIB Deviation: Slight delay in final delivery of the tool due to unplanned medical leaves in the company. System to be operational by M36.	Partially implemented: all data ingestion systems and analysis engine in place, results evaluation and operator User Interface still to be developed in the final months of AIDOaRt.

Table 15 Capabilities Implementation Status of the HIB_logAnalyzer Solution

3.8.3 Useful Resources

Online demo of a TAMUS demonstration system available at: <http://aidoart.tamus.io/>.

3.9 Solution - EMF Views (IMTA)

3.9.1 Overview

Solution: EMF Views		Partner: IMTA
Current TRL: 3-4	License: EPL-2.0 / GPL-3.0 (dual license)	Contacts: hugo.bruneliere@imt-atlantique.fr
Online Resource: https://www.atlanmod.org/emfviews		Collaborators: Hugo Bruneliere (researcher at IMTA), James Pontes Miranda (PhD student at IMTA), Massimo Tisi (associate-professor at IMTA)
Description: EMF Views is an Eclipse/EMF-based framework that allows specifying viewpoints, gathering and interrelating concepts from one or several existing metamodels, and obtaining views on corresponding sets of models (that conform to these metamodels). All kinds of metamodels and models can be possibly handled with EMF Views, including the ones used in the context of the model-based engineering of CPS. In order to do so, EMF Views relies on a generic model		

virtualization API that is applied similarly at both metamodel and model levels (to represent viewpoints and views respectively). From a ViewPoint Definition Language (VPDL) file, a virtual metamodel representing the viewpoint is computed automatically based on the contributing metamodels. This virtual metamodel/viewpoint is then used to obtain a corresponding virtual model/view from a set of corresponding contributing models. In both cases, the required extra-data (including the inter-model relationships) is stored in a separate weaving model.

Improvement: EMF Views was initially developed in past projects (including MegaM@Rt¹) and, in this context, has already been applied in realistic scenarios involving different kinds of large models (both runtime and design time ones) in order to reach TRL 3. In AIDOaRt, EMF Views has been consolidated by applying it in the context of the VCE use case as part of a model-based tool chain for systems engineering. This notably required fixing various bugs and improving the support for standard modelling languages such as SysML. EMF Views has also been extended by recently integrating an ML-based support for the automated inference of inter-model relations in model views. The work on such a recommendation system complements the other provided capabilities and practical application in order to reach TRL 4 or almost.

Intended Users: EMF Views is intended for any software and systems architect or engineer that works in the context of the model-based engineering of CPSs. It is particularly relevant in practice in all situations where several models can be created, used and evolved at different steps/iterations of a CPS model-based engineering process.

Satisfied Requirement:

- GR Mod 01 - Use AI techniques for verification of specifications and high-level models
- GR Mod 04 - Use semi-automatic model synthesis for design- and run-time verification
- GR Mod 05 - Use ML-based approach for model abstraction techniques

Satisfied Use Case Requirements:

- VCE_UCS_01 - Modeling system, software, data architectures

AI-Augmented tool set components implemented:

- Engagement & Analysis - The latest developments in EMF Views rely on the use of Machine Learning techniques (more precisely Heterogeneous Graph Neural Networks /HGNNs) for inference purposes.
- AI for Modeling - The goal of these latest developments is notably to provide an ML-based recommendation support for the appropriate modelling and building of model views.

Task:

T4.2 and T4.3 for the improvements and new developments in the context of EMF Views.
T4.4 for their application in the context of practical use case scenarios.

Use within AIDOaRt Use Cases

Already planned use:

Volvo Construction Equipment (VCE), mostly in the context of VCE_UCS_01 - "Modeling system, software, data architectures".

Potential foreseen links:

Some topics for possible collaboration are being discussed with Clearys (CSY) but there is no concrete plan yet.

¹ MegaM@art ECSEL H2020 (Grant Agreement ID 737494). <https://cordis.europa.eu/project/id/737494>



Use within AIDOaRt challenges:

VCE challenge “Modeling recommendations and process mining for system architecture descriptions”, started in practice from hackathon #3 up to now (cf. also challenge “FMU continued” in hackathon #4 and challenge “Combined architecture modelling and analysis in the eclipse tooling framework” in hackathon #5). However, some collaborative work was already performed with VCE and others during the hackathon #1 et #2.

Table 16 Synopsis table of the EMF Views Solution

In the standard EMF Views approach (cf. the lower part of the figure hereafter), the view engineer has to provide two artefacts: a *Viewpoint definition* at the metamodel level and a *View definition* at the model level. In EMF Views, these two artefacts can be partially generated from a specification by using the provided ViewPoint Definition Language (VPDL). Then, the *View Builder* takes these two artefacts as inputs and builds a virtual model that materialises the specified *Model View*.

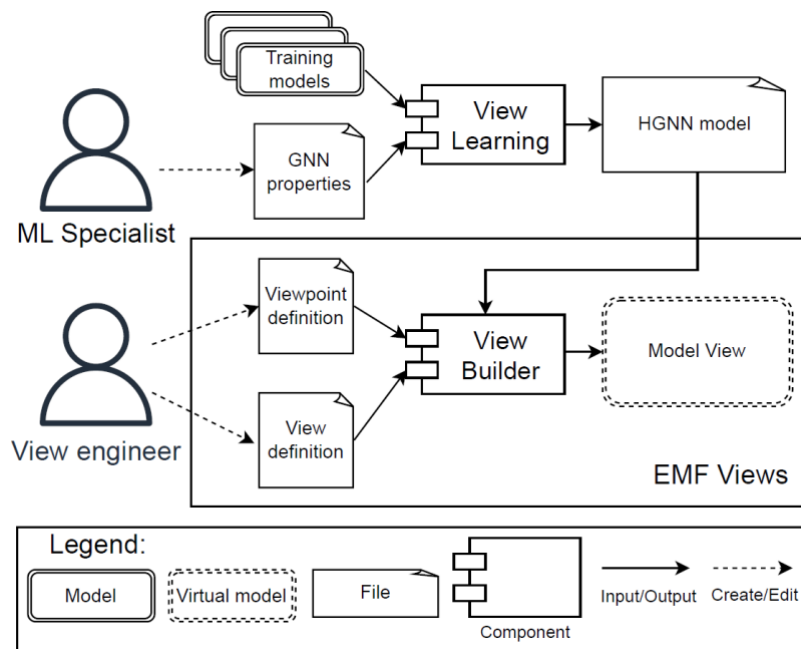


Figure 18 Overview of the ML-based extension for EMF Views developed in AIDOaRt

In the extension we developed in AIDOaRt (cf. the upper part of the Figure 18), we propose to complement EMF Views with a new *View Learning* component to support the *View Builder* base component. A set of assignments for Graph Neural Network (GNN) properties is computed from the *Viewpoint definition*. They describe the architecture of the GNN and the hyperparameters for link prediction, including training and embedding. A *ML Specialist* can possibly edit the value of these properties, e.g. to fine-tune the learning step. *Training models* are also required, including existing links used as examples for learning. Such existing models can possibly come from different sources, e.g. legacy projects. Then, the *View Learning* component takes these two artefacts as inputs and generates a trained Heterogeneous Graph Neural Network (HGNN) model. The set of inter-model links are computed by the *View Builder* component using the HGNN model, before constructing the corresponding view.

Note that the EMF Views solution already supports delegating the computation of inter-model links to external tools. Hence, our approach can reuse the standard structure of the *Viewpoint definition* and the standard *View Builder* component from EMF Views with no modifications. Moreover, the approach aims at decoupling the contributions of the *View engineer* and the *ML specialist*. Thus, the *ML specialist* can support the engineer by working on improving the accuracy and relevance of the inferred links without affecting the original *Viewpoint definition* and *View definition* made by the *View engineer*. Overall, we intend to make the use of ML as transparent as possible from the *View engineer* perspective by delegating the ML integration and execution to our approach (and possible ML-specific optimizations to the *ML specialist*).

We refer in what follows to a small example where a Users model contains personal information extracted from a social network, while a Movies model contains information from a film database. The goal is to automatically compute a model view containing users, movies, and links connecting each user with *the movies they probably watched*. We suppose we have a data set describing *other users* with information about watched movies, coming from the MovieLens data set² for instance. We aim to learn a mathematical relation between each user and their watched movies from this data set. This relation is then applied in the view to compute new inter-model links between our users and movies they probably watched. For this purpose, we use HGNNs in VPDL.

```

create view watched as

select Users.User[id, name],
       Movies.Movie.*,
       Users.User join Movies.Movie
       as watched

from 'http://paper/movies' as Movies,
     'http://paper/users' as Users

where "{s.id}<~s.movies~>{t.id, t.genres.value}"
      for watched

```

Figure 19 Extended VPDL for the recommendation example

The Figure 19 shows a snippet of our viewpoint specification in Extended VPDL for computing recommendations on this example. The *create* and *from* parts are standard VPDL. However, the *where* part contains a specific expression indicating, for each inter-model relation, the properties of the two models and the training relation to be considered for learning.

3.9.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
Model Viewpoint and View computation:	Implementation Level: Implemented	As introduced earlier, this tool capability was already partially

² <https://dl.acm.org/doi/10.1145/2827872>

<p>Be able to compute a given view in a scalable way, based on a previously specified viewpoint and a corresponding set of metamodels and models.</p>	<p>Estimated Delivery Date: MS6 (M28) License: EPL 2.0 / GPL 3.0 Deviation: No particular deviation</p>	<p>available in the past versions of EMF Views.</p> <p>However, updates have been performed (as some bugs have been detected on given view operations) in order to better support the building of viewpoints and views over particular types of models (e.g., UML/SysML ones).</p> <p>During the last phase of the project, we extended EMF Views and the associated VPD language with some AI-related support.</p>
<p>Model Viewpoint and View update: Be able to dynamically and efficiently update an already computed view.</p>	<p>Implementation Level: Partially Implemented Estimated Delivery Date: MS7 (M37) License: EPL 2.0 / GPL 3.0 Deviation: Obtaining satisfying results with the use of the ML/HGNNs took longer than initially expected due to the complexity of properly parameterizing and training such AI models. However, we have now set up a proper ML infrastructure we can now more easily reuse for further experiments and research works.</p>	<p>As introduced earlier, this tool capability is now partially available in the current version of EMF Views.</p> <p>We already obtained promising results by integrating the use of Machine Learning techniques (more specifically HGNNs) with EMF Views in order to improve the semi-automated computation and/or update of some elements of the views.</p> <p>This work will be continued as new developments are already ongoing, with more experimental data and improved results. These are planned to appear in future scientific publications.</p>

Table 17 Capabilities Implementation Status of the EMF Views Solution

3.9.3 Useful Resources

Main website: <https://www.atlanmod.org/emfviews>

Source code repository: <https://github.com/atlanmod/emfviews>

Installation instructions: <https://github.com/atlanmod/emfviews#readme>

User manual, tutorials: <https://www.atlanmod.org/emfviews/manual>

Related publications:

- Johan Cederbladh, Luca Berardinelli, Hugo Bruneliere, Antonio Cicchetti, Mohammadhadi Dehghani, Claudio Di Sipio, James Pontes Miranda, Abbas Rahimi, Riccardo Rubei, Jagadish Suryadevara. Towards Automating Model-Based Systems Engineering in Industry - An Experience Report. The 18th Annual IEEE International Systems Conference (SYSCON 2024), Apr 2024, Montreal, Canada. [hal-04448172](#)
- James Pontes Miranda, Hugo Bruneliere, Massimo Tisi, Gerson Sunyé. Towards the Integration Support for Machine Learning of Inter-Model Relations in Model Views. The 39th ACM/SIGAPP Symposium on Applied Computing (SAC '24), Apr 2024, Avila, Spain. [hal-04346770](#).
- Hugo Bruneliere, Florent Marchand de Kerchove, Gwendal Daniel, Sina Madani, Dimitris Kolovos, Jordi Cabot. Scalable Model Views over Heterogeneous Modeling Technologies and Resources. Software and Systems Modeling, 2020, 19 (4), pp.827-851. [10.1007/s10270-020-00794-6](#).
- Romina Eramo, Florent Marchand de Kerchove, Maximilien Colange, Michele Tucci, Julien Ouy, Hugo Bruneliere, Davide Di Ruscio. Model-driven Design-Runtime Interaction in Safety Critical System Development: an Experience Report. The Journal of Object Technology, Chair of Software Engineering, 2019, The 15th European Conference on Modelling Foundations and Applications, 18 (2), pp.1:1-22. [10.5381/jot.2019.18.2.a1](#).
- Hugo Bruneliere, Erik Burger, Jordi Cabot, Manuel Wimmer. A Feature-based Survey of Model View Approaches. Software and Systems Modeling, 2019, 18 (3), pp.1931-1952. [10.1007/s10270-017-0622-9](#).

3.10 Solution - INT-DET (INT)

3.10.1 Overview

Solution: INT-DET		Partner: INT
Current TRL: 4	License: GNU General Public License v3.0	Contacts: katia.diblasio@intecs.it luca.morlino@intecs.it
Online Resource: n/a	Collaborators: Tiziana Fanni (ABI), Fabio Marongiu (ABI)	
Description: INT-DET is a real-time object detection and tracking solution based on the deep learning architecture. It supports prediction capabilities and it is designed to give intelligent driving assistance. The inputs are images that are part of a video stream from cameras installed on intelligent vehicles. The output consists of the same image as the input superimposed with the detection of objects (vehicles, pedestrians, etc.).		
Improvement: From an initial TRL 2, when research to understand existing techniques and technologies for object detection were performed and a conceptual model based on the identified requirements and available resources was developed, we identified areas for improvement and optimization of a SOTA algorithm based on a deep-learning approach. Continuous iteration on the design and optimization process, among which adjusting anchor boxes, increasing data augmentation techniques and modifying the feature pyramid network, and evaluating the model's performance on blind test set led us to the final result.		
Intended Users: The ability to detect object from images or videos has wide-ranging applications across various domains, improving safety, efficiency, and user experience in numerous contexts so INT-DET could therefore be used, even without any prior knowledge of artificial intelligence, by		

modellers, industrial practitioners and software architects working on topics such as Autonomous Vehicles, Robotics, Surveillance and Security and Remote Sensing.	
Satisfied Requirement: Generic requirements: N/A Use Case requirements: <ul style="list-style-type: none"> • ABI_R04 - Use ML for video elaboration • ABI_R11 - The system shall detect relevant object 	
AI-Augmented tool set components implemented: <ul style="list-style-type: none"> • <u>Engagement & Analysis</u>: INT-DET relies on the use of AI/ML techniques to support the analysis of the environment in which the CPS moves (inference on camera image stream) 	Task: T4.2
Use within AIDOaRt Use Cases	Already planned use: Abinsula UC, mostly in the context of ABI_UCS1 - Functionality Verification, ABI_UCS2 - Test definition, ABI_UCS3 - Compliance verification
Use within AIDOaRt challenges: <ul style="list-style-type: none"> • 3rd Hackathon: Adoption of AI and ML techniques for image processing in the automotive context • 5th Hackathon: Adoption of AI and ML techniques for image processing in the automotive context [update] 	

Table 18 Synopsis table of the INT-DET Solution

In order to develop a fast and high-performance object detector, INT-DET, a tailored version of YOLOv3 [YOLOv3] was selected. This architecture falls under the single-stage object detector category in which the entire image is processed in a single pass and the class probabilities and the bounding box coordinates are learned all at once. Another key feature is that YOLOv3 divides the input image into multiple grids and makes predictions at three different scales: this is crucial in helping to recognise objects of interest (vehicles, pedestrians, etc.) even when they are far away. YOLOv3 is also optimised for real-time object detection on various hardware platforms, making it suitable for safety-critical applications such as assisting the driver.

This architecture was trained and tested on the KITTI dataset [KITTI], a widely used computer vision dataset for benchmarking and evaluating algorithms related to autonomous driving and robotics. It provides a diverse collection of sensor data collected from a moving vehicle in real-world urban traffic scenarios.

In Figure 20 and an Figure 21 overview of the performance achieved is shown. In addition to achieving high precision and recall values, it is important to note the trend of the Mean Average Precision (mAP) metric: it represents a precision and recall trade-off and is obtained by averaging the Average Precision (AP) values across all categories, calculated in turn by taking the area under the precision-recall curve, to provide an overall assessment of the model's performance.

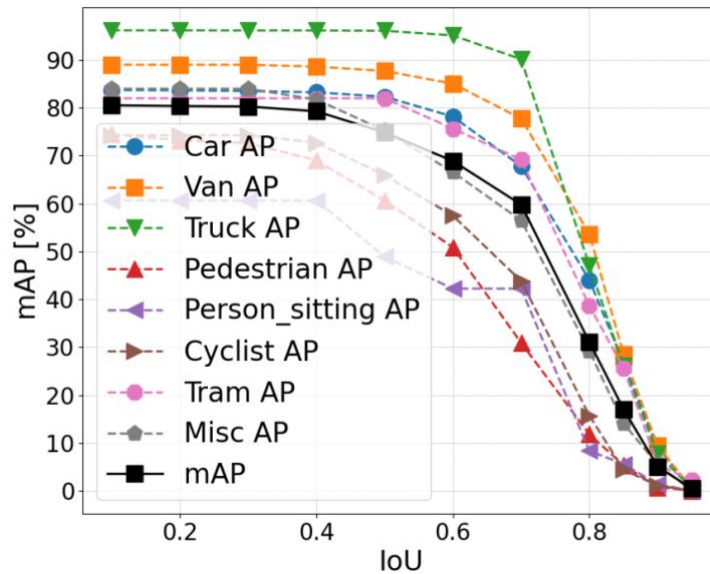


Figure 20 Model performance on the test set in terms of AP and mAP values; mAP@0.5 is 74.83%

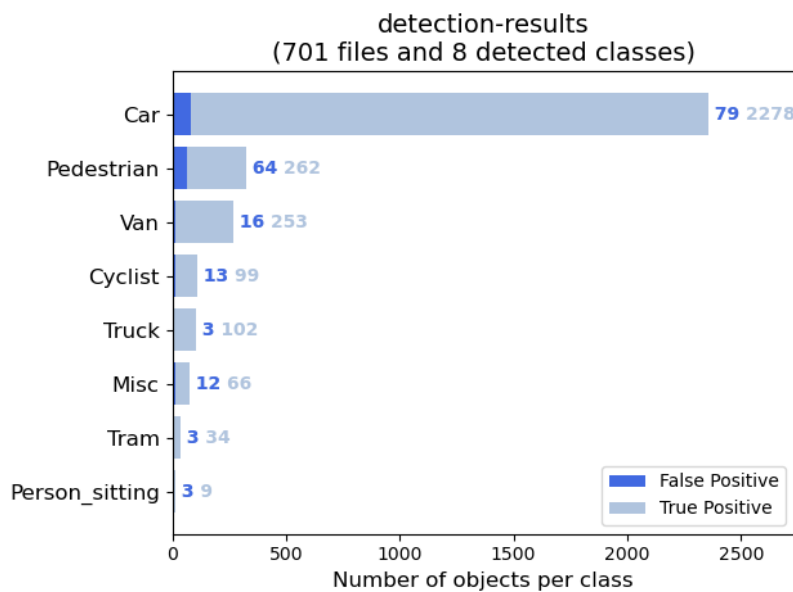


Figure 21 Model performance on the test set in terms of True Positive and False Positive; Precision is 94.14%, Recall is 80.81% and F1-score is 86.97%.

3.10.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
Real-Time Object Detection: Real-time object detection is the task of doing object detection in real-time with fast inference while maintaining a base level of accuracy. Given the initial state of a target in the first frame of a video sequence, the aim of	Implementation Level: Implemented Estimated Delivery Date: MS6 (M28) License: GNU General Public License v3.0	Deployment on CPU/GPU target platform completed. Deployment on FPGA target platform being finalised, difficulties encountered in model quantization.

<p>Visual Object Tracking is to automatically obtain the states of the object in the subsequent video frames. The challenge is to build a real-time system that is reliable and explainable for safety-critical applications.</p>	<p>Deviation: Deployment on FPGA not fully implemented.</p>	
---	--	--

Table 19 Capabilities Implementation Status of the INT-DET Solution

3.11 Solution - INT-DEPTH (INT)

3.11.1 Overview

Solution: INT-DEPTH		Partner: INT
Current TRL: 3/4	License: Proprietary	Contacts: katia.diblasio@intecs.it luca.morlino@intecs.it
Online Resource: n/a		Collaborators: n/a
<p>Description: INT-DEPTH is an AI-based system for depth perception. It is designed to give intelligent driving assistance. This component supports the deduction capabilities and it is based on the deep learning architecture. The inputs are images that are part of a video stream from two cameras installed on intelligent vehicles, forming stereo image pairs. The output consists of an estimate of the depth of objects in the scene provided through a disparity map</p>		
<p>Improvement: From an initial TRL 2, when research to understand existing techniques and technologies for depth estimation were performed and a conceptual model based on the identified requirements and available resources was developed, we identified areas for improvement and optimization of a SOTA algorithm based on a deep-learning approach. We then re-implemented this approach from scratch and performed experiments with different variations of the chosen architecture. Validation tests and a full evaluation of the model's performance remain to be completed in order to achieve a TRL 4.</p>		
<p>Intended Users: The ability to estimate depth from images or videos has wide-ranging applications across various domains, improving safety, efficiency, and user experience in numerous contexts so INT-DEPTH could therefore be used, even without any prior knowledge of artificial intelligence, by modellers, industrial practitioners and software architects working on topics such as Autonomous Vehicles, Robotics, Surveillance and Security and Remote Sensing.</p>		
<p>Satisfied Requirement: Generic requirements: N/A Use Case requirements:</p> <ul style="list-style-type: none"> ● ABI_R04 - Use ML for video elaboration ● ABI_R12 - The system shall estimate the vehicles approaching speed 		
<p>AI-Augmented tool set components implemented:</p> <ul style="list-style-type: none"> ● <u>Engagement & Analysis:</u> INT-DEPTH relies on the use of AI/ML techniques to support the analysis of the environment in which the CPS moves (inference on camera image stream) 		Task: T4.2
Use within AIDOaRt Use Cases	<p>Already planned use: Abinsula UC, mostly in the context of ABI_UCS1 - Functionality Verification, ABI_UCS2 - Test definition, ABI_UCS3 - Compliance verification</p>	

Use within AIDOaRt challenges: This tool was not used during the AIDOaRt challenges, only tested in ABI_UCS1, ABI_UCS2, ABI_UCS3.

Table 20 Synopsis table of the INT-DEPTH Solution

In order to develop a fast and high-performance stereo depth estimation tool, INT-DEPTH, leveraging MobileStereoNet architecture [MSNet] was developed. The MobileStereoNet model was initially trained solely on the KITTI dataset [KITTI], which provides real-world stereo image pairs with corresponding ground truth depth maps obtained from LiDaR scans. Standard training procedures including data pre-processing, network initialization, and optimization are employed. Experimentation with variations of MobileStereoNet architecture was conducted to enhance feature representation and depth estimation accuracy. Following what was done in the development of INT-DET, various augmentation techniques such as random cropping, rotation, and scaling are applied to augment the KITTI dataset, enriching the diversity of training samples. In addition, customised loss functions incorporating geometric constraints and disparity smoothness penalties are devised to encourage more accurate depth predictions. Greater performance advancement was achieved conducting a two-stage training: the model was firstly pre-trained on the SceneFlow dataset [SceneFlow], a large-scale synthetic dataset containing diverse stereo image pairs, to initialise the model with informative features; secondly, the pre-trained model was fine-tuned on the KITTI dataset to adapt its representations to real-world scenes and specific challenges encountered in the dataset. Training and evaluation protocols are established using standard metrics such as mean absolute error (MAE), root mean square error (RMSE), and disparity error. Qualitative assessments including visualisations of depth maps and disparity images corroborate the efficacy of the proposed approach. A quantitative evaluation and validation of the robustness of the approach followed will soon be finalised.

3.11.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
Depth Estimation: The depth estimation estimates a dense depth map from RGB images, retrieving distance information from the camera. As the previous point, the challenge is to build a real-time system that is reliable and explainable for safety-critical applications.	Implementation Level: Partially Implemented Estimated Delivery Date: M57 (M37) License: Proprietary Deviation: Deployment on FPGA not implemented.	Development completed. A final testing and validation phase remains to be completed after which deployment on CPU/GPU platform will be performed. The possibility of deployment on FPGAs is being studied.

Table 21 Capabilities Implementation Status of the INT-DEPTH Solution

3.12 Solution - a2k-runman (ITI)

3.12.1 Overview

Solution: a2k-runman		Partner: ITI
Current TRL: 4	License: Proprietary ITI	Contacts: ssaiez@iti.es
Online Resource: N/A	Collaborators: PRO	

<p>Description: The a2k-runman solution aims to collect information regarding system performance at run-time and to apply adaptive modifications to the system configuration to optimise resource usage. In particular, the a2k/detection service has been developed to analyse system and sensor data to detect and/or predict critical situations and to automatically correct them. Subsequently, this information is relayed to the a2k/tuning service, responsible for adjusting the system configuration to manage these abnormal conditions safely.</p>	
<p>Improvement: At the start of the project, the TRL was 2. Throughout the project, we implemented a demonstration proof of concept and validated the components with simulated data, thus advancing the TRL to level 4. In this project, regression models have been adapted using sliding windows to perform prediction tasks and thus anticipate problems of overload or lack of resources. In the last year the entire methodology has been optimised and validated using new simulated datasets. Additionally, one of the models has been deployed, and queries have been conducted to verify that the entire process is functioning correctly.</p>	
<p>Intended Users: Engineers who are designing and validating complex CPS. Within the PRO use case scenario 5 the algorithms have been developed in consultation with their system architects and end users.</p>	
<p>Satisfied Requirement:</p> <ul style="list-style-type: none"> ● GR MOD 01 – Using AI techniques for verification of specifications and high-level models. ● GR MOD 03 – Use ML-based models for early detection of system failures and recovery in self-healing. ● GR MOD 04 – Use semi-automatic model synthesis for design and run-time verification. ● GR MOD 06 – Use AI methods for easy configuration. ● PRO_R06 - Self healing and self-learning solution to minimise the downtime of the platform by detecting and correcting problems, automatically avoiding the manual recovery of the problems. ● PRO_R08 - Detect and predict high/low resources demand based on AI. 	
<p>AI-Augmented tool set components implemented:</p> <ul style="list-style-type: none"> ● Ingestion & Handling ● AI for Monitoring ● Automation 	<p>Task: T4.2 Engagement & Analysis</p>
<p>Use within AIDOaRt Use Cases</p>	<p>Already planned use: PRO_UCS5 — Resizing of resources based on current workload</p> <p>Potential foreseen links: Industrial partners who are using distributed CPS under highly dynamic operating conditions where processing demands and data input streams are continuously varying throughout the period of operation.</p>
<p>Use within AIDOaRt challenges: The solution was initially proposed in the first hackathon. Throughout the remaining hackathons we developed and refined the solution with feedback from the PRO use case engineers.</p>	

Table 22 Synopsis table of the a2k-runman Solution

As previously mentioned, A2k-runman aims to gather data on system performance and leverage the a2k/detection service for analysing system and sensor data to identify or anticipate abnormal situations. This data is then transmitted to the a2k/tuning service, which is tasked with modifying the system configuration to address these conditions and ensure safe operation effectively. To this end, progress has been made by working with a real-time simulator and AI regression and prediction methods to evaluate different CPU frequency control algorithms so that the component can adjust the processor frequency considering the workload demands, thereby controlling the power consumption

of the processor hardware. The overarching goal is to minimise power consumption, for example, for processors connected to IoT devices at the network's edge.

The real-time simulator simulates a schedule, assigning tasks to a processor to execute each one until completion. This simulator is modelled as a set of tasks $\tau = \{\tau_1, \tau_2, \dots, \tau_n\}$. The parameters of τ_i include its period (the time between successive requests) T_i , deadline D_i , and worst-case execution time (WCET) C_i . A task set is called feasible if the deadline of each task is always satisfied. When these tasks start to run, they also have time-dependent dynamic parameters, such as the remaining time, which refers to the processing time required to complete the task from the current moment until its termination.

During the simulation, the simulator collects data from these tasks and stores it in a dataset. The CPU optima frequency is a function of these parameters, and its calculation requires a heavy computation. This cost can be reduced considering the type of application being executed. Through training focused on this application, we can get an artificial intelligence module to perform these operations at a reduced cost.

An algorithm that calculates the optimal frequency using simulation outputs is developed to train the AI module. This frequency is used to label the data generated by the simulator. The AI methods then use these labels to train the models. After the training, an application (Optimization module) that uses this module can be deployed to obtain the optimal frequency and set the CPU to this value through an interface. Figure 22 describes these training and deployment phases.

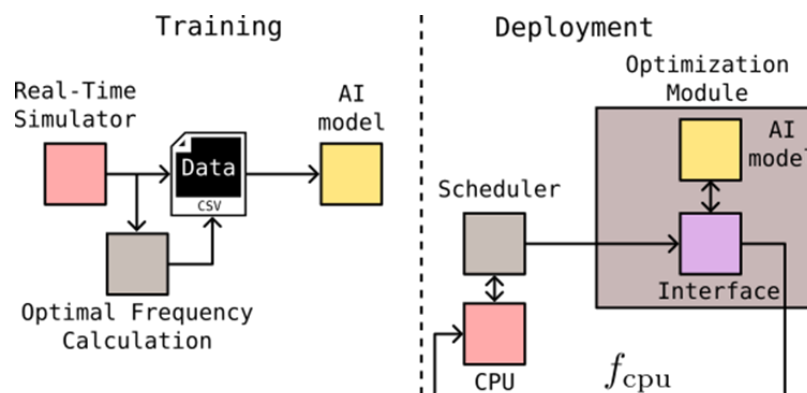


Figure 22 Data flow diagram between the real-time simulator and AI methods

Regarding the AI methods, the work has been structured in two phases, and two simulated datasets have been used; a simple one that simulates the execution of 3 tasks on a processor has been used to debug and implement improvements in the process, and one more complex with ten tasks that has been used to validate all the algorithms.

In the first phase, seven regression models were evaluated and compared: Linear Regression, Ridge Regression, Lasso Regression, ElasticNet Regression, Support Vector Regression, Random Forest Regression, and XGBoost Regression. The main idea was to find the most appropriate model to estimate the optimal frequency of the processor on which the different tasks are being executed so that all tasks can be completed without missing deadlines and optimising energy consumption. This

phase aims to contribute to the dynamic optimization of processor performance while ensuring energy efficiency and meeting real-time requirements. During the validation process of this phase, the best results were obtained using non-linear models, especially with the Random Forest and XGBoost models (Figure 23).

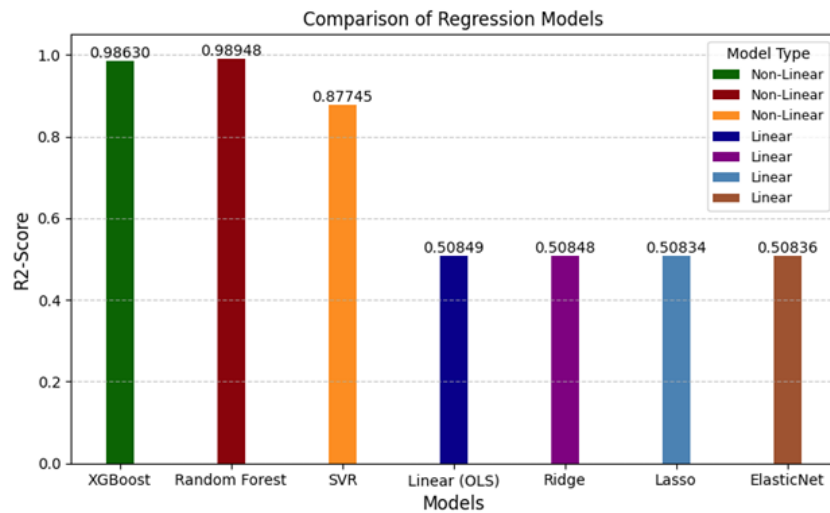


Figure 23 R² - score values obtained for the different regression models (with hyperparameter tuning)

In the second phase, prediction models were developed based on the regression models with the best performance identified in the initial phase. These models were constructed using sliding windows, moving forward through time. Each observation comprised information from multiple time instants (window size: $t_{-n}, \dots, t_{-2}, t_{-1}$) to predict the subsequent instant (t_{+1}). The process was then adapted to predict multiple time instants ($t_{+1}, t_{+2}, t_{+3}, \dots$) using two approaches: a direct multi-output method, where all future time instants are predicted simultaneously as independent single-output regression problems, and a chained multi-output method, where the prediction task is divided into a sequence of dependent single-output regression problems. During the validation process of this phase, the entire process was repeated for both the XGBoost and the Random Forest models, obtaining good results in both cases, although slightly better for the XGBoost model (Figure 24).

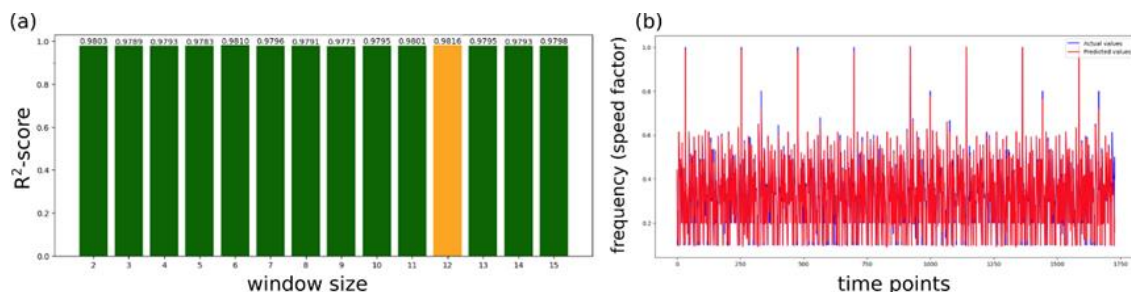


Figure 24 Performance of the XGBoost model in predicting a time instant

In (a), the R²-score values for different window sizes are shown. The window size with which the best result has been obtained is highlighted in orange. (b) Prediction results for the test set using the optimal window size.

The necessary pipeline has also been developed to prepare the different models (Regression, Simple Prediction, Multiple Predictions) for their deployment. In addition, one of the models has been deployed, and queries have been carried out to verify that the process is working correctly. Specifically, the first phase model has been deployed, used to estimate the optimal frequency of a processor in which different tasks are being executed. A demo has been prepared to show the results obtained when querying this deployed model (Useful Resources). All the information corresponding to the deployed model is collected in a Model Card and shown in the demo.

3.12.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
a2k/tuning: Automatic criticality level / operational mode change management based on real time data collection and analysis using adaptive learning algorithms	Implementation Level: Implemented Estimated Delivery Date: MS7 (M37) License: Proprietary ITI Deviation: The component has not been possible to validate with real data due to confidentiality issues. All analyses have been performed with simulated data.	This component will automatically adjust the clock frequency of a processing component considering the predicted workload demands. We are using a real-time simulator (for training) and a prediction method based on AI to do this.

Table 23 Capabilities Implementation Status of the a2k-runman Solution

3.12.3 Useful Resources

A demo has been prepared to show the results obtained when we query one of the developed and deployed models. This model has been trained to predict the optimal frequency of a processor in which different tasks are being executed so that all tasks can be executed without losing deadlines and optimising the power consumed. **Demo Link:** [here](#).

3.13 Solution - a2k-depman (ITI)

3.13.1 Overview

Solution: a2k-depman		Partner: ITI
Current TRL: 4	License: Propriety ITI	Contacts: ssaez@iti.es
Online Resource: N/A	Collaborators: PRO	
Description: a2k-depman is a dynamic deployment manager component. The objective of this component is to assign software services to processing nodes in a distributed, heterogeneous CPSs. The deployment of services to nodes is performed to satisfy multiple competing performance		

<p>objectives and constraints. The inputs to the component are specifications of non-functional objectives to be optimised while the outputs are the assignment of services to nodes. We use a multi-objective genetic algorithm to perform the optimisation in conjunction with neural networks to model the processing loads of the individual nodes according to incoming service requests.</p>	
<p>Improvement: TRL at the start of the project was 2. During the project we have implemented a demonstration proof of concept and validated the component with simulated data and thus we have advanced the TRL to level 4. Additionally, we have developed a time series neural network for modelling the CPU usage of processing nodes in terms of the incoming rate of service requests and the expected resources needed to service these requests.</p>	
<p>Intended Users: Engineers who are designing and validating complex CPS. Within the PRO use case the solution was developed in collaboration with their system architects and end users.</p>	
<p>Satisfied Requirement:</p> <ul style="list-style-type: none"> ● GR Mod 01- Use AI techniques for verification of specifications and high-level models. ● GR Mod 04 – Use semi-automatic model synthesis for design and run-time verification. ● GR Mod 05 – Use ML-based approach for model extraction techniques. ● GR Mod 06 – Use AI-based methods for easy configuration. ● GR Mod 07 – Use a model-based approach for ML representation. ● PRO_R06 - Self healing and self-learning solution to minimise the downtime of the platform by detecting and correcting problems, automatically avoiding the manual recovery of the problems. ● PRO_R08 - Detect and predict high/low resources demand based on AI. 	
<p>AI-Augmented tool set components implemented:</p> <ul style="list-style-type: none"> ● AI for Modelling 	<p>Task: T4.2 Engagement & Analysis</p>
<p>Use within AIDOaRt Use Cases</p>	<p>Already planned use: PRO_UCS5 — Resizing of resources based on current workload</p> <p>Potential foreseen links: N/A</p>
<p>Use within AIDOaRt challenges: Smart Port Monitoring Platform (Hackathons 3, 4 & 5).</p>	

Table 24 Synopsis table of the a2k-depman Solution

The goal of the a2k-depman(a2k/optimiser) solution is to automate the deployment of software micro-services to processing nodes under dynamically varying rates of service requests. We consider the case of a complex, heterogeneous CPS where several processing nodes are interconnected by communications links. The types of these processing nodes can range from so-called *edge-devices* which are directly connected to physical sensors and actuators, through to powerful, *cloud-based* data warehouses and high-performance compute nodes. Figure 25 shows a schematic of a typical CPS under consideration. In this diagram the dark blue nodes are physical devices, the green nodes are gateways, the cyan ones are intermediate processing nodes, and finally the red node is a cloud-based high-performance compute node.

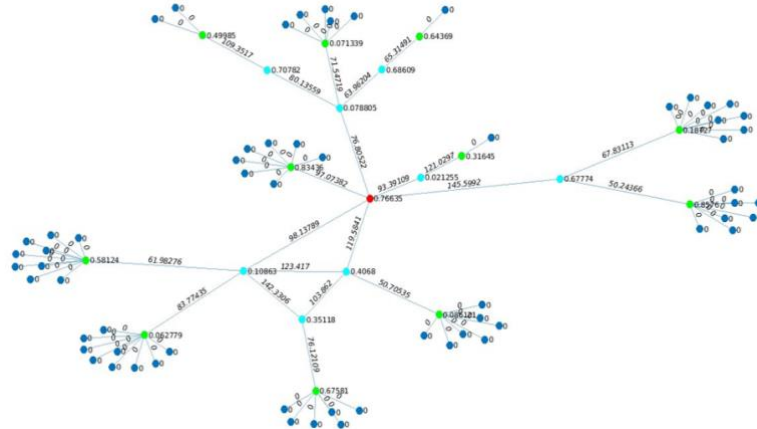


Figure 25 Typical CPS network Schematic of the Simulated Smart Port Infrastructure

Within such a system there arises the problem of where are the best processing nodes to locate the software services. On the one hand a service might be located close to an edge device so that communications latency is minimised. But this might be at the expense of time required to complete the software task because edge devices are likely not as powerful as the CPUs or GPUs used in cloud-based processors.

To this end we initially developed a high-level model for such a CPS. This is composed of two parts. Firstly, there is a model for the hardware platform of the system. This is effectively a graph data structure where the nodes are the processing elements, and the edges are the communications links. We then assign properties to each node and edge of the graph. For the processing nodes the properties might include items such as, for example, CPU & memory resources, clock speed, power consumption, and financial cost of use. In the case of the communications links the properties would include latency and data throughput rates.

The second modelling component defines how the software micro-services are allocated to the processing nodes and how the software services communicate with each other (i.e. the call graph). This software (or application) model also contains properties such as time required to execute the service, deadline for completing the service, resources required by the service and so forth.

Given these system properties we then developed a set of algorithms for computing non-functional performance metrics for the global system architecture. These metrics include latency between issuing a service request and receiving a response, overall power consumption, and financial cost. We also developed an improved version of the compositional performance analysis algorithm to compute these metrics.

We identified a set of constraints defined by the application requirements. Such constraints might include a specification that certain software services are only allowed to execute on certain processing nodes. In addition, there are usually timing constraints such as certain services must complete their tasks within a certain time frame (deadline).

Thus, effectively we have a multi-objective, constrained optimisation problem. To solve this, we developed a genetic algorithm optimiser to allocate the software services to processing nodes. Since the problem has multiple competing objectives (e.g. financial cost vs response time) the output of the optimiser is a *set* of potential solutions. These can be depicted by the so-called Pareto front. An

example of such a pareto set is shown in the Figure 26 below (Objective 1 is Latency and Objective 2 is financial cost). It is subsequently left up to the system designer (or an AI system) to decide on the best solution according to requirements.

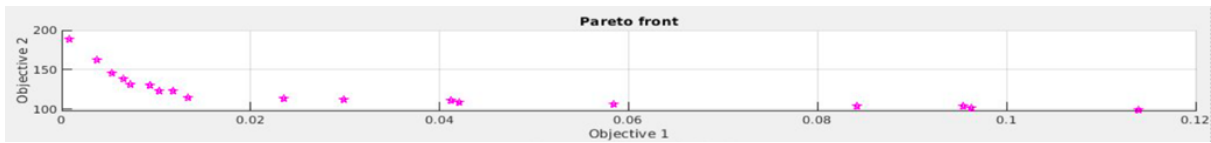


Figure 26 Pareto Front After Optimisation

Finally, we also implemented a LSTM neural network module to model the relationships between the input traffic to a processing node and the resulting CPU resource usage. We trained this network using simulated input traffic. The resulting model can then be employed to predict the overall system performance metrics under various traffic conditions and thus subsequently used in the service allocation. In this way services can be dynamically allocated to different processing nodes under varying incoming service request loads.

3.13.2 Capabilities Implementation Status

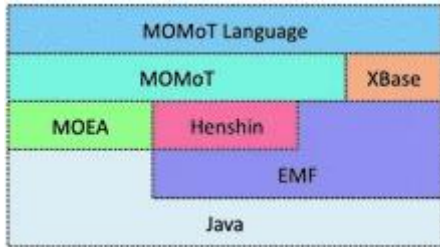
Capability Name & Description	Implementation Status	Comments / Release notes
a2k/optimiser: Deployment assistance in heterogeneous distributed real-time systems using multi-objective optimisation and machine learning	Implementation Level: Implemented Estimated Delivery Date: MS7 (M37) License: Proprietary ITI Deviation: The component has not been possible to validate with real data due to confidentiality issues. Efforts are being made to resolve these problems before the end of the project.	The component offers services for analytic computation of non-functional performance metrics such as latency and throughput. The component offers a multi-objective optimisation algorithm with non-linear constraints for allocating micro-services to processing nodes. The component includes neural network based learned models for prediction of CPU resource usage.

Table 25 Capabilities Implementation Status of the a2k-depman Solution

3.14 Solution - MOMOT (JKU)

3.14.1 Overview

Solution: MOMoT		Partner: JKU]
Current TRL: 4	License: EPL v2	Contacts: luca.berardinelli@jku.at manuel.wimmer@jku.at

Online Resource: https://martin-fleck.github.io/momot/ https://github.com/RL4MT/RL4MT	Collaborators: Johan Cederbladh (VCE)
Description: MOMoT: A framework that combines model-driven engineering and search-based software engineering to optimise model transformation. Features: It supports different model transformation languages, optimization algorithms, and quality objectives. It also provides a graphical interface for visualising and analysing the solutions. Application: It can be applied to various model transformation problems, such as modularization, refactoring, synthesis, and orchestration of transformation rules. Model-driven optimization is a technique that uses model transformations to optimise models according to some objectives. The first version of MOMoT relies on meta-heuristic search algorithms, such as genetic algorithms, to guide the transformation process. However, reinforcement learning (RL) methods have recently shown promising results for solving optimization problems in other domains. MOMoT has been extended by a novel approach that applies RL for in-place model transformations. It uses both value-based and policy-based RL techniques to learn how to apply transformation rules that improve the model quality.	
Improvement: MOMoT has been identified as a tool to extend the AIDOaRt solution proposed for the VCE Use Case. In particular, the goal is to provide decision support to select valid simulation configurations based on Functional Mockup Units (FMU). In order to use the tool, an ad-hoc Ecore-based (auxiliary) FMU metamodel and multi-objective optimization functions have been developed to optimise the configuration parameters for simulating the battery of in a battery-powered vehicle.	
Intended Users: Model-driven experts (tool usage and maintenance), system/software engineers using model-driven artefacts in their engineering process.	
Satisfied Requirement: <ul style="list-style-type: none"> ● GR Mod 04 - Use semi-automatic model synthesis for design- and run-time verification ● VCE_R01 - Use automated reasoning and ML techniques for verification of specifications and high-level models ● VCE_R02- AI/ML method for auto-adjusting model parameters w.r.t. similarity of execution traces of a Digital Twin with a CPS 	
Tasks T4.2 and T4.3 for the improvements and new developments in the context of MOMoT. T4.4 for their application in the context of practical use case scenarios (see Hackathons).	
AI-Augmented tool set components implemented: <ul style="list-style-type: none"> ● AI for Modeling <div style="text-align: center;">  <p>The diagram illustrates the MOMoT architecture as a layered stack. At the base is Java. Above it is EMF. The next layer contains MOEA and Henshin. Above that is MOMoT, which includes XBase. The top layer is MOMoT Language.</p> </div> <p><i>Figure 27 MOMoT architecture</i></p>	

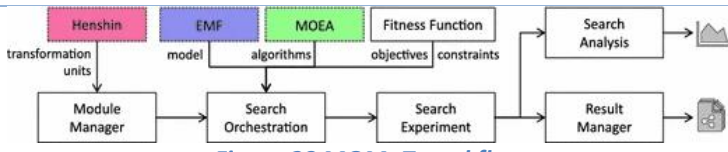
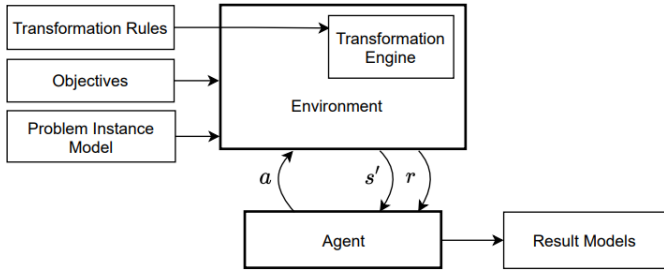
 <p>Figure 28 MOMoT workflow</p>	
 <p>Figure 29 Integration of RL in MOMoT</p>	
Use within AIDOaRt Use Cases	<p>Already planned use: VCE_UCS_01 - Model based architecture (framework) development VCE_UCS_02 - Validation and verification of architecture models. Optimization of Functional Mock-up Unit (FMU)'s parameterization</p> <p>Potential foreseen links: AVL_ODP_R01 Implement approaches of learning data-driven models based on project data (especially Key Performance Index (KPI) and parameter value evolution) provided by AVL. The models are utilised to compute the maturity of a specific KPI result in a specific project at a specific point in time in a standardised and objective way. AVL_ODP_R02 Implement approaches of data-driven models based on project data (especially Key Performance Index (KPI) and parameter value evolution) provided by AVL. The models are utilized to forecast the KPI (and parameter) value evolution in the project. AVL_ODP_R03 Implement approaches of data-driven models based on project data (especially Key Performance Index (KPI) and parameter value evolution) provided by AVL. The models are utilized to assess the information gain for specific experiments in order to identify experiments that provide little information gain and can thus be e.g. skipped in future projects.</p>
<p>Use within AIDOaRt challenges: 2nd Hackathon - Optimization of Development Process (merged with MOMoT for FMI Challenge) 3rd Hackathon - Architecture Optimization (merged with MOMoT for FMI Challenge) 5th Hackathon - MOMoT for FMI Challenge</p>	

Table 26 Synopsis table of the MOMoT Solution

The MOMoT Framework integrates model-driven engineering and search-based software engineering to optimise model transformations. It leverages Ecore meta-models to represent problem domains and employs graph transformation rules to optimise problem instances. Additionally, the framework extends its capabilities by incorporating two distinct Reinforcement Learning (RL) approaches: value-based and policy-based.

Key Features



- **Problem Representation:** MOMoT uses Ecore meta-models to define problem domains.
- **Graph Transformation Rules:** Problem instances (models conforming to the given Ecore meta-models) are optimised through executing (in-place) graph transformation rules by a Transformation Engine.

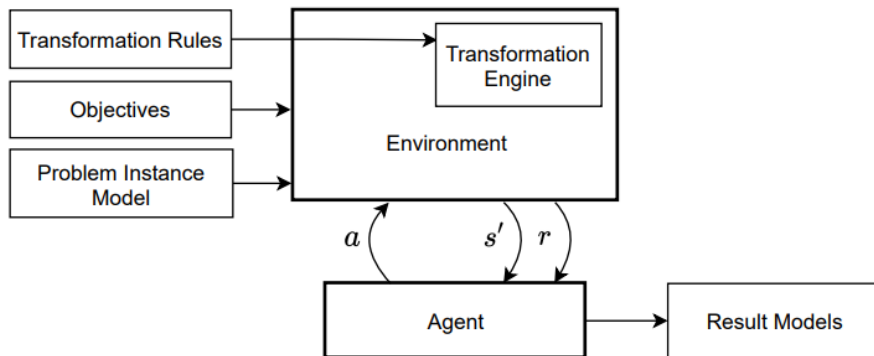


Figure 30 Integration of RL into MT frameworks

The most recent version of MOMoT applies reinforcement learning (RL) for in-place model transformations. For this purpose, the model transformation process is formalised as a Markov decision process (MDP). A MDP is a mathematical framework for modelling sequential decision making under uncertainty, where an Agent interacts with an Environment and receives rewards (r) for its actions (a).

To map the model transformation process to a MDP, the following components need to be defined:

- **States:** The states are the **models** in their current composition, which capture all the relevant information from the transformation history.
- **Actions:** The actions are the **rule applications** that change the model composition, which depend on the current state and the available rules.
- **Transitions:** The transitions are the probabilities of moving from one state to another after applying an action, which are deterministic in the case of model transformations.
- **Rewards:** The rewards are the immediate or expected **values of the objectives evaluated on the resulting models**, which reflect the quality of the transformation steps.

Once represented as a model transformation process as an MDP, two algorithms are executed by the Agent: Value-based methods are model-free RL approaches that try to derive a value function from the obtained rewards of actions within certain states. Policy-based methods are also model-free RL approaches that aim to directly learn a policy through function estimation without consulting a value function. The MOMoT extension with RL is described in the reported publications.

3.14.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
-------------------------------	-----------------------	--------------------------

AI-augmented MOMOT: Reinforcement learning for in-place model transformation. See Eisenberg, Martin, et al. "Towards reinforcement learning for in-place model transformations." 2021 ACM/IEEE 24th International Conference on Model Driven Engineering Languages and Systems (MODELS). IEEE, 2021.	Implementation Level: Implemented Estimated Delivery Date: MS7 (M37) License: Eclipse Public License - V2.0 Deviation: No deviations	The MOMoT tool augmented with RL is available at https://github.com/RL4MT/RL4MT
--	---	--

Table 27 Capabilities Implementation Status of the MOMOT Solution

3.14.3 Useful Resources

Publications:

- Fleck, Martin, Javier Troya, and Manuel Wimmer. "Search-based model transformations with MOMoT." *Theory and Practice of Model Transformations: 9th International Conference, ICMT 2016, Held as Part of STAF 2016, Vienna, Austria, July 4-5, 2016, Proceedings 9*. Springer International Publishing, 2016.
- Bill, Robert, et al. "A local and global tour on MOMoT." *Software & Systems Modeling* 18 (2019): 1017-1046.
- Eisenberg, Martin, et al. "Towards reinforcement learning for in-place model transformations." *2021 ACM/IEEE 24th International Conference on Model Driven Engineering Languages and Systems (MODELS)*. IEEE, 2021.

Repositories:

- <https://martin-fleck.github.io/momot/>
- <https://github.com/RL4MT/RL4MT>

3.15 Solution - GAN-Based Instance Model Generator (JKU)

3.15.1 Overview

Solution: GAN Model Generator		Partner: JKU
Current TRL: 3/4	License: MIT License	Contacts: Abbas.rahimi@jku.at luca.berardinelli@jku.at
Online Resource: https://github.com/AbbasRahimi/netgan/tree/Ecore_model_generator		Collaborators: n.a.
Description: The GAN-based Instance Model Generator is developed to address data augmentation needs in the MDE context. Due to the increasing demand for using MDE in different complicated systems, the need for developing new model-based solutions is also growing. The key input data required to design and develop such solutions are instance models. GAN-based instance model generator aims to mitigate this data shortage by producing a proper data set. It takes a metamodel and one big instance model of it as inputs, and then it can generate more Ecore-based models. The output data will be structurally realistic as generative adversarial networks are used as the generator engine.		

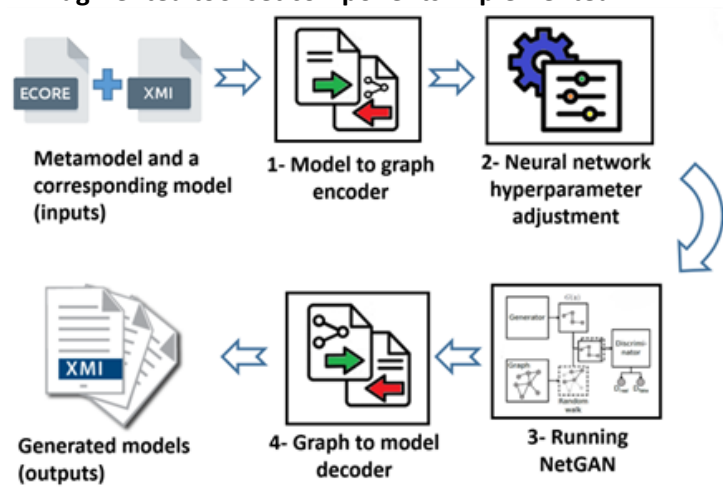
<p>Improvement: The GAN-based instance generator was initially developed to alleviate the deficiency of data within the context of Model-Driven Engineering. It supports specific categories of metamodels. In the context of AIDOaRt, endeavours were made to tailor its functionality towards generating XES instance models, whereby we encountered and rectified several bugs. Additionally, we focused on enhancing the encoding/decoding components to support a broader spectrum of metamodels; however, it remains a work in progress.</p>	
<p>Intended Users: The potential intended users are all MDE experts/engineers and researchers who need to produce more instance models to trigger/proceed with their work.</p>	
<p>Satisfied Requirement:</p> <ul style="list-style-type: none"> GR Mod 04 - Use semi-automatic model synthesis for design- and run-time verification 	
<p>AI-Augmented tool set components implemented:</p>  <p>Task: T4.2 and T4.3 for the improvements and new developments in the context of GAN.</p>	
<p>Use within AIDOaRt Use Cases</p>	<p>Already planned use: n/a</p>
	<p>Potential foreseen links: VCE_UCS_01 - Model based architecture (framework) development VCE_UCS_02 - Validation and verification of architecture models.</p>
<p>Use within AIDOaRt challenges: The GAN generator has not been used in AIDOaRt challenges after an internal investigation. The main limitation is that the NN we reused (NetGAN) is specifically designed to operate on homogeneous graphs. Consequently, our proposed framework is limited to generating models that adhere to a metamodel that respects certain specifications. In particular, the input metamodel can have only one type of relationship between each pair of EClasses. By changing the generator NN or enhancing the encoder and decoder, we hope to overcome this constraint.</p>	

Table 28 Synopsis table of the GAN-Based Instance Model Generator Solution

Figure 31 shows an overview of the GAN-based framework that can generate new instance models in 4 steps. In the first step, the model-to-graph encoder extracts the structural feature of the input metamodel and creates the graph representing the input instance model. Then, the user should adjust the neural network hyperparameters according to the input graph features like the number of relationships and the graph depth. Next, it is possible to run the training phase and then ask the trained neural network to generate new realistic graphs. Finally, the graph-to-model decoder converts the generated graphs to output models.

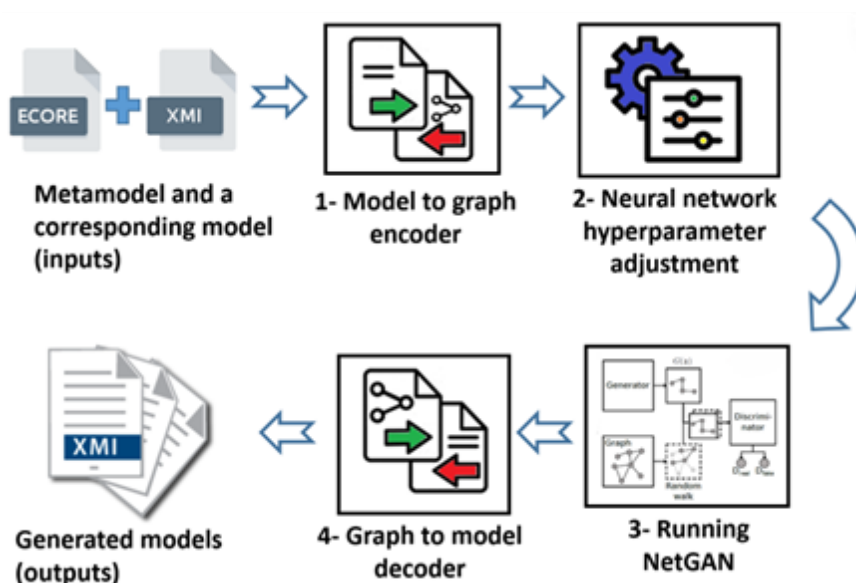


Figure 31 Overview of the GAN-Based instance model generator

3.15.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
Instance_Generator	Implementation Level: Implemented Estimated Delivery Date: MS7 (M37) License: MIT License	<p>The tool has been implemented and validated, as demonstrated in the related publication. However, its application to the AIDOaRt use cases is not feasible due to the inherent structural complexity of the utilized metamodels (for instance, the XES metamodel), which obstructs the encoding process</p> <p>The future plan is to revise the data encoding to support different types of metamodels and make the tool more easy-to-use by adding a graphical configuration setting.</p>

Table 29 Capabilities Implementation Status of the GAN-Based Instance Model Generator Solution

3.15.3 Useful Resources

Source code repository: https://github.com/AbbasRahimi/netgan/tree/Ecore_model_generator

Related publications:

- A. Rahimi, M. Tisi, S. K. Rahimi, and L. Berardinelli, "Towards Generating Structurally Realistic Models by Generative Adversarial Networks," in MDE Intelligence: 5th Workshop on Artificial

Intelligence and Model-driven Engineering, Västerås, Sweden: ACM/IEEE 26th International Conference on Model-Driven Engineering Languages and Systems, Oct. 2023. doi: 10.1109/MODELS-C59198.2023.00098.

3.16 Solution - Requirements Ambiguity Checker (MDU)

3.16.1 Overview

Solution: Ambiguity Checker		Partner: Alstom
Current TRL: 3/4	License: Open Source (GPL)	Contacts: lodiana.begiri@mdu.se calkin.suero.montero@mdu.se
Online Resource: https://github.com/GitRE2024/AmbiguityChecker/blob/main/Prototype_mockup.pdf (mockup of the tool)		Collaborators: n.a.
<p>Description: The solution's key features are described below.</p> <ul style="list-style-type: none"> • NLP Processing: NLP techniques are used to work with textual requirements. • AI and ML: binary classifiers are trained and used to categorise the requirements as ambiguous or not, by providing a level of confidence and an explanation for the classification. • User-Friendly Interface: a web prototype is currently underway, where users can upload the requirements to check for presence of ambiguity. <p>How It Works:</p> <ul style="list-style-type: none"> • Input: Using the web interface, the users can upload requirements written in English as either excel file or manually type the requirement to check for the presence of ambiguity. • Processing: the input entered is processed by employing NLP techniques. • Classification: ML/AI models are trained and used to classify the requirements as ambiguous or not. The models have learned from the data in the training phase and are capable of classification on new requirements (user input). • Output: For each requirement, the user is provided with a classification class (Ambiguous or Not Ambiguous), a confidence level of such classification, and an explanation that includes terms that add to the ambiguity. The user has the possibility to provide their own validation directly after the classification and feedback/ comments. Based on the user validation and the model outcomes metrics (True Positive, True Negatives, False Positives, False Negatives) can be calculated to assess the model's performance. <p>Improvement: The ambiguity checker model has been constructed, and initial tests to ensure its functionality have been performed using railway data and an expert; hence, the TRL level is 4.</p> <p>Intended Users: The intended users may include requirement engineers with expertise in requirements (such as rail-specific requirements) and their analysis. The overall goal is to assist in automatically analysing and categorising requirements based on their ambiguity, once the models have learned from the requirements and comments provided by such experts in the field.</p>		

<p>Satisfied Requirement:</p> <ul style="list-style-type: none"> GR RE 03 - Analyse the textual requirements defined in formal language from the railway domain provided by Alstom/BT and make suggestions or prescriptions to the Requirements Engineer and System Engineer. GR RE 04 - Verify the consistency of the requirements. 	
<p>AI-Augmented tool set components implemented:</p> <ul style="list-style-type: none"> <u>AI for Requirements Engineering</u> - AI/ML-based capabilities to support the requirements engineering phase of the system development, namely AI/NLP methods for requirements similarity check and requirements allocation. 	<p>Tasks:</p> <ul style="list-style-type: none"> T4.2 for the extraction and analysis of requirements from unstructured specification text documents. T4.4 for the validation on partners' data and use case scenarios.
<p>Use within AIDOaRt Use Cases</p>	<p>Already planned use: BT_UCS1 - Requirements analysis and recommendation of most suitable action per requirement.</p> <p>Potential foreseen links: None</p>
<p>Use within AIDOaRt challenges:</p> <p>3rd Hackathon in Valencia: Requirements Analysis, Processing and Response</p> <p>4th Hackathon in Vasteras: Achieving Explainable AI to Support Decision-making during Requirement Engineering Analysis</p>	

Table 30 Synopsis table of the Requirements Ambiguity Checker Solution

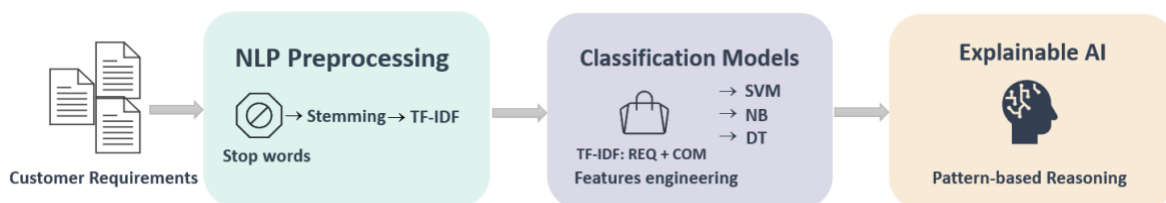


Figure 32 Ambiguous Requirement Classification Pipeline with Explanations

A general overview of the approach is shown in Figure 32. The natural language requirements undergo a pipeline that involves NLP for text processing tasks like removing stop words, stemming, and extracting lexical features. Porter stemmer is applied for stemming or removing different variations of the words and returning each word in its base form, and TF-IDF is used to convert the natural language requirements into numerical vectors that can then be utilised by ML techniques for binary classification. TF-IDF gives each word a weight depending on its frequency in the document and rarity across all documents. Classification models such as SVM, Decision Trees and Naive Bayes are trained on the extracted lexical feature from both requirements and the comments given by the experts and are then used to identify ambiguous requirements. Following the classification, explanations are given to experts. The explanations highlight key terms influencing the categorization decisions made by the model and indicating potential ambiguity. The concept of explainability adopted is that of local

interpretability where for each requirement a list of terms that might contribute to the classification of ambiguous requirements is given.

3.16.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
checkRequirementsAmbiguity: Requirements Ambiguity Checker is a solution based on natural language processing (NLP) and machine learning that can automatically detect ambiguous requirements. It takes a requirements document as an input and the output is a requirements ambiguity report. The report provides a predicted class for each requirement as ambiguous or not, a classification confidence level, extracted terms utilised by the model to make the classification decision, as well as it allows the user to validate the output and provide comments.	Implementation Level: Partially Implemented Estimated Delivery Date: MS8 (M42) License: to be specified Deviation: n.a.	The next phases would allow the model to be improved based on feedback from the initial validation, which included an expert; this would imply the adoption of more sophisticated models such as LLM. Furthermore, the future iteration of the model will incorporate the feedback received from the expert from the current version.

Table 31 Capabilities Implementation Status of the Requirements Ambiguity Checker Solution

3.16.3 Useful Resources

- An article has been submitted to the 32nd IEEE International Requirements Engineering 2024 conference and is now under review.
- A detailed description may be found at <https://github.com/GitRE2024/AmbiguityChecker>.

3.17 Solution - TATAT (PRO)

3.17.1 Overview

Solution: TATAT (Traceability and Test Automation Tools)		Partner: Prodevelop
Current TRL: 5	License: Open Source (GPL)	Contacts: evillanueva@prodevelop.es itorres@prodevelop.es
Online Resource: Not available		Collaborators: None
Description: TATAT (Traceability and Test Automation Tools) was initially designed to support error traceability, through the execution of integration and acceptance tests, for web applications deployed on premise. Its main objective is to automate the validation of infrastructure deployment by automatically running tests, so that the infrastructure is automatically evaluated after each deployment. The main advantage is that the tool allows unifying the automation of the execution of tests deployed with different testing tools, as it offers the possibility of launching multiple tests automatically.		

<p>Improvement: The main functionalities that have been incorporated during the AIDOaRt project are:</p> <p>Cloud deployment testing.</p> <ul style="list-style-type: none"> • Testing of node deployments and computing services. • Testing of Docker deployments. • Testing of service operation (access via different protocols). • Testing security and access to different services. <p>It also includes the development of different plugins, which allow interaction with both the testing tool and the test case management tool (TestLink), as well as with the project management tool (JIRA). After Y2, tests were carried out to check that all functionalities were responsive and to correct those that were not. With all this progress, we went from an initial TRL of 3-4 to a final TRL of level 5-6.</p>	
<p>Intended Users: The tool can be used by any user who wishes to carry out testing, since, as mentioned above, it integrates plugins compatible with a multitude of tools that facilitate the launching of different tests to validate tools. Although for the moment it is only available for internal use at Prodevelop.</p>	
<p>Satisfied Requirement:</p> <ul style="list-style-type: none"> • PRO_R04 - Automatically test the architecture (infrastructure tests) 	
<p>AI-Augmented tool set components implemented:</p> <ul style="list-style-type: none"> • Automation (AIOps Engineering) 	<p>Task:</p> <p>T4.3 for the developments and the improvements of the tool.</p> <p>T4.4 for its application in the PRO_UCS02.</p>
<p>Use within AIDOaRt Use Cases</p>	<p>Already planned use: PRO_UCS02</p> <p>Potential foreseen links: Adaptable to other use cases and partners, without specifying any particular one.</p>
<p>Use within AIDOaRt challenges: This tool was not used during the AIDOaRt challenges, only tested in the PRO_UCS02 and not be tested in other use cases or challenges.</p>	

Table 32 Synopsis table of the TATAT Solution

As mentioned, the new features since deliverable D.4.2, where the development work on the tool had already been completed, testing has been carried out to ensure that all the new functionalities added throughout the project work correctly.

3.17.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
<p>TestAutomationlink:</p> <p>Service that unifies the automation of the execution of tests implemented with different testing tools.</p>	<p>Implementation Level:</p> <p>Implemented</p> <p>Estimated Delivery Date: MS4 (M20).</p> <p>License: Open Source (GPL).</p> <p>Deviation: No deviations reported.</p>	<p>The TATAT tool covers the defined needs, as it unifies the automation of tests that each user can define according to the requirements they want to validate.</p>

Table 33 Capabilities Implementation Status of the TATAT Solution

3.18 Solution - CRT (QEN)

3.18.1 Overview

Solution: CRT		Partner: QEN
Current TRL: 6	License: commercial	Contacts: knupponen@copado.com nsharma@copado.com
Online Resource: https://www.copado.com/product-overview/copado-robotic-testing		Collaborators: Westermo
<p>Description: Copado Robotic Testing (CRT) is an advanced, AI-implemented test automation solution designed to streamline and enhance the testing process across web, mobile, and native desktop platforms. Key features of CRT include Guided Authoring, Flow Editor, and Recorder, which collectively simplify test creation by eliminating the need for scripting or coding. Its Self-Healing Tests address the issue of flaky tests by automatically updating test scripts. Performance Change Detection alerts users to any drops in functionality performance, while Live Testing provides real-time test run results.</p>		
<p>Improvement: Majority of the development has been focused on QEditor and CRTQI, but CRT itself has introduced support for Apple Silicon (Mx processors), enabling on-premise robots and local authoring to function seamlessly on the latest Apple hardware, thus broadening the platform's compatibility and performance efficiency. The release of the nCino Automated Testing Accelerator marks another significant advancement, offering specialised tools for testing within the nCino environment, thereby streamlining and optimising test processes for financial services software.</p>		
<p>Intended Users: Developers, testers, and quality assurance professionals seeking to streamline and enhance their testing processes through automated, AI-powered solutions.</p>		
<p>Satisfied Requirement:</p> <ul style="list-style-type: none"> ● GR Mod 01 - Use AI techniques for verification of specifications and high-level models ● GR Mod 02 – Use formal models, automated reasoning and/or ML techniques for test generation. ● GR RE 04 - Verify the consistency of the requirements ● W_UCS_3 - AI-powered root cause analysis w.r.t functional issues 		
AI-Augmented tool set components implemented:		Task: T4.2
<ul style="list-style-type: none"> ● AI for Testing 		
Use within AIDOaRt Use Cases	Already planned use: WMO_CS10	
<p>Use within AIDOaRt challenges:</p> <p>#2 L'Aquila: Exploring test results data (WMO)</p> <p>#3 Valencia: Test case dependencies (WMO)</p> <p>#4 Västerås: Performance Data Exploration (WMO)</p> <p>#5 Linz: Anomaly detection and Seasonal quality metrics (WMO)</p>		

Table 34 Synopsis table of the CRT Solution

Copado Robotic Testing (CRT) is a cloud-based software testing solution that leverages artificial intelligence and robotic process automation to streamline the test automation process across web, mobile, and desktop applications. It is designed to simplify the creation, execution, and management of test cases, making it an accessible tool for both technical and non-technical users alike. CRT's

approach to testing is centred around enhancing efficiency, reducing manual effort, and improving software quality.

Performance change detection is another key feature, alerting users to any significant changes in application performance early in the development cycle. Live testing provides real-time feedback on test execution, allowing for immediate issue identification and resolution. CRT supports parallel execution of multiple tests, reducing overall testing time, and utilizes AI for smart predictions to automatically generate test steps. Comprehensive test reporting, including video recordings and screenshots, offers clear evidence of test outcomes.

The benefits of using Copado Robotic Testing include increased efficiency in the testing process, enabling faster release cycles, and scalability through its cloud-based infrastructure. The low-code approach and guided authoring make it easy for team members of varying technical levels to contribute to testing efforts. Early defect detection and comprehensive test coverage contribute to higher quality software releases, while the reduction in manual testing and maintenance efforts leads to significant cost savings.

CRT is particularly useful for end-to-end testing of complex workflows across different platforms and environments, regression testing to ensure new changes do not break existing functionality, performance testing to ensure applications meet required benchmarks, and mobile application testing across different devices and operating systems without the need for a physical device lab.

3.18.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
<p>Cloud test execution platform: CRT is a cloud-based hyper scalable test automation platform. It can provision test automation infrastructure on the cloud with a click of a button. CRT executes and optimises test runs with AI and provides detailed execution reports and logs. It leverages AI in test execution monitoring and can detect change points in test execution runs. In addition, the platform supports for example test generation by applying combinatorial testing among other techniques.</p>	<p>Implementation Level: Implemented Estimated Delivery Date: MS7 (M37) License: https://www.copado.com/company-legal-agreements/ Deviation:</p>	<p>https://docs.copado.com/articles/#!release-notes-publication/copado-robotic-testing-latest</p>

Table 35 Capabilities Implementation Status of the CRT Solution

3.18.3 Useful Resources

- <https://docs.copado.com/articles/#!copado-robotic-testing-publication/copado-robotic-testing>
- <https://www.youtube.com/channel/UCo2lXZeJ8qNyIbbWWCG9z3Q>

3.19 Solution - CRTQI (QEN)

3.19.1 Overview

Solution: CRTQI		Partner: QEN]
Current TRL: 6	License: commercial	Contacts: knupponen@copado.com nsharma@copado.com
Online Resource: https://docs.copado.com/articles/#!copado-robotic-testing-publication/robotic-testing-analytics		Collaborators: Westermo
<p>Description: Copado Robotic Testing's Quality Intelligence and Analytics feature provides a comprehensive overview of the testing landscape, enabling teams to make informed decisions based on actionable insights. This feature allows users to define custom metrics by filtering desired data points or using formulas, facilitating the creation of metrics like Test Pass Rate by dividing the number of passed tests by the total executed tests. The Quality Intelligence Dashboard is a pivotal component, offering a clear and general vision of project test cases. Users can sort project data using filters and select date ranges to view relevant data, distinguishing the latest runs and their related test data. This dashboard supports filtering by test suites and date ranges, enhancing the visibility and management of test outcomes. Additionally, the analytics component is designed to make quality visible, providing insights into test data and behaviours, which helps in identifying trends, pinpointing issues, and improving test strategies over time. The Quality Intelligence feature is integral to maintaining high standards of software quality, ensuring that teams can quickly address issues and maintain a focus on continuous improvement</p>		
<p>Improvement: Key aspects of CRT's Quality Intelligence and Analytics include the introduction and refinement of dashboards that provide comprehensive overviews of project test cases, enabling users to filter and sort data according to various parameters such as test suites and date ranges. These dashboards are designed to help teams quickly identify trends, pinpoint issues, and assess the impact of changes on software quality.</p>		
<p>Another significant area of development is the ability to define custom metrics, which allows teams to tailor their quality metrics to match specific project needs or goals. This customization capability supports a more nuanced analysis of test data, facilitating targeted improvements and strategic decision-making. Additionally, CRT's focus on actionable insights means that the analytics tools are not just about data presentation but also about enabling teams to take specific actions based on the insights gathered. This could include identifying flaky tests, understanding the root causes of failures, and prioritising areas for improvement based on the impact on overall quality.</p>		
<p>Intended Users: Developers, testers, and quality assurance professionals seeking to streamline and enhance their testing processes through automated, AI-powered solutions.</p>		

Satisfied Requirement:	
<ul style="list-style-type: none"> ● GR Mod 01 - Use AI techniques for verification of specifications and high-level models ● GR Mod 02 – Use formal models, automated reasoning and/or ML techniques for test generation. ● GR RE 04 - Verify the consistency of the requirements ● W_UCS_3 - AI-powered root cause analysis w.r.t functional issues 	
AI-Augmented tool set components implemented:	Task: T4.2
<ul style="list-style-type: none"> ● AI for Testing 	
Use within AIDoArt Use Cases	Already planned use: WMO_CS10
Use within AIDoArt challenges:	
#2 L'Aquila: Exploring test results data (WMO)	
#3 Valencia: Test case dependencies (WMO)	
#4: Västerås: Performance Data Exploration (WMO)	
#5 Linz: Anomaly detection and Seasonal quality metrics (WMO)	

Table 36 Synopsis table of the CRTQI Solution

Copado Robotic Testing Quality Intelligence (CRTQI) is an integral component of the Copado Robotic Testing suite, designed to provide deep insights and analytics on the testing process. It focuses on monitoring and analysing various aspects of test execution, such as execution times, results, and trends over time. By examining these elements, CRTQI aims to identify statistically significant changes that could indicate potential issues within the application being tested or the testing infrastructure itself. At the heart of CRTQI is its ability to analyse trends in test execution. This involves a detailed examination of how test execution times and outcomes evolve, allowing teams to spot any deviations that might suggest problems like performance degradation in the application or inefficiencies in the testing setup. This proactive approach to monitoring helps in pre-empting potential issues before they escalate, ensuring the application's performance and reliability remain high.

CRTQI's monitoring mechanisms are not just about identifying problems; they also play a crucial role in continuous improvement efforts. By providing a clear view of testing trends and outcomes, teams can make informed decisions on where to focus their testing and development efforts. This data-driven approach to quality assurance helps in optimizing testing strategies, improving test coverage, and ultimately enhancing the quality of the software being developed.

The benefits of using CRTQI extend beyond problem detection and continuous improvement. It also offers a platform for better decision-making regarding resource allocation, testing priorities, and development focus. By understanding the dynamics of test results and execution times, teams can prioritize areas that need immediate attention, allocate resources more efficiently, and streamline their testing processes for better efficiency and effectiveness.

3.19.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
<p>QI for DevOps: Quality Intelligence (QI) for DevOps is a DevOps data analytics and integration solution that provides you with tried and true DevOps metrics that allow you to: (1) Understand the current status across all phases of the DevOps process (DevOps - development and operations), (2) Understand how different factors affect each other (leading indicators), (3) Solve issues proactively rather than after-the- fact, and (4) Start measuring DevOps on day 1 by just configuring the dashboard and integrations – without any massive infrastructure development projects. QI for DevOps gathers data from your DevOps data sources and presents it in a form that can be easily understood by everyone in the organization.</p>	<p>Implementation Level: Implemented Estimated Delivery Date: MS4 (M20) License: https://www.copado.com/company-legal-agreements/ Deviation: No deviations.</p>	<p>https://docs.copado.com/articles/#!release-notes-publication/copado-robotic-testing-latest</p>

Table 37 Capabilities Implementation Status of the CRTQI Solution

3.19.3 Useful Resources

- <https://docs.copado.com/articles/#!copado-robotic-testing-publication/copado-robotic-testing>
- <https://www.youtube.com/channel/UCo2lXZeI8qNylbbWWCG9z3Q>

3.20 Solution - QEDITOR (QEN)

3.20.1 Overview

Solution: QEDITOR		Partner: QEN
Current TRL: 6	License: commercial	Contacts: knupponen@copado.com nsharma@copado.com
Collaborators: Westermo		
<p>Description: QEditor, a central component of Copado Robotic Testing (CRT), is an integrated development environment designed to facilitate the creation, editing, and management of automated test cases with ease and efficiency. It leverages the power of QWords, which are natural language keywords, to simplify the test authoring process, making it accessible even to those with minimal coding experience. QEditor supports a wide array of features to aid in test development, including real-time error detection, predictive and completion suggestions based on machine learning algorithms, and the ability to interact with dynamic metadata from the instance of the application being tested. This environment is not only about simplifying the creation of test cases</p>		

but also about enhancing their quality and effectiveness through intelligent suggestions and error detection. CRT's approach to test authoring overall emphasizes ease of use, efficiency, and the democratization of testing. It includes Guided Authoring, which simplifies test creation with a low-code approach, and the Flow Editor, which provides a visual representation of test flows, further reducing the need for detailed scripting knowledge. The Recorder feature allows users to capture user interactions with the application and automatically generate test scripts, significantly speeding up the test creation process. Additionally, CRT's Self-Healing Tests and Performance Change Detection features ensure that tests remain relevant and accurate over time, even as applications evolve. Together, QEditor and CRT's suite of test authoring tools represent a comprehensive solution for creating, managing, and executing automated tests. They are designed to support teams in maintaining high-quality software through efficient and effective testing processes, making advanced testing capabilities accessible to a broader range of users, including those without extensive technical backgrounds

Improvement: QEditor within Copado Robotic Testing (CRT) has seen significant enhancements aimed at improving the user experience and expanding its capabilities in test authoring. One of the notable developments is the introduction of the "Intent-based Testing" Panel, currently in beta, which is powered by CopadoGPT. This AI-driven feature is designed to guide users through the test authoring process by understanding their intent, thereby improving the accuracy and efficiency of test creation. This development represents a leap forward in making test authoring more intuitive and aligned with natural language processing technologies. The introduction of the Flow Editor is also noteworthy. The Flow Editor is a revolutionary tool for script-less test automation. It allows users to create test cases by simply creating a flow of their test steps, significantly reducing the need for scripting knowledge and making test automation more accessible to users with varying levels of technical expertise.

Intended Users: Developers, testers, and quality assurance professionals seeking to streamline and enhance their testing processes through automated, AI-powered solutions.

Satisfied Requirement:

- GR Mod 01 - Use AI techniques for verification of specifications and high-level models
- GR Mod 02 – Use formal models, automated reasoning and/or ML techniques for test generation.
- GR RE 04 - Verify the consistency of the requirements

AI-Augmented tool set components implemented:

- AI for Testing

Task: T4.2

Use within AIDOaRt Use Cases

Already planned use: WMO_CS10

Use within AIDOaRt challenges:

- #2 L'Aquila: Exploring test results data (WMO)
- #3 Valencia: Test case dependencies (WMO)
- #4: Västerås: Performance Data Exploration (WMO)
- #5 Linz: Anomaly detection and Seasonal quality metrics (WMO)

Table 38 Synopsis table of the QEditor Solution

Copado Robotic Testing (CRT) QEditor is a sophisticated feature within the CRT suite designed to streamline the process of creating, managing, and executing test cases. This feature is tailored to accommodate both technical and non-technical users, making the development of automated tests accessible to a broader range of team members. CRT QEditor emphasizes simplicity, efficiency, and



flexibility, allowing users to quickly generate robust test scripts without the need for extensive coding knowledge.

The essence of CRT QEditor lies in its user-friendly interface and tools that facilitate the rapid creation of test cases. Among these tools are the Guided Authoring system and the Flow Editor, both of which employ a low-code approach to test development. This approach significantly reduces the barrier to entry for test automation, enabling users to focus on the logic and objectives of their tests rather than the intricacies of scripting languages.

Another pivotal feature of CRT QEditor is the Recorder, which captures user interactions with the application under test and automatically generates corresponding test scripts. This not only accelerates the test creation process but also ensures that tests accurately reflect real user scenarios. Additionally, CRT QEditor includes capabilities for Self-Healing Tests, which automatically update test scripts in response to changes in the application, thereby reducing maintenance efforts and enhancing test reliability.

CRT QEditor also supports Data-Driven Testing, allowing users to easily parameterize tests with different data sets for more comprehensive coverage. This feature is particularly useful for validating application behaviour across various scenarios and data conditions. Moreover, the integration of Smart Predictions utilizes AI to suggest test steps based on the application's context, further simplifying the test creation process.

The benefits of CRT QEditor extend to improved efficiency in the testing cycle, enabling faster identification and resolution of defects. By making test automation more accessible and less time-consuming, teams can allocate more resources to areas that enhance the application's value and user experience. Furthermore, the comprehensive test coverage and the ability to quickly adapt tests to application changes contribute to higher software quality and a more reliable release process.

3.20.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
<p>QEditor - AI-assisted test authoring: QEditor is a highly advanced test development environment for QWord and Robot Framework test scripts. It leverages a massive amount of test case data and using AI/ML it guides you through the scripting process and helps you to create scripts by constantly monitoring and analysing your actions. QEditor supports both progression and regression testing. It has a support for live testing the application and it supports test case recording. QEditor is provided as a cloud native solution but can also be installed locally as a VS Code extension</p>	<p>Implementation Level: Implemented Estimated Delivery Date: MS5 (M31) License: https://www.copado.com/company-legal-agreements/ Deviation: Any deviation.</p>	<p>https://docs.copado.com/articles/#!release-notes-publication/copado-robotic-testing-latest</p>

Table 39 Capabilities Implementation Status of the QEDITOR Solution

3.20.3 Useful Resources

- <https://docs.copado.com/articles/#!copado-robotic-testing-publication/copado-robotic-testing>
- <https://www.youtube.com/channel/UCo2lXZei8qNylbbWWCG9z3Q>

3.21 Solution - LogGrouper (RISE)

3.21.1 Overview

Solution: LogGrouper		Partner: Westermo
Current TRL: 4	License: Proprietary RISE	Contacts: muhammad.abbas@ri.se sarmad.bashir@ri.se
Online Resource: N/A		Collaborators: Westermo, RISE
<p>Description: LogGrouper tool is developed to streamline the analysis of nightly test execution failure logs. LogGrouper intends to significantly reduce the manual efforts required to identify common root causes of test failures by automating the grouping of failure logs. The tool employs natural language processing and clustering techniques to ensure meaningful grouping of logs. The inputs are failed test cases and their log files and the output is identified as similar clusters.</p>		
<p>Improvement: The development of the LogGrouper prototype started in Y1 of AIDOaRt. We continued evolving the various modules of the approach in Y2 and Y3.</p>		
<p>Intended Users: LogGrouper is developed to aid software developers in analysing the failing test logs and getting holistic insights into similar failing test cases.</p>		
<p>Satisfied Requirement:</p> <p>Generic Requirements:</p> <ul style="list-style-type: none"> • GR Mon 2.6 - monitor and identify root causes to anomalies • GR Test 4 - automated evaluation of test results <p>UC requirement:</p> <ul style="list-style-type: none"> • W_UCS_2 - AI-powered root cause analysis w.r.t functional issues 		
<p>AI-Augmented tool set components implemented:</p> <ul style="list-style-type: none"> • AI for monitoring • AI for Testing 		Task: T4.2
<p>Use within AIDOaRt Use Cases</p>	<p>Already planned use: Westermo, mostly in the context of [W_UCS_2] "AI-powered root cause analysis w.r.t. functional issues"</p>	
	<p>Potential foreseen links: n/a</p>	
<p>Use within AIDOaRt challenges: 1st Hackathon: 500 nights of testing</p>		

Table 40 Synopsis table of the LogGrouper Solution

The implementation of large industrial software systems requires continuous integration and testing. The repeated execution of test suites on different variants of the system generates a huge number of test execution logs that engineers must review to analyse root cause for failing tests. In addition, there could be duplicate logs for failing test cases that are executed on different systems. In this regard, the manual review of failure logs is a tedious activity and creates overhead for developers and test

engineers. Therefore, we propose a tool LogGrouper to simplify the exploration of test results to show the root causes that may be shared by several different test cases. LogGrouper communicates with a system to get a list of failing tests and then collects the related logs. The logs are pre-processed (e.g., for removal of stop words and lemmatization), vectorized, and clustered. For vectorization of the logs, we use information retrieval based TFIDF technique to represent them as features. Additionally, the grouping of the similar failure logs as feature vectors is based on Density-Based Spatial Clustering of Application with Noise (DBSCAN) machine learning algorithm. The flow of the proposed approach is illustrated in Figure 33 below. Furthermore, Figure 34 shows the results from the developed tool.

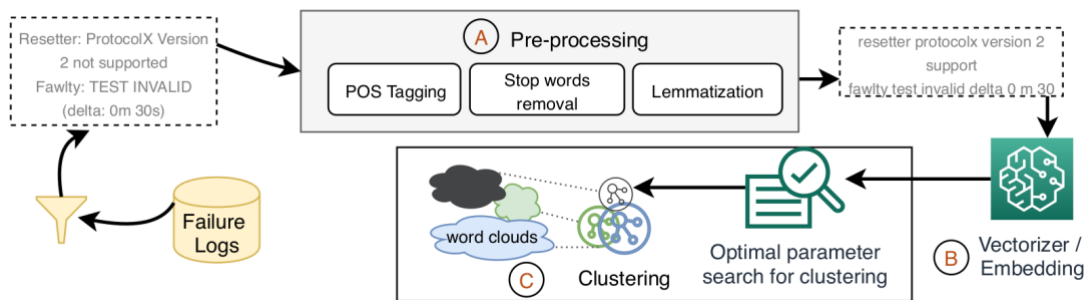


Figure 33 Overall process of LogGrouper

cluster_number	number_of_items	outcomes_id	Generated 2023-03-08 08:00
0	12	<ul style="list-style-type: none"> 62598800 62603636 62599572 62598950 62598835 62598965 62599131 62600860 62600967 62603677 62598905 62599502 	<ul style="list-style-type: none"> packets max allowed protocolX lb eth stream packet checks failed test protocolX fail packets max allowed protocolY lb eth stream packet checks failed test protocolY fail packets max allowed protocolZ lb eth stream packet checks failed test
1	9	<ul style="list-style-type: none"> 62603209 62598741 62599138 62601873 62592222 62598910 62578918 62598860 62599388 	<ul style="list-style-type: none"> packets max allowed protocolX lb eth stream packet checks failed test protocolX fail packets max allowed protocolY lb eth stream packet checks failed test protocolY fail packets max allowed protocolZ lb eth stream packet checks failed test
2	5	<ul style="list-style-type: none"> 62599009 62598984 62577869 62577711 62608876 	<ul style="list-style-type: none"> packets max allowed protocolX lb eth stream packet checks failed test protocolX fail packets max allowed protocolY lb eth stream packet checks failed test protocolY fail packets max allowed protocolZ lb eth stream packet checks failed test

Figure 34 Anonymised screenshot of one clustering

3.21.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
GroupLogs: Grouping of failed test cases logs for root cause analysis	Implementation Level: Implemented Estimated Delivery Date: MS8 (M42) License: Proprietary RISE Deviation: No particular deviation	The solution is provided to Westermo as an API endpoint for integration in their DevOps pipeline.

Table 41 Capabilities Implementation Status of the LogGrouper Solution

3.21.3 Useful Resources

- Abbas, Hamayouni, Moghadam, Saadatmand, and Strandberg. (2023) Making Sense of Failure Logs in an Industrial DevOps Environment. In Springer ITNG.

3.22 Solution - BugIdentifier (RISE)

3.22.1 Overview

Solution: BugIdentifier		Partner: Westermo
Current TRL: 3	License: Proprietary RISE	Contacts: sarmad.bashir@ri.se muhammad.abbas@ri.se
Online Resource: n/a	Collaborators: Westermo, RISE	
<p>Description: The BugIdentifier tool addresses the challenge of automated identification and ranking of bug-inducing commits. BugIdentifier is an offline tool that consists of three phases. In the first phase, the raw datasets, mainly git commit history and test execution results, are pre-processed using part-of-speech tagging, stop words removal, and lemmatization. After the pre-processing, the failing test cases are mapped to their commits, and groups are created based on the date and feature branch filter to reduce the search space. Furthermore, a feature matrix is created based on the most relevant features of the git commit history, e.g., the number of commit changes and similarity between the commit message and related failed test execution results. The similarity is calculated through a deep learning transformer model that captures the semantic relationships between the sentences. In the final phase, the commits are ranked based on the overall score of the feature matrix.</p>		
<p>Improvement: The development of the BugIdentifier prototype started in Y2 of AIDOaRt. We continued evolving the various modules of the approach in Y3.</p>		
<p>Intended Users: BugIdentifier is developed to aid software developers in analysing the failing test logs and related commits and getting direct links to the source code changes that may have caused the test to fail.</p>		
<p>Satisfied Requirement: Generic Requirements:</p> <ul style="list-style-type: none"> • GR Mon 1.1: The system shall access off-line data. The tool reads archived log files and test results data. • GR Test 4: The system shall perform automated evaluation of testing results. The purpose of the tool is to propose root causes for failing tests. 		

<ul style="list-style-type: none"> ● GR Test 5: The system shall report or fix the problems detected in the testing phase. Again, the purpose of the tool is to propose root causes for failing tests. 	
UC requirement:	
<ul style="list-style-type: none"> ● W_UCS_1 - AI-augmented DevOps development process ● W_UCS_2 - AI-powered root cause analysis w.r.t functional issues ● W_UCS_4 - AI-powered log analysis/ root cause analysis 	
AI-Augmented tool set components implemented: <ul style="list-style-type: none"> ● AI for monitoring ● AI for Testing 	Task: T4.2
Use within AIDOaRt Use Cases	Already planned use: [W_UC1, W_UC2, W_UC4]
	Potential foreseen links: n/a
Use within AIDOaRt challenges: Hackathon 3: Identifying bug-inducing code changes, Hackathon 4: Bug-Inducing Commits: Westermo and RISE	

Table 42 Synopsis table of the BugIdentifier Solution

The BugIdentifier tool employs a natural language processing technique and heuristics to conduct log-level analysis for the purpose of filtering out relevant attributes associated with each commit. The first step of the approach is to perform pre-processing of log files extracted from the WeOS, Fawltly test framework, and test execution logs from Westermo’s DevOps setup. The aim is to parse the data to introduce consistency and make sense of the logs for extracting the relevant features and ranking the bug-inducing commits based on their correlation. The filtering mechanism is based on two parameters; date range and specific feature branch of the source code changes (commits). Once the search space is reduced based on the filtering parameters, the relevant extracted features for WeOS and Fawltly framework (commit messages, the number of additions and subtractions in source code, etc.) are associated with each commit and test execution logs (when filtering out critical and error log entries). The ranking of the commits relies on the correlation between the extracted features. Figure 35 illustrates the flow of the BugIdentifier tool.

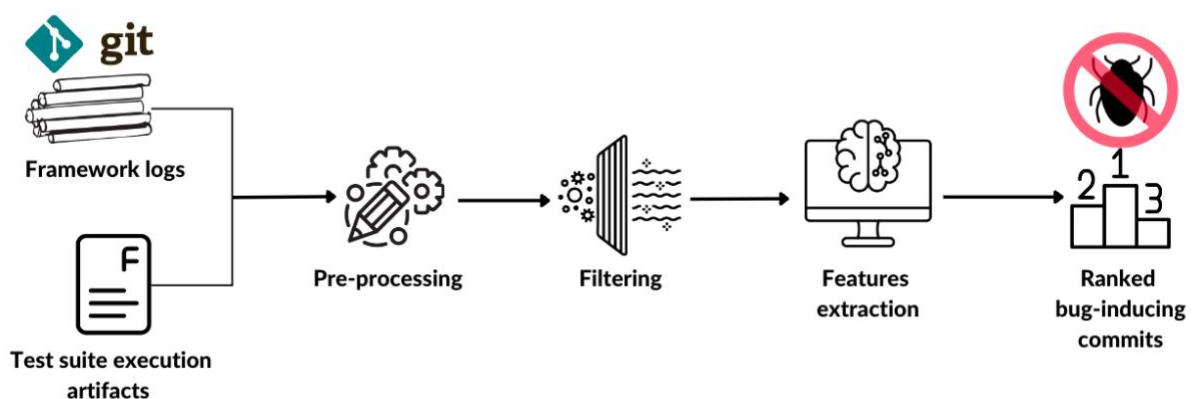


Figure 35 Overview of BugIdentifier workflow

3.22.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
BugInducingCommit: Mapping source code changes with failed test cases	Implementation Level: Implemented Estimated Delivery Date: MS8 (M42) License: Proprietary RISE Deviation: No particular deviation	The solution is provided to Westermo for integration.

Table 43 Capabilities Implementation Status of the BugIdentifier SolutionBug

3.23 Solution - DataAggregator (ROTECH)

3.23.1 Overview

Solution: DataAggregator		Partner: RoTechnology	
Current TRL: 4	License: Proprietary ROTECH	Contacts: diego.grimani@rotechnology.it	
Online Resource: n/a		Collaborators: TEKNE	
Description: Data Aggregator provides to help them make better decisions, improve process efficiency and, finally, understand performance of the Platform. Consistent evolution and the usability of the data play a key role too. Data Aggregator collects and consolidates data from various sources, data aggregation provides decision-makers with a comprehensive view of relevant information. This enables them to make informed decisions based on accurate, up-to-date data rather than relying on fragmented or outdated information. Data Aggregator simplifies the task of accessing, processing, and analysing large volumes of data. This solution automates, also, the process, saving time and reducing the potential for errors.			
Improvement: Data aggregator solution Provides help to make better decisions, improve process efficiency and finally, understand performance of the Platform. It starts from TRL 3 .			
Intended Users: System integrators in automotive and electronics sectors on vehicle customization.			
Satisfied Requirement: <ul style="list-style-type: none"> ● GR Mod 04: Use semi-automatic model synthesis for design- and run-time verification ● GR Cod 03: The AIDoArt Framework imports/exports variables/procedures between code and models 			
AI-Augmented tool set components implemented: n/a		Task: <ul style="list-style-type: none"> ● T4.3 	
Use within AIDoArt Use Cases	Already planned use: TEK_UCS_03 - Operating life monitoring		
	Potential foreseen links: n/a		
Use within AIDoArt challenges: Operating Life Monitoring			

Table 44 Synopsis table of the DataAggregator Solution

Data Aggregator: provides to help them make better decisions, improve process efficiency and, finally, understand performance of the Platform. Consistent evolution and the usability of the data play a key role too. Data Aggregator's Decision support 4 techniques configurations:

1. **In-comm aggregation:** Uses a multi-hop system for the process of gathering and routing tests information inside the Data Aggregator

2. **Tree-based approach:** Uses an aggregation tree, mapping and putting the data from leaves to root (source and sink nodes respectively)
3. **Cluster-based approach:** This approach is used to collect large amounts of data across the entire Platform
4. **Multi-path approach:** Uses partially aggregated data sent to the root or parent internal table which then can send the data along various paths

Implementing a communication protocol between an on-board platform and a remote platform that includes features for secure communication using encryption and decryption algorithms. The target of this protocol is to facilitate the exchange of data and information between the two platforms, enabling better decision-making, improved process efficiency, and a deeper understanding of the platform's performance

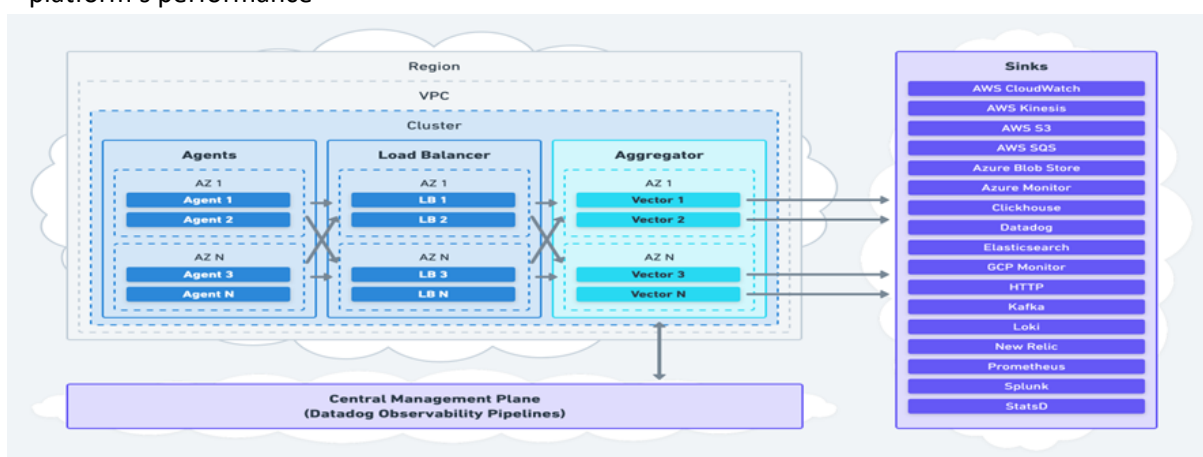


Figure 36 Overview DataAggregator workflow

3.23.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
Data aggregation: This solution is composed of one or more Relational and Non-relational Databases in order to store and aggregate data for Big Data management and Machine Learning purposes. Moreover, it will also include replication techniques based on the master - slave replication or any other type related to the chosen Databases.	Implementation Level: Implemented Estimated Delivery Date: MS6 (M28) License: Free-license Deviation: No deviation	The solution has been implemented and is currently in performance testing with the selected tools.

Table 45 Capabilities Implementation Status of the DataAggregator Solution

3.24 Solution - Modelio (SOFT)

3.24.1 Overview

Solution: Modelio - NLP4RE		Partner: SOFT
Current TRL: 4	License: Apache Public License (APL)	Contacts: bilal.said@docaposte.fr
Online Resource: http://Modelio.org		Collaborators: Bilal Said (Research Project Manager) Alessandra Bagnato (Research Scientist) Andrey Sadovykh (Research Scientist)
<p>Description: The Modelio NLP4RE prototype streamlines the workflow of system engineers by automatically processing and analysing textual requirements. The prototype leverages traditional machine learning and state-of-the-art language models to extract valuable insights from requirements documents. Namely, it supports:</p> <ul style="list-style-type: none"> ● Requirement identification and extraction: Utilising NLP techniques to automatically identify and extract requirements from textual documents, eliminating manual effort. ● Requirement similarity detection: Leveraging past project data to identify similar requirements, promoting knowledge reuse, and reducing redundancy. ● Requirement classification: Employing machine learning models to automatically classify requirements as functional or non-functional, further categorised within subcategories of non-functional requirements. ● Automatic team assignment: Utilising historical data and domain knowledge to suggest optimal team assignments for each requirement, promoting efficient workflow management. <p>This research aims to address the challenges faced by requirements engineers, particularly in managing large-scale projects with hundreds of requirements. The prototype facilitates this by harnessing knowledge from previous projects and domain expertise to deliver intelligent suggestions and automation opportunities.</p>		
<p>Improvement: The development of the NLP4RE prototype started at the end of Y1 of AIDOaRt when tackling the first hackathon challenges. We continued evolving the various modules during Y2 and Y3 to tackle additional hackathon challenges, and to integrate recent state-of-the-art methods and techniques in NLP. Recently, we built a web application to make the requirements semantic search feature easily accessible to end-users through an intuitive GUI.</p>		
<p>Intended Users: The NLP4RE prototype is designed for use by requirements engineers studying recurrent clients' bids or tenders with relatively similar or standard requirements specification documents and who are interested in getting insights on new bids based on responses to previous ones.</p>		
<p>Satisfied Requirement:</p> <ul style="list-style-type: none"> ● GR RE 03 - Analyse the requirements expressed in formal language and produce suggestions or prescriptions for the Requirements Engineer and the System Engineer. ● GR RE 04 - Verify the consistency of the requirements. <p>Satisfied Use Case Requirements:</p> <ul style="list-style-type: none"> ● BT_R01 - NLP contextual analysis of requirements and match against database of responses/solutions ● HIB_R01 - The AIDOaRt AI algorithms must be able to analyse log files (text) from the restaurant application. 		

<ul style="list-style-type: none"> HIB_R02 - The AIDOaRt solution will enable to process requirements expressed in natural language in Trello boards 	
AI-Augmented tool set components implemented: <ul style="list-style-type: none"> <u>AI for Requirements Engineering</u> - AI/ML-based capabilities to support the requirements engineering phase of the system development, namely AI/NLP methods for requirements similarity check and requirements allocation. 	Tasks: <ul style="list-style-type: none"> T4.2 for the extraction and analysis of requirements from unstructured specification text documents. T4.4 for the validation on partners' data and use case scenarios.
Use within AIDOaRt Use Cases	Already planned use: BT_UCS1 - Requirements analysis and recommendation of most suitable action per requirement. Potential foreseen links: HIB_UCS2 - Management and verification of the requirements
Use within AIDOaRt challenges: The NLP4RE module of Modelio has been initially specified and used in the challenge #2 "Recommendation system for RE" during hackathon #1. It was further enhanced to tackle deployment and integration aspects in challenge #8 "Requirements analysis, processing, and response" during hackathon #3. It was revised to incorporate explainability aspects for challenge #4 "Achieving Explainable AI to Support Decision-making during Requirement Engineering Analysis" during hackathon #4.	

Table 46 Synopsis table of the Modelio NLP4RE Solution

During the inaugural AIDOaRt hackathon, a collaborative effort was undertaken with Alstom/Bombardier (BT) to explore and address challenges within the railway industry's requirements engineering process. Specifically, the focus was on requirements semantic similarity and the automated team assignment for requirements. The NLP4RE prototype was then developed to leverage knowledge from past projects by identifying similar requirements (cf. Figure 37), thus informing optimal team allocation decisions. Initial evaluations demonstrated the potential for achieving an accuracy of approximately 70%. This can still be useful to requirements engineers to provide a first insight when looking into vast new tender documents.

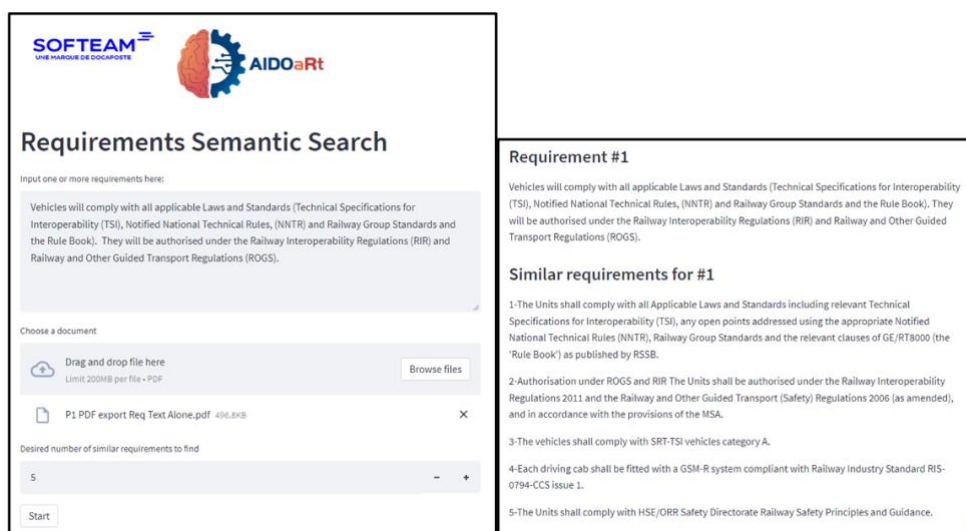


Figure 37 Modelio NLP4RE prototype web interface for Requirements Semantic Search

During the third hackathon, we focused on integrating our prototype in a toolchain with other solutions proposed by MDU and RISE. Whereas, in the fourth hackathon, we addressed the need to provide “explainable” insights to the requirements engineers and exchanged on this topic with MDU and BUT. For our semantic similarity check tool, an explanation of the high similarity between two different requirements can be shown to the end user by highlighting the common keywords and tokens.

In contrast with similar recent research work, our NLP4RE prototype is trained on real-world datasets provided by our industrial partners in the AIDOaRt project. This ensures the prototype delivers pertinent suggestions and actionable insights tailored to specific industry needs and diverse use cases. However, this desired capability remains limited when zero-shot learning is used with existing language models, especially since the requirements text is often technical (e.g., in the railway industry) and most words and tokens would fall Out-Of-the-Vocabulary (OOV) used to train the original models (typically generic text corpora curated from open sources, e.g., Wikipedia, historical Reuter news cables, etc.). This implies the necessity to fine-tune these language models on the industrial datasets to enhance their performance. Recently, we have been considering using embedding vector databases to gain efficiency. We are also exploring the potential of recently released Large Language Models (LLM), namely open-source models, and Retrieval Augmented Generation (RAG) in enhancing the relevance of our obtained results.

3.24.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release Notes
NLP4RE - A prototype for requirements identification, extraction, classification, and similarity check in unstructured text specification documents with NLP & AI/ML techniques.	Implementation Level: Implemented Estimated Delivery Date: MS6 (M28) License: Apache Public License (APL) Deviation: No particular deviation.	The current prototype allows the end-users to perform requirements similarity checks using an intuitive graphical user interface. The requirements identification, extraction, and classification features are only accessible programmatically.

Table 47 Capabilities Implementation Status of the Modelio Solution

3.24.3 Useful Resources

Main website: <http://Modelio.org>

Source code repository: <https://github.com/Modelio-R-D/NLP4RE>

Installation and execution instructions: <https://github.com/Modelio-R-D/NLP4RE#readme>

Related publications:

- Sadovykh, A., Yakovlev, K., Naumchev, A., Ivanov, V. (2024). Natural Language Processing with Machine Learning for Security Requirements Analysis: Practical Approaches. In: Sadovykh, A., Truscan, D., Mallouli, W., Cavalli, A.R., Seceleanu, C., Bagnato, A. (eds) CyberSecurity in a DevOps Environment. Springer, Cham. (https://doi.org/10.1007/978-3-031-42212-6_2)



- Ivanov, V., Sadovykh, A., Naumchev, A., Bagnato, A., Yakovlev, K. (2022). Extracting Software Requirements from Unstructured Documents. In: Burnaev, E., et al. Recent Trends in Analysis of Images, Social Networks, and Texts. AIST 2021. Communications in Computer and Information Science, vol 1573. Springer, Cham. (https://doi.org/10.1007/978-3-031-15168-2_2)

3.25 Solution - AALpy (TUG)

3.25.1 Overview

Solution: AALpy		Partner: TUG
Current TRL: 5	License: MIT	Contacts: benjamin.vonberg@ist.tugraz.at
Online Resource: https://github.com/DES-Lab/AALpy		Collaborators: Andrea Pferscher Benjamin von Berg
Description: AALpy is an automata learning library that supports learning behavioural models of reactive systems from a predefined set of execution traces (passive learning) or by interacting with the system (active learning).		
Improvement: At the inception of AIDOaRt, AALpy was at TRL 3. Since then, several learning algorithms have been implemented. The main development funded by AIDOaRt in the last year was the addition of a general framework for passive learning based on state merging.		
Intended Users: AALpy is intended for researchers and industry alike. Its high flexibility and extensibility make it a good starting point for researchers experimenting with new learning algorithms. Automata learning is used in validation and verification tasks through paradigms like learning-based testing. In this regard AALpy users benefit from a wide range of algorithms and its ease of use.		
Satisfied Requirement: Generic Requirements: <ul style="list-style-type: none"> • GR Mod 2 - Use formal models, automated reasoning and/or ML techniques for test generation • GR Mod 4 - Use semi-automatic model synthesis for design- and run-time verification. • GR Mon 2.2 - Monitoring in AIDOaRt could be done with the purpose of identifying deviations, anomalies or security events UC requirements: <ul style="list-style-type: none"> • AVL_RDE_R01 - Learning behavioural models of human driving behaviour from data • AVL_SEC_R01 - Use of automata learning and ML techniques to derive SUT models 		
AI-Augmented tool set components implemented: <ul style="list-style-type: none"> • Engagement and Analysis • Explainability • AI for Modeling • AI for Testing • AI for Monitoring 		Task: T4.2

Use within AIDoRt Use Cases	Already planned use: <ul style="list-style-type: none"> ● AVL_MBT_UCS1 ● AVL_RDE_USC3 ● AVL_SEC_UCS2
	Potential foreseen links: -
Use within AIDoRt challenges: <ul style="list-style-type: none"> ● Hackathon 1 - Real Driver Emission ● Hackathon 3 - Learning-based fuzzing of AGL ● Hackathon 3 - Testing ADAS functions ● Hackathon 4 - Learning-based fuzzing of AGL ● Hackathon 5 - Model-based testing of an Advanced Driver Assistance System 	

Table 48 Synopsis table of the AALpy Solution

Over the course of the project, AALpy has been improved by adding several automata learning algorithms and optimising existing implementations. The development of AALpy was also driven according to the needs of our project partners as is evident from the collaborations and hackathons. During the first hackathon, we investigated how passive automata learning can help the estimation of Real Driver Emission (RDE) data where the aim is to create a behavioural model of a human driver that reacts to environment stimuli. In this scenario, there is a data set of measurements collected from test drives from which human driver behaviour should be extracted. Since the available data is expected to be incomplete, we opted for a probabilistic model. Initial results showed promise but needed further work, which has been the starting point for two developments. For one, we developed a novel method for integrating domain knowledge about system modes into passive learning algorithms. An example in the context of RDE would be executing manoeuvres such as overtaking or decelerating due to a changing speed limit. While this approach is not officially a part of AALpy, a link is provided below. Secondly, we implemented a general framework for a whole class of passive learning algorithms that can be used to quickly implement a wide variety of algorithms and allows for easy experimentation. In the following hackathons we also contributed to a challenge that deals with testing Advanced Driver Assistance Systems (ADAS) and Automotive Grade Linux (AGL), which is an operating system for automotive applications. In both cases, we want to know whether a system exhibits any behaviour that violates a safety specification. An idea to avoid costly full factorial testing is to gradually build a model of the system while testing, in order to reduce sampling in regions that are expected to be non-critical, which corresponds to the paradigm of Learning-based Testing (LBT). This motivates the use of active automata learning in the two previously mentioned use cases. In the former, a significant challenge is how to incorporate domain knowledge into the learning process, which is currently ongoing work. In the latter, we developed a method for learning-based fuzzing on top of AALpy.

3.25.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
Automata learning of deterministic systems: Inference of a behavioural model of a black-box system that behaves deterministically. Following modelling formalisms are supported: deterministic finite automata, Mealy machines, Moore machines	Implementation Level: Implemented Estimated Delivery Date: MS7 (M37) License: MIT Deviation: -	Active and passive algorithms were implemented.
Automata learning of non-deterministic systems: Inference of a behavioural model of a black-box system that behaves non-deterministically. To provide efficiency, learning can also be combined with an abstraction mechanism. Following modelling formalisms are supported: Observable non-deterministic finite state machines (+ abstracted version)	Implementation Level: Partially Implemented Estimated Delivery Date: MS7 (M37) License: MIT Deviation: Passive algorithms not fully developed / implemented	Active algorithms were implemented. Passive algorithms are in development but have been put on hold due to a higher demand on passive learning for stochastic systems.
Automata learning of stochastic systems: Inference of a behavioral model of a black-box system that behaves stochastically	Implementation Level: Implemented Estimated Delivery Date: MS7 (M37) License: MIT Deviation: -	Active and passive algorithms were implemented.

Table 49 57 Capabilities Implementation Status of the AALpy Solution

3.25.3 Useful Resources

AALpy is hosted on GitHub, which also serves as the main resource for information about the library. It can be found here: <https://github.com/DES-Lab/AALpy>

Other useful resources include:

- Examples: <https://github.com/DES-Lab/AALpy/blob/master/Examples.py>
- Wiki: <https://github.com/DES-Lab/AALpy/wiki>
- Approach for integrating domain knowledge: <https://github.com/BenjaminVonBerg/hierarchical-automata-learning>

3.26 Solution - S3D (UCAN)

3.26.1 Overview

Solution: S3D - AICROSSIM		Partner: UCAN
Current TRL: 5	License: Open Source for research and academic purposes, under License Agreement for commercial use.	Contacts: davida@teisa.unican.es evillar@teisa.unican.es
Online Resource: https://s3d.unican.es/		Collaborators: TEK
<p>Description: CPSoS modelling, analysis and design framework able to select from a library of reusable components those to be used in the current project. Using them, the functional architecture is decided. A model of the HW/SW execution platform is also developed. By mapping the first to the latter, as many platform-specific solutions can be defined.</p>		
<p>Improvement: From our already existing tool at TRL 6, we have added a new feature. The initial TRL of the new feature was TRL 2. This new feature consists of instrumenting the code with the PAPI library to read the performance counters of a host machine, and training a ML algorithm that maps the performance metrics readed on the host to the performance expected on the target embedded system. A patent pending for approval, a transaction currently being written, and a proof of concept in the TEK's use case put the current TRL of this new feature between TRL 4 and TRL 5.</p>		
<p>Intended Users: This tool provides embedded system designers faster and more accurate simulations of the software to be deployed, even if the hardware platform is not ready yet. Intended users include current users of S3D plus at least the TEK Use case target users, mainly partners related to the automotive domain.</p>		
<p>Satisfied Requirement:</p> <ul style="list-style-type: none"> GR Mod 01 - Use AI techniques for verification of specifications and high-level models. GR Mod 04 - Use semi-automatic model synthesis for design- and run-time verification. <p>Satisfied Use Case Requirements:</p> <ul style="list-style-type: none"> TEK_R_102: The AIDoArT Framework verifies in a semi-automatic manner, at design time, with respect to the requirements, the adequacy (the response versus the resources) of the real components on-which/with-which the system architect has in mind to map/realise the architecture. TEK_R_103: The AIDoArT Framework synthesises in a semi-automatic manner the models needed for the verification at design time (the models that define both the tests and the results, i.e. the pass/fail criteria). TEK_R_104: The AIDoArT Framework interprets in a semi-automatic manner the results of the design time verification. TEK_R_001: The AIDoArT Framework imports/exports variables/procedures between code and models. 		
<p>AI-Augmented tool set components implemented:</p> <ul style="list-style-type: none"> AI for Modeling: The added functionality will provide relevant mechanisms to complete the modelling of software operations since it will provide better adjusted execution times 		Task: T4.3 and T5.3
Use within AIDoArT Use Cases	<p>Already planned use: TEK's use case simulation-based design exploration related scenarios</p> <p>Potential foreseen links: n/a</p>	

Use within AIDOaRt challenges: TEK Use case challenges, in particular Design choices exploration/verification - From 1° to 5° Hackathon.

Table 50 Synopsis table of the S3D Solution

The solution implemented in S3D - AICROSSIM is an extension of S3D simulation tools aimed to run in the simulator so that it can perform cross-platform software simulation using an NN trained to get performance in a target platform after the execution of the simulation in a host platform and the monitoring of carefully selected performance counters.

3.26.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
S3D: CPSoS modelling, analysis and design framework able to select from a library of reusable components those to be used in the current project. Using them, the functional architecture is decided. A model of the HW/SW execution platform is also developed. By mapping the first to the latter, as many platform-specific solutions can be defined.	Implementation Level: Partially Implemented Estimated Delivery Date: MS6 (M28) License: Open Source for research and academic purposes, under License Agreement for commercial use. Deviation: No deviations.	Automatic instrumentation of the code, i.e. automatically adding the PAPI library functions, has not been developed yet. Currently, for the performed tests, the instrumentation of the code has been done manually.

Table 51 Capabilities Implementation Status of the S3D Solution

3.26.3 Useful Resources

We are actively working on preparing a publication detailing the results and insights into its value. Also, a GitHub repository to provide access to the benchmarks and any other related materials.

3.27 Solution - SoSIM (UCAN)

3.27.1 Overview

Solution: SoSIM		Partner: UCAN
Current TRL: 4 and 5	License: Open Source for research and academic purposes, under License Agreement for commercial use.	Contacts: davida@teisa.unican.es ; evillar@teisa.unican.es
Online Resource: https://vippe.unican.es/		Collaborators: TEK
Description: CPSoS simulation and performance analysis tool able to simulate any of the platform-specific models defined in S3D. Non-functional metrics such as delays, throughputs and energy consumptions can be analysed. SoSIM is able to generate execution traces that can be analysed by specific tools.		
Improvement: From our already existing tool at TRL 6, we have added 2 new features. The initial TRL of the new features was TRL 1 and TRL 2. The first new feature consists of using a ML algorithm (a Neural Network) trained to predict the execution time of a piece of code in a particular embedded microprocessor. Particularly, the NN learns the abnormalities that can happen in a complex CPU		

<p>pipe, given the code to be executed. This technique does not take into account cache effects, but it has been proven more accurate than other static (i.e. without executing the code) techniques currently used on the state-of-the-art. Its final TRL is 4. The second feature is the dynamic simulation of the instrumented code with the PAPI library from the previously described S3D tool, including the NN estimations executed in a GPU in parallel with the rest of the simulation. Its final TRL level is TRL 5.</p>	
<p>Intended Users: This tool provides embedded system designers faster and more accurate simulations of the software to be deployed, even if the hardware platform is not ready yet. Intended users include current users of SoSIM plus at least the TEK Use case target users, mainly partners related to the automotive domain.</p>	
<p>Satisfied Requirement:</p> <ul style="list-style-type: none"> GR Mod 01 - Use AI techniques for verification of specifications and high-level models. GR Mod 04 - Use semi-automatic model synthesis for design- and run-time verification. 	
<p>AI-Augmented tool set components implemented:</p> <ul style="list-style-type: none"> Engagement and Analysis: The properties added to the modelling of software operations enable the validation of more complex models. 	<p>Task: T4.3 and T5.3</p>
<p>Use within AIDOaRt Use Cases</p>	<p>Already planned use: TEK's use case</p> <p>Potential foreseen links: n/a</p>
<p>Use within AIDOaRt challenges: Mainly TEK Use case challenges, in particular Design choices exploration/verification - From 1° to 5° Hackathon</p>	

Table 52 Synopsis table of the SoSIM Solution

3.27.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
<p>SoSIM: CPSoS simulation and performance analysis tool able to simulate any of the platform-specific models defined in S3D. Non-functional metrics such as delays, throughputs and energy consumptions can be analysed. SoSIM is able to generate execution traces that can be analysed by specific tools.</p>	<p>Implementation Level: Implemented</p> <p>Estimated Delivery Date: MS6 (M28)</p> <p>License: Open Source for research and academic purposes, under License Agreement for commercial use.</p> <p>Deviation: -</p>	<p>As future work, it is intended to provide a heuristic to automate and optimise the choice of the subset performance counters used on the simulation (since all of them cannot be used at the same time).</p>

Table 53 Capabilities Implementation Status of the SoSIM Solution

3.27.3 Useful Resources

We are actively working on preparing a publication detailing the results as well as comprehensive insights into its merits. Also, a GitHub repository to provide access to the benchmarks and any other related materials.

3.28 Solution - UNISS_SOL_01 (UNISS) - ReqH

3.28.1 Overview

Solution: ReqH		Partner: UNISS
Current TRL: 3	Licence: "Commons Clause" License Condition v1.0	Contacts: dguidotti@uniss.it lpandolfo@uniss.it
Online Resource: https://github.com/AIMet-Lab/AIDOaRt-UNISS-ReqH		Collaborators: Tiziana Fanni (ABI)
<p>Description: ReqH serves as a complementary tool to ReqV, focusing on streamlining the translation of natural language requirements into Property Specification Patterns (PSPs) format. This translation process is facilitated through the utilisation of Large Language Models (LLMs), which have demonstrated remarkable proficiency in understanding and generating human-like text. Implemented in Python language, ReqH takes as input a set of requirements expressed in natural language, typically contained within a txt file. Leveraging the capabilities of LLMs, ReqH then performs the translation into PSP format, generating an output txt file that encapsulates the transformed requirements. By automating this process, ReqH helps in reducing the burden on designers and mitigates the risk of errors inherent in manual translations.</p>		
<p>Improvement: ReqV was developed in past projects and it is at TRL 3-4, representing its proof-of-concept stage. ReqH began development during the AIDOaRt project and is currently at TRL 3.</p>		
<p>Intended Users: Both ReqH and ReqV support designers of safety critical CPSs, enabling them to formally verify functional requirement consistency without requiring specialised expertise in the field of formal methods.</p>		
<p>Satisfied Requirements:</p> <ul style="list-style-type: none"> ● GR RE 01 – The Requirement Management Tool of the AIDOaRt Framework translates requirements from semi-structured language to formal language. ● GR RE 04 – The Requirement Management Tool of the AIDOaRt Framework verifies the consistency of the requirements. ● GR Mod 01 – Use AI techniques for verification of specifications and high-level models. ● GR Mod 2.1 – Monitoring in AIDOaRt could be done with the purpose of identifying violations to pre-defined requirements (e.g. above 85% resource usage). 		
<p>Satisfied Use Case Requirements:</p> <ul style="list-style-type: none"> ● ABI_R05 – Use of automatic tools for compliance verification. 		
<p>AI-Augmented tool set components implemented:</p> <ul style="list-style-type: none"> ● <u>AI for Requirement</u> - It supports requirement consistency checking by leveraging on AI/ML techniques. 		<p>Task: T4.2 and T4.3 for the improvements and developments of the solution. T4.4 for its application in the context of practical use case scenarios.</p>
Use within AIDOaRt Use Cases	Already planned use: ABI	
	Potential foreseen links: Adaptable to various use cases without specificity to any particular one.	
<p>Use within AIDOaRt challenges:</p> <ul style="list-style-type: none"> ● Modelling properties constraints of the ABI UC – 1° Hackathon (Virtual). ● Modelling properties constraints of the ABI UC (continuation) – 4° Hackathon (Vasteras). 		

Table 54 Synopsis table of the UNISS_SOL_01 Solution - ReqH

ReqH is a tool designed to streamline the translation of natural language requirements into Property Specification Patterns (PSPs). It leverages Large Language Models (LLMs), which are known for their ability to understand and generate human-like text. Implemented in Python, ReqH takes input in the form of natural language requirements stored in a text file and translates them into PSP format. One of ReqH's key features is its integration with the LangChain library, available at <https://www.langchain.com/langchain>. This library offers users the flexibility to choose the LLM that best suits their needs. Additionally, LangChain allows users to adjust the contextual parameters provided to the selected LLM, enabling fine-tuning of the translation process.

The quality of ReqH's translation process depends on both the chosen context and the selected LLM. By allowing users to customise these parameters, ReqH ensures that the translation output is accurate and tailored to specific requirements. This flexibility empowers users to achieve optimal results across various use cases. On the other hand, it is crucial to understand that if the selected large language model or the contextual parameters provided to it are not appropriate for the task at hand, the translation output generated by ReqH may be significantly flawed. LLMs, while highly advanced, are not infallible and are susceptible to producing erroneous text.

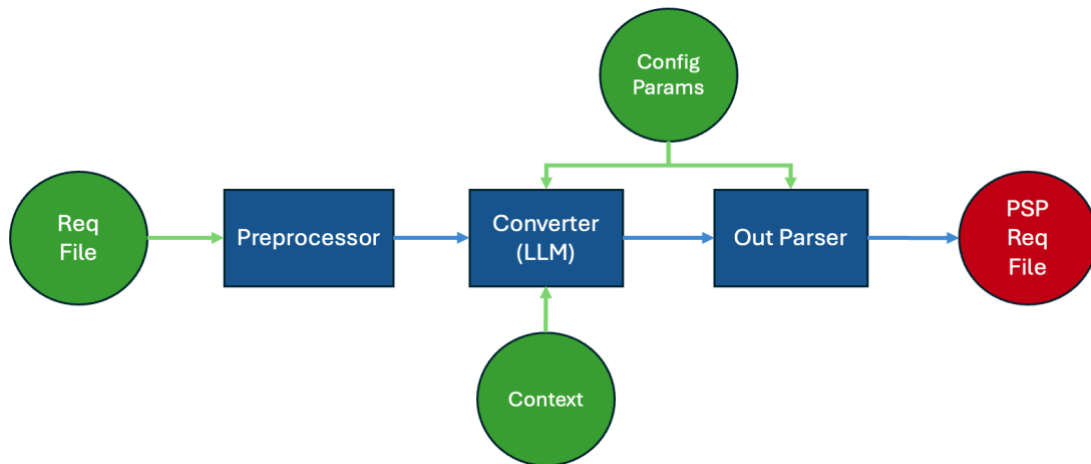


Figure 38 General Operational Framework for ReqH

Inputs and Outputs: ReqH requires as inputs a plain text file containing the requirements of interest, another plain text file containing the context the user wants to provide to the *Converter*, and a set of configuration parameters dictating various options for the *Converter* and the *Out Parser*. The output of ReqH is a plain text file containing the attempted translation of the requirements of interest into PSP format. Figure 38 depicts the operational framework of ReqH. Initially, the *Req File*, provided as a plain text document, undergoes pre-processing by the *Preprocessor* to conform it to the format expected by the *Converter*. Subsequently, the *Converter* processes the list of requirements, utilising both the provided *Context* and the *Config Params* to attempt translation into PSP format on a requirement-by-requirement basis. The results of this translation process are then passed to the *Out Parser*, which formats them according to the *Config Params* and generates the final *PSP Req File*.

3.28.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
Requirements consistency verification: It provides consistency verification of technical specifications in the context of safety critical systems. It also provides feedback to the designer when inconsistencies are detected	Implementation Level: Implemented Estimated Delivery Date: MS7 (M37) Licence: "Commons Clause" License Condition v1.0 Deviation: No deviations reported.	The capability of requirements consistency verification is fully covered by our tool, ReqV. It effectively performs consistency checks on technical PSPs, providing feedback to designers in case of any detected inconsistencies. Additionally, to address the specific needs of our UC provider collaborator, ReqH was developed to support the phase preceding the use of ReqV. It addresses the challenge of supporting the translation of natural language requirements into PSP, a crucial step before inputting requirements into ReqV for formal consistency checking.

Table 55 Capabilities Implementation Status of the UNISS_SOL_01 Solution - ReqH

3.28.3 Useful Resources

Interested users can visit our GitHub page (<https://github.com/AIMet-Lab/AIDOaRt-UNISS-ReqH>) for access to the codebase, documentation, and any related materials. We are actively working on preparing a publication detailing the ReqH capabilities and implementation, which will provide comprehensive insights into its usage and benefits. Additionally, we have plans to release video tutorials by the end of the project, offering step-by-step guidance on using the tool effectively.

3.29 Solution - UNISS_SOL_02 (UNISS) - NNVer

3.29.1 Overview

Solution: NNVer		Partner: UNISS
Current TRL: 3	Licence: "Commons Clause" License Condition v1.0	Contacts: dguidotti@uniss.it lpandolfo@uniss.it
Online Resource: https://github.com/AIMet-Lab/AIDOaRt-UNISS-NNVer		Collaborators: Tiziana Fanni (ABI)
Description: NNVer is a verification tool designed to enhance the reliability of Neural Networks (NNs) for safety critical CPSs. Our solution introduces "step-zero object detection" networks alongside intricate object detection NNs to simplify verification processes. By prioritising object classification over precise localization, NNVer ensures formal certification of step-zero models alongside the original object detection model, bolstering safety by alerting to object presence even when exact locations are unclear.		
Improvement: PyNever was initially developed before AIDOaRt project at TRL 3. During the AIDOaRt project PyNever reached TRL 4. NNVer began development from scratch during the AIDOaRt project and is currently at TRL 3.		

Intended Users: NNVer, tailored for the ABI Use Case, is ideal for developers and engineers working on safety-critical CPS. By simplifying verification of NNs, it streamlines development processes and enhances system reliability, benefiting users with reduced risk and improved safety standards.	
Satisfied Requirement: <ul style="list-style-type: none"> GR Mod 01 – Use AI techniques for verification of specifications and high-level models. 	
Satisfied Use Case Requirement: <ul style="list-style-type: none"> ABI_R02 – Use automated reasoning for verification of Deep Neural Network models. 	
AI-Augmented tool set components implemented: <u>Engagement & Analysis</u> - It relates to this component since the solution focuses on verification based on ML and Automated Reasoning approaches.	Task: T4.2 for the improvements and developments of the solution. T4.4 for its application in the context of practical use case scenarios.
Use within AIDOaRt Use Cases	Already planned use: ABI
	Potential foreseen links: There are potential foreseen links identified.
Use within AIDOaRt challenges: <ul style="list-style-type: none"> Formal verification of Neural Networks – 2° Hackathon (L'Aquila). Adopting AI/ML techniques for image processing in the automotive context – 3° Hackathon (Valencia). Formal Verification of Neural Networks: A "Step Zero" Approach for Vehicle Detection – 5° Hackathon (Linz). 	

Table 56 Synopsis table of the UNISS_SOL_02 Solution - NNVer

The focus of NNVer's development is to deliver verification capabilities, thereby ensuring the reliability of Neural Networks (NNs) for the ABI Rear-view Mirror Use Case, which represents a safety-critical CPS. Our solution is grounded in the acknowledgment of the inherent limitations in the scalability of contemporary state-of-the-art verification tools. To tackle this challenge, we propose the development of "step-zero object detection" networks to complement more intricate object detection neural networks (NNs) as a protective measure. These step-zero networks are specifically designed to focus solely on identifying the presence of objects within an image, without engaging in the precise spatial localization (i.e., prioritising object classification over detection). This intentional simplification makes the step-zero network less complex and thus more conducive to verification processes. Consequently, the final system comprises formally certified step-zero models alongside the original object detection model.

The rationale is that in scenarios where, for instance, the object detection model fails to recognize another car in an image, the step-zero model intervenes by alerting the driver to the presence of the car, even if their exact location is indiscernible. This could enhance the safety of such a critical CPS. Furthermore, in our efforts to streamline the verification process, we adopt a strategic approach whereby we abstain from verifying the convolutional layers within the step-zero network. Instead, we liken these layers to a feature extractor akin to SIFT (Scale-Invariant Feature Transform) or similar

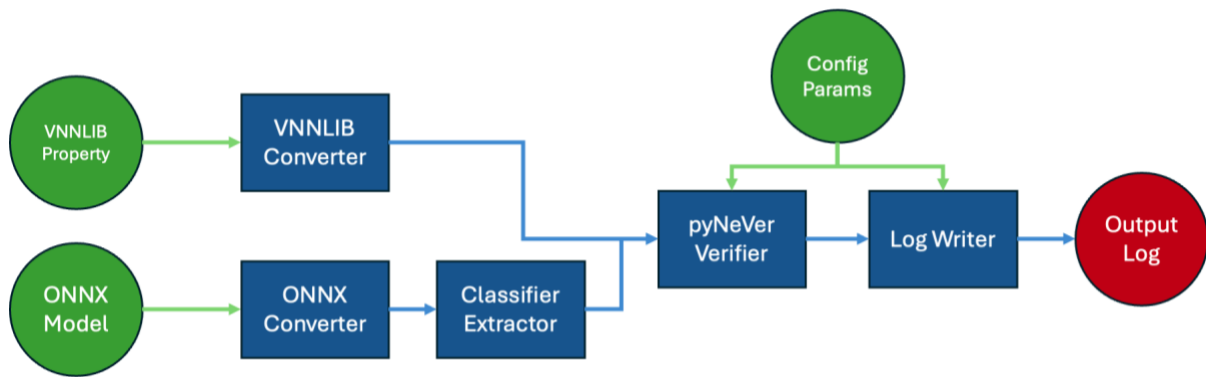


Figure 39 General Operational Framework for NNVer

methodologies. Consequently, our verification efforts are focused on scrutinising the fully-connected layers comprising the classifier.

Inputs and Outputs: NNVer requires as inputs a Neural Network (NN) saved in the standard ONNX format, a property of interest formulated as a VNNLIB property (in compliance with the VNNLIB standard) presented as an SMT-LIB file, configuration parameters dictating the utilisation of verification algorithms, and configuration parameters governing the output log to be generated.

Figure 39 illustrates the operational framework of NNVer. The *ONNX Model*, supplied in a .onnx file format, undergoes conversion by the *ONNX Converter* to adapt it into an internal representation compatible with the *pyNeVer Verifier*. Subsequently, the model is processed by the *Classifier Extractor*, which analyses it and, if necessary, isolates the sub-components corresponding to the classifier. Simultaneously, the corresponding *VNNLIB Property*, provided as a .smtlib file, is converted by the *VNNLIB Converter* into an internal representation compatible with the *pyNeVer Verifier*, contingent upon its compliance with the VNNLIB standard. Both the model and the property are then submitted to the *pyNeVer Verifier*, which, guided by the *Configuration Parameters*, selects an appropriate verification strategy and commences formal verification to assess the model's adherence to the property of interest. The outcomes of this verification process are sent to the *Log Writer*, which, based on the provided *Configuration Parameters*, records pertinent information in a corresponding *Output Log*.

3.29.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
NN verification: It provides property verification in Neural Networks (NN)	Implementation Level: Implemented Estimated Delivery Date: MS7 (M37) Licence: "Commons Clause" License Condition v1.0 Deviation: Deviations occurred due to the complexity of the verification problem (at best NPcomplete), demanding extra personnel efforts. Mitigation involved	NNVer enhances NNs reliability for safety critical CPSs by providing property verification. Introducing "step-zero object detection" networks alongside intricate NNs, NNVer simplifies verification, prioritising object classification. Formal

	intensified collaboration, resource allocation, and alternative approaches to streamline verification. These actions aimed to avoid delays and ensure progress.	certification of step-zero models ensures safety by alerting to object presence even with unclear exact locations.
--	---	--

Table 57 Capabilities Implementation Status of the UNISS_SOL_02 Solution

3.29.3 Useful Resources

- Guidotti, D., Pandolfo, L., & Pulina, L. (2024, to be published). Formal Verification of Neural Networks: A "Step Zero" Approach for Vehicle Detection. In 2024 37th International Conference on Industrial, Engineering & Other Applications of Applied Intelligent Systems (IEA/AIE).
- Guidotti, D., Pandolfo, L., & Pulina, L. (2023, November). Verifying Neural Networks with Non-Linear SMT Solvers: A Short Status Report. In 2023 IEEE 35th International Conference on Tools with Artificial Intelligence (ICTAI) (pp. 423-428). IEEE.
- Guidotti, D., Pandolfo, L., & Pulina, L. (2023, October). Verifying Neural Networks with SMT: An Experimental Evaluation. In 2023 IEEE 19th International Conference on e-Science (e-Science) (pp. 1-2). IEEE.
- Eramo, R., Fanni, T., Guidotti, D., Pandolfo, L., Pulina, L., & Zedda, K. (2022). Verification of Neural Networks: Challenges and Perspectives in the AIDOaRt Project.

Also, Interested users can visit our GitHub page (<https://github.com/AIMet-Lab/AIDOaRt-UNISS-NNVer>) for access to the codebase, documentation, and any related materials. We are releasing some video tutorials offering step-by-step guidance on using the tool effectively.

3.30 Solution - UNISS_SOL_03 (UNISS) - ReqT

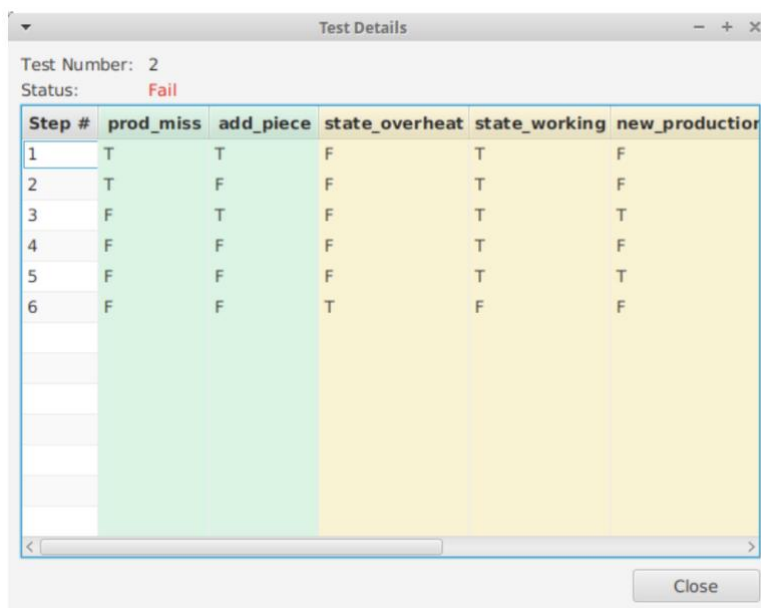
3.30.1 Overview

Solution: ReqT		Partner: UNISS
Current TRL: 3-4	Licence: General Public Licence (GPL)	Contacts: lpulina@uniss.it
Online Resource: https://github.com/AIMet-Lab/AIDOaRt-UNISS-ReqT		
Description: ReqT is a tool for requirements-based test suites generation. The key feature of this tool is to provide automatic requirement-based test generation, execution and evaluation of black-box reactive systems. As input, it takes a set of verified requirements encoded in Linear Temporal Logic (LTL), while the output consists in a list of traces (i.e., sequences of inputs and outputs assignments) executed on the System Under Test (SUT) and the corresponding evaluation of each test (passed/failed).		
Improvement: ReqT was previously developed as part of the European projects CERBERO and FitOptiVis, achieving a TRL 3-4. The intention was to extend the framework within the scope of the AIDOaRt project, aiming to enhance it with additional test oracles, exploration strategies, and increased expressiveness in the input language through the addition of numerical constraints. However, research efforts in this area did not yield the expected results. Furthermore, challenges		

such as unsuccessful personnel recruitment hindered progress and prevented the generation of new outcomes to present.	
Intended Users: Software developers without any knowledge of formal methods and logical languages.	
Satisfied Requirement:	
<ul style="list-style-type: none"> GR Mod 02 – Use formal models, automated reasoning and/or ML techniques for test generation. GR Test 01 – AI/ML techniques for test case generation from high level models. GR Test 04 – Evaluation of testing results. GR Test 05 – Report or fix the problems detected in the testing phase. 	
Satisfied Use Case Requirement:	
<ul style="list-style-type: none"> ABI_R03 – Use automated reasoning and ML techniques for generation of optimal test suites. 	
AI-Augmented tool set components implemented: <ul style="list-style-type: none"> <u>AI for Testing</u> - It enables the employment of AI/ML techniques to support the testing phase of the system development. 	Task: T4.2 for the improvements of the solution.
Use within AIDOaRt Use Cases	Already planned use: ABI Potential foreseen links: Adaptable to various use cases without specificity to any particular one.
Use within AIDOaRt challenges: There are no challenges to report.	

Table 58 Synopsis table of the UNISS_SOL_03 Solution - ReqT

ReqT provides a Graphical User Interface (GUI) – see Figure 40 – to perform automatic testing with respect to a formal specification. The user needs to indicate the specifications to use and the system to test, plus few optional parameters to set the algorithm.



Step #	prod_miss	add_piece	state_overheat	state_working	new_production
1	T	T	F	T	F
2	T	F	F	T	F
3	F	T	F	T	T
4	F	F	F	T	F
5	F	F	F	T	T
6	F	F	T	F	F

Figure 40 An example of the ReqT GUI during the execution of testing

Testing Framework. In order to test black-box systems, ReqT adopts the framework presented below. The System Under Test (SUT) is the system we want to test, and ReqT interacts with it during execution, probing some inputs and evaluating the produced output. The tests are generated starting from a formal specification. In particular, we assume that the specification is composed of a list of LTL formulas or PSPs requirements, plus the declaration of the set input and output properties. To write and verify the correctness of requirements you can use ReqH and ReqV.

The main components of the framework are represented in Figure 41 and described in the following:

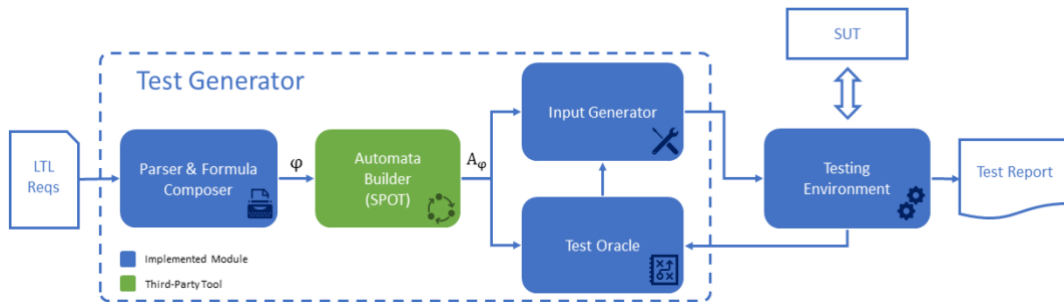


Figure 41 General Operational Framework for ReqT

- **Parser.** It reads the input specification, creates the intermediate data structures and builds the conjunction of requirements.
- **Automata Builder.** It builds a Büchi or equivalent automaton representation of the input specification.
- **Input Generator.** It chooses which inputs to execute on the SUT.
- **Test Oracle.** It evaluates the output produced by the SUT and checks if it satisfies the specifications.
- **Testing Environment.** It is responsible for orchestrating the interaction between the different components. It queries the Input Generator for new tests and executes them on the SUT. It collects the output and passes it to Oracle for evaluation. If the test is complete, it stores the final verdict and resets the environment to start a new test.
- **System Under Test.** ReqT has to communicate with the System Under Test (SUT) in order to collect outputs and evaluate the tests status.

3.30.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
Test generation: The service provides automatic test suites generation	Implementation Level: Implemented Estimated Delivery Date: MS7 (M37) Licence: General Public Licence (GPL) Deviation: Despite initial ambitions to improve ReqT within the AIDOaRt project, several challenges hindered progress and impeded the realisation of the intended improvements. One obstacle was the unsuccessful	ReqT simplifies test setup by specifying the formal specification and target system. Its Testing Framework, comprising components like Parser, Automata Builder, and Test Oracle, automates test generation and evaluation against specifications, ensuring

	<p>recruitment of personnel, which likely resulted in a lack of necessary expertise and resources required to drive the project forward effectively. Additionally, the complexity of integrating new features such as test oracles and exploration strategies into the existing framework may have proven more formidable than initially anticipated. Consequently, these factors prevent the generation of desired outcomes within the project's scope.</p>	<p>comprehensive coverage. ReqT seamlessly interacts with the System Under Test to collect outputs and assess test status, streamlining the testing process.</p>
--	--	--

Table 59 Capabilities Implementation Status of the UNISS_SOL_03 Solution - ReqT

3.30.3 Useful Resources

- Narizzano, M., Pulina, L., Tacchella, A., & Vuotto, S. (2020). Automated requirements-based testing of black-box reactive systems. In NASA Formal Methods: 12th International Symposium, NFM 2020, Moffett Field, CA, USA, May 11–15, 2020, Proceedings 12 (pp. 153-169). Springer International Publishing.
- Video Presentation: <https://www.youtube.com/watch?v=yH0jMBJ9Y0I&t=210s>.

3.31 Solution - UNISS_SOL_04 (UNISS) - OSPCheck

3.31.1 Overview

Solution: OSPCheck		Partner: UNISS	
Current TRL: 3	Licence: “Commons Clause” License Condition v1.0	Contacts: lpandolfo@uniss.it dguidotti@uniss.it	
Online Resource: https://github.com/AIMet-Lab/AIDOaRt-UNISS-OSPCheck		Collaborators: Tiziana Fanni (ABI)	
Description: OSPCheck (Ontology SMT Properties Checker) is a tool designed to verify the consistency of properties within a given system model by using formal methods. This tool permits translating system model properties, expressed in OWL ontology language, to Satisfiability Modulo Theories (SMT) format. The output of the encoding is then used by SAT solvers to obtain all possible satisfying solutions.			
Improvement: OSPCheck has been developed from scratch during the AIDOaRt project. Our focus in Y2 was primarily on modelling the properties of the ABI use case. During the current period, significant progress has been made with the development of the encoder component.			
Intended Users: This tool supports designers and architect engineers of safety critical CPSs, enabling them to formally verify the consistency of a model's properties.			
Satisfied Requirements: <ul style="list-style-type: none"> • GR Mod 01 – Use AI techniques for verification of specifications and high-level models. • GR Mon 2.1 – Monitoring in AIDOaRt could be done with the purpose of identifying violations to pre-defined requirements (e.g. above 85% resource usage). 			

Satisfied Use Case Requirements: <ul style="list-style-type: none"> ABI_R01 – Use automated reasoning and ML techniques for verification of specifications and high-level models. ABI_R05 – Use of automatic tools for compliance verification. 	
AI-Augmented tool set components implemented: <ul style="list-style-type: none"> <u>AI for Requirement</u> - It supports consistency checking of specifications at the early design stage. <u>Engagement and Analysis</u> - It employs AI/ML to support the verification of system model properties. 	Task: T4.2 and T4.3 for the improvements and developments of the solution. T4.4 for its application in the context of practical use case scenarios.
Use within AIDOaRt Use Cases	Already planned use: ABI Potential foreseen links: There are currently none specifically identified.
Use within AIDOaRt challenges: <ul style="list-style-type: none"> Modelling properties constraints of the ABI UC – 1° Hackathon (Virtual). Modelling properties constraints of the ABI UC (continuation) – 4° Hackathon (Vasteras). 	

Table 60 Synopsis table of the UNISS_SOL_04 Solution .- OSPCheck

The development of OSPCheck involved a technical process aimed at ensuring the consistency checking of the ABI use case model specifications, which represents an example of safety-critical CPS. Traditional system modelling approaches, such as those based on UML notations, typically provide well-defined syntax but lack precise semantics, hindering formal verification efforts. This ambiguity in semantics undermines the effectiveness of formal verification techniques, necessitating a more rigorous and formal approach. To address these challenges, OSPCheck leveraged an ontology-based approach as an alternative to traditional system modelling and property specification, since ontology offers a structured representation with clear and unambiguous semantics. This allows for more rigorous formal verification of system specifications, enhancing the reliability and safety of the resulting models.

In the following, we provide detailed insights into the development of OSPCheck by highlighting technical aspects and functionality.

Figure 42 depicts the general framework for OSPCheck. **Input.** OSPCheck accepts system model properties expressed in Web Ontology Language (OWL). OWL is a widely used ontology language for representing knowledge about a domain and is suitable for expressing complex relationships and properties within a system model. Unlike domain-specific languages or other notations, ontology easily allows users to model their system properties, as there are different tools available, such as OWLGrEd, that enable graphical development of ontologies without requiring expertise in formal languages.

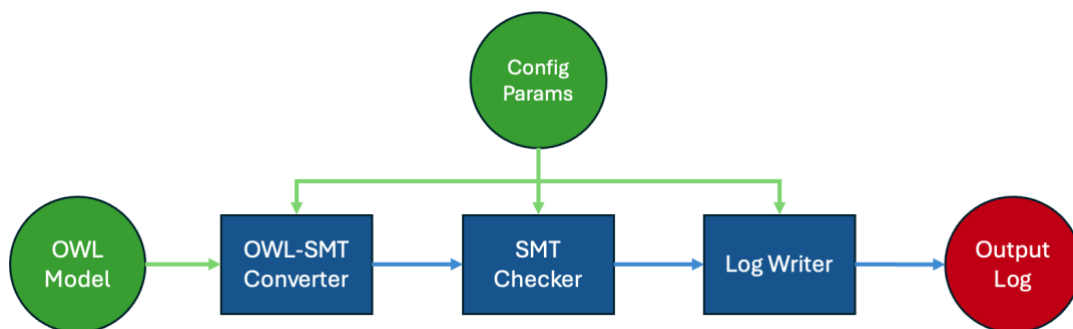


Figure 42 General Operational Framework for OSPCheck

OSPCheck includes a component responsible for translating system model properties from OWL to SMT-Lib format, namely *OWL-SMT Converter*. SMT-Lib is a standardised format for describing problems in the domain of SMT, providing a common interface for interacting with several solvers. During the encoding process, OSPCheck translates the OWL ontology-based representation of system model properties into an instance of SMT-Lib. This encoding ensures that the properties are represented in a form that can be efficiently processed by SAT solvers. Subsequently, the encoded system model properties are then passed to the *SMT Checker* module for executing the analysis. During this step, SAT solvers analyse the encoded system model properties and attempt to find satisfying solutions that meet the specified constraints. OSPCheck leverages the capabilities of SAT solvers to obtain all possible satisfying solutions for the given system model properties. During these steps, the tool utilises the *Config Params*. OSPCheck includes feedback mechanisms to provide users with insights into the consistency checking process, collected within the *LogWriter* module. Finally, the results of this consistency checking process are then passed to the *Output Log*, which reports the consistency status of the system model properties and details of any inconsistencies detected.

3.31.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
Property verification: It provides property verification for checking whether a model of a system meets given specifications. In case of inconsistencies, the solution will send feedback to the designer	Implementation Level: Partially Implemented Estimated Delivery Date: MS7 (M37) Licence: “Commons Clause” License Condition v1.0 Deviation: No deviations to report. The solution is in the final stage of its development and will be ready to be tested before the end of the project.	As reported earlier, this tool has been developed from scratch during the AIDOaRt project. Currently, the tool supports the ABI model. We are working on generalising the tool to be able to handle any OWL ontology expressed in the OWL 2 DL language.

Table 61 Capabilities Implementation Status of the UNISS_SOL_04 Solution - OSPCheck

3.31.3 Useful Resources

- Pandolfo, L., Pulina, L., & Vuotto, S. (2021). SMT-based consistency checking of configuration-based components specifications. IEEE Access, 9, 83718-83726.

There are no other specific resources directly associated with the OSPCheck, however interested users can visit our GitHub page (<https://github.com/AIMet-Lab/AIDOaRt-UNISS-OSPCheck>) for access to the codebase, documentation, and any related materials. We are actively working on preparing a publication detailing the tool’s capabilities and implementation, which will provide comprehensive insights into its usage and benefits. Furthermore, we have plans to release some video tutorials by the end of the project, offering step-by-step guidance on using the tool.

3.32 Solution - HEPYCODE (UNIVAQ)

3.32.1 Overview

Solution: HEPYCODE		Partner: UNIVAQ
Current TRL: 4	License: GPLv3	Contacts: vittoriano.muttillo@guest.univaq.it , luigi.pomante@univaq.it , marco.santic@guest.univaq.it
Online Resource: HEPYCODE AIDOaRt Repository: https://github.com/hepsycode/HEPSYCODE-AIDOaRt HEPYCODE Official Website: www.hepsycode.com		Collaborators: Luca Berardinelli (JKU), Riccardo Rubei (UNIVAQ), Claudio Di Sipio (UNIVAQ), Abbas Rahimi (JKU)
<p>Description: HEPYCODE (ESL HW /SW CO-DEsign of HETerogeneous Parallel Dedicated SYstems) is a tool designed to shorten the development time for embedded applications.</p> <p>Main functionalities: 1) AI/ML for prediction and evaluation/estimating metrics such as performance and energy consumption. 2) Design Space Exploration (DSE) of different design options. 3) Design methodologies are needed from different industrial domains such as automotive, aerospace, and smart cities. HEPYCODE key features implemented are:</p> <ul style="list-style-type: none"> ● System-level design space exploration: HEPYCODE helps define the hardware/software (HW/SW) partitioning and map them to a suitable heterogeneous multiprocessor architecture. Furthermore, HEPYCODE proposes feasible solutions based on system requirements and application models. ● AI/ML integration: HEPYCODE utilises AI/ML for modelling, performance and energy consumption simulations and predictions, and design space exploration throughout the whole design process. ● Metrics and data management: HEPYCODE supports defining metrics, extracting data (through simulation or real-world data), and managing data for AI/ML activities. ● Improved implementation and accuracy: HEPYCODE leverages AI/ML to find alternative architectural solutions and reduce simulation errors and execution time while achieving better accuracy. 		
<p>Improvement: HEPYCODE has successfully transitioned from TRL 2 to TRL 4 by integrating AI/ML techniques into its HW/SW co-design flow. A thorough evaluation of various AI/ML algorithms was conducted to identify those most suitable for the specific activity within the design flow. Improvements compared to the initial status included algorithm accuracy, computational complexity, and compatibility with existing HEPYCODE tool and framework. Respect to Y2, selected AI/ML algorithms were fully integrated into the HEPYCODE toolset at various stages of the HW/SW co-design flow. All the implemented features are fully integrated into a single, user-friendly environment. This provides a cohesive and efficient platform for HW/SW co-design, eliminating the need to switch between different tools. Finally, HEPYCODE is available as open-source software on a public GitHub repository. This makes it freely accessible to researchers and developers, promoting collaboration and innovation in the field of embedded systems and CPSs design.</p>		
<p>Intended Users: Embedded systems designers and CPS designers, Industrial practitioners (i.e., TEK)</p>		
<p>Satisfied Generic Requirement:</p> <ul style="list-style-type: none"> ● GR Mod 04: Use semi-automatic model synthesis for design- and run-time verification 		

<ul style="list-style-type: none"> GR Cod 03: The AIDoArt Framework imports/exports variables/procedures between code and models. <p>Satisfied Use Case Requirements:</p> <ul style="list-style-type: none"> TEK_R_101: The AIDoArt Framework verifies, in a semi-automatic manner, at design time, with respect to the requirements, the coverage of the architectural models. TEK_R_102: The AIDoArt Framework verifies in a semi-automatic manner, at design time, with respect to the requirements, the adequacy (the response versus the resources) of the real components on-which/with-which the system architect has in mind to map/realise the architecture. TEK_R_103: The AIDoArt Framework synthesises in a semi-automatic manner the models needed for the verification at design time (the models that define both the tests and the results, i.e. the pass/fail criteria). TEK_R_104: The AIDoArt Framework interprets in a semi-automatic manner the results of the design time verification. TEK_R_001: The AIDoArt Framework imports/exports variables/procedures between code and models. 	
<p>AI-Augmented tool set components implemented:</p> <ul style="list-style-type: none"> Data Handling Capabilities: Integration of a well-known HW/SW Co-Design methodology able to capture system behaviours from input data set. Model-Based Capabilities: MDE approaches and principles applied to HW/SW Co-Design methodologies. Ingestion & Handling: Data metrics definition and data analysis and selection. Engagement & Analysis: Design Space Exploration activity to find alternative patterns using metrics, data and AI/ML algorithms useful for system improvements. AI for Code: Automatic code generation and AI/ML analysis starting from DSL/UML models. AI for Modeling: XES models generated from <i>Modeling Process Mining</i> tool (JKU) are used by <i>MORGAN</i> recommender system (UNIVAQ) to support automated modelling routines. 	<p>Task: T4.1 and T4.2 for the integration and development of the AI/ML solutions. T4.4 for its application in the context of practical use case scenarios (i.e., TEKNE TEK_UCS_01).</p>
<p>Use within AIDoArt Use Cases</p>	<p>Already planned use: Design choices verification — Before the implementation; on models and/or with rapid prototyping; environment: simulated/emulated (TEK_UCS_01).</p>
	<p>Potential foreseen links: We see possible connections with the <i>MORGAN</i> tool from UNIVAQ and the <i>Modeling Process Mining</i> tool from JKU.</p>
<p>Use within AIDoArt challenges: Design choices exploration/verification - From 1° to 5° Hackathon</p>	

Table 62 Synopsis table of the HEPHYCODE Solution

HEPSYCODE's development and implementation encompass a wide range of capabilities, addressing key challenges in embedded systems design, as presented below.

Model-Driven DevOps approach for HW/SW Co-Design: HEPHYCODE provides a unified environment for HW/SW co-design, supporting high-level abstraction to focus on system functionalities. This

approach streamlines the design workflow and facilitates rapid prototyping and exploration of different design options. HEPSYCODE leverages model-driven engineering principles for design reuse and quality improvement. AI/ML algorithms provide intelligent support during modelling, including automated model generation, verification, and optimization. External tools such as *MORGAN (UNIVAQ)* and *Modeling Process Mining (JKU)* have been integrated to enhance continuous modelling capabilities.

Performance Simulation and Predictions: HEPSYCODE employs ML techniques to predict performance and power consumption, reducing simulation time while maintaining high accuracy. Comprehensive analysis were conducted to compare the performance of ML algorithms for predicting various performance metrics on different processors and a small subset of results are summarised in Figure 43.

Processor	Time	Target	DT	DT	RF	RF	SVM	SVM	Best
			Error (%)	Train (s)	Error (%)	Train (s)	Error (%)	Train (s)	
ARMV4T	SVM	Clock Cycles	0.45%	6.47	1.01%	92.81	18.40%	349.8	DT/RF
ARMV6-M	SVM	Clock Cycles	0.45%	7.22	2.50%	125.52	21.73%	641.57	DT
Leon3	SVM	Clock Cycles	1.77%	6.28	1.02%	91	13.68%	502.24	RF
Atmega328p	DT	Clock Cycles	19.23%	7.43	9.96%	137.9	99.70%	339.66	RF

Figure 43 HEPSYCODE error percentage of the ML algorithm for processor performance estimations. DT: Decision Tree, RF: Random Forest, SVM: Support Vector Machine, NN: Neural Network

The subset analysis in Figure 43 shown DF in achieving both high accuracy and reasonable training and prediction times for most target processor architectures. RF also demonstrated promising results. SVM offered slowest training time but faster prediction speeds, while its accuracy was consistently lower than DT and RF, rendering SVM unsuitable for most prediction tasks. NN, though not included in Figure 43, perform significantly worse than the other algorithms evaluated because it depends on the parameters and datasets. Additional tests will be conducted in future.

Design Space Alternatives Exploration: During AIDOaRt, UNIVAQ analysed and modified HEPSYCODE's existing tools for design space exploration. These tools search for feasible architectural and mapping elements that satisfy imposed constraints. The goal was to improve their performance and speed up the exploration process. An evolutionary algorithm called Non-Dominated Sorting Genetic Algorithm II (NSGA-II) was integrated into the toolset. NSGA-II utilises a Pareto-based approach to solve multi-objective optimization problems, allowing for more efficient exploration of the design space. To further accelerate the Design Space Exploration phase while maintaining acceptable accuracy levels, a tool was developed that utilises machine learning models to extract potential solutions belonging to the Pareto front. The selected ML algorithms are: 1) Multi-label Support Vector Machine with polynomial kernel; 2) Standard multi-label Random Forest classifier; 3) Random Forest classifier with 500 trees and a maximum of 16 leaves; 4) Neural network with 3 layers: 24 nodes in the first layer, 12 nodes in the second, and the 3 nodes in the third. The tool architectures developed during the AIDOaRt project are presented in Figure 44.

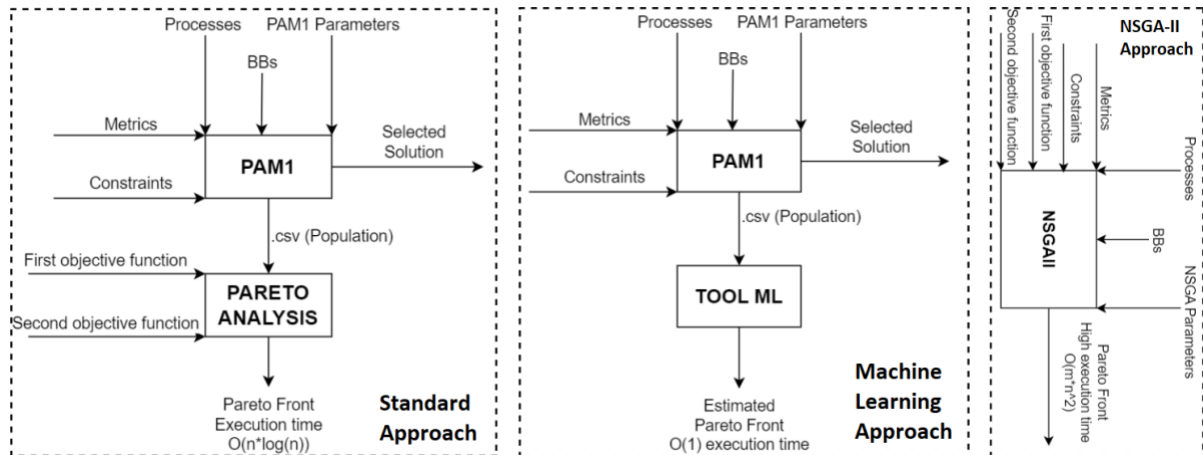


Figure 44 HEPHYCODE Design Space Exploration Capabilities

These approaches significantly reduced the computational time required for DSE, while the performance of the implemented tools was evaluated using benchmark applications from the literature. The results demonstrated significant improvements in terms of both speed and accuracy compared to the original HEPHYCODE tool, as shown in Figure 45.

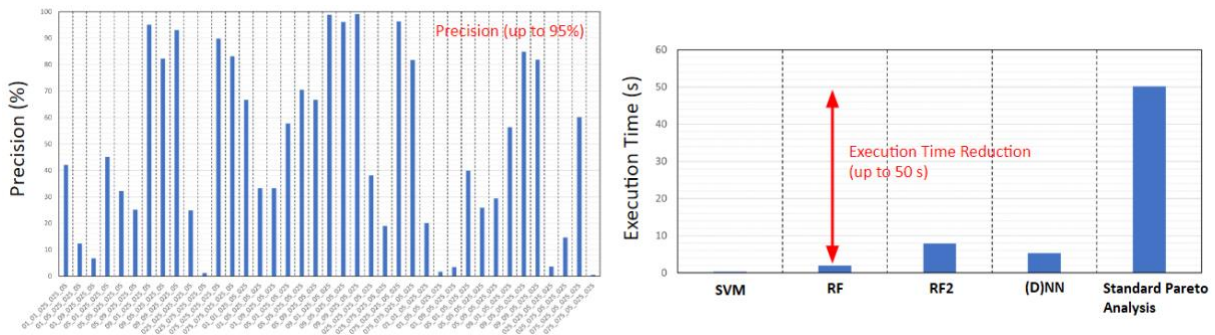


Figure 45 HEPHYCODE Improvement Results

3.32.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
Performance Simulation and Predictions: HEPHYCODE Design Flow techniques (from high-level representation to the low-level implementation), considering AI/ML techniques for simulations and predictions	Implementation Level: Implemented Estimated Delivery Date: MS7 (M37) License: GPL v3 Deviation: No deviations to report.	AI/ML algorithms were used to predict the performance and energy consumption of various design solutions, reducing simulation errors and improving accuracy.
Design Space Alternatives Exploration: HEPHYCODE Design Flow techniques (from high-level	Implementation Level: Implemented Estimated Delivery Date: MS7 (M37)	AI/ML algorithms were employed to identify (pareto) sub-optimal hardware and software components for each

representation to the low-level implementation), considering AI/ML techniques for design space exploration	License: GPL v3 Deviation: No deviations to report.	task. ML-based techniques were also used to fine-tune design space parameters for further performance and energy improvements.
Model-Driven DevOps approach for HW/SW Co-Design: HEPSYCODE Design Flow techniques (from high-level representation to the low-level implementation), considering DevOps and Standard MDE principle into an automatic co-design framework	Implementation Level: Implemented Estimated Delivery Date: MS7 (M37) License: GPL v3 Deviation: No deviations to report.	HEPSYCODE incorporates external tools for tracing, monitoring, and suggests appropriate modelling actions at runtime. This enables detailed tracking of design progress, providing valuable insights into the modelling process, and continuously helping designers make informed decisions. HEPSYCODE also provides an open-source environment that is freely accessible and encourages high collaboration through Git version control systems.

Table 63 Capabilities Implementation Status of the HEPSYCODE Solution

3.32.3 Useful Resources

Official HEPSYCODE WebSite: <https://www.hepsycode.com/>

Official AIDoArT HEPSYCODE Repository: <https://github.com/hepsycode/HEPSYCODE-AIDoArT>

Publications:

- V. Muttillio, L. Pomante, M. Santic, G. Valente, SystemC-based Co-Simulation/Analysis for System-Level Hardware/Software Co-Design, Computers and Electrical Engineering, Volume 110, 2023.

3.33 Solution - MORGAN (UNIVAQ)

3.33.1 Overview

Solution: MORGAN		Partner: UNIVAQ
Current TRL: 4	License: Apache 2.0	Contacts: Claudio Di Sipio claudio.disipio@univaq.it
Online Resource: https://github.com/MDEGroup/MORGAN		Collaborators: Abbas Raimi, Luca Berardinelli, MohammadHadi Dehghan (JKU)
Description: MORGAN is a recommender system that supports modelling operations by using graph kernels. Given the model under construction, the system encodes the models as graphs and suggests the most similar classes and structural features, i.e., attributes and relationships.		

<p>Improvement: In the context of the project, we employ MORGAN to automatize the specification of SysML model in the VCE use case. In particular, we combined our solution with the MER tool provided by JKU solution provider. In particular, MER focuses on retrieving user’s operations captured during the modelling activity and encodes them as traces. Afterwards, MORGAN is trained with those traces to recommend the next operation or similar ones. Furthermore, we are integrating the two tools in a Docker container, thus facilitating the usage for non-expert users. In particular, the MORGAN environment will be provided as a container to communicate with MER in order to ingest traces and provide recommendations.</p>		
<p>Intended Users: Modelers, industrial practitioners, software architects</p>		
<p>Satisfied Requirement:</p> <ul style="list-style-type: none"> ● GR Mod 01 – Use AI techniques for verification of specifications and high-level models. <p>Satisfied Use Case Requirement:</p> <ul style="list-style-type: none"> ● VCE_RO1 – Use automated reasoning and ML techniques for verification of specification and high-level models 		
<p>AI-Augmented tool set components implemented:</p> <ul style="list-style-type: none"> ● AI for Modeling: MORGAN will provide relevant model operations that can be used to complete the model under construction. In particular, it can support operations described in the XES metamodel provided by JKU using a graph kernel as an engine. ● Ingestion and Handling: Given the input model, MORGAN employs a tailored parser to extract structural features. In particular the tool supports the data handling for the XES traces produced by MER tool. ● Engagement and analysis: MORGAN encodes relevant features using a set of standard NLP techniques i.e. stemming, dash removal. Afterward, this data is encoded and used to produce a list of graphs representing the model elements and their corresponding structural features. 		<p>Task: T4.1, T4.2, T4.3</p>
<p>Use within AIDOaRt Use Cases</p>	<p>Already planned use: VCE_UCS_01: In the context of VCE use case, MORGAN can help industrial practitioners by providing relevant SysML operations given the model under construction. To this end, we combine MORGAN capabilities with the MER tool provided by JKU solution provider.</p>	
	<p>Potential foreseen links: We see possible connection with HEPHYCODE tool and the TEKNE use case “Agile process and Electric/Electronic Architecture of a vehicle for professional applications.”</p>	
<p>Use within AIDOaRt challenges:</p> <ul style="list-style-type: none"> ● CH4 - Modeling patterns for AI enhanced architecture modeling (2nd Hackathon) ● CH3 - Modeling recommendations and process mining for system architecture descriptions (3rd Hackathon) ● CH1 - VCE-Hackathon-Integrating modeling assistant to design CPSs (4th Hackathon) ● CH1 - Combined architecture modelling and analysis in the eclipse tooling framework (5th Hackathon) 		

Table 64 Synopsis table of the MORGAN Solution

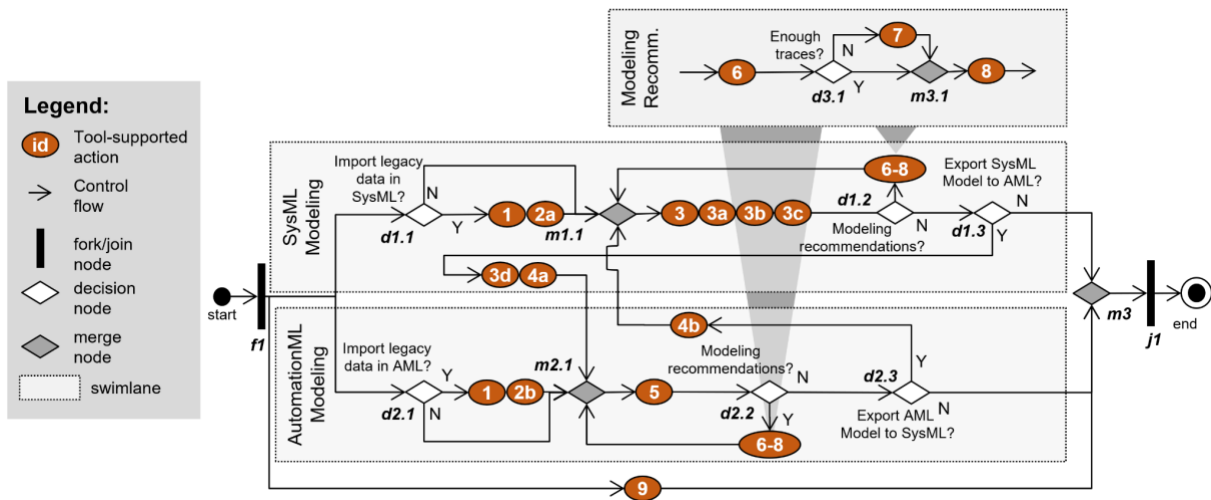


Figure 46 MORGAN Solution Architecture

The overall engineering workflow is depicted in Figure 46. In particular, MORGAN is trained by a dataset of XES traces compliant with the XES metamodel brought by the MER tool. MER can generate them (6) when VCE experts use SysML or AutomationML editors (3,5) or generated by EMF-compliant model generators (7). Due to the lack of large-scale dataset, we exploited VIATRA model generator techniques to obtain an initial dataset composed of a decent amount of XES traces if real traces are missing. In addition, we mimic the modeller's behaviour by collecting real XES traces with the Sirus-based editor equipped with the MER. In such a way, the MER is able to produce meaningful operations that can be used to feed the AI-based component.

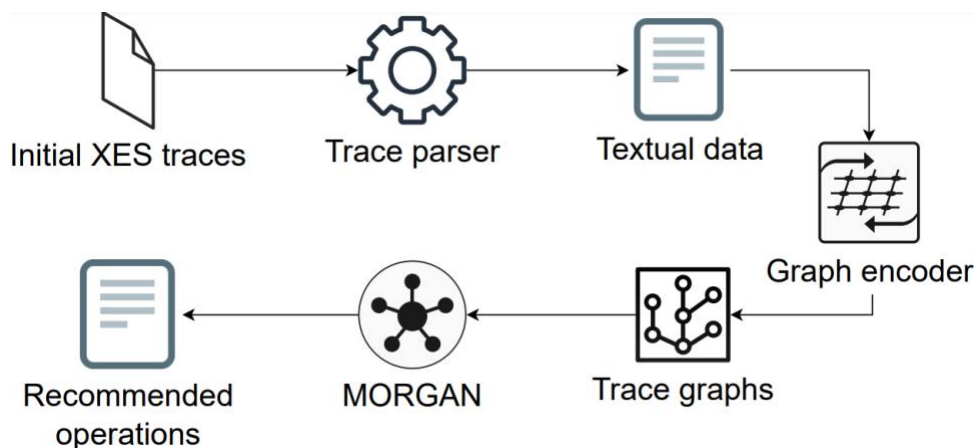


Figure 47 MORGAN Architecture

The overall architecture of MORGAN is depicted in Figure 47. In the scope of the project, we modify the MORGAN's architecture by introducing a trace parser to extract relevant information from XES traces. In such a way, we obtain a textual-based representation that is used by the Graph-encoder to produce a list of trace graphs. To this end, the encoder extracts different features for each event, i.e., the type of event and the affected artefacts. Each graph is constructed by employing a set of Natural Language Processing (NLP) techniques, i.e., stemming, and dash-removal. Afterward, we reuse the

same underpinning algorithm presented in our previous work, i.e., the Graph Kernel. In particular, we assess the graph similarity between the XES graphs using the Weisfeiler-Lehman algorithm, provided by the Grakel Python library. The outcomes of the component are the ranked list of similar operations given the context of the modeller, i.e., the initial XES trace. We eventually integrated MER and MORGAN solutions in a Docker environment, thus making the whole tool deployable in any execution environment.

3.33.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status
Recommender of model and metamodel elements: MORGAN is an approach based on graph kernels to support the specification of model and metamodel elements	Implementation Level: Implemented Estimated Delivery Date: MS7 (M37) License: Apache 2.0 Deviation: Due to the lack of real models, we generated one using VIATRA. Furthermore, we mimic the modeler's behaviour by collecting real XES traces with the Sirius-based editor equipped with the MER.

Table 65 Capabilities Implementation Status of the MORGAN Solution

3.33.3 Useful Resources

VCE-solution-framework <https://github.com/AIDOaRt-VCE-Team/Solution-framework>

MORGAN GitHub project: <https://github.com/claudioDsi/MORGAN-MER>

Publications:

- Di Sipio, C., Di Rocco, J., Di Ruscio, D., and Nguyen, P. T. (2023). MORGAN: A modeling recommender system based on graph kernel. *Software and Systems Modeling*, 22(5), 1427–1449. <https://doi.org/10.1007/s10270-023-01102-8>
- Cederbladh, J., Berardinelli, L., Bilic, D., Bruneliere, H., Cicchetti, A., Dehghani, M., Di Sipio, C., Miranda, J., Rahimi, A., and Rubei, R. (2024). Towards Automating Model-Based Systems Engineering in Industry—An Experience Report. Accepted at the 18th Annual IEEE International Systems Conference

3.34 Solution - TWIMO (UNITE)

3.34.1 Overview

Solution: TWIMO		Partner: UNITE
Current TRL: 4	License: GPL	Contacts: Romina Eramo (reramo@unite.it)
Online Resource: https://minds-group.github.io/TWIMO/		Collaborators: UNIVAQ, MDU, AVL
<p>Description: TWIMO provides a conceptual framework to define domain-specific notations, used for the definition of Human Driver Behavior models and ML-based services for its prediction or analysis. The tool also provides ML prediction methods (e.g., a Random Forest classifier was applied for the analysis and prediction of Human Driver Behavior in virtual environments).</p> <p>Improvement: The data model initially used (from the AVL RDE use case) has been extended by means of a complete metamodel language (i.e., domain-specific language) describing the Human Driver Behaviour domain. We extended the framework with a Digital Twin framework focusing on modelling the behaviour of considered physical systems to make predictions by using several ML techniques. We improved the initial implementation by refining the trained ML model. We have improved the TRL from 3 to 4.</p> <p>Intended Users: Modelers, industrial practitioners (e.g., AVL users), software architects</p> <p>Satisfied Requirement:</p> <ul style="list-style-type: none"> ● AVL RDE R01 - development of an ML model that, based on real driving recordings, is trained to simulate human-like driving given a target route, vehicle, and traffic conditions ● AVL RDE R02 - development of an AI method for providing better statistics of the environment based on the statistics of the real driving recording and data from digital map service ● AVL RDE R03 - automate multi-source data analysis of the real driving test data such that the relevant features of the driver behavior can be clustered (e.g. highway driving, low-speed driving, cornering, braking, acceleration, ...) ● AVL RDE R05 - analysis of basic driver attributes, including acceleration/deceleration histogram, breaking behavior, cornering behavior, and max speed preference. ● GR Mod 04 - use of semi-automatic model synthesis for design- and run-time verification ● GR Mon 05 - monitoring tool using human-defined parameters ● GR Mod 06 - use of AI-based methods for easy configuration 		
<p>AI-Augmented tool set components implemented: Engagement and Analysis: data in the form of driving cycles is analysed and transformed into a predictive behavioural model of human driving behaviour.</p> <ul style="list-style-type: none"> ● AI for modeling: suitable abstraction of the recorded data are defined, the method can be applied to other data sets to create models of different driver styles ● AI for testing: AI-based solutions for Real Driving Emissions testing are developed 		<p>Task: T4.2: AIDOaRt solutions for Engagement & Analysis</p>
Use within AIDOaRt Use Cases	Already planned use: AVL_RDE_UCS1, AVL_RDE_UCS3	
	Potential foreseen links:	
Use within AIDOaRt challenges: CH4 “Real Driver Emission” (1° Internat hackathon)		

Table 66 Synopsis table of the TWIMO Solution

TWIMO proposes both a conceptual modelling framework for Real Driving Prediction and a ML-based service for the prediction of Human Driver Behaviour.

TWIMO conceptual modelling framework: The TWIMO framework is a Digital Twin framework to define domain-specific notations, used for the definition of human driver behaviour and ML-based services (see Figure 48). It does not discuss particular tools or technologies, but it categorises the different roles and the relationships of artefacts on a conceptual level; in particular, it aims at designing components, classes and their relationships for the prediction of driver behaviour. The objective of this framework is to enable predictions through the analysis of historical data and updated data. It is composed of three main parts: i) the digital twin to reason over time about the spatial model (it is the end-users' entry point to manipulate the framework), ii) the spatial model and predictors configuration (part I of the figure below) and iii) the domain-specific classes and relations describing the driving behaviour metamodel language we considered to use the framework (part II - it depicts a fragment of the metamodel).

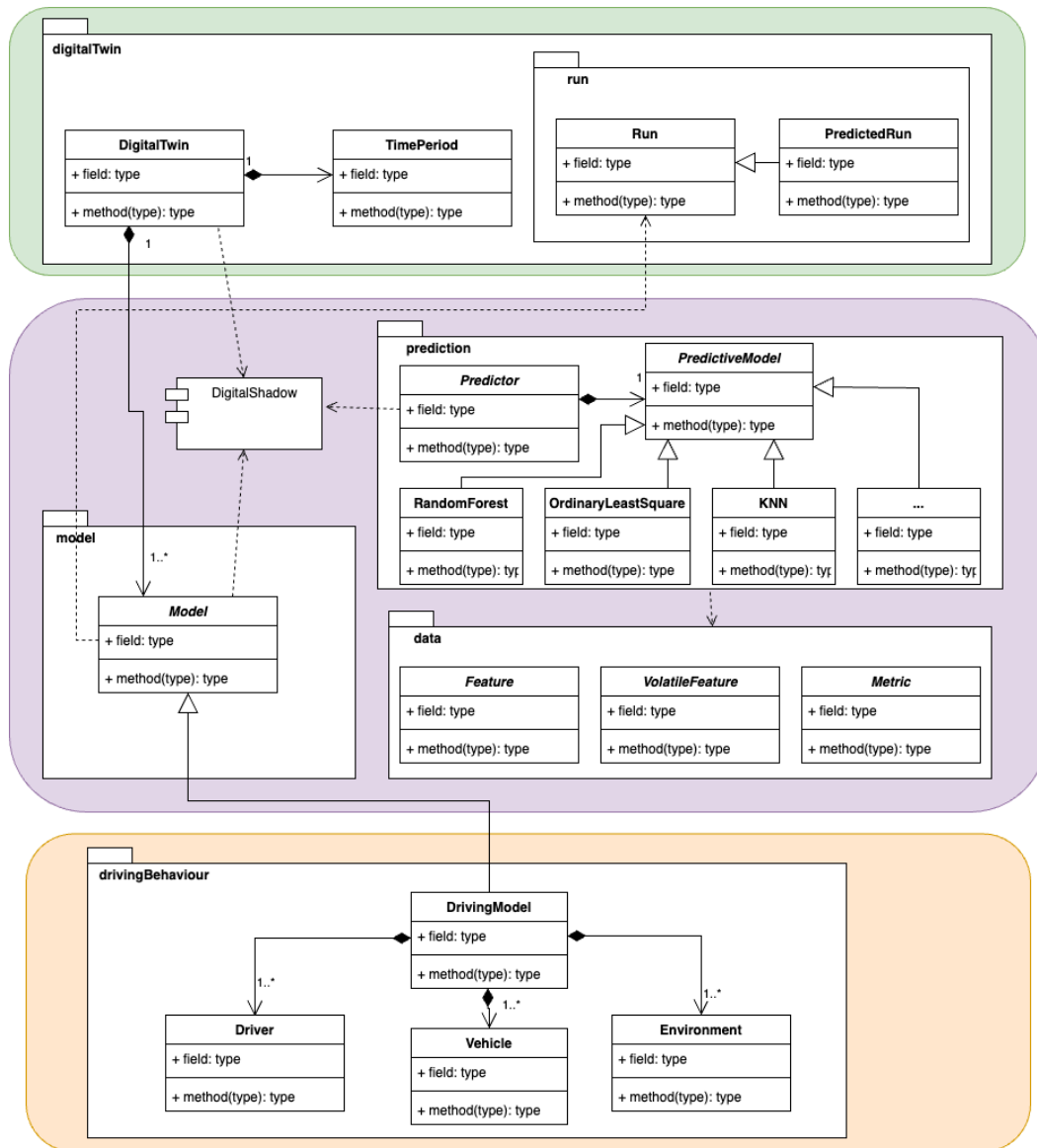


Figure 48 TWIMO Digital Twin conceptual framework

Figure 49 depicts an instance of the TWIMO framework for the AVL RDE use case. It is composed of two main parts: i) the spatial model and predictors configuration (part I of the figure below) and ii) the domain-specific classes and relations describing the driving behaviour metamodel language we considered to use the framework (part II).

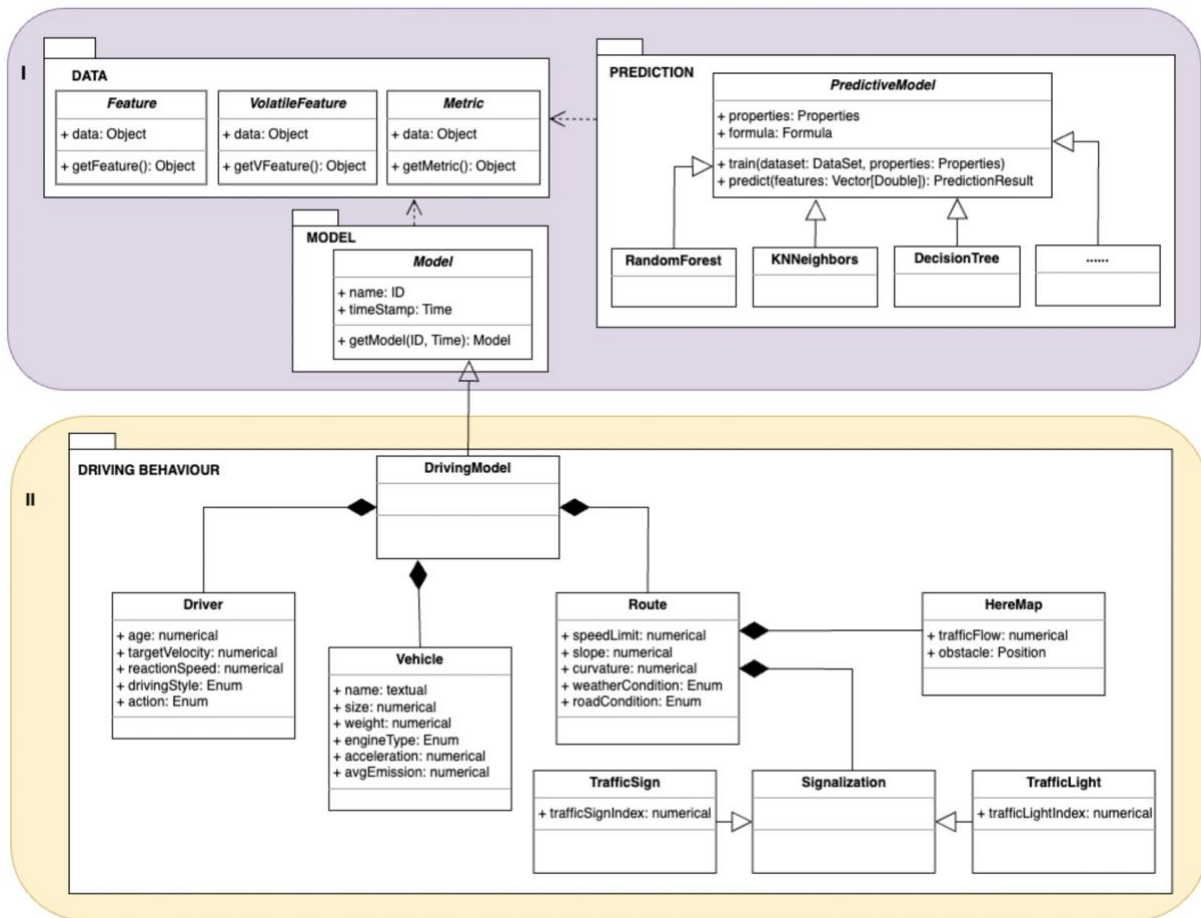


Figure 49 TWIMO framework instance for the AVL RDE UC

The proposed framework can be used for the prediction of Real Driving Behaviour by using different ML techniques (as illustrated in the Prediction components).

The proposed framework has been applied by UNITE (in collaboration with UNIVAQ and MDU) to the AVL-RDE use case, following the step below.

Data Acquisition: The data and their description that are used in this research work are provided by AVL company. The data was collected from special equipment placed on a car. Data refers to several drivers and different driving paths: highways, mountains, etc. The data collected is real driving recordings in time series format (time-based data on vehicle speed, throttle/brake pedals, curvature, road gradient, GPS coordinates, etc.).

Features Selection: To reduce the unnecessary computational cost, we filter out all the channels that are not relevant for modelling human driver behaviour. To that end, we took a close look at the channels in the dataset and found that there are three types of channels: 1. channels (resp. their corresponding data) are collected by two-way (snapshots and real driving recordings), 2. channels (volatile features like distance) are calculated from other features (speed and time), and 3. metric channels (such as speed).

Pre-processing: When we explored the features' data in the dataset, we found that each feature's data has a different scale. All features' data in the dataset are re-scaled using this standardisation except the velocity_kmh_raw feature as it represents the target feature or class.

Driver Behaviour Prediction: In the literature, there are many different ML algorithms for classification problems and there is no certain ML algorithm fit to all datasets. Choosing a ML algorithm depends on the type and size of the dataset of interest. In this research work, we compare the performance of mostly widely used ML algorithms using Python scikit-learn packages to determine which algorithm is the best fit for the collected dataset. These algorithms are GradientBoosting, DecisionTree, RandomForest, LogisticRegression, KNeighbors, GaussianNB, LinearSVM, and AdaBoost. The target algorithm should be able to predict the driver behaviour on the basis of the real dataset available and selected features on any arbitrary test route.

3.34.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
Domain-specific modeling:	Implementation Level: Implemented Estimated Delivery Date: MS7 (M37) License: GPL Deviation: No deviations to report.	A conceptual framework for modeling and prediction, including the "Human Driver Behaviour" metamodel, is proposed
MLAnalysis:	Implementation Level: Implemented Estimated Delivery Date: MS7 (M37) License: GPL Deviation: No deviations to report.	ML algorithms are used to predict the driver behaviour

Table 67 Capabilities Implementation Status of the TWIMO Solution

4 AIDOaRt AI-Augmented tool set Solutions in Use Cases

In this section, we provide an extensive overview of the current application of the solutions (referred to as tools) from the AIDOaRt AI-Augmented tool set in the context of specific use case challenges. The main objective is to provide a clear and detailed insight into how AIDOaRt solutions have been deployed in real-world situations, addressing the challenges of use cases and related CPSs.

To support the comprehension of this section, the following Table 68 summarises all the AIDOaRt case study titles, their corresponding IDs, and the related use case providers.

ID	Case Study	Provider
ABI_CS01	Safety critical systems in the automotive domain using disruption technology	ABI
AVL_CS02	AI supported Digital Twin Synthesis supporting secure vehicle development and testing for novel propulsion systems	AVL
ALSTOM_CS03	DevOps for Railway Propulsion System Design	ALSTOM
CAM_CS04	AI for Traffic Monitoring Systems	CAMEA
CSY_CS05	Machine learning in interactive proving	CSY
HIB_CS06	AI DevOps in the restaurants business	HIB
PRO_CS07	Smart Port Platform Monitoring (SPPM)	PRO
TEK_CS08	Agile process and Electric/Electronic Architecture of a vehicle for professional applications	TEK
VCE_CS09	Data modelling to support product development cost and efficiency	VCE
WMO_CS10	Automated continuous decision making in testing of robust and industrial-grade network equipment	WMO

Table 68 AIDOaRt Case Studies

4.1 Application in ABI Challenge – Formal Verification of NNs: A “Step Zero” Approach for Vehicle Detection – UNISS (UNISS_SOL_01 - NNVer)

Within the framework of the AIDOaRt project, the Abinsula Virtual Rear Mirror Case Study requires the development of neural networks. These networks aim to enhance an electronic rear-view mirror by providing informative alerts and suggestions, thereby augmenting driver situational awareness. Given the implicit safety-critical nature of these tasks, it is imperative that the behaviour of the models developed in this context can be certified through formal verification. Recognizing the complexity of current state-of-the-art *Object Detection* models and the limitations in scalability of existing verification methodologies, our focus in this challenge is related to the verification of models which serve as foundational building blocks, reinforcing the reliability of traditional object detection systems. To bridge the gap between the complexity of standard object detection neural networks and the need for formal verification, we propose a “step zero” object detection approach. This approach seeks to classify whether an image contains a vehicle, essentially serving as a foundational building block for bolstering the reliability of object detection systems. By shifting our focus to this preliminary stage, we aim to navigate the challenges posed by the immense complexity of object detection neural networks, thus enhancing their overall dependability. Instead of following the traditional complexity associated with object detection networks, our method centres on the fundamental task of classifying whether an image contains a vehicle. In Figure 50, it is a graphical representation of a generic CNN model, in

which we highlight where our verification task is focused. We verify the local robustness of the classifier. While it may not rival the object detection capabilities of more intricate networks, what sets our approach apart is its formal verifiability. This verification process can be leveraged to guarantee that our "step zero" networks, although potentially less powerful, adhere to stringent reliability standards, rendering them well-suited for safety-critical applications such as automotive systems. Consequently, we have chosen to train our models using the Vehicle Detection Image Set³, a dataset comprising 17,760 colour images measuring 64 by 64 pixels, which can be classified as either containing vehicles or not. For our experimental evaluation, we used NNs with identical architectures in the feature extraction phase. This initial part of the network leverages convolutional and pooling layers to extract essential numeric features from the input image. Specifically, we adopted a feature extractor composed of six blocks, each consisting of a convolutional layer, a max-pooling layer, and a ReLU layer. Following the feature extraction phase, our networks employed three distinct classifiers.

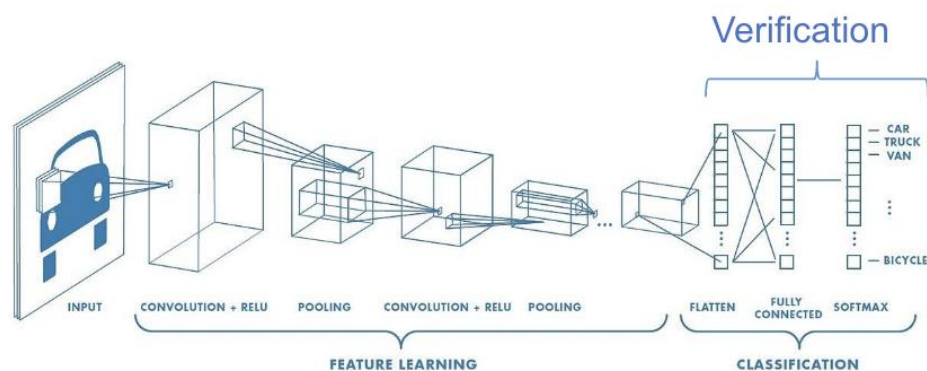


Figure 50 Image Source: Towards Data Science

Each classifier had two fully connected layers with ReLU activation functions, and a single fully connected output layer. The first classifier had 64 hidden neurons in the first layer and 32 in the second. The second had 128 in the first and 64 in the second, while the third had 256 in the first and 128 in the second. All three classifiers shared an output layer with two neurons for the classes of interest. The feature extractor's input was a matrix representing images (64x64x3 pixels), while the classifiers received vectors of 1024 numerical features. Model training utilised pyNeVer⁴, a tool based on PyTorch⁵, employing the Adam optimizer with a learning rate of 0.001 for 25 epochs. Testing and validation sets comprised 20% and 30% of the dataset, respectively, with batch sizes of 128 for training and 64 for validation. A scheduler decreased the learning rate by 0.5 after five epochs without reducing validation loss. The Cross-Entropy Loss function was used, and accuracy measured model performance in image classification. For further details on hyperparameters and training, refer to the PyTorch documentation.

³ Vehicle Detection Image Set: <https://www.kaggle.com/datasets/brsdincer/vehicle-detection-image-set>. Accessed: 2024-02-13.

⁴ <https://github.com/NeVerTools/pyNeVer/tree/main>

⁵ <https://pytorch.org/>

Model ID	Accuracy	Epsilon	Result	Time
VC_[64-32]	99.94%	0.001	FALSE	113.23
		0.01	FALSE	111.24
		0.1	FALSE	111.33
VC_[128-64]	99.92%	0.001	FALSE	192.84
		0.01	FALSE	193.54
		0.1	FALSE	210.28
VC_[256-128]	99.97%	0.001	FALSE	375.09
		0.01	FALSE	382.58
		0.1	FALSE	462.65

Table 69 Results of our experimental evaluation. Model ID represents the identifier assigned to a certain NN. Accuracy reports the percentage of samples classified correctly by each model on the test set. Epsilon represent the value for ϵ considered for the pro

The evaluation, covering accuracy and verifiability aspects, employed pyNeVer to successfully verify the robustness of classifiers against local adversarial perturbations. As shown in Table 1, all our models demonstrate a satisfactory level of accuracy. Interestingly, it is worth noting that accuracy does not appear to have a direct correlation with the complexity of the models. Regarding the verification of our models, when employing pyNeVer’s over-approximate algorithm, it successfully verifies the local robustness of our models across all considered epsilon values. However, it is evident that the verification time increases significantly with both the NN’s size and the property input space. In this context, pyNeVer managed to certify the robustness of our classifier components against adversarial perturbations.

4.2 Application in ABI Challenge – Adoption of AI and ML techniques for image processing in the automotive context – INT (INT-DET, INT-DEPTH)

Within the framework of the AIDOaRt project, Abinsula presents a Virtual Rear Mirror Case Study in which multiple cooperative cameras are used to capture the context outside the vehicle, by means of AI-based technology, providing informative alerts and suggestions, thereby augmenting driver situational awareness. In this context, integrating an object detection tool like INT-DET with a stereo depth estimation tool like INT-DEPTH enables the development of a guidance support system with robust perception capabilities, allowing for safer and more reliable navigation and decision-making in dynamic environments. How to achieve this integration? On one hand INT-DET can identify and localise various objects, such as pedestrians, vehicles, traffic signs, and obstacles, within the scene captured by the camera; on the other hand, INT-DEPTH provides depth information for each detected object, enabling the system to understand the spatial relationship and distance of objects relative to the vehicle or robot. By combining the object detection results with depth information, the guidance support system can make informed decisions regarding navigation, path planning, obstacle avoidance,

and collision prevention. For example, the system can prioritise objects based on their distance and size, calculate safe trajectories, and provide real-time feedback to the user or autonomous system for safe and efficient navigation. Additionally, the system can enhance situational awareness by detecting objects that may not be directly visible but can still pose a potential hazard, such as low-profile obstacles or objects in blind spots. It is worth emphasising that with INT-DET, in contrast to the model developed in Challenge 4.1 linked to the same Case Studies, the aim here is to obtain a model with a good trade-off between performance and efficiency: high performance (see Figure 20 and Figure 21) to have a model capable of recognising as many objects as possible, good efficiency to be able to have a fast model that can be deployed on FPGA-type architectures more suitable for embedded devices.

4.3 Application in AVL Challenge – Real-Driver Emission (AVL_RDE) – TUG (AALpy)

One application of the AALpy automata learning library in the AIDoArt project is a subtask from the domain of Real Driving Emissions (RDE). The goal of this use case is to obtain a model of human driving behaviour that can be used during testing in various environments (simulation, Hardware-in-the-Loop, testbeds) without compromising realism to obtain KPIs such as emissions or battery usage. Given a formal description of the road and other environment features, the model should produce a velocity profile resembling human driving behaviour along the given road. Since there is more than one possible way of driving along a road, the model should be generative. While such a model could potentially be handcrafted by domain experts, this is a time consuming and expensive process. Instead, we apply a data driven approach by learning a driver model from a (small) set of recorded test drives.

TUG developed a prototype for this task based on passive automata learning for inferring probabilistic automata using AALpy. The prototype is able to learn a model from a data set and sample velocity profiles from a specific model and route.

AALpy is used primarily for its model learning capabilities. To learn a model, the provided data is first pre-processed. This also includes an abstraction process, during which features are selected and discretized, which is required since the formalisms currently implemented in AALpy do not support continuous domains. We then use learning algorithms provided by AALpy such as IOAlergia to learn automaton models for different high-level actions of the driver (i.e. acceleration and deceleration manoeuvres). Those high-level actions are introduced to compensate for limitations of the original IOAlergia algorithm which can occur when data is scarce. Those models are later combined into a single model, which can then be queried to produce velocity profiles for a specific route. Smoothing is required to account for the discretization process. An example of such generated behaviours for a given route is shown in Figure 51 along with real measurement data for comparison. The model that produced those traces was inferred from a set of 21 test drives. The approach could be validated by domain experts at AVL by comparing the traces generated using this method with a rule-based baseline provided by AVL.

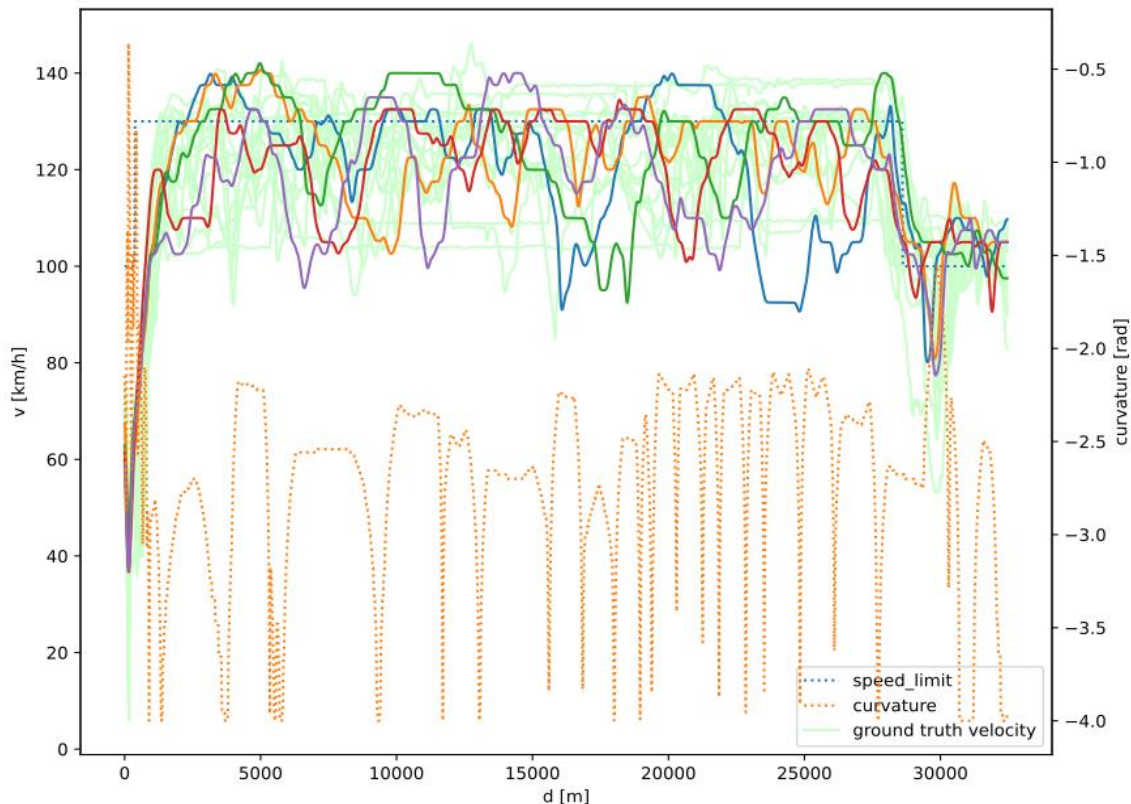


Figure 51 Measured behaviour (light green), behaviour generated from a learned model (solid lines) and road features (dashed) for a given route

4.4 Application in AVL Challenge – Model-based Testing (AVL_MBT, Testing ADAS functions) – TUG (AALpy)

AALpy is employed in the MBT use case of AVL. The goal of this use case is to efficiently test Advanced Driver Assistance Systems (ADAS), which are systems designed to handle specific driving tasks, such as highway driving. This is especially important since the safety of human lives depends on them while they also exhibit a high complexity due to the tight integration of cyber and physical components, which constitutes a case of safety-critical CPS.

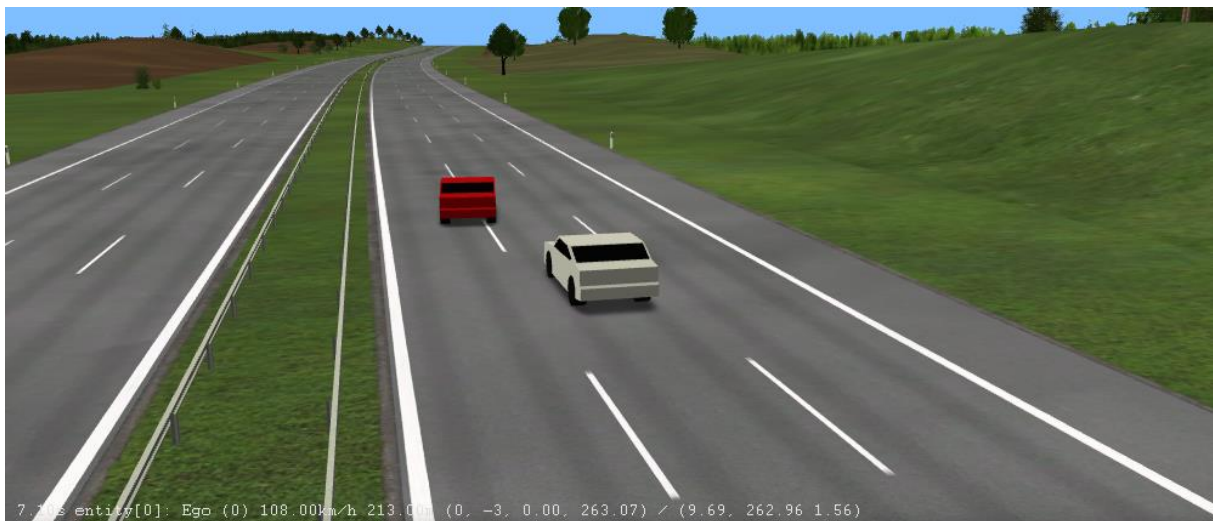
Testing ADAS is commonly done using parameterized scenario specifications that describe situations relevant to the operational domain of the ADAS. This is done using the OpenSCENARIO description language. Safety violations can be specified as ranges of KPIs such as a time to collision below 0.5 seconds.

In this use case, we employ a two staged testing approach. We first perform a global search that is based on an OpenSCENARIO file, searching for parameter combinations that result in critical behaviour. While we used genetic algorithms for this step, it could also be based on other methods such as Active Design of Experiments (ADoE), which is developed by AVL. In the second stage, we use parameter combinations identified in the global search as starting points for further testing. In particular, we are interested in slight variations of the scenario that might not be covered by the parameterized scenario description, which can be seen as testing for robustness. This is done by

defining a set of actions for the other traffic participants that can be chained into complex behaviours. This enables us to apply active automata learning from AALpy where we model how the KPIs change over time in response to the actions of the other traffic participants. Active automata learning methods commonly try to create a complete model of the system. This is undesirable in this use case, since the state space of the underlying model is infinite and might cover many cases that are of no interest. To account for the former issue, we stop the learning algorithm early which is acceptable since we are only interested in slight variations of the original scenario. The latter problem can be addressed by careful abstraction.

The use of AALpy results in (incomplete) behavioural models that are approximations of the behaviour of the ADAS from specific starting points. Such models can be analysed for clusters of states associated with critical KPIs which are more likely to represent similar failure modes or critical regions. The models could also be used to sample more action sequences with a bias for critical states. While this is unlikely to reveal new failure modes, it can increase the understanding of critical regions. Also, the behaviour in a new global search point could be approximated using models corresponding to points close by in the parameter space.

A visualisation of an action sequence executed after partially applying a scenario with parameters found by the global search is shown in Figure 52. It shows a cut-in scenario where the vehicle controlled by the ADAS function (in white) is overtaken by a second car (in red) that is controlled by action sequences.



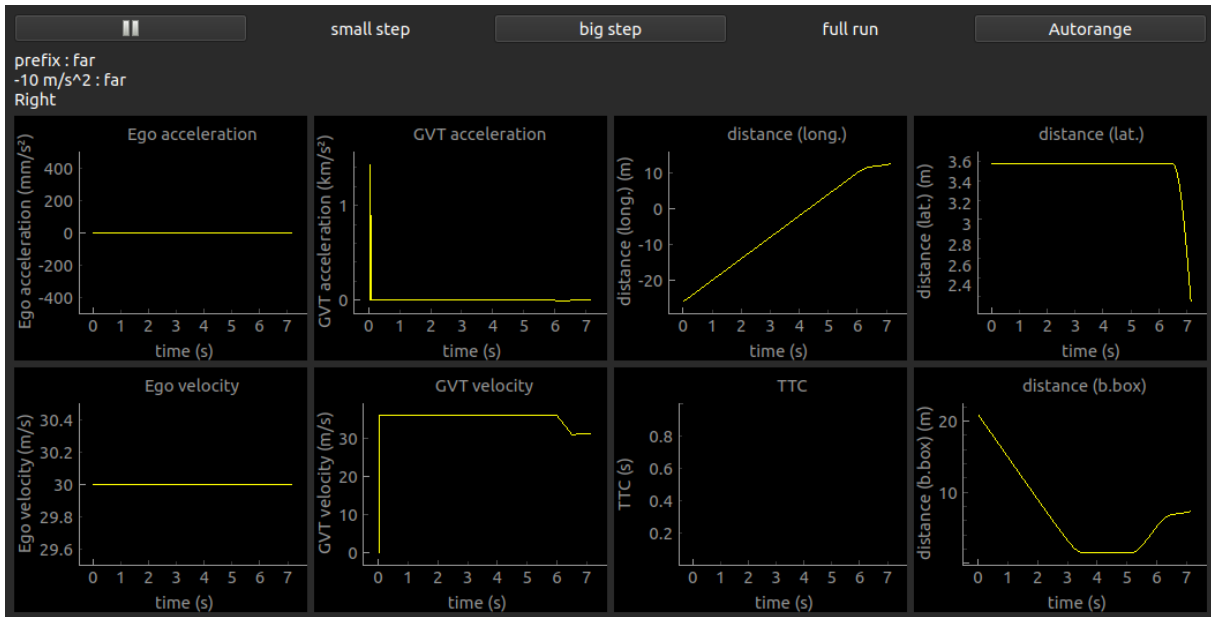


Figure 52 Visualization of an action sequence being executed with a 3D view and plots of KPIs

4.5 Application in AVL Challenge – Security Testing (AVL_SEC, Learning-based fuzzing of AGL) – TUG (AALpy)

One application of the AALpy library in the AIDOaRt project is security testing for Automotive Grade Linux (AGL), which is an open-source operating system for automotive applications. The development of AGL is a joint effort of industries from across the automotive domain, aiming to create a standard platform for in-vehicle software solutions and is thus expected to be shipped as part of millions of cars worldwide. Due to the wide exposure and the fact that cars are inherently safety critical, high requirements with respect to security have to be met, raising the need for thorough security testing. In this challenge, we aim to test the different communication interfaces of an AGL demonstrator which are shown in Figure 53. More specifically, we tested the Vehicle Signaling Specification (VSS) server and a Bluetooth interface. The VSS server is responsible for the communication between Electronic Control Units (ECU), which is especially critical.

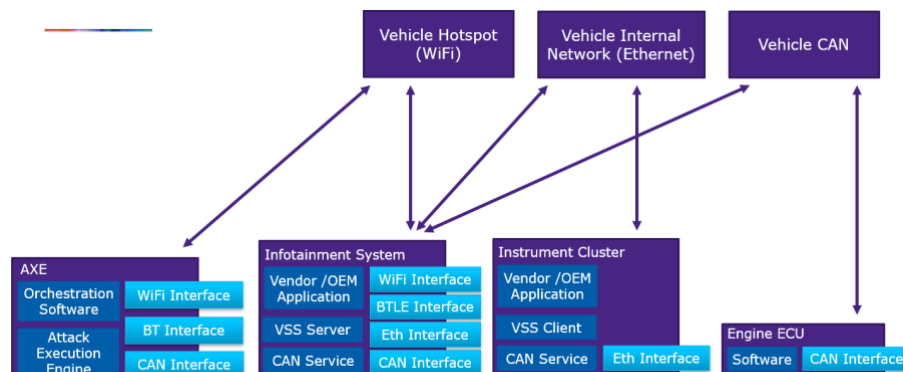


Figure 53 Communication between different subsystems in a car

This was done in a two-stage approach. First, we employed active automata learning via AALpy to learn an abstract behavioural model of the system under test. We then used this model as a basis for Model-Based Fuzzing, where a high number of invalid or unexpected inputs is tested starting from different states in the behavioural model to uncover new behaviour that might violate security assumptions. This learning-based fuzzing technique has the benefit that the model learned with AALpy counteracts the random nature of fuzz testing by initialising the system to meaningful states that might be hard to find by executing random input. This process is illustrated in Figure 54.

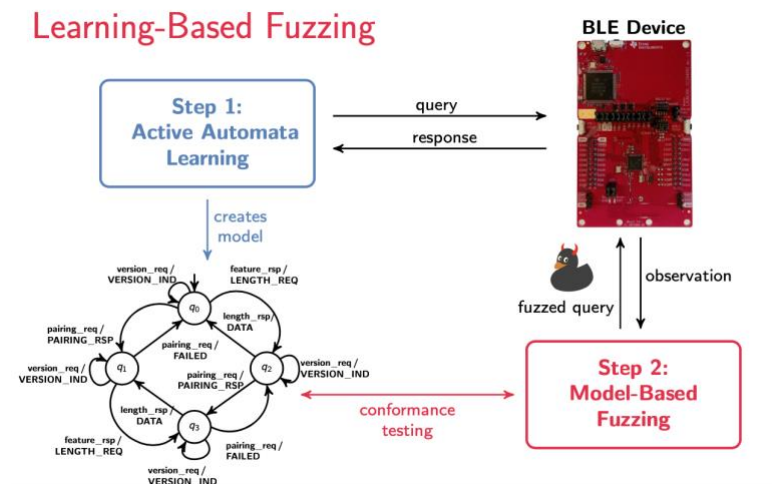


Figure 54 Learning-based fuzzing approach exemplified using a Bluetooth device

4.6 Application in AVL Challenge - Test Case Validation (AVL_TCV, ADAS Test Space Reduction) - AIT (DTSynth), ABO (STGEM)

DTSynth has been successfully applied in the use case challenge “Parameter space reduction in the TCV_USE_CASE” with respect to the fourth and fifth Hackathon. The solution implements a novel generative machine learning approach for the automatic generation of critical test cases within a given traffic scenario and aims on recommending suitable parameters, as demonstrated in the use case challenge for the use case AVL_TCV_UCS2 – SCENIUS parameter recommender. This approach contrasts with the analytical approach that has been developed in the AIDOaRt project. Analytically validating and recommending test scenario parameters, or parameter ranges, is only feasible if the scenarios’ complexity is low enough. However, if the complexity increases analytically recommending a parameter set with respect to some measure of interest gets out of hand quickly. Therefore, we started to look into approaches that provide more flexibility and we developed a generative AI method for recommending parameters/parameter ranges. This involves producing parameter values that initialise the traffic scenario for input into a simulator. Here, a test case is deemed critical when the resulting simulation involves high-risk situations, such as the ego vehicle approaching dangerously close to a pedestrian, for example, cf. Figure 55.

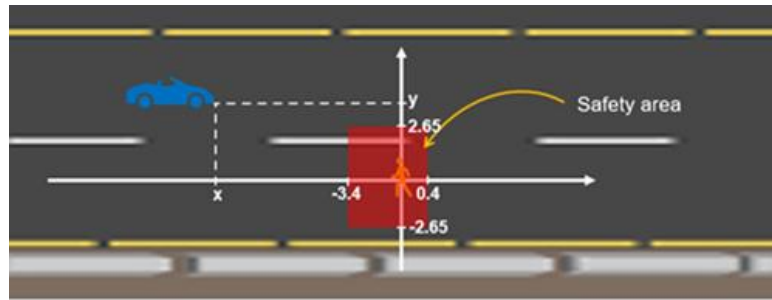


Figure 55 Example Safety Area

We used the generator trained to generate 1000 test cases for a pedestrian traffic scenario. Figure 56 illustrates these generated test cases, indicating their corresponding y-coordinates and time values. The visualisation utilises the parallel coordinates technique, enabling a high-dimensional representation where each y-axis corresponds to a specific dimension. Each test case appears as a set of connected lines, depicting its values on the axes. The top and bottom lines serve as boundary markers for variable validity and are not indicative of generated test cases.

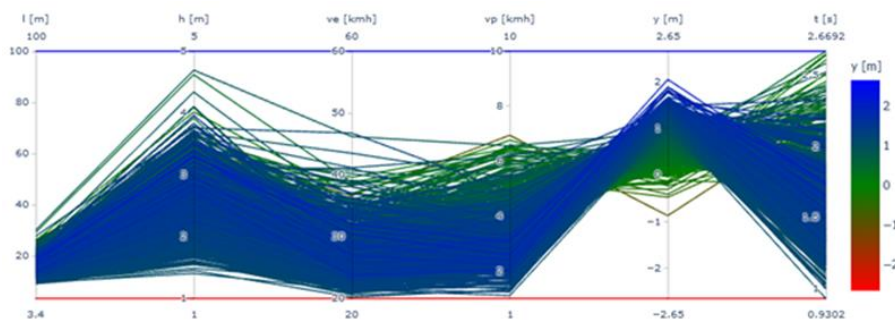


Figure 56 Parameter coverage of generated test cases

It is evident that the generated test cases did not cover the complete validity ranges of all scenario variables. This limitation can be attributed partially to the quality of the test case generator and potentially to the absence of solutions in certain regions of the input search space. This shortfall might arise due to the validity range limits being set without considering dynamic relationships between the variables. However, the generated test cases all fulfil the criticality assumption.

The STGEM tool developed at ABO can be applied to parameter space reduction. The use of the tool has been proposed in the use case. STGEM contains an implementation of the WOGAN algorithm [WOGAN] developed during the project, and this algorithm trains a generator that can sample tests from the set of critical tests given a requirement. The algorithm works by training a Wasserstein Generative Adversarial network targeting sampling from the critical set. The WOGAN algorithm has shown good performance in standard CPS benchmarks in the research literature. For example, WOGAN

was used in the SBST 2022⁶ and SBFT 2023⁷ CPS tool competitions to validate the lane keep assist of a self-driving car. The algorithm was ranked among the top entries in the competition. This established performance shows that the WOGAN algorithm could be successfully used in the use case.

4.7 Application in VCE Challenge - MOMoT for FMU-based simulation optimization

The challenge aims at extending the framework developed for architecture specification and analysis recently accepted for publication at SysCon⁸ supporting prescriptive modelling in SysML v1.x and AutomationML standards. The challenge assumes the existence of FMUs of the system under study corresponding to SysML or AutomationML modelling elements (e.g., a block or internal elements, respectively).

By utilising the Functional Mockup Interface (FMI), standard simulations created in various tools like Simulink and OpenModelica can operate independently. Additionally, by integrating several other enabling technologies, we can automate the parameterisation and execution of FMI-supported simulations for custom optimisation.

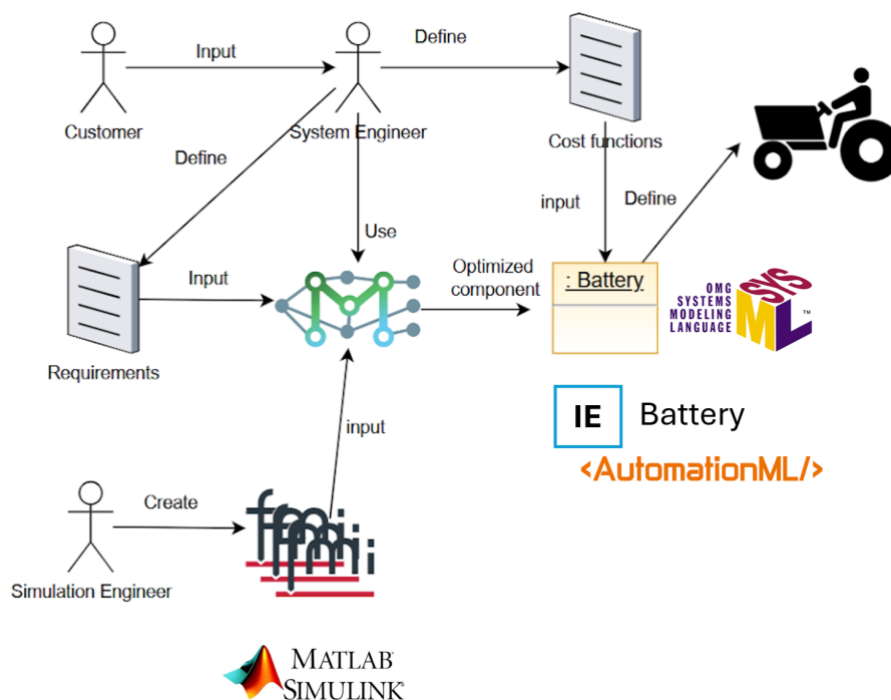


Figure 57 MOMoT for FMU-based simulation optimization scenario

⁶ Jarkko Peltomäki, Frankie Spencer, Ivan Porres, WOGAN at the SBST 2022 CPS Tool Competition. SBST@ICSE 2022: 53-54. <https://dl.acm.org/doi/10.1145/3526072.3527535>

⁷ Jesper Winsten, Ivan Porres, WOGAN at the SBFT 2023 Tool Competition - Cyber-Physical Systems Track. SBFT 2023: 43-44. <https://ieeexplore.ieee.org/document/10190409>

⁸ https://www.researchgate.net/publication/378148318_Towards_Automating_Model-Based_Systems_Engineering_in_Industry_-_An_Experience_Report

Figure 57 sketched the artefacts, steps (labelled arrows) and stakeholders involved in MOMoT for the FMU-based simulation optimization scenario. A battery is a component of a VCE vehicle and it can be modelled in SysML and/or AutomationML by the system engineer. The simulation engineer creates the corresponding simulation model of the battery. The simulation model can be of arbitrary complexity and different cost functions can be considered to suitably tune the simulation.

MOMoT can support the selection of parameter values. It represents internally the FMU as a Ecore-based model, implements an ad-hoc multi/single optimization function and returns an optimised component configuration.

The challenge started during the 5th hackathon and the collaboration is still ongoing. A video showing MOMoT in action is available in the AIDoArT shared folder https://drive.google.com/open?id=1SWoFSc8q_-KTermf9Ta2xpSQPg0ikqHJ&usp=drive_fs.

A substantial overlap with the AVL case study “Optimization of Development Process” and associated hackathon challenges has been identified since the target of optimising simulation model parameters is the same. Thus, AVL will contribute a more complex simulation model (FMU) consisting of several components (e.g., a powertrain or vehicle simulation model) with more parameters to optimise for validating the solution which has been developed for the VCE challenge.

4.8 Application in VCE Challenge - Keptn for FMI

This challenge offers the possibility to leverage use of DevOps pipelines for continuous delivery of FMUs for simulating system components (e.g., batteries).

As simulation with the FMI standard abstracts files into standalone black box representations, a.k.a., Functional Mockup Units (FMUs), it makes the standard suitable for automation. Often when dealing with simulation there is a need to perform many different simulation executions with varying parametrics. Manual execution of these tasks is inefficient, necessitating automation. Moreover, mere parameter variation automation may not suffice; an algorithm should be employed to optimise parameters and reduce the number of required simulation runs. For instance, consider the [challenge of MOMoT for FMU-based simulation](#).

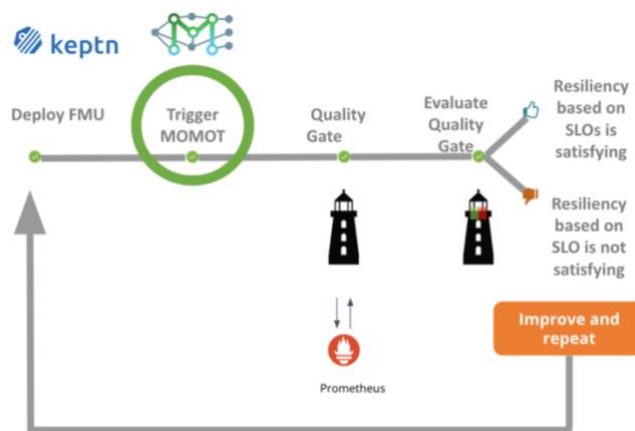


Figure 58 integrating Keptn with parameter optimization of FMI simulations.

In this challenge we explored the Keptn tool to use as a DevOps pipeline. We assume that different deliveries occur whenever new FMUs become available or when existing FMUs are updated with new parameterizations. These FMUs are then deployed into a target simulation environment. The challenge itself is to provide a successful delivery of a DevOps integration running Volvo CE reference simulations in the AIDOaRt context, an example is available in figure below.

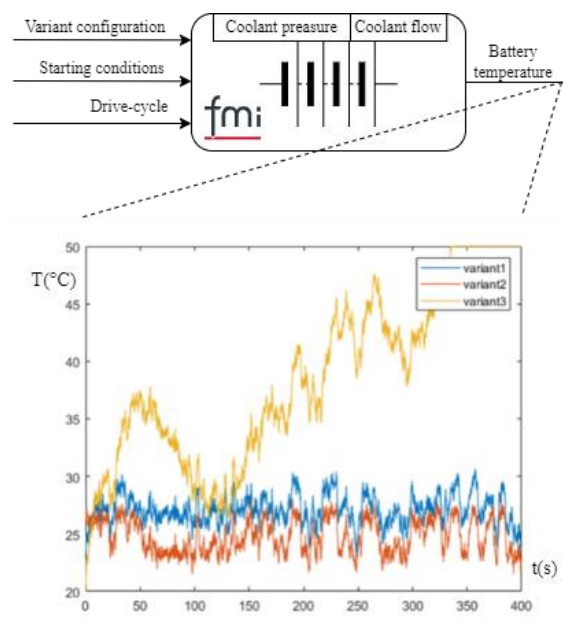


Figure 59 Multiple parameters variants simulation of a thermal management system using Keptn and FMI

4.9 Application in ALSTOM/BT Challenge – Automate Parameterization of Thermal Model

The ALSTOM/BT_UCS2 is aimed to parameterise the machine thermal model in a controller with a certain level of accuracy using machine learning algorithms. Accurate prediction of temperature in critical parts of a machine is important for improving the reliability in real time operation. Hence, temperature rise tests are conducted in the test platform in an emulated environment prior to the real deployment. The controller is embedded with a reduced order lumped parameter thermal network (LPTN) model, to which thermal parameters are fed, calculated based on analytical estimations. During the test procedure the parameters are fine-tuned manually against the measurement data. The challenge to obtain the accurate thermal parameters is linked to the dynamics associated with drive cycle operation and inputs from other interlinked components.

Within the AIDOaRt project framework, the parameterization procedure is proposed to be automated using the AVL CAMEO Active DOE tool. ML/AI algorithms such as symbolic regression and reinforcement learning algorithms within the Active DoE tool can capture the nonlinearity in the system, hence, can determine those control parameters of the system. This is achieved by learning the relation between control parameters and performance KPI. Both these algorithms are implemented.

State-Of-The-Art Solution

The temperature itself follows a first or second order linear RC differential equation for any fixed state, and it can be modelled as the output of a linear steady state system with system inputs given by cooling air, and rotor/stator losses. Non-linearity of these so-called *equivalent circuit models* is achieved by allowing the model parameters to vary over the model state. The state-of-the-art solution makes use of this setup, in the sense that parameter identification tests are performed for a number of operating points (with fixed states) independently. For each operating point the model is assumed to be linear, such that parameter identification procedures can be used locally. Then, assuming that the model parameters change smoothly across the state space, they can be extrapolated from those operating points with dedicated tests.

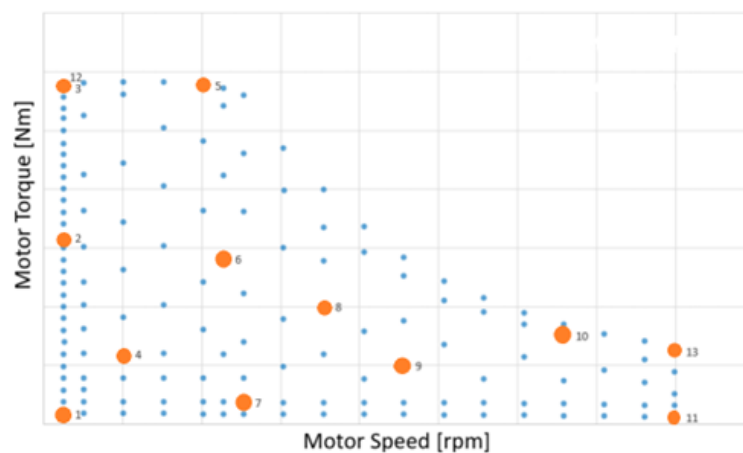


Figure 60 Model parameters are calculated for fixed operating points (orange), model parameters are extrapolated for the remaining operating points (blue)

The following results from AVL CAMEO show the model coefficients, as well as the RC time constant and the steady state prediction as a function of the state given by motor speed and torque. Parameters are calculated for a selected number of operating points where data is available using a dataset provided by ALSTOM. A neural network is then trained on the model coefficients, mapping motor speed/torque values to new, unobserved operating points.

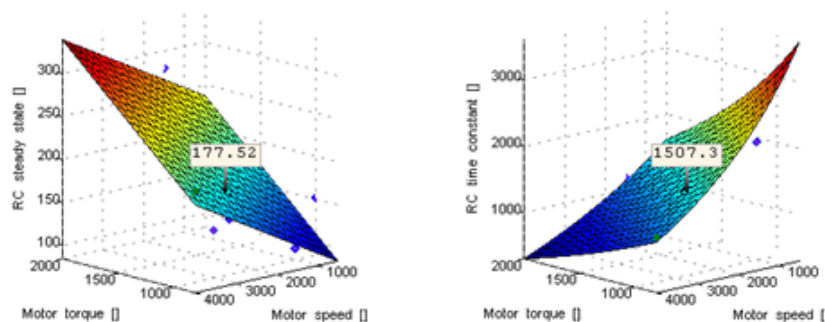


Figure 61 Parameter coverage of generated test cases.

The problem using the state-of-the-art solutions is that parameter-identification has to be performed locally for dedicated operating points, which requires a lot of time. Moreover, it is desirable to use on-board or real driving data from the component's usage. This cycle data usually does not contain any fixed steady state, which means that parameters cannot be identified independently. Instead, the parameter maps/functions (as functions of the state) have to be identified using generic input data, and no standard parameter identification procedures as in the linear case exist.

Active DoE:

AVL Active DoE is a Reinforcement-Learning-type procedure which can be used in order to calibrate or optimise generic components based on input/output data. In this case, a Simulink/FMU model is prepared by the user, the model parameters and targets (e.g. difference model vs data) are exposed to CAMEO via an interface. An intelligent testing approach determines the optimal model parameters. This approach has been used to determine the unknown parameters of the temperature model. However, since this generic approach does not take any prior knowledge of the model into account, the procedure is extremely slow and full parameter identification may take up to several hours. For this reason, an alternative, semi-physical approach was considered.

Symbolic Regression:

This approach provides a promising combination of data-based machine learning methods with the possibility of entering physical knowledge into the procedure. A physical formula is entered by the user - either for the temperature or its derivative as a function of the parameters. An optimization algorithm determines the unknown formula coefficients. For example, such a formula may take this form:

$$C_s \frac{dT_s}{dt} = \lambda_{1s}(T_s - T_{env}) + \lambda_{12s}(T_s - T_r) + (1 + \alpha_{R1} \cdot (T_1 - 20)) \cdot I_s^2 + K_{stray} \cdot P_{stray} + K_{harm} \cdot P_{harm}$$

In this case, the stator temperature derivative, dT_s/dt , is modelled as a function of known inputs (green) and unknown parameter functions (yellow), and The target is to determine the unknown parameters from the model inputs.

Symbolic regression models allow the user to enter a formula, and the unknown parameter functions are identified using a genetic algorithm. Results for the dataset provided by ALSTOM are shown in Figure XY16. Here, three models were trained on three different cycle datasets. All models are evaluated using the first dataset as verification data. It can be seen that the stator temperature can be modelled up to an accuracy between 1.1 and 4 degrees Celsius, measured by Median Absolute Deviation (MAD).

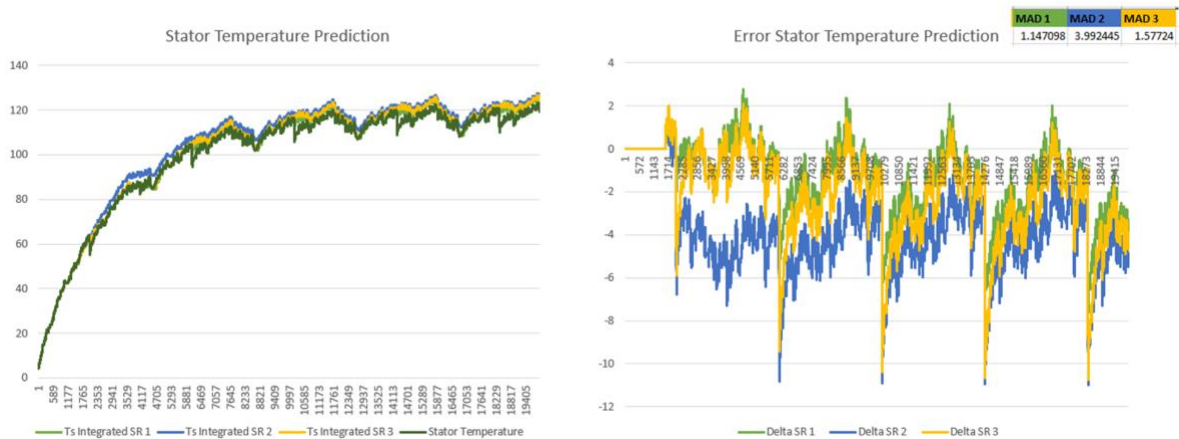


Figure 62 Results for the stator temperature model, obtained using Symbolic Regression

4.10 Application in WMO Challenge - Enhanced Test Results Exploration with Natural Language Processing - RISE LogGrouper and BIC-Tool

At Westermo, nightly automated regression testing is an important activity for quality control. In order to speed up this part of the DevOps feedback loop, Westermo desires AI-support for root cause analysis of functional issues and for the repetitive and time-consuming task of humans investigating test results by reading log files.

RISE is developing two tools for these purposes (see image Figure 63).

- First LogGrouper, a tool that identifies similarities in test case executions. LogGrouper has the goal to simplify the exploration of test results by grouping log files that seem to show the same error instantiated by multiple test executions, possibly on different test systems by different test cases, and maybe even on different code projects. LogGrouper is running in Westermo's environment, produces clustering using on-line data. However, work on parameter settings for strictness of the grouping, and the integration with other systems has not been finalised.
- Second is the BugIdentifier tool, which also uses information from code changes to identify and rank links from failing tests to bug-inducing commits. A first complete prototype of the tool has been developed, and further implementation is ongoing. Based on the input filters (date and branch) the tool suggests code changes that might have caused the failing tests.



Figure 63 Tools from RISE. LogGrouper (left) clusters failing tests based on similarities in log files from test execution. BugIdentifier Tool (right) proposes code changes that could have caused failing tests

With respect to the AIDOaRt architecture, these two tools are examples of AI for Testing and AI for Monitoring, and they simplify Engagement and Analysis.

The tools require a significant amount of data engineering: Data collection, data management and data representation. In the first phase, the raw datasets, mainly git commit history and test execution results, are pre-processed using part-of-speech tagging, stop words removal, and lemmatization. After the pre-processing, the failing test cases are mapped to their commits, and groups are created based on the date and feature branch filter to reduce the search space.

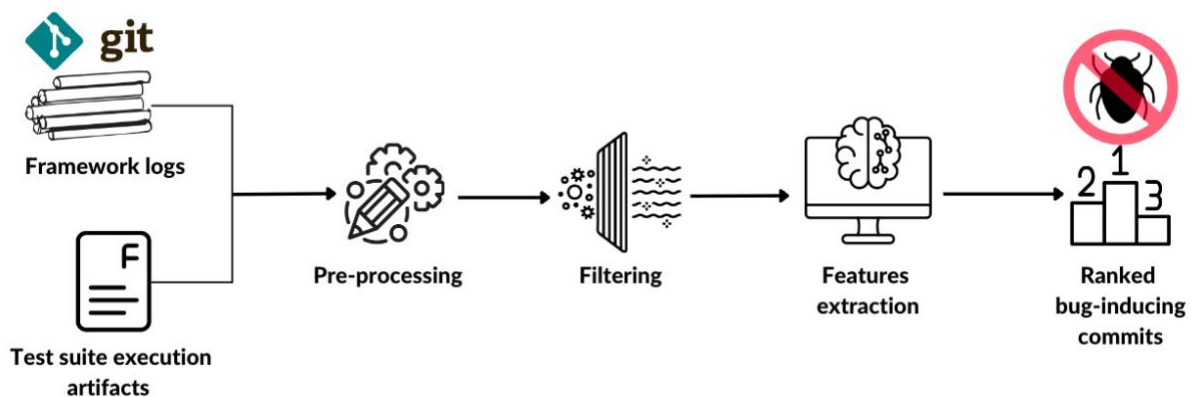


Figure 64 The BugIdentifier Tool Natural Language Processing Pipeline

Currently, the LogGrouper and BugIdentifier tools are implemented as prototypes and LogGrouper is running in Westermo’s DevOps environment. Next steps include discussions on whether we could add more data as input, a possible further integration into Westermo’s DevOps environment of the BugIdentifier Tool, and to conduct an evaluation of the BugIdentifier with colleagues at Westermo.

4.11 Application in WMO Challenge – Performance Data Exploration – Copado CRT

Numerous products from Westermo Network Technologies are powered by their proprietary Westermo Operating System (WeOS). Changes to the WeOS software are integrated frequently throughout the day. Following these updates, a suite of automated regression tests is conducted overnight via the Fawlty test framework. This testing is performed on dedicated servers, from which we periodically harvest performance metrics using a node exporter, tracking variables such as memory and CPU usage, along with network load.

At times, the servers that facilitate these tests encounter issues. For instance, we encountered a situation where excessive network traffic generated to test the WeOS's resilience ended up overwhelming a server. It was a result of the server allocating too much disk space to traffic data and consuming nearly all available CPU and memory resources, thereby hindering normal testing operations. This issue was swiftly resolved by our DevOps team, allowing the usual testing processes to resume.

The challenge at hand involves delving into this performance data to uncover ways of detecting anomalies automatically. The ultimate aim is to enhance our understanding of how we can effectively implement a blend of rule-based, statistical, and AI-powered anomaly detection within Westermo Network Technologies. This initiative draws inspiration from recent academic research (Torshizi, 2022) and aims to build upon it.

For this project, Westermo entrusted Copado with an extensive array of performance data from their testing infrastructure. This dataset was rich with various critical operational metrics, providing a detailed snapshot of system health and efficiency.

Our initial step was a thorough examination of the data to isolate significant patterns and identify key performance intervals. We scrutinised periods of high demand to determine the system's capacity and response. Part of our analysis also included a study of the metrics' seasonality to ascertain if any usage patterns were predictable.

Beyond routine analysis, we sought to pinpoint anomalies that deviated from normal patterns, which could signal underlying issues or unexpected system events. Our comparative analysis also extended to assessing correlations within and across the computers in the testing network, helping us understand the interplay of different performance indicators.

We paid particular attention to detecting shifts in the data trends and identifying points where resources neared full capacity, which could indicate potential constraints or concerns within the system.

Our analytical arsenal included a blend of traditional statistical methods and advanced machine learning algorithms, ranging from Granger causality tests and Pearson correlation coefficients to extreme gradient boosting models. This multifaceted approach ensured a robust and nuanced understanding of the performance data.

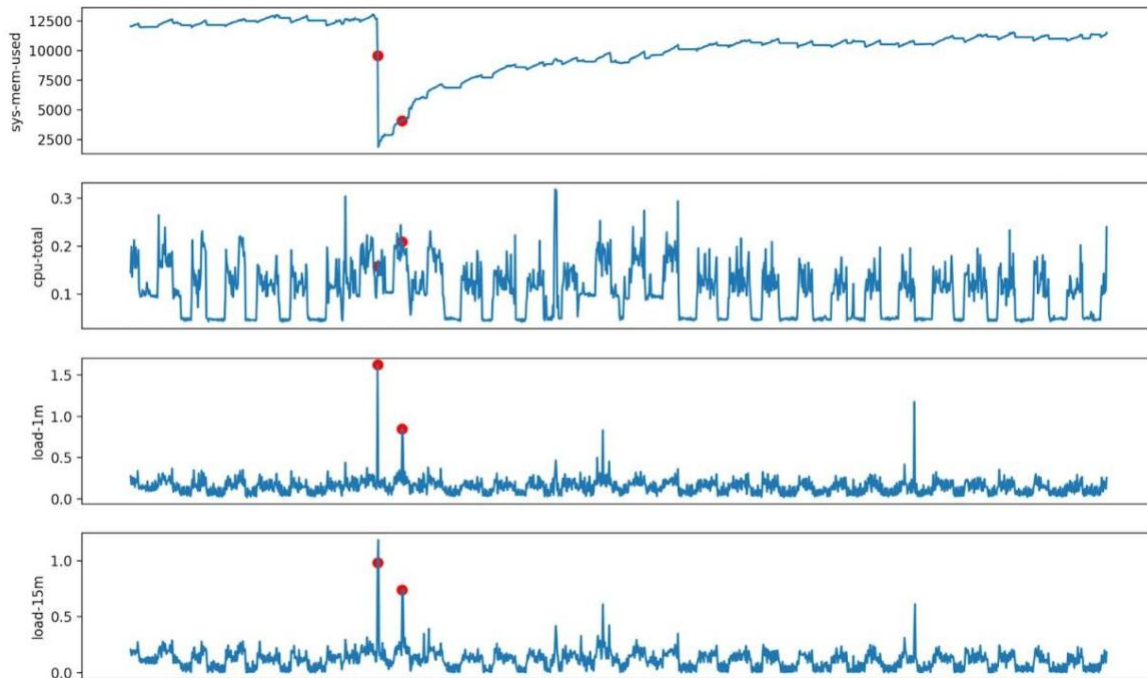


Figure 65 A month of performance data from one test system. Each curve represents one metric over time. Highlighted with red dots are anomalies detected by Copado using the decision forest algorithm

4.12 Application in WMO Challenge – Online Test Case Selection and Prioritization – ABO (STGEM)

Many WMO products run the WMO Operating system (WeOS). In the software development process, WeOS code changes are added several times a day. During the night, the new software undergoes automated nightly regression testing by the Fawltly test framework. This framework is also developed in an agile software development process, and also receives frequent code changes. In late 2022, WMO released the WMO test results data set⁹ and in this challenge, ABO continues exploring this data to work on test selection for regression testing.

Regression testing in software development checks if new software features affect existing ones. Regression testing is a key task in continuous development and integration, where software is built in small increments and new features are integrated as soon as possible. It is therefore important that developers are notified about possible faults quickly. In this article, we propose a test case prioritisation schema that combines the use of a static and a dynamic prioritisation algorithm. The dynamic prioritisation algorithm rearranges the order of execution of tests on the fly, while the tests are being executed. We propose to use a conditional probability dynamic algorithm for this. We evaluate our solution on three industrial datasets and utilise Average Percentage of Fault Detection for that. The main findings are that our dynamic prioritisation algorithm can: a) be applied with any static algorithm that assigns a priority score to each test case b) can improve the performance of the

⁹ <https://github.com/westermo/test-results-dataset>

static algorithm if there are failure correlations between test cases c) can also reduce the performance of the static algorithm, but only when the static scheduling is performed at a near optimal level. Our code was originally developed for an AIDOaRt Hackathon and we evaluated our solution on a dataset provided by WMO. Then, to evaluate our solution on more industrial datasets, we modified our code to adjust it to different data formats. As additional datasets we chose Paint Control and IOF/ROL which were created by ABB Robotics Norway [SGMM2017]. To effectively evaluate the Dynamic Test Case Prioritization approach, we compare our solution to Optimal, Worst, and Random algorithms. The Optimal algorithm creates an optimal schedule, i.e., it always schedules all failing tests before the passing ones, thus the dynamic algorithm can either reduce or provide the same performance as the Optimal algorithm. The Worst algorithm acts in the opposite way, and the dynamic scheduler can only improve or provide the same performance as the Worst algorithm. The Random algorithm, as the name reveals, creates a schedule in a completely random manner. Due to the non-deterministic nature of the Random algorithm, we execute it 30 times for each evaluated cycle. The dynamic approach can reduce, improve, or provide the same performance as the Random algorithm. The comparison of performance of pure static and dynamic approaches applied to a static one for Optimal, Random, and Worst algorithms is demonstrated in Figure 66.

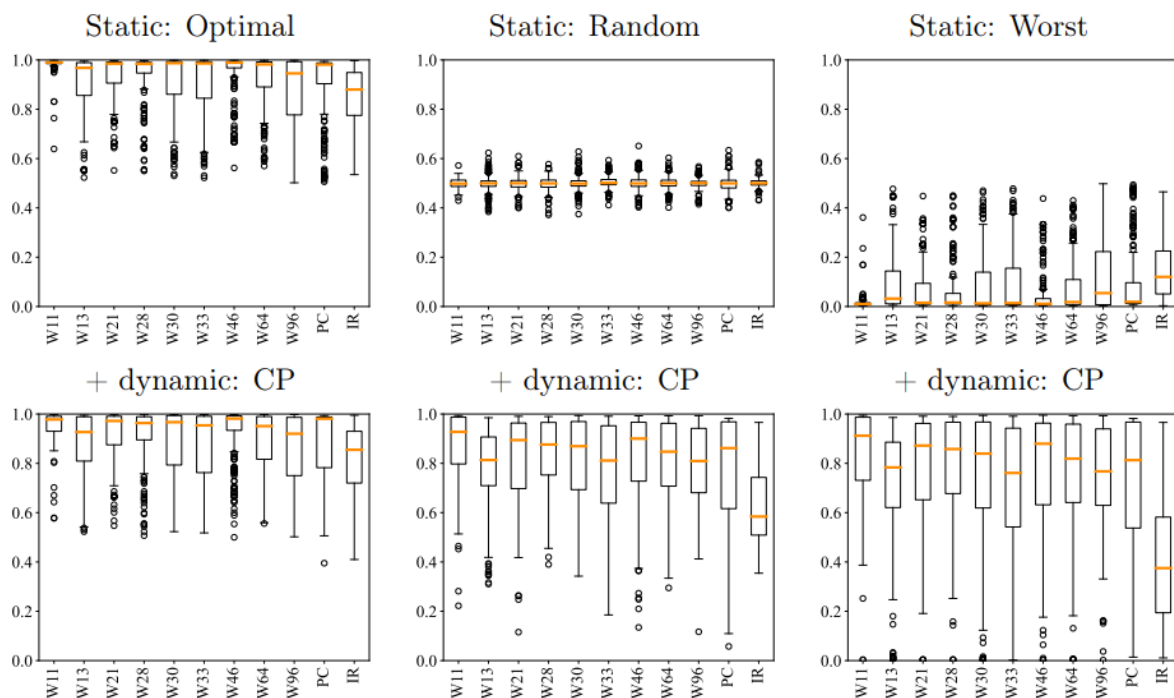


Figure 66 Comparison for APFD for static and dynamic prioritisation algorithms (higher values are better). The first nine systems are from the Westermo dataset, the last two are from Paint Control and IOF/ROL.

For the industry, our approach helps to dynamically adjust the schedule of test cases and execute possibly failing test cases earlier, thus decreasing the waiting time for developers. When evaluated with the APFD metric, the dynamic algorithm demonstrated a noticeable increase in performance when applied to Random and Worst algorithms which means that it succeeded in scheduling failing test cases earlier. The limitations of our approach are that constant values are applied to choose

history length as well as to adjust scores of correlated test cases. As a possible direction for the future work, these values should be adjusted dynamically for each test system. In addition to that, the dynamic approach should be applied to existing static TCP baselines. Our publication for the proposed solution was presented at the AST conference¹⁰ and the pre-print is available here: [TSP2024].

4.13 Application in TEKNE Challenge – Design choice exploration/verification – UNIVAQ (HEPSYCODE), UCAN (S3D)

The TEKNE challenge focused on the application of AI/ML techniques within an agile process for the Electric/Electronic architecture of a professional-grade vehicle. Specifically, the challenge deals with the diagnostics and prognostics of the vehicle's power electronics, which can be considered a CPS. In the TEKNE challenge, UNIVAQ leveraged its expertise in AI/ML and design space exploration to play a significant role in several aspects of the solution:

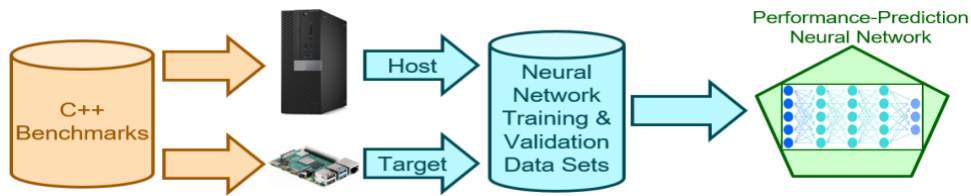
- First, UNIVAQ utilised HEPsim2, a SystemC-based TLM simulation framework, to evaluate the timing and performance characteristics of various design solutions under consideration. This simulation provided valuable insights into the system's behaviour before physical implementation.
- Regarding Design Space Exploration, UNIVAQ leveraged AI/ML techniques to explore the design space for the CPS system. This process involved identifying potential configurations, evaluating their performance, and selecting optimal choices. To achieve this, UNIVAQ employed AI/ML algorithms to predict the Pareto front for hardware-software (HW/SW) partitioning and mapping. This Pareto front represents a set of (sub-)optimal solutions offering trade-offs between objectives like performance, power consumption, and cost.

UNIVAQ explored a wide range of ML algorithms, including Support Vector Machines (SVM), Random Forests, and Deep Neural Networks (DNNs), to address various challenges within the DSE process. UNIVAQ further contributed by preparing and cleaning datasets for training and evaluating ML models. Additionally, UNIVAQ has been involved in developing and utilising multi-label classification algorithms, where a single data point can belong to multiple classes simultaneously. This might be relevant if certain design options need to be classified based on multiple criteria.

UCAN focuses on predicting the timing performance of the TEKNE Use Case using ML approaches. Two approaches have been investigated:

- Phase-granularity run-time analysis: This approach involves analysing the application behaviour divided in phases. Each phase is a section of the application code in which its performance (i.e. execution time) in the final (target) processor, is going to be predicted. To do so, the characteristics of the target processor on a representative set of benchmarks are extracted. By relating the characteristics of the processor where the system design is going to be simulated (the host) with the target, a neural network is trained, as shown in the following figure:

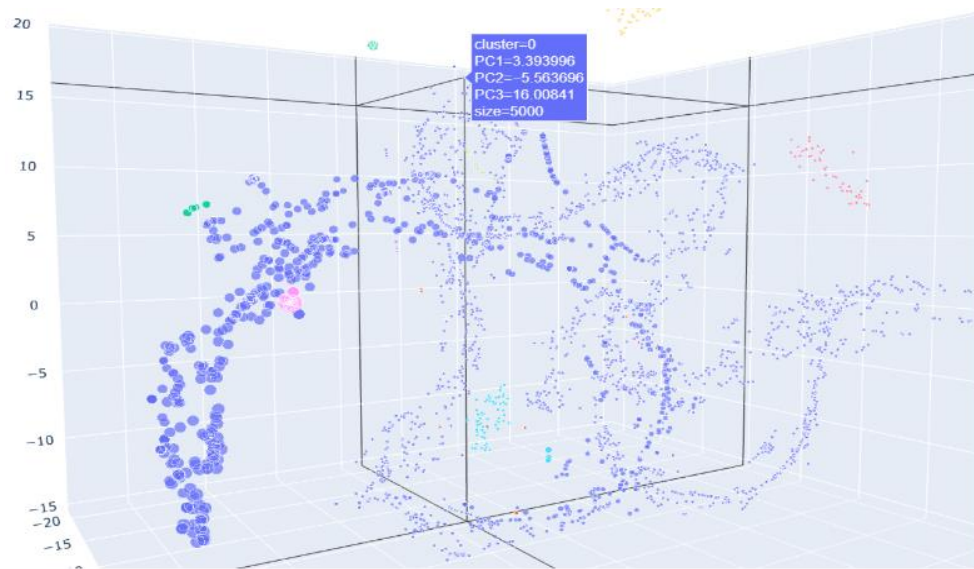
¹⁰ <https://conf.researchr.org/home/ast-2024>



The Neural Network will be used during system simulation to execute the application code in the host processor but as it would be executed by the target, as shown below:



The prediction-space is too large that the different solutions found have been clustered in several clouds:



Each one requires a specific NN. Results obtained so far are the following:

mape	(50, 425)	(50, 800)	(50, 1175)	(50, 1550)
(50, 425)	15,52	15,86	17,67	20,62
(50, 800)	15,67	15,33	15,60	18,16
(50, 1175)	17,83	16,94	16,05	16,36
(50, 1550)	15,27	15,44	14,98	15,11

Which shows a good estimation accuracy of each NN in its own problem-size but also outside it (around 15%).

- Basic-Block-granularity static analysis and back-annotation: This approach involves statically analysing the application code at the basic block level to extract relevant information. This information is then "back-annotated" to the code, allowing for cross-compiled simulation of

the timing performance. The performance figures obtained are very good. The following Table shows the accuracy obtained in three benchmark functions of the number of cycles in each basic block predicted by the NN against the most usual techniques:

Benchmark	Bubble sort	sha256	dijkstra	mean error
1 cycle per instruction	28.97%	29.71%	40.81%	33.16%
Constant number of cycles per instruction	22.16%	20.90%	1.81%	14.96%
Specific number of cycles for each instruction	10.06%	13.37%	6.01%	9.81%
Proposed NN	9.67%	9.04%	5.47%	8.06%

As it can be seen, the mean error provided by the NN is smaller than the state-of-the-art techniques used nowadays.

During the Hackathon, the application of both techniques to the TEKNE use case were discussed. Preliminary results were analysed. Nevertheless, the final performance-directed solution will be proposed once the final versions of the tools are released.

In the final validation activity for TEK_UCS_01, planned in the WP5 tasks, TEKNE will provide real-world scenario data to serve as a test bench. UNIVAQ will then integrate this data into the HEPHYCODE modelling framework and SystemC code to extract results simulating the chosen real-world scenario. UCAN will further refine the AI/ML approach by leveraging the larger dataset and a new model to reduce errors and improve accuracy. This refinement will involve verification and validation using both benchmark functions derived from real-world data and out-of-benchmark training data.

UNIVAQ will leverage previously established AI/ML performance prediction models (timing, power, area/size, etc.) developed during the AIDoRt project. These ML-based cross-platform prediction frameworks will be integrated into HEPHYM2 for performance analysis, replacing computationally expensive TLM/ISS components with an AI/ML-powered performance simulator. The predictions will then be compared against UCAN's AI/ML approach.

Overall, the TEKNE challenge showcases the potential of AI/ML in the development of complex automotive CPS systems, particularly in enhancing diagnostics, prognostics, and timing performance prediction for the vehicle's power electronics.

4.14 Application in AVL Challenge – Real-Driver Emission (AVL_RDE) – UNITE (TWIMO), UNIVAQ, MDU

Figure 67 depicts an instance of the TWIMO framework for the AVL RDE use case, developed by UNITE, UNIVAQ, and MDU. It is composed of two main parts: i) the spatial model and predictors configuration (part I of the figure below) and ii) the domain-specific classes and relations describing the driving behaviour metamodel language we considered to use the framework (part II).

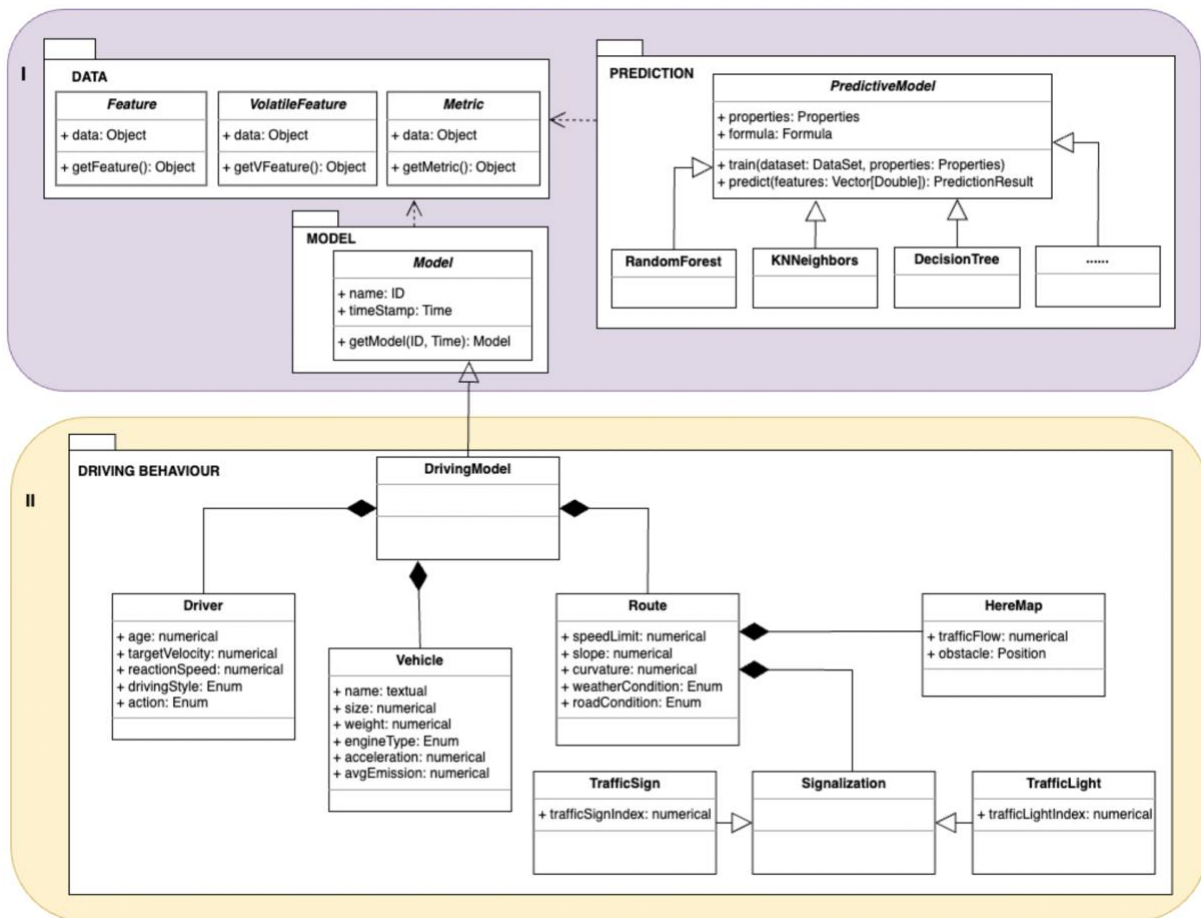


Figure 67 TWIMO framework instance for the AVL RDE UC

The proposed framework can be used for the prediction of Real Driving Behaviour by using different ML techniques (as illustrated in the Prediction components).

The proposed framework has been applied by UNITE (in collaboration with UNIVAQ and MDU) to the AVL-RDE use case, following the step below.

Data Acquisition: The data and their description that are used in this research work are provided by AVL company. The data was collected from special equipment placed on a car. Data refers to several drivers and different driving paths: highways, mountains, etc. The data collected is real driving recordings in time series format (time-based data on vehicle speed, throttle/brake pedals, curvature, road gradient, GPS coordinates, etc.).

Features Selection: To reduce the unnecessary computational cost, we filter out all the channels that are not relevant for modelling human driver behaviour. To that end, we took a close look at the channels in the dataset and found that there are three types of channels: 1. channels (resp. their corresponding data) are collected by two-way (snapshots and real driving recordings), 2. channels (volatile features like distance) are calculated from other features (speed and time), and 3. metric channels (such as speed).

Pre-processing: When we explored the features' data in the dataset, we found that each feature's data has a different scale. All features' data in the dataset are re-scaled using this standardisation except the velocity_kmh_raw feature as it represents the target feature or class.

Driver Behaviour Prediction: In the literature, there are many different ML algorithms for classification problems and there is no certain ML algorithm fit to all datasets. Choosing a ML algorithm depends on the type and size of the dataset of interest. In this research work, we compare the performance of mostly widely used ML algorithms using Python scikit-learn packages to determine which algorithm is the best fit for the collected dataset. These algorithms are GradientBoosting, DecisionTree, RandomForest, LogisticRegression, KNeighbors, GaussianNB, LinearSVM, and AdaBoost. The target algorithm should be able to predict the driver behaviour on the basis of the real dataset available and selected features on any arbitrary test route.

4.15 Application in TEKNE challenge – Operating Life – ROTECH

Data Aggregator simplifies the task of accessing, processing, and analysing large volumes of data. This solution automates, also, the process, saving time and reducing the potential for errors. The context of operating life monitoring is the following:

- Services in an environment where data are partly simulated;
- Measured data collected and pre-processed (cleaning, filtering, features extraction) to obtain “monitoring data”;
- Data transferred to the remote computing and data storage aggregator whose resources are available to run a full capabilities PHM system.

General Challenge Goals:

- **Bridger:** provides the communication between the On-board Platform and Remote Platform. It introduces the features of secure communication using an encryption and decryption algorithm;
- **ConvHandler:** provides data filtering and cleaning of the raw data;
- **Data Aggregator:** provides to help them make better decisions, improve process efficiency and, finally, understand performance of the Platform. Consistent evolution and the usability of the data play a key role too.

There are four techniques:

- **In-comm aggregation:** Uses a multi-hop system for the process of gathering and routing information inside the Data Aggregator;
- **Tree-based approach:** An aggregation tree is constructed, mapping put the data from leaves to roots (source and sink nodes respectively);
- **Cluster-based approach:** This approach is used to collate larger amounts of data on the entire Platform.

Multi-path approach: In this approach, partially aggregated data is sent to the root or parent internal table which then can send the data down various paths.

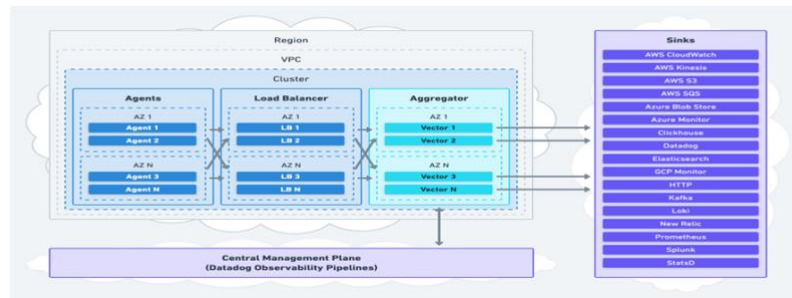
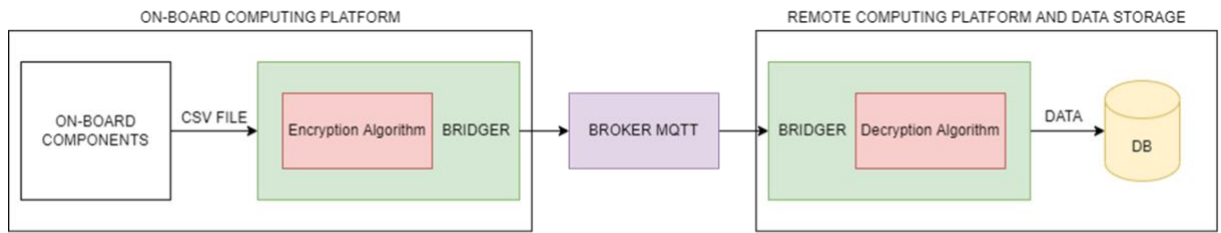


Figure 68 Operating life monitoring flow

4.16 Application in CAMEA Challenge – Power-Aware Radar Configuration – ABO (STGEM)

This challenge deals with a traffic monitoring system, which is a complex solution consisting of various sensors and components. CAMEA uses a radar-on-chip platform that is a highly integrated solution with a radar signal part and processing cores embedded in silicon. The radar sensor has to be configured before each start-up. The configuration is quite complex and some of its parameters can have an influence on power consumption and heat dissipation of the radar platform. There are also some constraints on the configuration parameters that need to be fulfilled and some requirements pertaining to the environment sensing properties (e.g., maximum range, range resolution, angular resolution, and velocity resolution) that need to be met.

The main objective is to reduce the average power consumption of the device by using AI/ML to find optimal parameters for the device configuration and setup. Such systems can be then deployed in the field with the possibility of autonomous operation, e.g., with battery supply or solar power. Reducing heat dissipation is the second objective as it should correlate with lowering the average power consumption.

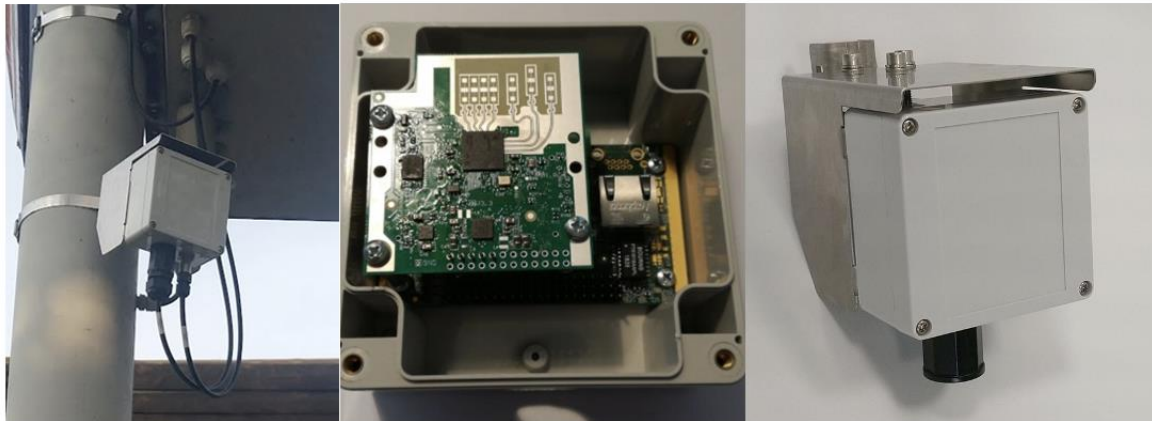


Figure 69 CAMEA traffic monitoring system (radar sensor)

The STGEM tool by ABO uses a general-purpose black-box online test generation algorithm called WOGAN [WOGAN] based on Wasserstein generative adversarial networks (WGAN). Initially, the algorithm generates a batch of random tests to be executed on the SUT, from which both the analyser and WGAN are trained. The main loop works as follows. First, a set of candidate tests are produced by the WGAN generator. Then these tests are validated by the analyser according to a minimum fitness threshold. The best test is then selected for execution. After that, the models used by the generator and analyser are retrained with the new ground truth obtained from the last test execution. This loop continues until the budget has been exhausted.

The tool is tasked to generate combinations of configuration parameters that try to minimise average power consumption and heat dissipation while providing the similar levels of performance. A device configuration is first generated/modified in the supporting SW, then checked for its validity and sent to the real physical device. Measurements (monitor data) for the physical device are then periodically sent back and analysed. Configuration testing is an iterative process of testing and refining various configurations online on the radar sensor.

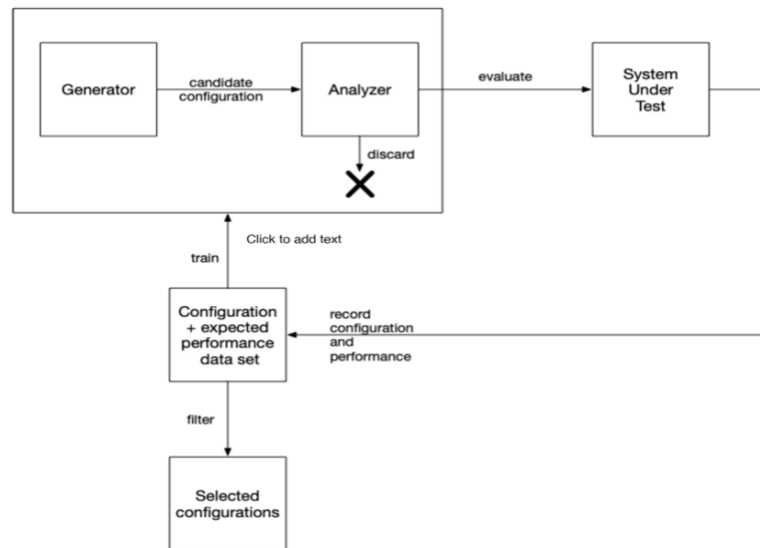


Figure 70 Overview of the proposed configuration generation and testing approach

The STGEM tool automatically creates test suites using no knowledge and generates as diverse test suites as possible in order to detect a varied set of faults and help development in a continuous integration setting. The novelty of our approach is to bring in true AI/ML (neural networks, generative networks etc.) into DEVOPS processes concerning the validation of CPSs in place of more traditional optimization-based methods.

4.17 Application in PRO Challenge – Automatic Resizing of Smart Port Computing Resources – PRO & ITI

We have employed ITI’s a2k/depman component in the smart port monitoring use case provided by the partner PRO – particularly the challenge related to automatic resizing of smart port computing resources. The objective is to dynamically resize the smart port computing resources in accordance with predicted incoming sensor traffic demands, which in turn depends on the amount of ship loading and unloading at different locations in the port at various times of the day.

The overall workflow is to use a simulation of the port computing infrastructure and measure its performance under different traffic conditions. In Figure 71 the System block is a model of the port infrastructure in terms of the computing hardware and software (both local and cloud servers), the Static Traffic Scenarios block is a simulator of the port sensor traffic. Within this block, we can define the number of sensors, their data rates, their message sizes, and their physical locations within the port area. The Performance Measurement block calculates the resulting performance of the system under the defined input traffic conditions. The Deployment Optimiser block is a component which attempts to adjust the system to maximise the performance. System adjustments include changing the numbers of replicas of software components and services, changing the locations of the software components, enabling more cloud-based resources (virtual machines, memory, etc.), and so forth.

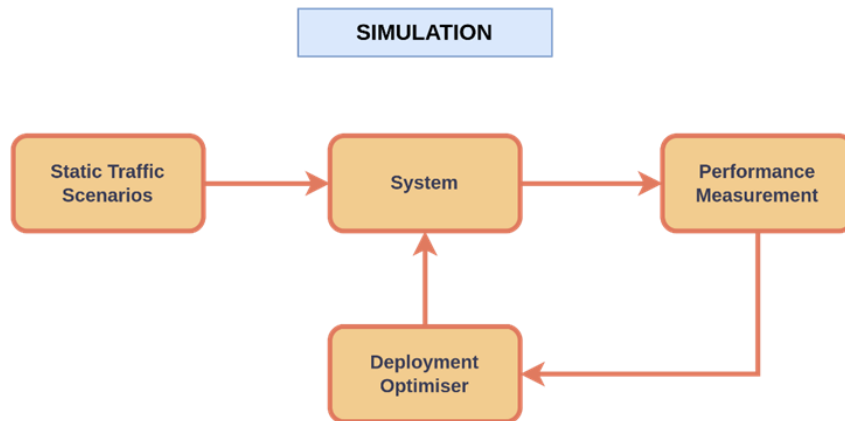


Figure 71 Architecture for Training the Smart Port Resizing

At present, we are using a multi-objective, genetic algorithm-based optimiser. This tool can handle several competing performance objectives under non-linear constraints. The goal of the optimiser is to find the best infrastructure architecture for a given static traffic scenario. This optimal deployment can be stored in a database for future recall. Multiple simulations under differing traffic conditions can be run and a learning machine is trained to match the incoming traffic loads with the optimal service deployment.

At run-time the architecture is shown in Figure 72. In this diagram we have introduced two new components. The “Pre-trained Learning Machine” is a component such as a neural network classifier which monitors the port sensor traffic (or predictions thereof) and selects the best system deployment to handle this traffic load. The second new component is the “Deployment Orchestrator”. This implements the logic to invoke system architecture changes (resizing) according to what is commanded by the Learning Machine.

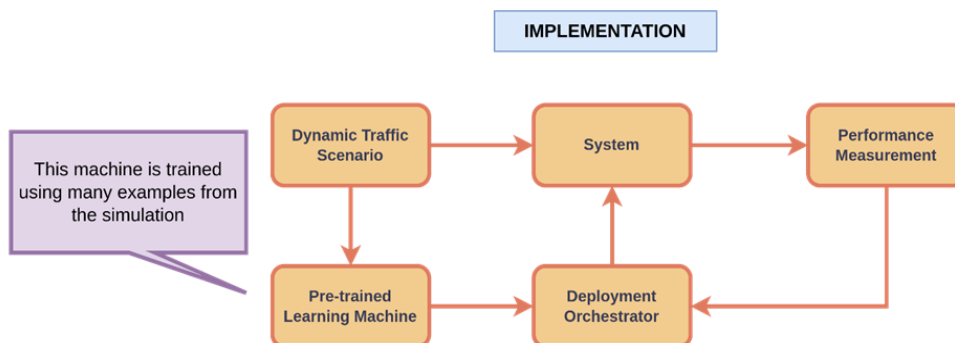
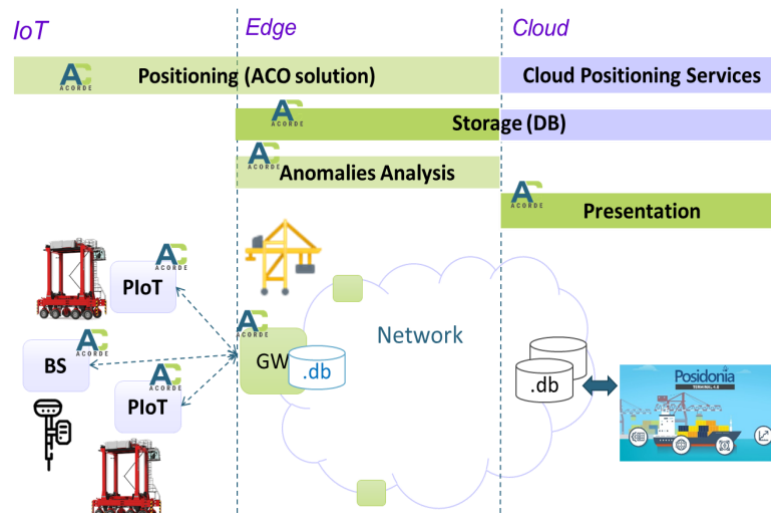


Figure 72 Run-time Dynamic Optimiser for the Smart Port

4.18 Application in PRO Challenge – ACORDE

ACORDE is intensively collaborating with Prodevelop to enable, in the time framework of the project, a piloting. This piloting requires and enables the collaboration of AIDOaRt with another running project, aerOS (<https://aeros-project.eu/>), where Prodevelop is involved.

The aerOS project aims to tackle the challenge of “transparently utilising the resources on the edge-to-cloud computing continuum for enabling applications in an effective manner while incorporating multiple services.”. This target is perfectly synergistic with some of the innovations brought by the AIDOaRt ACORDE-Prodevelop collaboration. It is sketched in the figure below, with emphasis on the contributions of ACORDE. These contributions comprise HW/SW monitoring infrastructure at IoT/Edge levels, and service solutions at the whole computing continuum range (IoT/Edge/Cloud). In light green developments requiring custom design and implementation are represented. In dark green, solutions where ACORDE has enabled adaptation/integration of services on the edge (i.e. edge database service) or has done specific configurations or plugins (e.g., dashboards for presentation at the cloud side).

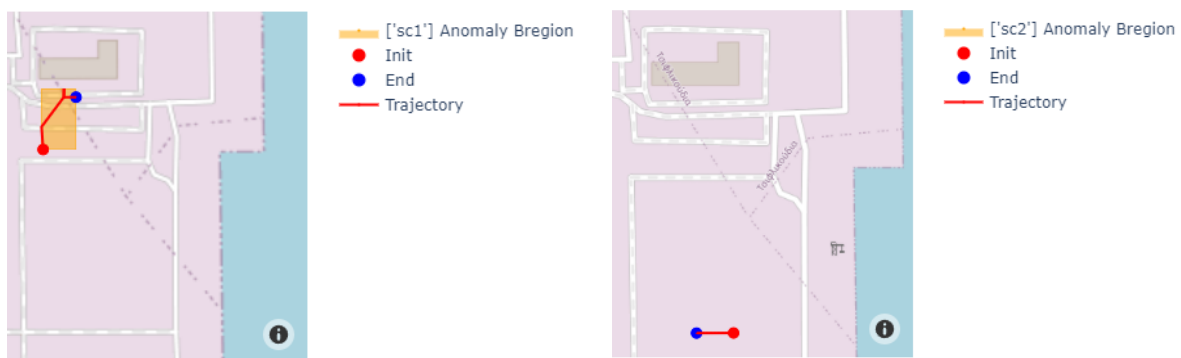


With regard to the HW infrastructure, ACORDE positioning solution developed in AIDOaRt is an optimal conjugation in terms of cost, precision and accuracy. The solution architecture has been adapted from the former proposal, which relied on a beacons network for resilience and accuracy. That solution was devised presuming STS cranes as the asset to be located. However, further collaboration with Prodevelop and the port stakeholders involved in the aerOS project, lead to an even more ambitious location target, to cover assets as the straddle carriers, which have far more mobility, and which require higher update rates (10Hz) on the positioning information. In the piloting being set up, a former phase is being tackled within the AIDOaRt timeframe, where the positioning sensing elements (PloT and BS) are being validated, by integrating them within the communications infrastructure already deployed on the port.

With regard to the application/SW services, as reported in section 3.2, application and configuration software of the base station (BS) and of the PloTs of the SW are allowing the smooth configuration and connection of the industrial positioning sensing solution to a 3rd party, already deployed infrastructure. Industrial interfaces (Modbus) as required by Prodevelop, are facilitating its smoother

connection to the industrial monitoring network. At the edge side, a modular gateway which can substitute conventional deployed one has been developed. This gateway is more suitable for continuum computing architectures, once it is able to run database and data analysis services smoothly at the edge, which brings relevant benefits for advanced monitoring solutions over the computing continuum. For instance, the edge data can keep local data irrelevant for the cloud. Edge analysis of edge (local) data, e.g. Anomalies Analysis services developed by ACORDE, enables smart decisions on which data to share with the cloud.

Figure below shows graphical reports on the Location Anomalies Analysis case introduced in section 3.2. This way port operator is enabled with automatic detection of anomalies, which otherwise could have been unnoticed, especially in subtle cases where the time variable is involved in the anomalous pattern. Therefore, it enhances sensitivity and autonomy on detection of port operations. Anomaly detection adds and complements to the detection of violation of pre-established rules. This means that the anomaly a), i.e. the SC going through the truck docking area, may represent an unallowed path (according to port operation manual) that could have been detected relatively easily with a conventional imperative algorithm. However, it could be also the case that such a path is allowed according to port formal operation rules, but that the straddle carriers always avoid such a path in its normal operation for pragmatic reasons (e.g., to avoid truck traffic and gain on safety and efficiency). Then, the DL-technology of the LAA service developed enables the capture of the “normality model” of the daily operation, and detect when that “normality” is broken, regardless of if it leads to any issue on the port operation or not. In any case, if any eventual issue happens, the maybe related anomaly is detected.



a) Location Anomaly detected on straddle carrier 1 related to location components.

b) Location Anomaly detected on straddle carrier 2 related to location in time.

Moreover, report figures also show the xAI capability implemented for root cause analysis. As the analysis is holistic, i.e. done for all the 2D location related variables of all the monitored SCs, the detected anomaly refers to all the system. The xAI analysis capability consists in assessing the contribution to the anomaly at the variable (location component) and also at the device level. The latter enables assessing if one device, or subset of devices among the monitored ones, is/are the main source of the anomaly. This xAI capability of the LAA service enables the port manager to properly guess the actual straddle carrier which originated each anomaly.

5 AIDOaRt Internal Hackathons Report

5.1 Introduction

This section describes the activity conducted during the AIDOaRt Internal Hackathons. They represent an intra-project activity that aims at enabling collaborations among partners and encouraging technical development progress by creating positive synergies between Use Case Providers (UCPs) and Solution Providers (SPs). The general objective of the hackathon was to develop experimental solutions that contribute to the AIDOaRt framework. In particular, each solution is an instance of the AIDOART architecture and meets the framework integration requirements [AIDOART 5.2].

We organised five internal Hackathons in conjunction with the plenary meetings of the project as follows:

- 1° AIDOaRt Internal Hackathon (2° Plenary meeting) Apr 2022 (Virtual)
- 2° AIDOaRt Internal Hackathon (3° Plenary meeting) Oct 2022 (L'Aquila, IT)
- 3° AIDOaRt Internal Hackathon (4° Plenary meeting) Feb 2023 (Valencia, ES)
- 4° AIDOaRt Internal Hackathon (5° Plenary meeting) May 2023 (Vasteras, SE)
- 5° AIDOaRt Internal Hackathon (6° Plenary meeting) Dec 2023 (Linz, AT)

After a planning phase, each group presented its solution that had been voted on by the other participants. The winner of each hackathon has been used as a showcase demo for the reviewing process in the corresponding deliverables.

This report is structured as follows. Section 5.2 gives an overview of the Hackathon organisation process in terms of required inputs and expected outputs. Section 5.3 describes the call for challenges template that has been used to define the groups and the main goal for each challenge. We conclude the report by summarising and discussing key findings from the hackathons, including both quantitative data and qualitative insights, in Section 5.4 and Section 5.5.

5.2 Hackathon Organization

The hackathons were organised as one-day events in the following way. Before the event, the organisers published the hackathon goal and the “call for challenges”. Afterward, the UCPs and SPs created the working teams based on the proposed topic. Each team included at least one technical person(s) from the UCP and the SPs, responsible for all the technical questions and providing detailed insights about the use case (scenario) and the challenge. Such information was collected in a shared document.

The goal was to identify a problem, define its boundaries, and implement a solution that could be experimented in 4 hours of joint work. Some partners failed to find a team due to different factors, e.g., none of the proposed challenges fit with their capabilities or needs. In this case, those participants published the challenge and asked for interested partners (in the “Request for additional members” field in the shared document). The interested partners answered in the “Expressions of interest” field in the shared document. During the event, the challenge and teams were pitched with a 5-minute presentation as the first step. Then, the teams moved to separate virtual/physical rooms to start the work. At the end of the day, the teams eventually pitched the result with a 7-minute

demo/presentation showing the progress, the encountered issues, and the future plans. In parallel, all the participants voted via live polls online (e.g., mentimeter.com) to rank the presentations, in terms of different factors. The organisers presented the results of the Hackathon and the two most voted challenges have been used as a showcase during the deliverables.

5.3 Call for Teams Challenges

As said, UCPs and SPs were invited to create working groups (i.e., teams) and provide a challenge for the specific hackathon edition. In the scope of the project, a challenge is a well-defined and limited experiment related to an AIDOaRt use case that can be explored/conducted in almost 4 hours.

Requirements:

On the day of the event, it was fundamental (and mandatory) to have concrete materials/artefacts (inputs) to be used in order to implement the experiment: actual models, source code, tools/environments, etc.

Each team was composed of at least one technical person from UC who participated in the hackathon, provided the inputs, and replied to related questions, and technical persons from SPs who participated in the hackathon and developed the experiment.

Topics:

- Possible topics related to the challenge:
- Model-driven engineering and artificial intelligence for DevOps
- AI/ML engagement and analysis solutions for the improvements of DevOps practices
- Data monitoring and management solution for AIOPS
- Automating development and operations by exploiting MDE
- Accountability and explainability for AI-based DevOps
- Framework integration aspects
- Further topics that belong to the scope of AIDOaRt

Note that the challenges could be new or the continuation of the previous hackathon challenges.

5.4 Hackathon report

To summarise the results, we conducted an internal report among the partners that participated in the hackathons. This section presents this report on the activities carried out during the hackathons, in terms of partner participations, challenges and solutions defined, and development activities.

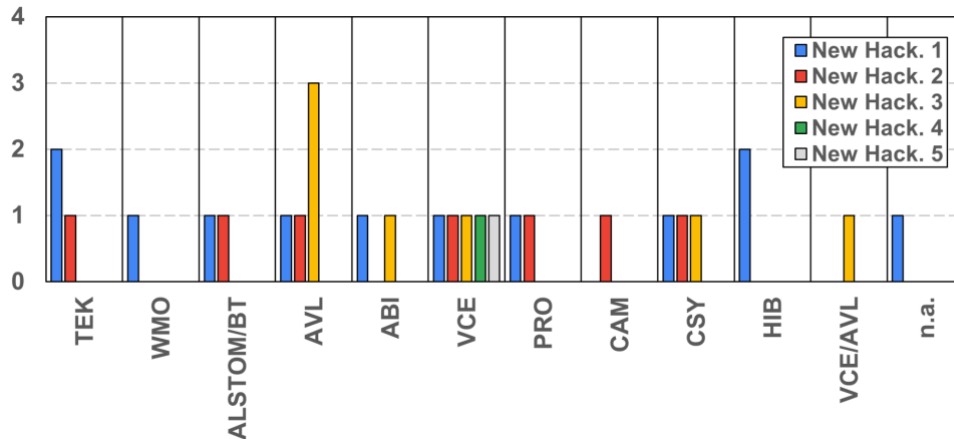


Figure 73 UCPs Proposed New Challenge per Hackathons

Firstly, Figure 73 reports the list of new challenges proposed in each hackathon, divided by UCPs. As expected, most UCPs proposed new challenges between the first and second hackathon. VCE can be considered an outlier as it proposed one new challenge for each hackathon. The last bar of the chart (i.e., with n.a. label) represent the challenges proposed by a SP (i.e., JKU and AST during the 1st Hackathon) that did not find any UCPs with this issue, so the challenge died in that edition.

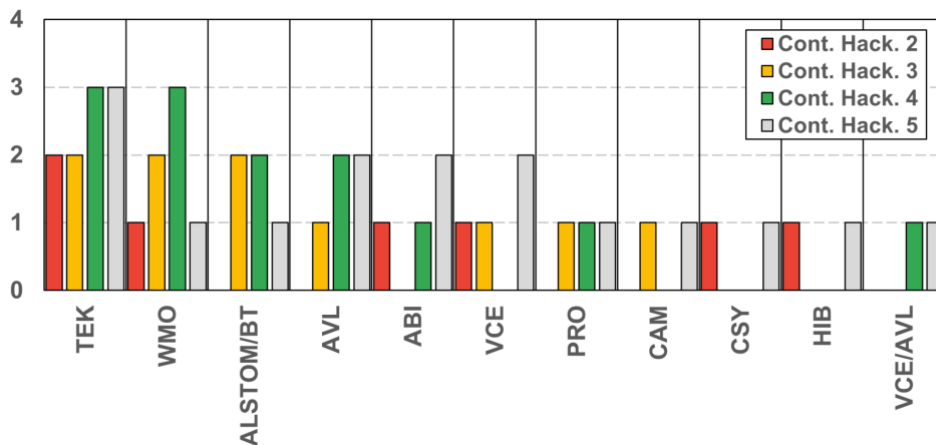


Figure 74 UCPs Continuous Number of Challenges per Hackathon

Figure 74 shows which challenges from previous hackathons were continued in later ones. Unlike Figure 73, the plot starts from the second hackathon. This is because UCPs introduced most of the new challenges in the first hackathon, then focused on continuing those challenges in the remaining hackathons. TEKNE and Westermo, for example, kept working on challenges across all hackathons editions.

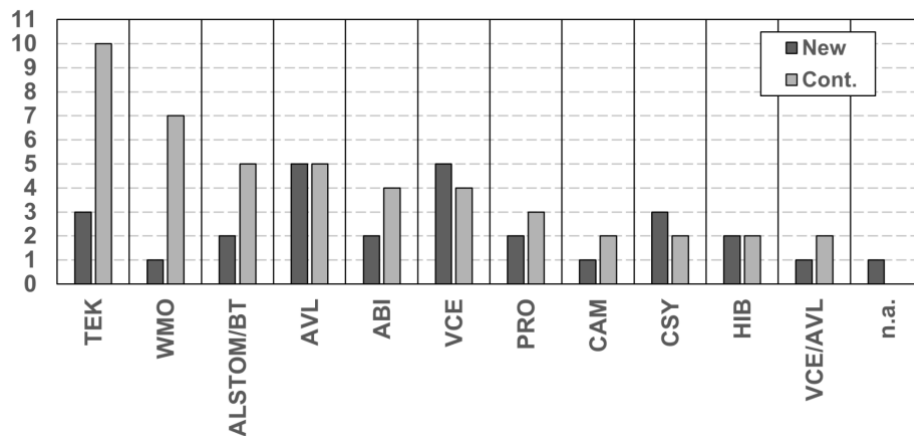


Figure 75 UCPs Total Number of New Vs Continuous Challenge

Figure 75 combines the information from Figure 73 and Figure 74. The figure highlights a clear trend: UCPs typically prefer to continue working on the challenges they originally proposed, rather than suggesting entirely new ones. However, there are a few exceptions where UCPs proposed a similar or even higher number of new challenges, mostly to look for SPs (e.g., CSYs) or to solve and address requirements and issues that were not considered at the beginning of the project but emerged during the hackathons (e.g., VCE).

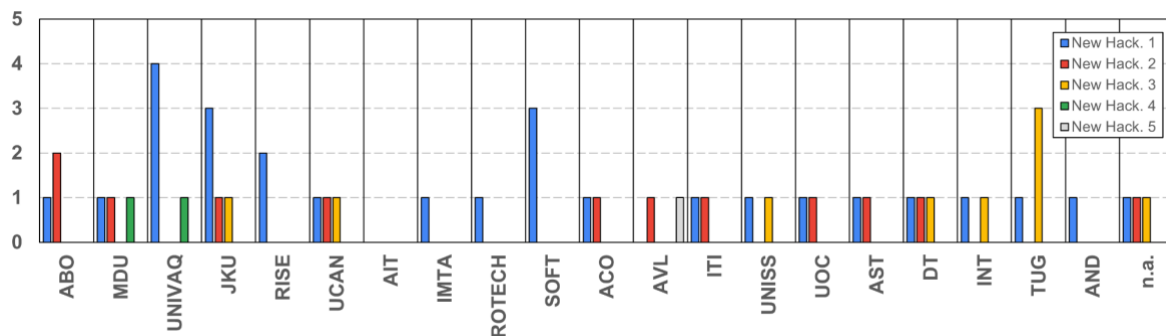


Figure 76 SPs Proposed New Challenge per Hackathons

Figure 76 shows statistics for new challenges proposed by UCPs which were attended by SPs. Similar to Figure 73 for UCPs, most SP attended challenges during the first hackathon. One exception is AVL, which attended a challenge also in the 5th hackathon. AVL is an exception also because it is both a UCP and an SP within the AIDOaRt project. Finally, the last bars of the chart (i.e., with n.a. label) represent the challenges proposed by the SPs (i.e., JKU and AST) or UCPs (i.e., CSY) that did not find any UCPs or SPs, respectively.

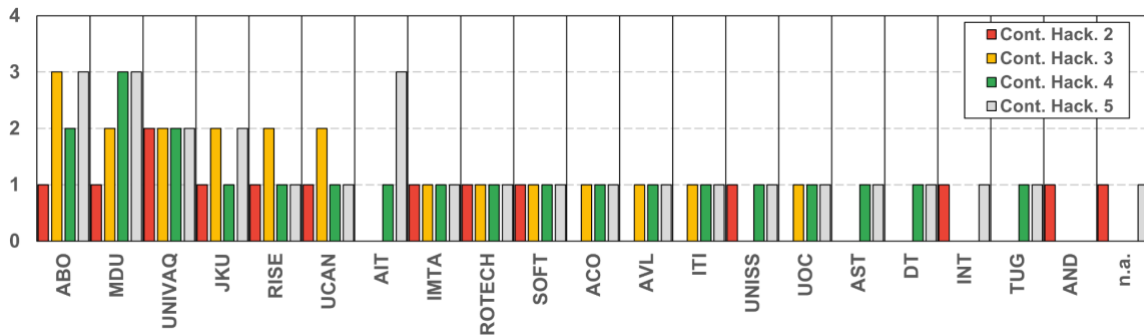


Figure 77 SPs Continuous Number of Challenges per Hackathon

Figure 77 shows which challenges from previous hackathons were continued by SPs in later Hackathons. Similar to Figure 74 for new challenges, some SP continued their challenges across all editions. For example, UNIVAQ continued working on two challenges they introduced in the first hackathon, while also participating in a new challenge proposed by VCE in the fourth hackathon.

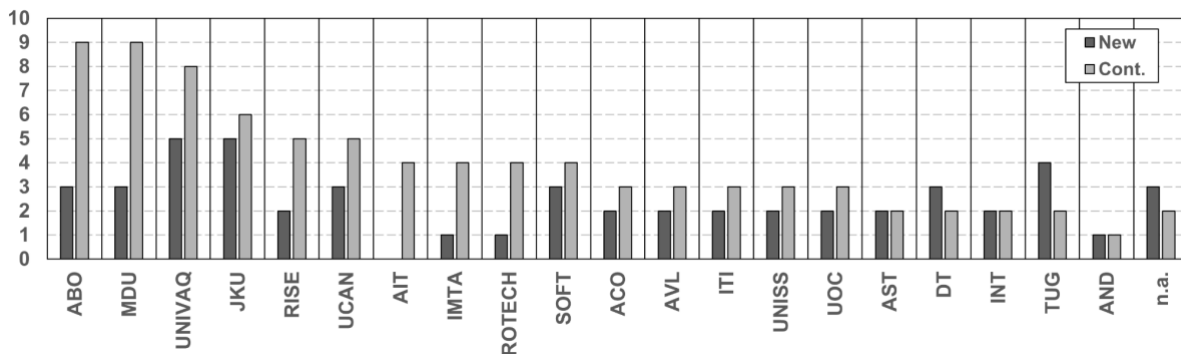


Figure 78 SPs Total Number of New Vs Continuous Challenge

Figure 78 combines the information from Figure 76 and Figure 77, summarising how SPs participated in challenges across the hackathons. Notably, most SPs continued working on more challenges than they initially attended. For example, ABO, MDU, and UNIVAQ persevered with 8 or 9 continuous challenges across all hackathons. In contrast, TUG focused on new ideas, participating in 4 new challenges while continuing only 1 existing ones over 2 hackathons.

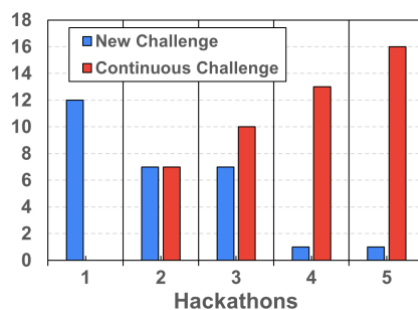


Figure 79 Total Number of New Vs Continuous Challenges

Figure 79 shows the overall trends in new versus continued challenges. As expected, the number of new challenges proposed decreases as the number of continued challenges increases.

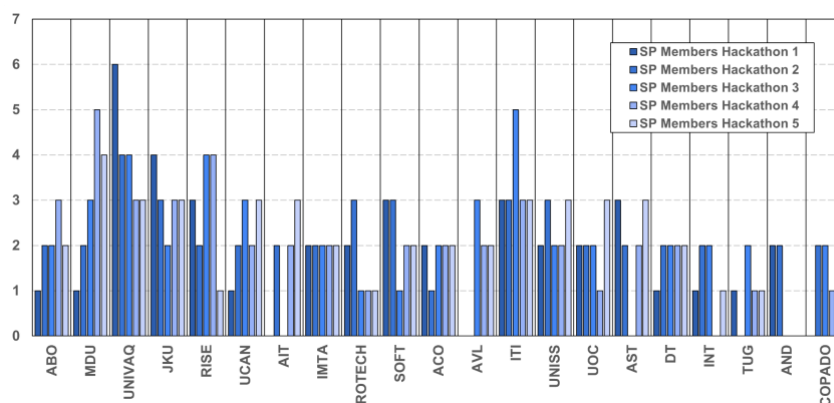


Figure 80 Number of SP Participants (i.e., Members) per Hackathon

Figure 80 shows the number of participants for each SP across the hackathons. The number of participants generally increased for most SPs, likely due to the growing number of challenges they faced. However, UNIVAQ saw a decrease in participants, possibly because they dropped some challenges.

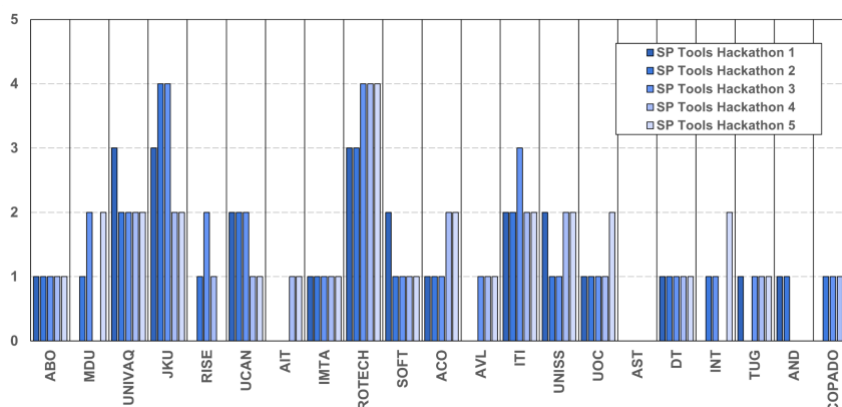


Figure 81 Number of SP Tools Implemented/Improved per Hackathon

Figure 81 shows the number of tools employed in each hackathon per SPs. It is worth noting that there is a general increasing trend in the SP tools exploitation, likely due to the increasing needs of UCPs across the hackathons.

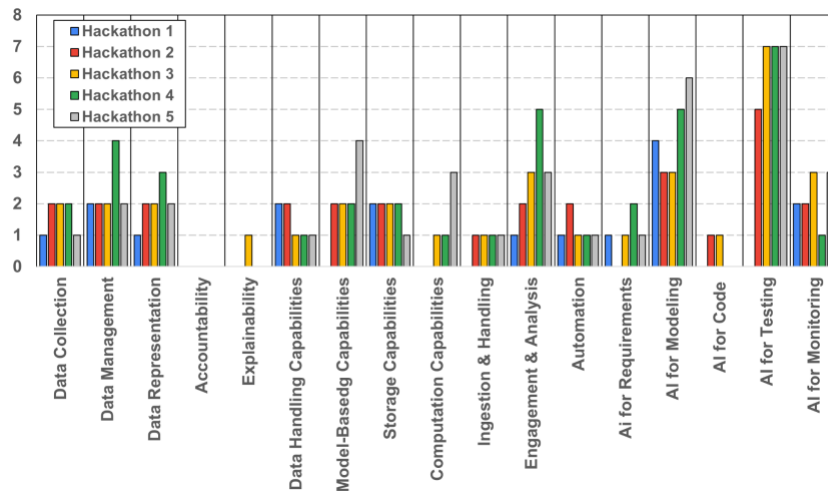


Figure 82 Total Number of Implemented AIDOaRt Components per Hackathon

Figure 82 shows the AIDOaRt framework components exploited during the hackathons. Notably, there is a higher number for “AI for Modeling” and “AI for Testing”, while less number can be found for “Accountability”, “Explainability”, and “AI for code”. This result is in line with the project results.

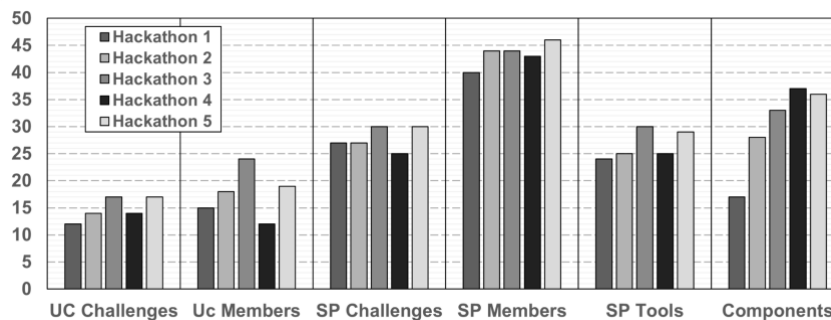


Figure 83 Final Hackathons Statistics

Finally, Figure 83 summarises all the key statistics. We see a general upward trend in challenges, UCP and SP participation, tools used, and implemented AIDOaRt components across all the hackathons. These results show that the project was able to maintain constant and coherent progress concerning the pre-established objectives, from the point of view of intermediate products and collaboration in hackathons. Moreover, the increase in “Components” likely reflects the complex requirements and challenges defined by UCs. Overall, these results are aligned well with the goals and objectives of the AIDOaRt project.

5.5 Hackathon survey

To further discuss the qualitative aspects, we conduct an internal survey to collect the feedback of the participants on various aspects of the Hackathons. This survey is based on 37 answers, 28 (24.3%) were from SPs and 9 (75.7%) from UCs. Note that at least one answer from each partner was provided. The survey was structured as follows. Section 5.5.1 reports the participants role in the project and their domain area. Section 5.5.2 concerns participation in the hackathon events in terms of team

composition and the generic process adopted. Meanwhile, we discuss in detail how the partners implement their solution in Section 5.5.3 and how it impacts the hackathon continuation in Section 5.5.4. We conclude the survey by asking additional suggestions or takeaways in Section 5.5.5.

5.5.1 Participant Role and Domain Areas

Figure 84 represents the Work Packages (WPs) of the project and the type of organisation respectively. Concerning the WPs, we see that most of the partners involved in the hackathon contribute to the WP4, which is devoted to the actual implementation of the AI-augmented solutions. Concerning the involved organisations, 21 (58.8%) are universities, followed by 9 (24.3%) large companies, 7 (18.9%) from SME and 7 (18.9%) from research institutes.

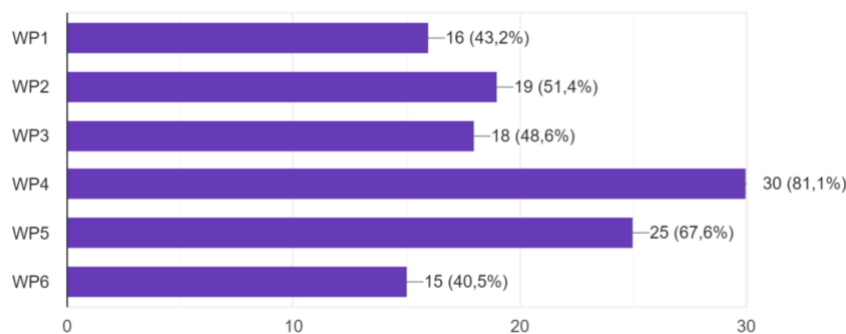


Figure 84 WPs participations

To better characterise the participants in terms of domain area, we asked them to indicate the general domain area, such as Automotive, Railway, or Manufacturing, and the specific ones, e.g., Model-Driven Engineering, or IoT. The results are reported in Figure 85 and Figure 86. We report that the main domain area is Automotive, represented by 11 participants, followed by Manufacturing and Railway. The three main areas, considered both industrial and academic partners, are AI/ML, Model-Driven Engineering, and Cyber-Physical Systems, followed by Testing, DevOps and IoT.

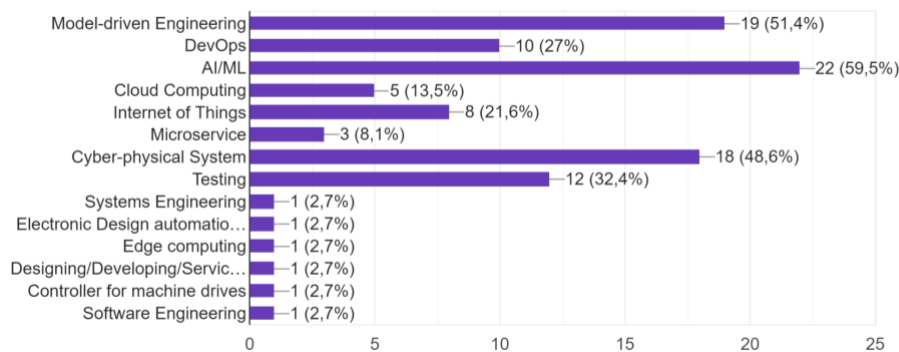


Figure 85 Research or business areas

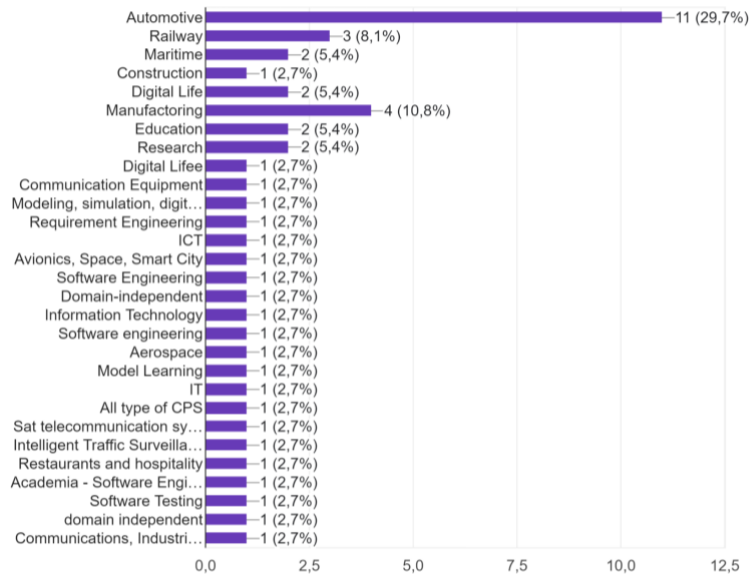


Figure 86 Application domain

Afterward, we asked the participants to declare their level of expertise in the indicated application domain areas, as shown in Figure 87.

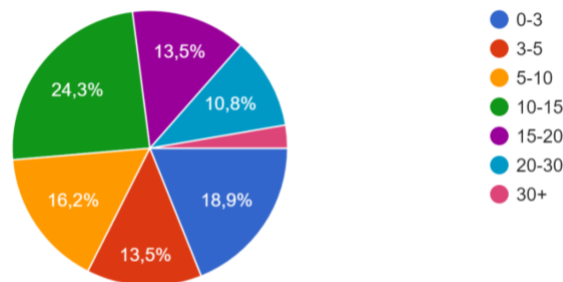


Figure 87 Expertise (in years) in the application domain areas

Furthermore, we assess the expertise level in the three key areas of the project, i.e., DevOps, MDE, and AI that are reported in Figure 88 a), b), and c). Notably, most of the hackathon participants declare more than 5 years in their domain area but less than 3 years in the key areas of the project.

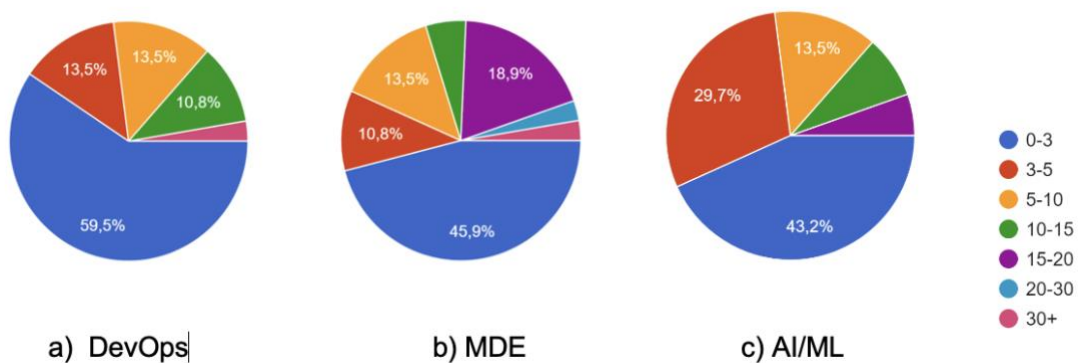


Figure 88 Expertise (in years) in the three key areas

5.5.2 Experience in the AIDOaRt Internal Hackathons

This section describes the general engagement by reporting the attendance of the survey participants through the different hackathons in Figure 89. Overall, the participants who started the hackathons continued working until the last one.

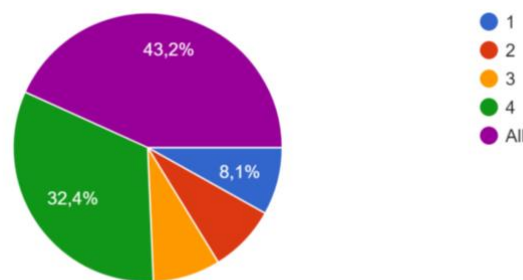


Figure 89 Hackathon attendance (no. of editions)

Furthermore, only a small percentage of the teams was limited to one area, resulting in an adequate level of interdisciplinary during the events (i.e. 89.2% of the survey participants answered that they worked in an interdisciplinary team).

To further analyse the overall organisation, we asked a set of open questions to assess how the teams were formed and to what extent the UCPs and SPs contributed to the overall solution development. Altogether, we report that most of the partners defined the challenges and the corresponding teams in the first hackathon. Concerning the contributions, the UCPs narrowed down the scope of the challenge while SPs reuse or develop technical solutions to address the overall goal.

5.5.3 Participation in the AIDOaRt Internal Hackathons

This section aims to report the planned activities and the process carried out to develop the envisioned solution. We adapt the widely adopted Diamond Diagram methodology [AIDOART 5.2] to characterise the hackathons in the context of the project. In particular, each hackathon encompasses four different phases, i.e., Challenge definition, Solution Definition, Solution Development, and Delivery Phase.

We carefully checked the current literature to identify design patterns and methodologies commonly used in hackathons [FIL2017]. Since the participants were free to choose the methodology that fits the proposed challenge, we didn't recommend a precise pattern for each phase. Instead, we left the possibility to specify the patterns used during each event. Figure 90 shows that the most frequent pattern used for the challenge definition phase was Brainstorming, followed by Brainwriting and Scenarios.

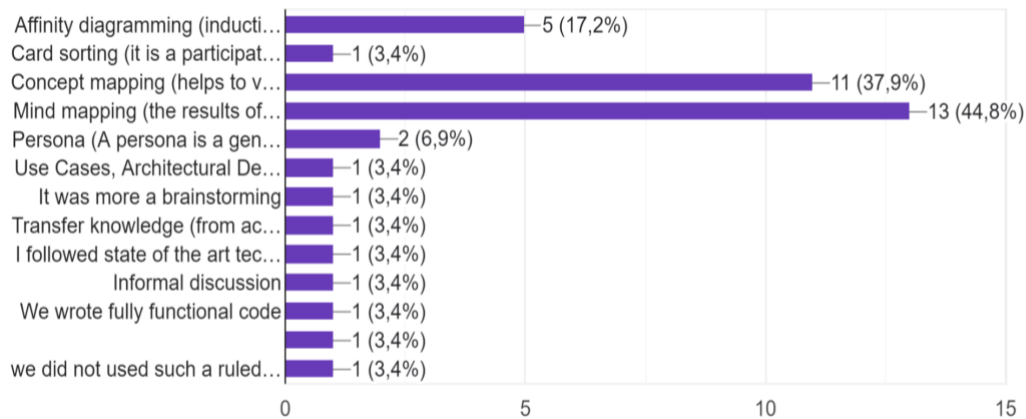


Figure 90 Challenge definition phase

To define the solution, using the mapping techniques, i.e., MindMapping and ConceptMapping, results as the most spread strategy as reported in Figure 91. Interestingly, brainstorming has been adopted as the major methodology in the Solution.

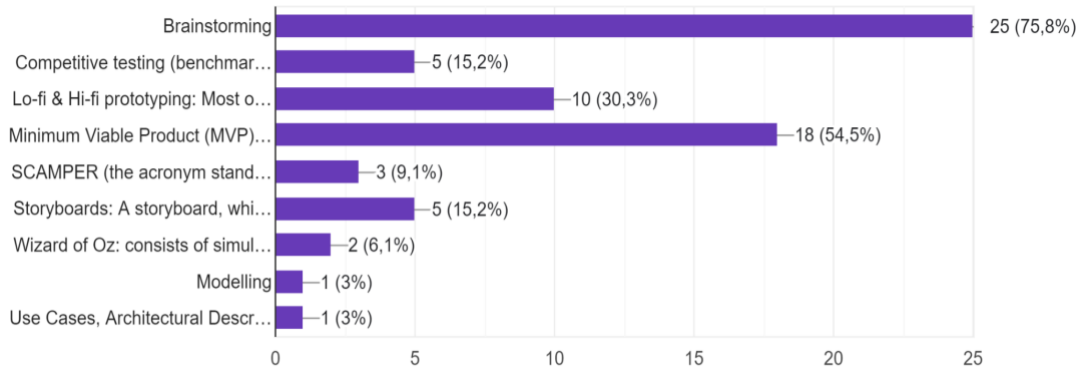


Figure 91 Solution definition phase

Development phase even though 18 partners also adopted the Minimum Viable Product (MVP) to reuse their internal know-how. Furthermore, Figure 92 shows that the participants implement the Lo-fi&Hi-fi prototyping to test their solution. Since the delivery phase has been fixed as a pitch, we asked open questions to assess the overall satisfaction instead of defining a set of closed questions. We report that different participants deliver the solution as an early-stage prototype and the collaborations lead to different scientific articles published in peer-reviewed venues relevant to the project topics.

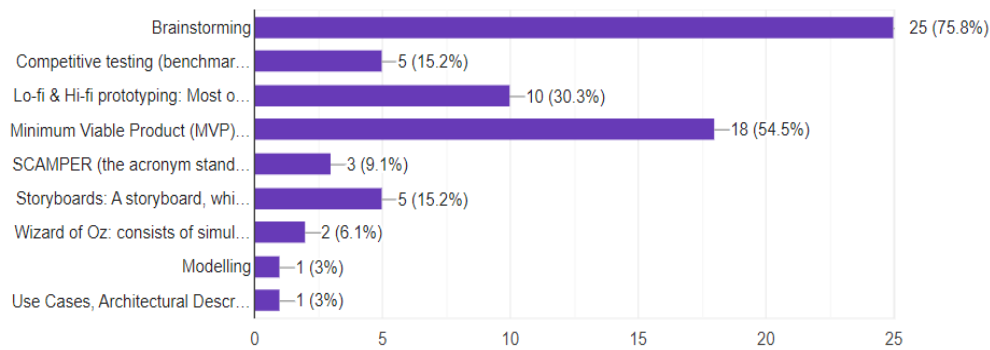


Figure 92 Solution development phase

At the end of these questions, we asked the participants to give an overall judgement of the process in terms of satisfaction. Figure 93 reports the judgement for the four conducted phases, i.e., Challenge definition, Solution Definition, Solution development, and Deliver phase.

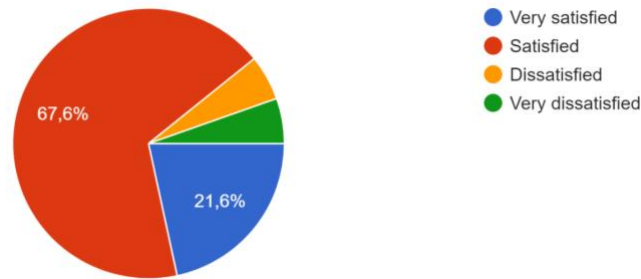


Figure 93 Overall satisfaction

In particular, the most challenging phase is the actual development of the conceived solution while the simpler phase to handle was the challenge definition as reported in Figure 94 a) and b) respectively. Overall, the conceived process was satisfying for most of the partners as shown in Figure 93, with only four participants who expressed negative judgments, i.e., two dissatisfied and two very dissatisfied.

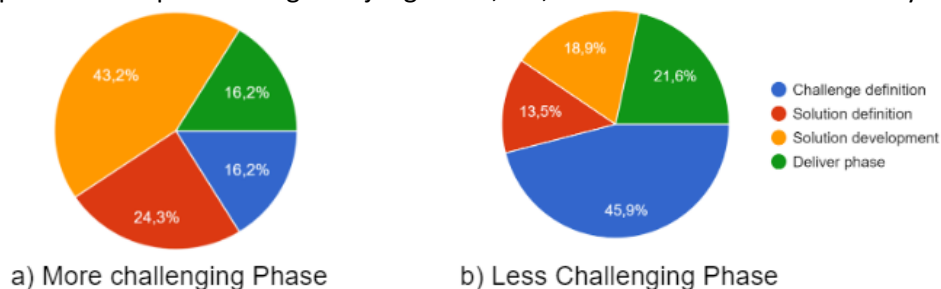


Figure 94 Phases satisfaction

5.5.4 Hackathon continuation

This section reports the results in assessing the continuation of the various challenges throughout the different hackathon events. It is worth noting that most of the survey participants (67.6%) answered that there have been continued challenges. Furthermore, Figure 95 reports a few cases of deviation from the initially adopted methodology, i.e. 27 participants (73%) declared no deviation from the

originally conceived process. Meanwhile, 4 participants (10.8%) reported some deviation in the development solution, mostly to improve the overall performance or due to some external issues.

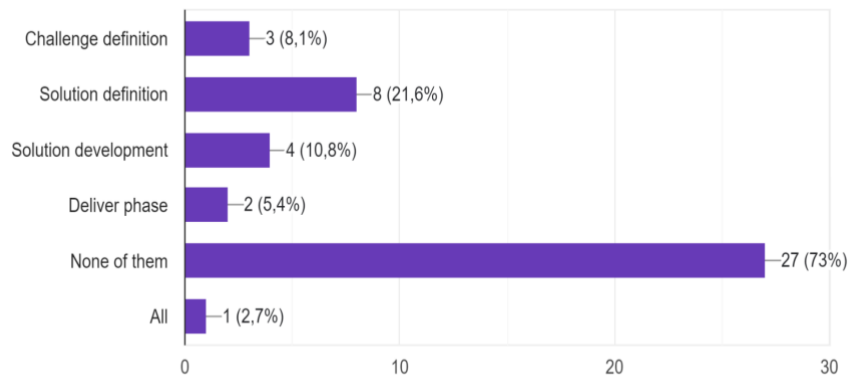


Figure 95 Methodology deviation

To better understand the root of the aforementioned deviations, we identify a set of well-founded issues that are typically encountered during hackathons [GAMA2023]. In particular, Figure 96 and Figure 97 show the positive factors and negative factors respectively. Among the positive factors that foster the challenge continuation, the leading role of UCPs plays a crucial role, followed by the technical preparation of SPs. Meanwhile, changes that occurred in the team composition negatively impacted the challenge continuation, resulting in the challenge interruption or modification.

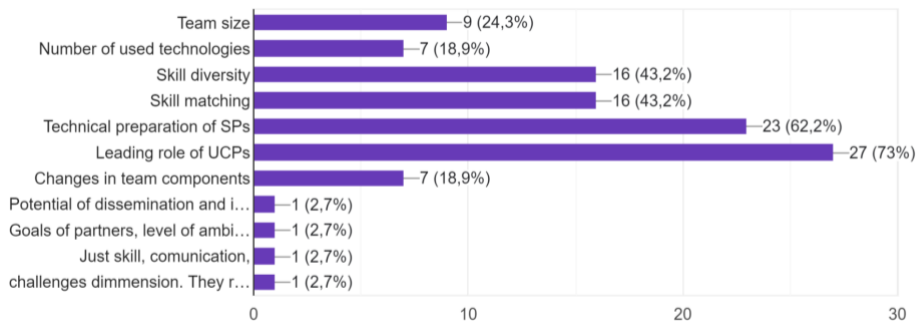


Figure 96 Positive factors

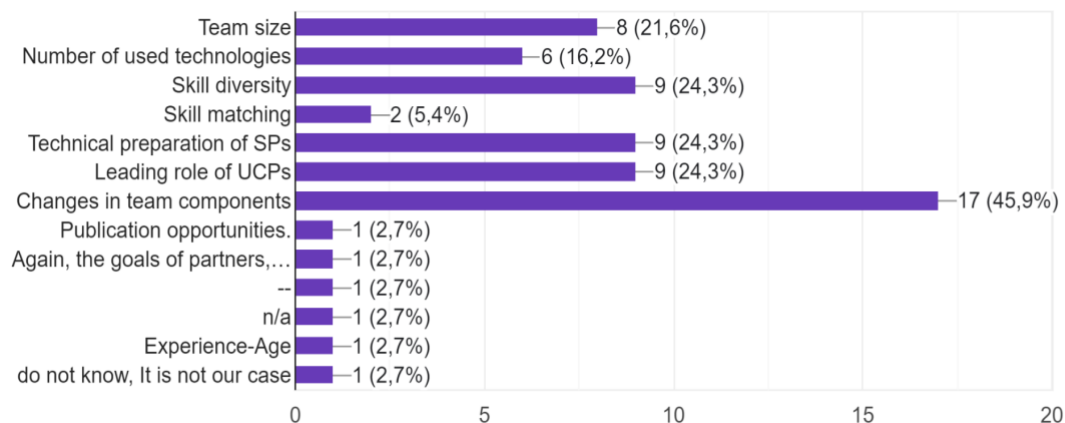


Figure 97 Negative factors

5.5.5 Other information

We eventually asked the participants to give additional feedback, mainly focused on collecting suggestions to improve the hackathons or reporting how the hackathons events contribute to advancing the project deadlines. This additional information can be found in a separate CSV file [HACK-RESULTS].

6 Conclusions

In this deliverable, we have presented the status of the activities carried out in WP4 (M25-M37). The work reported in D4.3 is in the direct continuation of the WP4 activities that already resulted in the deliverable [AIDOART-D4.1] providing the initial specification of the AIDOaRt AI-Augmented tool set, and the deliverable [AIDOART-D4.2] describing an intermediate version of the tool set.

The main objective of D4.3 was thus to consolidate the work carried out during the whole duration of WP4 and to refine the description and status of the AIDOaRt AI-Augmented tool set according to the latest information collected from both Use Case Providers and Solution Providers. In particular, D4.3 showcased the latest advancements of the tools compared to the previous deliverable [AIDOART-D4.2]. Our efforts have been directed towards elucidating the contexts and purposes for which these solutions have been implemented within the realm of CPSs. By detailing the progress made in tools development, we aim to underscore the significant strides taken since D4.2 and provide a comprehensive overview of the innovative solutions tailored for addressing CPS challenges. Another goal of D4.3 was also to report on more recent practical applications of solutions from the AIDOaRt tool set in the context of project's use cases and of their related CPSs as well as to provide some statistics and reports about the hackathons' events and. Finally, in the Appendix the mapping of the related solutions to the interface level of that AI-Augmented tool set and the mapping of the Use Case Requirements and the Use Case Data Requirements to respective interfaces has been reported.

As the concluding deliverable in this series, D4.3 represents the culmination of WP4 efforts, offering the current snapshot of the solutions developed. The present status of the AIDOaRt tool set demonstrates overall favourable progress both in terms of developments and applications. As evidenced in D4.3, and also noted in preceding deliverables, the array of solutions identified effectively addresses a diverse range of use case challenges and associated scenarios. At this juncture in the project's timeline, there is a noticeable increase in the utilisation of different components of the AIDOaRt AI-Augmented across various use cases, reflecting the growing practical application and relevance of the tool set within real-world contexts.

Nevertheless, the solution providers will continue in their efforts to refine their individual solutions, engaging in ongoing collaboration with the use case providers to address various scenarios throughout the project's duration. This ongoing collaboration ensures the continued consolidation and practical implementation of the diverse solutions within the AIDOaRt AI-augmented tool set. These efforts will be further advanced within the context of WP5, facilitating the consolidation and integration of these solutions into the AIDOaRt framework and across diverse use cases. The outcomes of these continuous advancements will be integrated in the forthcoming deliverables D5.8 and D5.9. Importantly, this integration process will likely extend beyond the project's conclusion, ensuring the sustained evolution and application of the AIDOaRt AI-Augmented tool set.

7 References

- [AIDOART-D1.1] AIDOaRt consortium: Use cases requirements specification, 2021. (Access restricted to the AIDOaRt project's consortium)
- [AIDOART-D1.3] AIDOaRt consortium: Use cases scenarios and evaluation criteria definition, 2021. (Access restricted to the AIDOaRt project's consortium)
- [AIDOART-D1.4] AIDOaRt consortium: Architecture Specification Final Version, 2022. (Access restricted to the AIDOaRt project's consortium)
- [AIDOART-D2.1] AIDOaRt consortium: Data collection and representation - Initial Version, 2022. (Available online at: <https://www.aidoart.eu/download/18.7d39d59b181260bc66b12343/1654680192979/D2.1.pdf>)
- [AIDOART-D2.2] AIDOaRt consortium: Data collection and representation - Intermediate Version, 2022. (Available online at: <https://www.aidoart.eu/download/18.3022ca0a184b9251f0a105bb/1670242565559/AIDOaRt%20D2.2%20Data%20Collection%20and%20Representation%20-%20Interim%20Version.pdf>)
- [AIDOART-D3.1] AIDOaRt consortium: Report on Foundations of MDE and AIOPS for DevOps, 2021. (Available online at: <https://www.aidoart.eu/download/18.1b4dc71217eaa69c2fa5499d/1644265570830/D3.1.%20Report%20on%20Foundations%20of%20MDE%20and%20AIOPS.pdf>)
- [AIDOART-D3.3] AIDOaRt consortium: AIDOaRt Core Infrastructure and Framework – intermediate version, 2022. (Available online at: https://www.aidoart.eu/download/18.3022ca0a184b9251f0a105bd/1670242714911/AIDOaRt_D3-3%20_AIDOaRt-Core-Infrastructure-and-Framework-Intermediate-Version.pdf)
- [AIDOART-D4.1] AIDOaRt consortium: AIDOaRt AI-Augmented tool set - initial version, 2022. (Available online at: <https://www.aidoart.eu/download/18.3022ca0a184b9251f0a105c0/1670243232895/D4.1%20AIDOaRt%20AI-Augmented%20tool%20set%20-%20%20Initial%20Version.pdf>)
- [AIDOART-D4.2] AIDOaRt consortium: AIDOaRt AI-Augmented tool set - intermediate version, 2023 (this document).

- [AIDOART-D5.1] AIDOaRt consortium: AIDOaRt Integration Approach, 2022 (Available online at: <https://www.aidoart.eu/download/18.7d39d59b181260bc66b12340/1654680145599/D5.1.pdf>)
- [AIDOART-D5.2] AIDOaRt consortium: AIDOaRt Integrated Framework, 2023 (Available online at: <https://sites.mdu.se/download/18.4f6730fe18b37a742932d69/1697541266402/D5.2.pdf>)
- [FIL2017] Filippova, A., Trainer, E., & Herbsleb, J. (2017). From Diversity by Numbers to Diversity as Process: Supporting Inclusiveness in Software Development Teams with Brainstorming. <https://doi.org/10.1184/R1/6622319.v1>
- [GAMA2023] Gama, K., Valença, G., Alessio, P., Formiga, R., Neves, A., & Lacerda, N. (2023). The Developers' Design Thinking Toolbox in Hackathons: A Study on the Recurring Design Methods in Software Development Marathons. *International Journal of Human-Computer Interaction*, 39(12), 2269–2291. <https://doi.org/10.1080/10447318.2022.2075601>
- [HACK-RESULTS] Hackathon report sheet and Hackathon report summary and charts
- [HACK-SURVEY] Hackathon survey answer's link.
- [NOLTE2018] Nolte, A., Pe-Than, E. P. P., Filippova, A., Bird, C., Scallen, S., & Herbsleb, J. D. (2018). You Hacked and Now What? - Exploring Outcomes of a Corporate Hackathon. *Proceedings of the ACM on Human-Computer Interaction*, 2(CSCW), 129:1-129:23. <https://doi.org/10.1145/3274398>
- [RiFa21] G. Rimassa and F. M. Facca. Digital Autonomy in the Computing Continuum: From Cloud to Edge to IoT for European Data. 11Nov. 2021. Available at <https://www.h-cloud.eu/news/highlights-of-the-ec-workshop-digital-autonomy-in-the-computing-continuum/>
- [Zero-Shot] W. Alhoshan, L. Zhao, A. Ferrari, and K. J. Letsholo, "A zero-shot learning approach to classifying requirements: A preliminary study," in *International Working Conference on Requirements Engineering: Foundation for Software Quality*. Springer, 2022, pp. 52–59
- [YOLOv3] J. Redmon, and A. Farhadi, "YOLOv3: An Incremental Improvement", 2018. <https://arxiv.org/abs/1804.02767>
- [KITTI] A. Geiger, P. Lenz, C. Stiller, R. Urtasun, "Vision meets robotics: The KITTI dataset", *The International Journal of Robotics Research*. 2013; 32(11): pp. 1231-1237
- [MSNet] F. Shamsafar, S. Woerz, R. Rahim, and A. Zell, "Mobile Stereonet: Towards lightweight deep networks for stereo matching", in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 2417-2426



- [SceneFlow] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, & T. Brox, "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation", in Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4040-4048.
- [SGMM2017] H. Spieker, A. Gotlieb, D. Marijan, and M. Mossige, "Reinforcement learning for automatic test case prioritization and selection in continuous integration", in Proceedings of the 26th ACM SIGSOFT International Symposium on Software Testing and Analysis, ISSTA 2017, pp. 12–22.
- [TSP2024] A. Torbunova, P. E. Strandberg, and I. Porres, "Dynamic Test Case Prioritization in Industrial Test Result Datasets", arXiv preprint arXiv:2402.02925, 2024.
- [WOGAN] Jarkko Peltomäki, Frankie Spencer, and Ivan Porres. 2023. Wasserstein generative adversarial networks for online test generation for cyber physical systems. In Proceedings of the 15th Workshop on Search-Based Software Testing (SBST '22). Association for Computing Machinery, New York, NY, USA, 1–5. <https://doi.org/10.1145/3526072.3527522>

Appendix

A. Mapping Solutions to AIDOaRt AI-Augmented tool set Components

In this section, we present the list of solution components realising the functional specification of the AI-Augmented tool set components. This mapping has been defined by the Solutions Providers based on the potential of their proposed solutions in fulfilling the description, specification, and functional interfaces of each component of the AIDOaRt Framework Architecture.

i. Mapping to Ingestion & Handling

In this section, we present the list of solution components realising the "Ingestion & Handling" component of the AI-Augmented tool set. Each solution name is followed by the partner acronym between parentheses. The full partners names can be found in the Partners Acronyms table in the preamble of this document.

Solution Name	Rationale
HIB_logAnalyzer (HIB)	Acquire data to get the maximum quantity of data for the subsequent AI operations on the logs.
a2k-runman (ITI)	Acquired and filtered data will be used as input to the anomaly detection algorithm.
HEPSYCODE (UNIVAQ)	UNIVAQ contributes to the ingestion of data by means of metrics definition and data analysis and selection. The data will be used for AI/ML algorithms both for prediction and learning activities.
MORGAN (UNIVAQ)	Given the input model, MORGAN employs a tailored parser to extract structural features. In particular the tool supports the data handling for metamodels and models expressed as ecore and XMI respectively.

Table 70 Solutions Mapping to the "Ingestion & Handling" Component

Solution Name	Rationale
HIB_logAnalyzer (HIB)	Acquire data to get the maximum quantity of data for the subsequent AI operations on the logs.
a2k-runman (ITI)	The a2k/monitoring service is used to manage the collection and pre-processing system performance and sensor data at run-time.

Table 71 Solutions Mapping to the "IF-CONTINUOUS-MONITORING" Interface

Solution Name	Rationale
HEPSYCODE (UNIVAQ)	UNIVAQ helps with data ingestion by defining metrics and analysing and selecting data. The data is used by AI/ML algorithms for both learning and prediction tasks. The AI/ML algorithms use information that is shared across all design phases.

MORGAN (UNIVAQ)	Our solution provides data selection capabilities by using a set of dedicated parsers for each type of model artefact supported, i.e. SysML, XES traces, and AutomationML.
------------------------	--

Table 72 Solutions Mapping to the "IF-DATA-SELECTION" Interface

Solution Name	Rationale
HEPSYCODE (UNIVAQ)	Cleaning and statistical activities are introduced in HEPSYCODE to prepare the datasets, while the information is passed through the engagement and analysis components.

Table 73 Solutions Mapping to the "ML-based Dataset Balancing" Interface

ii. Mapping to Engagement & Analysis

In this section, we present the list of solution components realising the "Engagement & Analysis" component of the AI-Augmented tool set. Each solution name is followed by the partner acronym between parentheses. The full partners names can be found in the Partners Acronyms table in the preamble of this document.

Solution Name	Rationale
Position Monitoring for Industrial Environment (ACO)	ACORDE aims to exploit AI/ML techniques to detect anomalies based on the monitored data, considering multiple sources and solving challenges like huge amounts of data, difficult to process in real time, or too many or too frequent alarms. Capabilities to predict potential alarms are also of interest. All this can impact or lead to better and faster reactions, and thus improvements in the control loop of the monitored system.
INT-DEPTH (INT)	This component supports the deduction capabilities based on deep learning architecture. Particularly, it allows retrieval of the depth estimation of objects (vehicles, pedestrians, etc.). This estimation is based on images coming from cameras installed in intelligent vehicles.
INT-DET (INT)	This component supports the prediction capabilities based on deep learning architecture. Particularly, it allows the detection of objects (vehicles, pedestrians, etc.) from images coming from cameras installed in intelligent vehicles.
a2k-depman (ITI)	Multiobjective optimization for system design space exploration possibilities. Analysis of monitoring data to effect mode changes at run time.
a2k-runman (ITI)	The component will analyse the monitoring data to detect abnormal situations and determine the root cause of these events.
S3D (UCAN)	The tools suite in S3D (MAST) will allow performance estimations even at the analysis phase by using temporal budgets of initial behavioural models. Response times will be linked to behavioural models for concrete analysis contexts, this enables design space exploration and further optimization of design parameters.

Solution Name	Rationale
SoSIM (UCAN)	SoSIM will be explored as a source of valid training data to program run time validation strategies. This tool will handle and generate performance indicators that can be improved in design space exploration optimization processes.
UNISS_SOL_02 (UNISS)	This solution provides automated verification of Neural Networks (NN) using AI/ML techniques.
UNISS_SOL_04 (UNISS)	This solution provides automated formal consistency verification of a properties' system design expressed in a given Domain System Language (DSL) by using AI techniques.
HEPSYCODE (UNIVAQ)	UNIVAQ contributes to the use of ML and AI for correlation/aggregation of processor execution times. Analysis is used to predict function/component timing and possible energy/power consumption patterns. Other non-functional metrics will be analysed to improve system development at various levels of abstraction. UNIVAQ will integrate an offline design space exploration activity to find alternative patterns using metrics and data useful for system improvements.
MORGAN (UNIVAQ)	MORGAN encodes relevant features using a set of standard NLP techniques i.e. stemming, dash removal. Afterward, this data is encoded and used to produce a list of graphs representing the model elements and their corresponding structural features.
TWIMO (UNITE)	It provides ML-based analysis and prediction capability.

Table 74 Solutions Mapping to the "Engagement & Analysis" Component

Solution Name	Rationale
a2k-runman (ITI)	a2k-runman collects information on system performance. It uses the a2k/detection and a2k/tuning services to detect abnormal or critical situations and changes the system mode to manage these conditions in a safe manner.
a2k-depman (ITI)	The a2k/optimiser service provides insight into the expected performance of different system architectures. This is used to provide deployment assistance using multi-objective design space exploration algorithms assisted by ML methods.
UNISS_SOL_02 (UNISS)	UNISS_SOL_02 provides analysis capabilities such as verification of Neural Networks based on AI/ML techniques.
UNISS_SOL_04 (UNISS)	UNISS_SOL_04 supports the consistency verification process of a system model through the application of formal methods.
HEPSYCODE (UNIVAQ)	UNIVAQ introduces feature analysis and extraction to improve system estimation and performance evaluation (e.g., Random Forest Regressor, Extra Trees Regressor, Gradient Boosting Regressor, Ada Boost Regressor, PCA).

Table 75 Solutions Mapping to the "IF-INSIGHT-ANALYSIS" Interface

Solution Name	Rationale
a2k-runman (ITI)	The a2k/detection service provides some ML and AI algorithms which are used to analyse system performance and sensor data to detect and/or predict abnormal conditions. This information is passed to the a2k/tuning service which can change the system configuration in response to detected abnormal or fault conditions.
HEPSYCODE (UNIVAQ)	UNIVAQ analysed optimal configuration and better solution search techniques for multi-objective optimization and system performance simulation algorithms using different AI paradigms and approaches (such as metaheuristics, evolutionary algorithms, and swarm particle algorithms). Innovative ML paradigms and algorithms are used to introduce ML strategies to determine system design and evaluate system performance.
MORGAN (UNIVAQ)	Our solution provides predictive analysis capabilities that can support modellers during their activities, i.e., model specification, by suggesting relevant items given the active context.
TWIMO (UNITE)	

Table 76 Solutions Mapping to the "IF-PREDICTIVE-ANALYSIS" Interface

Solution Name	Rationale
a2k-runman (ITI)	A2K is a multi user software tool. Users can share models and data. There is also the possibility of importing and exporting models in formats which enable data exchange with tools from other vendors.

Table 77 Solutions Mapping to the "IF-COLLABORATION-SERVICE" Interface

Solution Name	Rationale
Position Monitoring for Industrial Environment (ACO)	ACORDE is investigating and deploying a combination of techniques for retrospective analysis of anomalies. The objective is to find the earliest anomalies in the event causa chain eventually leading to the issue that caused an alarm The analysis is based on three phases, based on three related definitions of anomaly.
a2k-runman (ITI)	The a2k/detection service implements anomaly detection for system performance and sensor data. It does this using various types of ML based detection algorithms. These are trained from historical data sets, both real and simulated (via the a2k/scheduling service).

Table 78 Solutions Mapping to the "AI/ML for Anomaly Detection" Interface

Solution Name	Rationale
TWIMO (UNITE)	TWIMO provides ML-based analysis and prediction capabilities on the human driver behaviour in the automotive domain

Table 79 Solutions Mapping to the "ML-based Prediction for Human–Machine Interaction (HMI)" Interface

Solution Name	Rationale
INT-DET (INT)	INT-DET provides an AI-based algorithm for the detection of objects in the street driving domain.

INT-DEPTH (INT)	INT-DEPTH provides an AI-based algorithm for the analysis (estimation) of objects depth in the street driving domain.
------------------------	---

Table 80 Solutions Mapping to the "ML-based Object Detection" Interface

Solution Name	Rationale
a2k-runman (ITI)	The a2k/detection service uses several AI and ML based algorithms to predict system performance from monitored run-time and historical data.
S3D (UCAN)	<p>S3D will support the system modelling and will include the mechanisms to invoke SoSIM modules as well as manage the intermediate models and data it needs to operate.</p> <p>There are two AI technologies being implemented as SoSIM modules. The first, will infer performance and energy predictions on a target processor taking as inputs concrete performance indicators obtained from the execution of the code in the host. Training a neural network with data from both, the host and the target platforms, and using the host HW performance registers to take the run time figures of merit in the host, the tool will be able to exploit modern GPU's capacity to bring into the simulation environment on-line data of the actual performance of the code in the target. This technique will enable the simulation of code subject to much more complex structures than the basic flat uninterrupted block of code. The second, will use AI to statically predict from the target binary (RISC-V) its performance figures (time and energy) when executed.</p>
SoSIM (UCAN)	<p>There are two AI technologies being implemented as SoSIM modules. The first, will infer performance and energy predictions on a target processor taking as inputs concrete performance indicators obtained from the execution of the code in the host. Training a neural network with data from both, the host and the target platforms, and using the host HW performance registers to take the run time figures of merit in the host, the tool will be able to exploit modern GPU's capacity to bring into the simulation environment on-line data of the actual performance of the code in the target. This technique will enable the simulation of code subject to much more complex structures than the basic flat uninterrupted block of code. The second, will use AI to statically predict from the target binary (RISC-V) its performance figures (time and energy) when executed.</p>
HEPSYCODE (UNIVAQ)	UNIVAQ helps correlate and aggregate processor execution times, power consumption, and resource utilisation, while predicting resource usage, energy/power consumption patterns, and functional/component timing using AI/ML approaches (e.g., FPGA area, memory allocation, bus utilisation). Hybrid supervised learning approaches using Convolutional Neural Network as feature extraction for various regression approaches (e.g. Support Vector Machine, Regression Trees, Random Forest, Linear and non-Linear Regression, and Deep Neural Networks) will be used to predict functional performance on multiple processors. UNIVAQ will investigate other non-functional indicators to improve system design at different levels of abstraction.

Table 81 Solutions Mapping to the "ML-based Prediction For Performance and Resource Utilisation" Interface

iii. Mapping to Automation

In this section, we present the list of solution components realising the "Automation" component of the AI-Augmented tool set. Each solution name is followed by the partner acronym between parentheses. The full partners names can be found in the Partners Acronyms table in the preamble of this document.

Solution Name	Rationale
Cloud expertise (AND)	AND offers their cloud expertise for every cloud related tasks that others might have. Most of the time when it comes to the cloud, automation is a huge part of the architecture. Creating an architecture that can automatically scale and change on demand is crucial when creating cloud native applications.
Infrastructure as Code (IaC) expertise (AND)	AND offers their expertise in IaC by providing help with building your infrastructure in a unified way. We have experts with knowledge in Terraform, Ansible and Puppet.
Mimir (AND)	<p>AI assistant for project management.</p> <p>It uses two step AI/ML analysis to estimate impact of the ticket to the project and offer improvements to the ticket content.</p> <p>Aim is to analyse Agile project's performance and provide advisory feedback for example based on user stories content.</p> <p>Mimir provides suggestions for better user stories, better estimations or give feedback about missing details in the issues based on the ML model.</p> <p>It also suggests changes to scrum project based on sprint retrospectives, velocity etc</p>
Keptn (DT)	<p>Keptn – pronounced captain – is a control-plane for DevOps automation of cloud-native applications.</p> <p>Keptn uses a declarative approach to build scalable automation for delivery and operations which can be scaled to a large number of services.</p> <p>Scriptless delivery for maintenance</p> <p>Keptn uses a simple, declarative approach that allows specifying DevOps automation flows like delivery or operations automation without scripting all the details. This definition can be shared across any number of micro services without the need to build individual pipelines and scripts.</p> <p>Separation of concerns for higher compliance</p> <p>Keptn separates the process defined by SREs from the actual tooling defined by DevOps engineers and the information about the artefacts.</p> <p>These definitions can be managed independently compared to traditional pipelines where everything is stored in a single file. This ensures that people cannot mistakenly break workflows and as they are managed in Git you will also have a full history of changes.</p> <p>Event-based automation for easy extensibility</p>

	<p>Keptn acts as a central control plane and uses well-defined CloudEvents for pretty much everything that can happen during continuous delivery and operations automation. Small services register for these events which makes the integration of tools simple and fast.</p> <p>Micro-service based tool integrations avoiding lock-in</p> <p>Keptn integrations translate well-defined CloudEvents into proprietary vendor APIs and hide complex automation for advanced tasks. This makes exchanging tools a simple configuration change rather than touching each individual pipeline.</p> <p>See all details at https://keptn.sh/.</p>
a2k-runman (ITI)	Output from the anomaly detection algorithms may be used to perform changes in the system to recover from faults and/or to optimise performance.
TATAT (PRO)	Thanks to TATAT, it is possible to automate the validation of a deployment. For each deployment a set of test can be executed
Constellation (SOFT)	Constellation allows to automate actions execution based on Modelio models status changes.
Modelio (SOFT)	Modelio may contribute to this component with the eventual "DevOps Connector" capability: to create connections between models and other DevOps Tools, s.a. ticketing systems (e.g., Jira), and CI (e.g., jenkins).

Table 82 Solutions Mapping to the "Automation" Component

Solution Name	Rationale
Keptn (DT)	Keptn's DevOps Automation as well can be used for the needed remediation actions, either for cloud backends as well as CPS systems.

Table 83 Solutions Mapping to the "IF-REMEDIATION" Interface

Solution Name	Rationale
Mimir (AND)	Mimir is a Scrum project assistant service that can be used as a plugin in project management tools to provide better quality tasks for the development teams. The AI learns to predict the impact of created tasks to the project and can auto generate solution suggestions to the task.
Keptn (DT)	Keptn is a DevOps automation for cloud-native applications with its CloudEvents, control-plan and quality gates adds value as one sees a convergence of CPS and cloud technologies (cf. https://fmi-standard.org as well as https://link.springer.com/chapter/10.1007/978-3-030-54997-8_17).
a2k-runman (ITI)	The a2k/tuning service uses the outputs from the a2k/detection service to detect and/or predict critical situations fault conditions and consequently invoke system mode changes to maintain performance.
TATAT (PRO)	After the execution of the automatic test, the results will be provided and integrated with the test orchestration tool used Interface Realisation IF-INFRA-COMPUTING-FROM-ARTIFACTS

Modelio (SOFT)	Currently, Modelio offers APIs that allows to support any kind of automation. Collaboration with AIDOaRt partners will be needed to determine potential implementation scopes.
Constellation (SOFT)	Constellation allows to automate tasks on Modelio models. By analysing current status of Modelio model, Constellation is able to launch preprogrammed tasks.

Table 84 Solutions Mapping to the "IF-RESPONSE-AUTOMATION" Interface

iv. Mapping to AI for Requirements Engineering

In this section, we present the list of solution components realising the "AI for Requirements Engineering" component of the AI-Augmented tool set. Each solution name is followed by the partner acronym between parentheses. The full partners names can be found in the Partners Acronyms table in the preamble of this document.

Solution Name	Rationale
Requirements Ambiguity Checker (MDU)	The requirements ambiguity checker aids in discovering terms that could cause potential misinterpretations in the requirements analysis. Moreover, the tool provides an explanation by means of a list of terms that are considered problematic for the requirement definition. This solution enables continual improvement in the requirements specification while also allowing the requirement engineer to participate in the validation and provide feedback.
VARA (RISE)	VARA automatically analyses natural language text requirements using NLP technique to identify similar requirements across various projects, and their corresponding implemented components.
Modelio (SOFT)	Modelio would contribute to this component with Natural Language Processing for Requirements Engineering (NLP4RE) capabilities, planned for development within AIDOaRt. For example, to identify, extract and classify (func, non-func) requirements from textual documents with NLP & AI/ML techniques.
UNISS_SOL_01 (UNISS)	This solution allows to support the requirement phase by checking the consistency of technical specifications using AI techniques.
UNISS_SOL_04 (UNISS)	This solution allows to verify the consistency of a set of properties' system design by using AI techniques with the purpose of identifying inconsistency to pre-defined specifications.

Table 85 Solutions Mapping to the "AI for Requirements Engineering" Component

Solution Name	Rationale
Requirements Ambiguity Checker (MDU)	The ambiguity checker is an NLP tool that receives as input a given textual requirement or an Excel file containing the requirements and provides as output whether the requirement is ambiguous or not, supporting the end-user's decision-making process.
VARA (RISE)	VARA supports the requirements engineering process, and enables automatic processing of textual requirements by using Natural Language Processing (NLP).

UNISS_SOL_01 (UNISS)	UNISS_SOL_01 supports the requirement engineering phase by checking the consistency of technical specifications using AI techniques.
-----------------------------	--

Table 86 Solutions Mapping to the "IF-AI-FOR-REQUIREMENTS-ENGINEERING" Interface

Solution Name	Rationale
VARA (RISE)	By using Natural Language Processing (NLP) and traceability information between requirements and previous project implementations, VARA is capable to perform a similarity analysis on a set of new requirements versus requirements from previous projects, and identify similar ones.
Modelio (SOFT)	Softeam uses AI/ML NLP techniques to evaluate the similarity of requirements in large specification documents. Within AIDOaRt, Softeam has developed a proof-of-concept tool for checking requirements similarities, using state-of-the-art sentence transformers.

Table 87 Solutions Mapping to the "AI for Requirements Similarity Check" Interface

Solution Name	Rationale
UNISS_SOL_04 (UNISS)	UNISS_SOL_04 provides automated formal consistency verification of a system design.

Table 88 Solutions Mapping to the "AI for Model Consistency Verification" Interface

Solution Name	Rationale
Modelio (SOFT)	Softeam uses AI/ML NLP techniques to classify requirements, e.g., as functional or non-functional requirements, or for assignment to various engineering teams. Within AIDOaRt, Softeam teamed up with other partners to develop a proof-of-concept prototype for requirements allocation, using various state-of-the-art language models and NLP/ML techniques.

Table 89 Solutions Mapping to the "AI for Requirements Allocation" Interface

Solution Name	Rationale
Requirements Ambiguity Checker (MDU)	Ambiguity Checker processes and analyses natural language requirements given in text format in order to identify the ones that can be interpreted differently (abstract or vague requirements), based on a trained classification model using real-world requirements labelled by an expert. The tool will give the reasons why a specific requirement was classified as ambiguous based on learnt patterns, enabling explainable AI.

Table 90 Solutions Mapping to the "AI for Requirements Ambiguity Check" Interface

Solution Name	Rationale
VARA (RISE)	VARA automatically processes natural language requirements, performs similarity analysis on them against the requirements from previous projects, identifies similarities, and ultimately recommends which components from previous projects can be reused for implementation of the new requirements.

Table 91 Solutions Mapping to the "AI for Reuse Analysis and Recommendation" Interface

v. Mapping to AI for Modeling

In this section, we present the list of solution components realising the "AI for Modeling" component of the AI-Augmented tool set. Each solution name is followed by the partner acronym between parentheses. The full partners names can be found in the Partners Acronyms table in the preamble of this document.

Solution Name	Rationale
Active DoE (AVL)	Active DoE is designed to model the unit under test using only a small amount of data.
EMF Views (IMTA)	EMF Views, that we plan to enhance thanks to the use of AI / Machine Learning techniques in the context of the AIDOaRt project, can be used in order to assist users during their modelling activities (e.g. by allowing to produce specific views on their models according to their needs and current tasks).
a2k-depman (ITI)	AI/ML methods are used to enhance multiobjective optimisation algorithms for design space exploration.
GAN-Based Instance Model Generator (JKU)	GAN-Based instance model generator aims to mitigate this shortage by producing a proper data set. As generative adversarial networks are used as the main part of the generator engine, the output data (generated Ecore-based models) will be structurally realistic. These data might be needed for feeding DL-based solutions or testing the newly developed tool
MOMOT (JKU)	MOMOT improves the performance of in-place model transformations by finding an optimal solution for orchestrating the transformation rules w.r.t. given user-specified metrics. The optimization process will be extended in order to support reinforcement learning algorithms.
Modelio (SOFT)	Modelio may support this component with two eventual capabilities: AI-Assisted Modelling: to generate model completion hints and suggestions to help analysts complete their models based on knowledge extracted from previous models. AI-Optimized Model Checking: to generate more complete and efficient consistency reports by analysing a model based on knowledge extracted from previous models.
AALpy (TUG)	AALpy allows the extraction of deterministic, non-deterministic and stochastic behavioural models from black-box systems.
HEPSYCODE (UNIVAQ)	HEPSYCODE model-based AI/ML approach integrated with DevOps and standard MDE principles.
MORGAN (UNIVAQ)	MORGAN will provide relevant model and metamodel artefacts that can be used to complete the model under construction. In particular, it can retrieve both classes and structural features i.e., attributes and relationships using a GNN as engine.
TWIMO (UNITE)	It offers advanced modelling and AI/ML analysis .

Table 92 Solutions Mapping to the "AI for Modeling" Component

Solution Name	Rationale
DTsynth (AIT)	DTsynth provides the possibility to learn the model, i. e., the finite state automata of a device and generate test scenarios for cyber physical systems and digital twins given a criticality measure.
Active DoE (AVL)	Starting from an initial unit-under-test (UUT) model with limited approximation approaches, the learning-based testing approach in combination with active DoE results in iteratively improved models by applying AI/ML approaches
EMF Views (IMTA)	EMF Views, that we plan to enhance thanks to the use of AI / Machine Learning techniques in the context of the AIDOaRt project (for improving view-model synchronisation), can be used in order to assist users during their modelling activities. This is notably the case when they want to produce specific views on their models according to their needs and activities in the context of their CPS design and development processes.
MOMOT (JKU)	MOMOT can be used for in-place model transformation to find the orchestration of executing transformation roles to reach the optimised form of the input(initial) model based on defined objectives.
Modelio (SOFT)	Currently, Modelio does not support any kind of modelling supported by AI. Collaboration with AIDOaRt partners will be needed for such implementation.
HEPSYCODE (UNIVAQ)	HEPSYCODE provides AI/ML approaches to enable model-based design and monitoring and improve code production. AI-driven model refinement algorithms with runtime tracking support and guide the designer during modelling. In addition, AI/ML algorithms are also used according to needs and objectives, such as performance evaluation, function execution time estimation, and area or size evaluation, to select the best behavioural solutions.
MORGAN (UNIVAQ)	Our solution provides AI for Modeling by exploiting graph kernel similarity algorithms. In such a way, MORGAN can support automation during the design of the system.
TWIMO (UNITE)	TWIMO provides modelling and analysis capabilities supporting the design phase of the system development. It provides a conceptual framework to define domain-specific notations for the definition of ML-based services.

Table 93 Solutions Mapping to the "IF-AI-FOR-MODELING" Interface

Solution Name	Rationale
MORGAN (UNIVAQ)	This interface supports the modelling activity by relying on the AI-based algorithm. In particular, MORGAN provides a graph-based kernel similarity to recommend modelling artefacts, including modelling operations.

Table 94 Solutions Mapping to the "AI-based Modeling Assistant" Interface

Solution Name	Rationale
Active DoE (AVL)	Design Space Exploration is relevant regarding finding the perfect parameterization of the unit under test (UUT) for a given UUT KPI by significantly reducing the design and test space applying the active DoE approach

a2k-depman (ITI)	The a2k-optimiser services uses multi-objective optimisation algorithms and ML methods to suggest suitable architectures for different conditions.
HEPSYCODE (UNIVAQ)	UNIVAQ investigates pareto-based solutions to multi-objective optimization problems using measurements and data valuable for system improvements. UNIVAQ will also conduct an offline design space exploration to find alternative solution patterns. To accelerate the solution space search process and overall design, UNIVAQ will use machine learning techniques during the design space exploration. Classification algorithms (e.g., Support Vector Classifier, Random Forest Classifier, Regression Trees Classifier, Deep Neural Network) will be used to quickly find feasible alternative solutions compared to classical approaches.

Table 95 Solutions Mapping to the "Design Space Explorer" Interface

Solution Name	Rationale
EMF Views (IMTA)	The building and updating of model views with EMF Views are being improved thanks to the use of some AI / Machine Learning techniques in order to automate the inference and generation of some of the required virtual elements and links. This way, we aim at better and more efficiently keeping the views and related contributing models synchronised during the whole CPS design and development process.

Table 96 Solutions Mapping to the "AI for View-Model Synchronisation" Interface

Solution Name	Rationale
Active DoE (AVL)	The applied AI/ML approach represents a robust automatic UUT parameter selection for an evolving model architecture for design space exploration.
AALpy (TUG)	AALpy incorporates a variety of automata learning methods that can be used to infer behavioural models of systems by interacting with the system (active learning) or analysing a predetermined set of observations (passive learning) such as system logs.

Table 97 Solutions Mapping to the "Model Learning" Interface

Solution Name	Rationale
GAN-Based Instance Model Generator (JKU)	GAN-based Instance Model Generator provides a model generator that leverages generative adversarial networks(GANs) to produce a collection of new realistic instance models for a given DSL.
Generative AI test scenario generator synth (AIT)	DTSynth uses this component to generate critical parameters sets for given test scenarios.

Table 98 Solutions Mapping to the "AI for Instance Model Generation" Interface

vi. Mapping to AI for Code

In this section, we present the list of solution components realising the "AI for Code" component of the AI-Augmented tool set. Each solution name is followed by the partner acronym between parentheses. The full partners names can be found in the Partners Acronyms table in the preamble of this document.

Solution Name	Rationale
Modelio (SOFT)	Modelio may support this component with two of its already existing capabilities: Code Generation: to generate customizable code from models Reverse engineering: to update models from edited generated code.
HEPSYCODE (UNIVAQ)	HEPSYCODE automatic code generation and AI/ML analysis starting from DSL/UML models.

Table 99 Solutions Mapping to the "AI for Code" Component

Solution Name	Rationale
Modelio (SOFT)	Currently, Modelio does not support any kind of code generation supported by AI. Collaboration with AIDoArT partners will be needed for such implementation.
HEPSYCODE (UNIVAQ)	HEPSYCODE provides a graphical modelling workbench that is part of the Sirius project.

Table 100 Solutions Mapping to the "IF-AI-FOR-CODE" Interface

Solution Name	Rationale
HEPSYCODE (UNIVAQ)	HEPSYCODE uses a Model2Text transformation that incorporates the Xtext framework and allows translation of the HEPSYCODE Modeling Language (HML) model into a simulatable CSP /SystemC model. This is used in the HW/SW co-design methodology at the system level. Starting from the HEPSYCODE metamodel, which follows the EMF Ecore metamodel specification, designers can use AI model tracking and predictive approaches to assist them in their modelling tasks, while code is automatically generated from the modified model. AI/ML methods are also used to select appropriate algorithms based on the specifications and goals of the analysis, such as evaluating performance, estimating the time it takes a function to execute, and determining the area or size of a system.

Table 101 Solutions Mapping to the "AI/ML Techniques for Functional Code Generation" Interface

vii. Mapping to AI for Testing

In this section, we present the list of solution components realising the "AI for Testing" component of the AI-Augmented tool set. Each solution name is followed by the partner acronym between parentheses. The full partners names can be found in the Partners Acronyms table in the preamble of this document.

Solution Name	Rationale
STGEM (ABO)	STGEM uses AI/ML techniques to automate and improve test generation, test selection, and test scheduling and execution components.
ESDE (ACO)	AI/ML helping to ensure that nightly tests can be completed in time with the maximum chances to find bugs and/or ensure the stability of the developed releases, applying techniques that enable the smart selection of relevant tests, either found in the SoA performed in the project or from innovations provided by AIDoArt partners.
DTsynth (AIT)	DTSynth uses generative AI methods to generate only test scenarios which are of interest.
Active DoE (AVL)	Active DoE is designed to select a small amount of data necessary to characterise the unit under the test and calibrate the control systems. As such, it can be used to create and manage AI-driven tests.
CRT (QEN)	<p>-- Test execution & test execution libraries --</p> <p>Qentinel offers test execution libraries for various types of application interfaces (web (open sourced), mobile and native application testing).</p> <p>Advanced automated UI interaction capabilities are offered through computer vision. QVision is the computer vision engine of CRT that interacts with the application UI by "seeing" the screen</p> <p>-- Test case maintenance through self-healing tests --</p> <p>Self-healing refers to automation that adapts tests to changes in the application. Typical cause for test failure is change in UI which cannot be handled by the test</p> <p>First line of defence in CRT through flexible test execution libraries. CRT test execution libraries apply various forms of resolution f.ex. in object detection with fallback</p> <p>Logical & non trivial UI changes can be healed using QEditors Live Testing environment. ML algorithm infers the most probably action to take confirming with user</p>
CRTQI (QEN)	DevOps emphasises the importance of feedback loops that allow you to see the problems as they occur. Measuring progress & quality in DevOps is essential but a challenge as setting up measuring systems is expensive, finding useful and actionable metrics is hard and there is an information overload – hard to see forest from trees.

	<p>Quality Intelligence for DevOps is a new cloud-based analytics solution that allows CRT users to measure and optimise DevOps</p> <ul style="list-style-type: none"> - Enables generation of the essential metrics based on Agile, DevOps and Flow Framework best practices with one click - Copado and Qentinel has pioneered using system dynamics modelling to predict quality of information systems based on patented Value Creation Model - CRT can present a summary of “everything” in one screen
QEDITOR (QEN)	<p>QEditor has trained ML algorithms to predict the next most likely test steps in your test cases. Based on your test cases and test flow so far, it makes predictive suggestions to what your next test steps is likely going to be</p> <p>Using normative models of test cases, editor identifies likely/unlikely test steps guiding test cases towards test automation best practices</p> <p>Provides advanced analytics by looking test cases making estimations on a test case’s probability of success, execution time, correlation with other test cases etc</p>
BugIdentifier (RISE)	<p>BugIdentifier uses AI/ML techniques to automate the linking of failed test cases with the related commits to support the root cause analysis in DevOps environment.</p>
LogGrouper (RISE)	<p>LogGrouper allows clustering of failure logs that share the same features using AI techniques to support the engineers while testing the system.</p>
DataAggregator (ROTECH)	<p>Provides to help them make better decisions, improve process efficiency and finally, understand performance of the Platform. Consistent evolution, and the usability of the data plays a vital role too.</p> <p>There are four technique</p> <p>In-comm aggregation: Uses a multi-hop system for the process of gathering and routing information inside the Data Aggregator</p> <ul style="list-style-type: none"> •Tree-based approach: An aggregation tree is constructed, mapping put the data from leaves to roots (source and sink nodes respectively) •Cluster-based approach: This approach is used to collate larger amounts of data on the entire Platform. <p>Multi-path approach: In this approach, partially aggregated data is sent to the root or parent internal table which then can send the data down various paths.</p>
Modelio (SOFT)	<p>Modelio may support this component with an AI-Enhanced Test Generation capability, that intends to generate test cases from a model and a requirements specification based on knowledge extracted from prior development projects.</p>
AALpy (TUG)	<p>AALpy allows to extract deterministic, non-deterministic and stochastic behavioural models from black-box systems, which can be used to model-check properties of the system under test which can be derived from its specification or an abstract model of the system.</p>

UNISS_SOL_03 (UNISS)	It enables the employment of AI/ML techniques to support the testing phase during the system development.
TWIMO (UNITE)	It offers validation capabilities.

Table 102 Solutions Mapping to the "AI for Testing" Component

Solution Name	Rationale
STGEM (ABO)	The STGEM (System Testing using Generative Models) tool uses novel AI/ML techniques to automate and improve the performance of several activities in the testing process including test generation, test selection and prioritisation, and test scheduling. STGEM can also be used to optimise configuration parameters that affect the performance of low-power devices. Its AI algorithms can explore the space of possible configurations efficiently and find configurations that minimise power consumption while maintaining the required performance of the device.
ESDE (ACO)	The work on AI for monitoring in ESDE is conceived as an offline monitoring, i.e., for failure or bug detection. The goal is to enable an overall DevOps environment based on VP which speeds up the testing of the developed firmware. To this respect, many tests might be needed, and some AI/ML test selection required, e.g., to make feasible a representative VP based regular (e.g., nightly builds).
DTsynth (AIT)	Our solution provides to actively/passively learn the finite state automata of a device under test. The digital twin shall provide means of reducing the test effort of CPS. It further allows learning and generating test scenarios/cases, i.e., for ADAS use cases.
Active DoE (AVL)	The Active_DoE approach (represented by AVL tool CAMEO) is based on an ML/AI approach to map unit-under-test (UUT) parameters to UUT KPIs. This approach is interactively applied during vehicle test execution cycles in connected test environments
CRT (QEN)	CRT offers multiple AI based testing mechanisms that are used in test execution, in test reporting and in the result analysis
CRTQI (QEN)	CRTQI offers a full blown analytical platform first summarising the key findings but offering a well versed dashboard UI for detailed analysis
QEDITOR (QEN)	QEDITOR offers various mechanisms based on AI for streamlining the test authoring process. The editor can predict users most probably actions, pin point deviations from normative test authoring style, run test case level analytics for insights, and more
LogGrouper (RISE)	LogGrouper supports the testing phase of system development by clustering scattered similar test execution logs of test suites to provide relevant information using AI techniques.
BugIdentifier (RISE)	BugIdentifier tool supports the testing phase of system development through mapping failing test cases to their source code changes in an automated fashion using NLP and AI methods.
Modelio (SOFT)	Currently, Modelio does not support any kind of system testing supported by AI. Collaboration with AIDOaRt partners will be needed for such implementation.

Solution Name	Rationale
UNISS_SOL_03 (UNISS)	UNISS_SOL_03 supports the testing phase by using AI/ML based techniques.
TWIMO (UNITE)	TWIMO supports the phase of testing, validation and verification of the system development, by providing ML-based analysis and prediction capabilities.

Table 103 Solutions Mapping to the "IF-AI-FOR-TESTING" Interface

Solution Name	Rationale
STGEM (ABO)	STGEM uses novel AI techniques to automate test suite generation. It improves the performance of several aspects of test suite generation including test input selection, test scheduling, and oracle synthesis. It achieves this by using machine learning algorithms to train both a test generator and a surrogate model of the system under test. STGEM can be used to generate a new test suite. Moreover, the surrogate model of the system under test can be queried for test selection purposes.
DTsynth (AIT)	The provided solution will provide new insights when digitally cloning the device under test. This will lead to new means for the generation of tests.
CRT (QEN)	CRT offers multiple AI based mechanisms for analysing the test suites such as deviation detection aiming to point out aspects of the suite that might have issues
QEDITOR (QEN)	QEDITOR offers various mechanisms based on AI for streamlining the test authoring process. Predictive engine can and is used to generate parts of the test cases forming test suites
UNISS_SOL_03 (UNISS)	UNISS_SOL_03 provides automatic test suite generation by using AI/ML techniques.

Table 104 Solutions Mapping to the "AI for Test Suite Generation" Interface

Solution Name	Rationale
STGEM (ABO)	STGEM creates machine learning models for both online and offline testing scenarios. In online testing, it uses adaptive learning algorithms during the testing process. In offline testing, it uses supervised learning algorithms between system integration builds.
Active DoE (AVL)	Starting from a lean Design-of-Experiment (DoE) applied for initial unit-under-test (UUT) tests, further test cases are derived from the initial test scenario to find the target output areas, where initial approximation was limited and to improve the quality of the approximation especially there by applying AI/ML learning approaches (static DoE -> active DoE).
QEDITOR (QEN)	QEDITOR offers various mechanisms based on AI for streamlining the test authoring process. The editor can predict users most probable actions, pin point deviations from normative test authoring style, run test case level analytics for insights, etc, all based on learning improving the results over time

AALpy (TUG)	In Learning-Based Testing, a model of a system under test is learned and checked against a specification of the system. This is covered by AALpy in the following way: AALpy provides capabilities to learn behavioural models of systems, to export those models to the format of the PRISM model checker and to invoke PRISM on those models. Depending on the need of use case providers this approach can be extended to other model checkers and other forms of LBT.
--------------------	---

Table 105 Solutions Mapping to the "Learning Based Testing" Interface

Solution Name	Rationale
CRT (QEN)	CRT offers a build in test generation capability offering mechanisms for f.ex. combinatorial test generation
QEDITOR (QEN)	QEDITOR offers a build in test generation capability offering mechanisms for f.ex. combinatorial test generation.
DataAggregator (ROTECH)	DataAggregator provides a method to automate unit test with a large number of possible automated tests thus decreasing the test execution time and reducing the manual effort.

Table 106 Solutions Mapping to the "AI for Unit Test Generation" Interface

Solution Name	Rationale
STGEM (ABO)	STGEM uses AI techniques for test case selection, prioritisation, and reduction. The tool can also be used to extend an existing test suite with tests previously generated by STGEM or another tool.
Active DoE (AVL)	By applying AI-based and learning-based testing methods, corner cases for unit-under-test (UUT) tests are more easily spotted, while areas with high approximation quality require fewer additional tests. In sum, significant test case reduction is achieved by applying the solution approach

Table 107 Solutions Mapping to the "AI for Test Case Reduction" Interface

viii. Mapping to AI for Monitoring

In this section, we present the list of solution components realising the "AI for Monitoring" component of the AI-Augmented tool set. Each solution name is followed by the partner acronym between parentheses. The full partners names can be found in the Partners Acronyms table in the preamble of this document.

Solution Name	Rationale
ESDE (ACO)	AI/ML analysis on the log and traces associated with the several implementation layers of an embedded system virtual model, in order to detect bugs or performance issues. The use of virtual models enables the integration in the DevOps flow.
HIB_logAnalyzer (HIB)	New AI algorithms to analyse system logs to find anomalies and alerts during the execution of the regular monitoring for the application.

a2k-runman (ITI)	New AI/ML algorithms for anomaly detection are being investigated for development in a2k.
CRT (QEN)	CRT includes numerous mechanisms for monitoring. A key component of CRT is used to analyse the trends in the test execution time durations which is used to see if there are statically significant changes in the test execution times, that might signal for example unfavourable changes in the application being tested or changes in the testing infrastructure.
CRTQI (QEN)	CRTQI encompasses a variety of tools for tracking and observation. A crucial element of CRTQI is dedicated to examining the patterns in test execution times, outcomes, and durations of other aspects. This analysis is aimed at identifying statistically significant shifts in these patterns, which could indicate, for instance, adverse modifications in the application under test or alterations in the testing infrastructure.
BugIdentifier (RISE)	BugIdentifier tool supports the monitoring phase of the system development process by providing relevant information of bug inducing commits based on test execution logs and failed test cases git information using NLP techniques.
LogGrouper (RISE)	LogGrouper supports the monitoring phase of system development by clustering the relevant logs information using AI techniques.

Table 108 Solutions Mapping to the "AI for Monitoring" Component

Solution Name	Rationale
HIB_logAnalyzer (HIB)	New AI algorithms to analyse system logs to find anomalies and alerts during the execution of the regular monitoring for the application.
a2k-runman (ITI)	The a2k/detection service uses several AI and ML based algorithms for anomaly detection of system performance and sensor data, which are provided by the a2k/monitoring service.
CRT (QEN)	CRT includes numerous mechanisms for monitoring. A key component of CRT is used to analyse the trends in the test execution time durations which is used to see if there are statically significant changes in the test execution times, that might signal for example unfavourable changes in the application being tested or changes in the testing infrastructure.
CRTQI (QEN)	CRTQI includes numerous mechanisms for monitoring. A key component of CRTQI is used to analyse the trends in the test execution time durations which is used to see if there are statically significant changes in the test execution times, that might signal for example unfavourable changes in the application being tested or changes in the testing infrastructure.
LogGrouper (RISE)	LogGrouper supports the monitoring phase of system development by clustering the relevant logs information using AI techniques.
BugIdentifier (RISE)	BugIdentifier tool supports the monitoring phase of the system development process by providing relevant information of bug inducing commits based on test execution logs and failed test cases git information using NLP techniques.

Table 109 Solutions Mapping to the "IF-AI-FOR-MONITORING" Interface

Solution Name	Rationale
HIB_logAnalyzer (HIB)	HIB_logAnalyzer uses Text Analytics to extract information from the captured logs (both from machine generated logs and from natural language sources such as comments by users).

Table 110 Solutions Mapping to the "AI for Text Analytics" Interface

Solution Name	Rationale
ESDE (ACO)	After a basic characterization of typical functional and performance bugs in the embedded domain (e.g., stack overflow), from the work on generic anomaly detection in ESDE, ACORDE expects to derive patterns that could be helpful for earlier detection of the afore mentioned functional and performance bugs.
HIB_logAnalyzer (HIB)	In HIB_logAnalyzer, logs are analysed to extract performance information from the different modules of the TAMUS application.

Table 111 Solutions Mapping to the "AI/ML based Functional / Performance Bug Detection" Interface

Solution Name	Rationale
ESDE (ACO)	ACORDE aims to experiment the application on the embedded domain of the same retrospective anomalies detection and analysis methodology developed for the distributed domain (showcased in the industrial positioning system). Specifically, the generic anomaly detection methods should be applicable to some extent (with some specific differences, like the type of monitored signals and data rate scales).
HIB_logAnalyzer (HIB)	HIB_logAnalyzer applies AI to the information in the logs to detect anomalies in the functioning of TAMUS.
CRT (QEN)	CRT includes numerous mechanisms for monitoring and anomaly detection. A key component of CRT is used to analyse the trends in the test execution time durations which is used to see if there are statically significant changes in the test execution times, that might signal for example unfavourable changes in the application being tested or changes in the testing infrastructure.
CRTQI (QEN)	CRTQI includes numerous mechanisms for monitoring and anomaly detection. A key component of CRTQI is used to analyse the trends in the test execution time durations which is used to see if there are statically significant changes in the test execution times, that might signal for example unfavourable changes in the application being tested or changes in the testing infrastructure.

Table 112 Solutions Mapping to the "AI/ML for Anomaly Detection" Interface

B. Mapping Use Case Requirements to AIDOaRt AI-Augmented tool set Components

In this section, we present the mapping of the use case requirements and data requirements to the AI-Augmented tool set components. This mapping has been defined by the Case Study Providers based on the potential of each component of the AIDOaRt Framework Architecture in satisfying each of their use case requirements and data requirements.

For the sake of clarity, we present these mappings per component. For each component, we list the correlated requirements in a separate section for each component, grouped in two tables. The first table lists the related use case requirements, and the second table lists the related use case data requirements.

Note that each requirement identifier is prefixed by the partner acronym. The full partners names can be found in the Partners Acronyms table in the preamble of this document.

ix. Mapping to Ingestion & Handling

In this section, we present the mapping of the use case requirements and data requirements to the "Ingestion & Handling" component of the AI-Augmented tool set.

Requirement ID	Requirement Description	Rationale
VCE_R02	AI/ML method for auto-adjusting model parameters w.r.t. similarity of execution traces of a Digital Twin with a CPS	AIOPS for Ingestion & handling component is expected to ensure continuous monitoring. VCE would use Ingestion & handling to provide the possibility of a continuous monitoring of data for AI/ML analysis tools.
AVL_RDE_R03	Automated multi-source data analysis of the real driving test data such that the relevant features of the driver behaviour can be clustered (e.g. highway driving, low speed driving, cornering, braking, acceleration,...). To be used for understanding the driving conditions.	The Ingestion & Handling component should allow the extraction of relevant features from massive, redundant, and noisy datasets. AVL would use the Ingestion & Handling component solution to cluster the extracted features of the driver's behaviour in order to better understand the driving conditions.
AVL_RDE_R05	Basic driver attributes can be analysed, including: <ul style="list-style-type: none"> - acceleration / deceleration histogram - breaking behaviour (preference of coast down vs active braking) - cornering behaviour (speed per curvature) - max speed preference 	The Ingestion & Handling component is expected to ensure that the extracted features can be analysed and visually presented in a user-friendly way. AVL would use Ingestion & Handling solution to analyse basic driver attributes including acceleration, braking, cornering behaviour or max speed preference.

Requirement ID	Requirement Description	Rationale
CSY_R02	Use reinforcement and deep learning techniques on proof theory and solving	CSY can use Ingestion and handling to manage real-time suggestions in or beside the Interactive prover tool. This would require analysing the stack of the prover while the user is manipulating it.
AVL_SEC_R05	Use AI (ML) methods to learn on the normal behaviour on a powertrain CAN	The Ingestion & Handling component is expected to ensure that relevant features can be extracted from massive, redundant, and noisy datasets. AVL would use Ingestion & Handling solution to structure the data on a CAN bus to sensibly define a baseline of a car's CAN communications to later detect anomalies.
AVL_SEC_R06	Use AI (ML) methods to learn detect abnormal behaviour on a CAN	The Ingestion & Handling component is expected to ensure that relevant features can be extracted from massive, redundant, and noisy datasets. AVL would use Ingestion & Handling solution to identify outliers in CAN communications in order to detect anomalies induced by our testing systems.
W_R_3	Extract data from steps in DevOps process.	The large amounts of log data from the DevOps process requires more than trivial ingestion, handling and storage in order to support e.g. search and retrieval of log files and logged data.
HIB_R01	The AIDOaRt AI algorithms must be able to analyse log files (text) from the restaurant application.	Since considerable amounts of logs will be processed by the system, it needs to collect and ingest all of this data for future operations.
W_R_2	Quality monitoring and predictions in DevOps process	For monitoring and predictions, some data needs to be ingested and handled.
CSY_R03	Use classification on project PO to predict the best tool for automatic proving	CSY can use Ingestion and handling to discover similarities and patterns in proof obligations. It can be used to adapt and direct the automatic treatments, or spread model changes initiated by a developer to the demonstrations of the affected proof obligations.

Table 113 Use Case Requirements Mapping to the "Ingestion & Handling" Component

Data Requirement ID	Data Requirement Description	Rationale
TEK_Data_01	Monitoring data of test execution and processing of results.	Data processing.
TEK_Data_02	Monitoring data of test execution and processing of results.	Data processing.
TEK_Data_03	Monitoring data for AI models for diagnostics and prognostics.	Data processing.

Table 114 Use Case Data Requirements Mapping to the "Ingestion & Handling" Component

Requirement ID	Requirement Description	Rationale
VCE_R02	AI/ML method for auto-adjusting model parameters w.r.t. similarity of execution traces of a Digital Twin with a CPS	The interface can enable the necessary measurements required for a high-quality digital twin implementation.
W_R_2	Quality monitoring and predictions in DevOps process	By continuous monitoring, quality shortcomings could be identified as they happen.
W_R_3	Extract data from steps in the DevOps process.	By continuous monitoring, quality shortcomings could be identified as they happen. We wish to monitor the DevOps process, some of the monitoring can be expected to be batch driven (check logs once a test session is complete), other aspects ought to be monitored continuously, e.g. disk space.
HIB_R01	The AIDOaRt AI algorithms must be able to analyse log files (text) from the restaurant application.	The HIBLA tool ingests the log data for monitoring.

Table 115 Use Case Requirements Mapping to the "IF-CONTINUOUS-MONITORING" Interface

Data Requirement ID	Data Requirement Description	Rationale
TEK_Data_01	Monitoring data of test execution and processing of results.	Monitoring data of test execution and processing of results.
TEK_Data_02	Monitoring data of test execution and processing of results.	Monitoring data of test execution and processing of results.
TEK_Data_03	Monitoring data for AI models for diagnostics and prognostics.	Monitoring data for AI models for diagnostics and prognostics.

Table 116 Use Case Data Requirements Mapping to the "IF-CONTINUOUS-MONITORING" Interface

Requirement ID	Requirement Description	Rationale
CSY_R02	Use reinforcement and deep learning techniques on proof theory and solving	This interface could serve the case study as many hypotheses would be irrelevant for most proof obligations.
CSY_R03	Use classification on project PO to predict the best tool for automatic proving	This interface could serve the case study as many hypotheses would be irrelevant for most proof obligations.
AVL_RDE_R05	Basic driver attributes can be analysed, including: - acceleration / deceleration histogram - breaking behaviour (preference of coast down vs active braking) - cornering behaviour (speed per curvature) - max speed preference	The IF-DATA-SELECTION is expected to allow user friendly data analysis and visualisation.

Table 117 Use Case Requirements Mapping to the "IF-DATA-SELECTION" Interface

Requirement ID	Requirement Description	Rationale
CSY_R02	Use reinforcement and deep learning techniques on proof theory and solving	This interface could serve the case study to extract meaningful relationships in proof obligations.
CSY_R03	Use classification on project PO to predict the best tool for automatic proving	This interface could serve the case study to extract meaningful relationships in proof obligations.
AVL_RDE_R03	Automated multi-source data analysis of the real driving test data such that the relevant features of the driver behaviour can be clustered (e.g. highway driving, low speed driving, cornering, braking, acceleration,...). To be used for understanding the driving conditions.	The IF-PATTERN-DISCOVERY component is expected to identify relevant features from big and noisy datasets.
AVL_SEC_R05	Use AI (ML) methods to learn on the normal behaviour on a powertrain CAN	AVL would use IF-PATTERN-DISCOVERY solution to structure the data on a CAN bus to sensibly define a baseline of a car's CAN communications to later detect anomalies.

Table 126 Use Case Requirements Mapping to the "IF-PATTERN-DISCOVERY" Interface

Requirement ID	Requirement Description	Rationale
AVL_SEC_R06	Use AI (ML) methods to learn detect abnormal behaviour on a CAN	AVL would use the Bug Pattern Discovery solution to identify outliers in CAN communications in order to detect anomalies induced by our testing systems.

Table 118 Use Case Requirements Mapping to the "Bug Pattern Discovery" Interface

x. Mapping to Engagement & Analysis

In this section, we present the mapping of the use case requirements and data requirements to the "Engagement & Analysis" component of the AI-Augmented tool set.

Requirement ID	Requirement Description	Rationale
VCE_R03	Use ML for predicting values which are actually not measurable	AIOPS for Engagement & analysis component is expected to predict values which are not measurable in the VCE context based on AI/ML techniques. VCE would use engagement & analysis to support monitoring activities in cases where monitoring is not feasible of certain values in the system.
AVL_RDE_R04	ML methodology for identifying parameters of the worst case conditions. Emission simulation/testing is time consuming, therefore only the most critical experiments should be evaluated. ML methodology should identify the critical experiments based on the provided driving profiles.	The Engagement & Analysis component is expected to ensure prediction capability in identifying critical parameters of the system. AVL would use an Engagement & Analysis solution to identify parameters of the worst-case driving profile in terms of emission.
AVL_MBT_R01	tool set to generate test cases based on an Unified Modeling Language (UML) model in order to make initial measurements to generate a surrogate model for large (dynamic and or combinatorial) systems. For example, uniform or Sobol sampling for continuous regression models or Amplitude Modulated Pseudo-Random Bit Sequences (APRBS) for dynamic models.	The Engagement & Analysis component is expected to ensure design space exploration. AVL would use an Engagement & Analysis solution to generate test case parameters based on the Unified Modeling Language model in order to perform initial measurements when generating a surrogate model for a large system.
AVL_TCV_R03	The new parameter values given by the SCENIUS parameter recommender must lead to critical situations which are not covered by the generated Tests from the SCENIUS test case generator.	The Engagement & Analysis component is expected to provide predictability in identifying critical system parameters. AVL would use an

Requirement ID	Requirement Description	Rationale
		Engagement & Analysis solution to identify parameters that are critical but not included in the initial set of training parameters.
AVL_ODP_R03	Implement approaches of data-driven models based on project data (especially Key Performance Index (KPI) and parameter value evolution) provided by AVL. The models are utilized to assess the information gain for specific experiments in order to identify experiments that provide little information gain and can thus be e.g. skipped in future projects.	The Engagement & Analysis component is expected to ensure analysis capabilities including root cause analysis, inference and deduction based on ML techniques. AVL would use an Engagement & Analysis solution to access experiments that provide little information gain and may be left out in future projects.
ABI_R02	Use automated reasoning for verification of Deep Neural Network models.	The Engagement & Analysis component is expected to ensure the automated verification of Neural Networks (NN) using AI/ML techniques. ABI would use AI/ML-based solutions for the verification of Deep Neural Network models.
ABI_R04	Use ML for video elaboration.	The Engagement & Analysis component is expected to provide solutions for intelligent driving assistance, by using AI/ML techniques. ABI would use AI/ML-based solutions for video elaboration in the context of intelligent driving assistance.
ABI_R09	Soiling detection to recognize if the camera is dirty.	
ABI_R10	The system shall adapt when driving into/out of a tunnel.	
ABI_R11	The system shall detect relevant objects.	
ABI_R12	The system shall estimate the vehicles approaching speed.	
AVL_SEC_R01	Use automata learning and ML techniques to derive SUT models	
AVL_SEC_R02	Use an ANN to perform plausibility checks on models	
AVL_SEC_R03	Train ANN on SUT topology discovery using test observation	Engagement & Analysis is expected to provide solutions for identifying particular spots in a model for plausibility and formal model checking.
AVL_SEC_R04	Use formal model checking methods to derive test cases out of a system model	

Requirement ID	Requirement Description	Rationale
AVL_SEC_R05	Use AI (ML) methods to learn on the normal behavior on a powertrain CAN	Engagement & Analysis is expected to provide solutions for identifying particular spots in the data to facilitate anomaly detection.
AVL_SEC_R06	Use AI (ML) methods to learn detect abnormal behavior on a CAN	
AVL_SEC_R07	Use intelligent fuzzing techniques on a CAN bus	
W_R_1	AI/ML-powered monitoring/automation of DevOps process	In order to support Westermo's use cases, some inferences, deductions or predictions are required.
W_R_2	Quality monitoring and predictions in DevOps process	
W_R_3	Extract data from steps in DevOps process.	
W_R_4	Log file storing, indexing, searching, clustering and comparing	
ABI_R01	Use automated reasoning and ML techniques for verification of specifications and high-level models.	The Engagement & Analysis component is expected to ensure the automated verification of specifications using AI/ML techniques. ABI would use AI/ML-based solutions for the verification of specifications.
TEK_R_102	The AIDOaRt Framework verifies in a semi-automatic manner, at design time, with respect to the requirements, the adequacy (the response versus the resources) of the real components on-which/with-which the system architect has in mind to map/realize the architecture.	TEK would use the Engagement & Analysis component in the use case scenario TEK_UCS_01 for supporting the design space exploration.
TEK_R_104	The AIDOaRt Framework interprets in a semi-automatic manner the results of the design time verification.	TEK would use the Engagement & Analysis component in the use case scenario TEK_UCS_01 for supporting the interpretation of the results obtained from the design space exploration.
TEK_R_203	The AIDOaRt Framework interprets, in a semi-automatic manner, the results of the runtime verification.	TEK would use the Engagement & Analysis component in the use case scenario TEK_UCS_01 for

Requirement ID	Requirement Description	Rationale
		supporting the interpretation of the results of the runtime test.
AVL_ODP_R02	Implement approaches of data-driven models based on project data (especially Key Performance Index (KPI) and parameter value evolution) provided by AVL. The models are utilized to forecast the KPI (and parameter) value evolution in the project.	The Engagement & Analysis component is expected to ensure analysis capabilities including root cause analysis, inference and deduction based on ML techniques. AVL would use Engagement & Analysis solution to forecast the KPI (and parameter) value evolution in the project.

Table 119 Use Case Requirements Mapping to the "Engagement & Analysis" Component

Data Requirement ID	Data Requirement Description	Rationale
W_DR_02	To identify non-trivial indicators for quality shortcomings, the test cases could be parsed with NLP.	For some artefacts it could be beneficial to ingest them with natural language processing.
PRO_Monitoring	The monitoring platform with collaboration with some AI algorithms will detect problems in the platform. Every time that a problem is found an alarm/notification will be generated.	Uses Engagement & Analysis to detect problems with the SPMP platform.
PRO_IoT	IoT devices periodically send data collected by the different sensors they contain. This data is sent via JSON messages. Regarding trucks and cranes, they send one message per second with information about the vehicle's operation/status. The important thing about this data is to verify that it is sent and that the messages are not lost. The content of the messages is not relevant to the purpose of the use case.	

Table 120 Use Case Data Requirements Mapping to the "Engagement & Analysis" Component

Requirement ID	Requirement Description	Rationale
ABI_R01	Use automated reasoning and ML techniques for verification of specifications and high-level models.	ABI_R01 is related to the use of automated reasoning and ML techniques for verification of specifications and high-level models. Therefore, it is related to the analysis capabilities, in terms of verification, given by IF-INSIGHT-ANALYSIS.
ABI_R02	Use automated reasoning for verification of Deep Neural Network models.	ABI_R02 is related to the use of automated reasoning for verification of Deep Neural Network models. Therefore, it is related to the analysis capabilities, in terms of verification, given by IF-INSIGHT-ANALYSIS.
AVL_SEC_R01	Use automata learning and ML techniques to derive SUT models	IF-INSIGHT-ANALYSIS is expected to provide solutions for identifying particular spots in a model for plausibility and formal model checking
AVL_SEC_R02	Use an ANN to perform plausibility checks on models	
AVL_SEC_R03	Train ANN on SUT topology discovery using test observation	
AVL_SEC_R04	Use formal model checking methods to derive test cases out of a system model	
AVL_ODP_R03	Implement approaches of data-driven models based on project data (especially Key Performance Index (KPI) and parameter value evolution) provided by AVL. The models are utilised to assess the information gain for specific experiments in order to identify experiments that provide little information gain and can thus be e.g. skipped in future projects.	AVL will use IF-INSIGHT-ANALYSIS to infer / deduce experiments that provide little information gain and may be left out in future projects.

Table 121 Use Case Requirements Mapping to the "IF-INSIGHT-ANALYSIS" Interface

Data Requirement ID	Data Requirement Description	Rationale
PRO_IoT	IoT devices periodically send data collected by the different sensors they contain. This data is sent via JSON messages. Regarding trucks and cranes, they send one message per second with information about the vehicle's operation/status.	Uses IF-INSIGHT-ANALYSIS to verify the service level agreements of the different sensors.

	The important thing about this data is to verify that it is sent and that the messages are not lost. The content of the messages is not relevant to the purpose of the use case.	
--	--	--

Table 122 Use Case Data Requirements Mapping to the "IF-INSIGHT-ANALYSIS" Interface

Requirement ID	Requirement Description	Rationale
VCE_R03	Use ML for predicting values which are actually not measurable	The interface can enable required predictions needed for analysis.
W_R_2	Quality monitoring and predictions in DevOps process	A quality monitoring tool could learn patterns relevant for predicting performance, which would be very relevant for W_R_2.
TEK_R_102	The AIDOaRt Framework verifies in a semi-automatic manner, at design time, with respect to the requirements, the adequacy (the response versus the resources) of the real components on-which/with-which the system architect has in mind to map/realize the architecture.	The interface supports the prediction capabilities (e.g. potential undesired scenarios), based on AI/ML techniques, algorithms and models trained and fed with different sources of data and can satisfy the TEK_R_102 requirement related to the AIDOaRt Framework that verifies in a semi-automatic manner, at design time, with respect to the requirements, the adequacy (the response versus the resources) of the real components on-which/with-which the system architect has in mind to map/realise the architecture.
TEK_R_104	The AIDOaRt Framework interprets in a semi-automatic manner the results of the design time verification.	The interface supports the prediction capabilities (e.g. potential undesired scenarios), based on AI/ML techniques, algorithms and models trained and fed with different sources of data and can satisfy the TEK_R_104 requirement related to the AIDOaRt Framework that interprets in a semi-automatic manner the results of the design time verification.
TEK_R_203	The AIDOaRt Framework interprets, in a semi-automatic manner, the results of the runtime verification.	The interface supports the prediction capabilities (e.g. potential undesired scenarios), based on AI/ML techniques, algorithms and models trained and fed with different sources of data and can satisfy the TEK_R_203 requirement related to the AIDOaRt Framework that interprets, in a semi-automatic manner, the results of the runtime verification.

Requirement ID	Requirement Description	Rationale
ABI_R04	Use ML for video elaboration.	ABI_R04 is related to the use of ML for video elaboration, in particular for ensuring proper visibility and avoiding obstacles. Therefore, it is related to the prediction capabilities (e.g. potential undesired scenarios) given by IF-PREDICTIVE-ANALYSIS.
ABI_R12	The system shall estimate the vehicles approaching speed.	ABI_R12 specifies that the system shall estimate the vehicles approaching speed. Therefore, it is related to the prediction capabilities (e.g. potential undesired scenarios) given by IF-PREDICTIVE-ANALYSIS.
ABI_R11	The system shall detect relevant objects.	ABI_R11 specifies that the system shall detect relevant objects while backing up. Therefore, it is related to the prediction capabilities (e.g. potential undesired scenarios) given by IF-PREDICTIVE-ANALYSIS.
ABI_R10	The system shall adapt when driving into/out of a tunnel.	ABI_R10 is related to the adaptation when driving into/out of a tunnel, to ensure visibility. Therefore, it is related to the prediction capabilities (e.g. potential undesired scenarios) given by IF-PREDICTIVE-ANALYSIS.
ABI_R09	Soiling detection to recognize if the camera is dirty.	ABI_R09 is related to the presence of soil on the camera, to ensure visibility. Therefore, it is related to the prediction capabilities (e.g. potential undesired scenarios) given by IF-PREDICTIVE-ANALYSIS.
AVL_RDE_R04	ML methodology for identifying parameters of the worst case conditions. Emission simulation/testing is time consuming, therefore only the most critical experiments should be evaluated. ML methodology should identify the critical experiments based on the provided driving profiles.	The Engagement & Analysis component should allow predictability in the identification of critical system parameters.
AVL_SEC_R03	Train ANN on SUT topology discovery using test observation	IF-PREDICTIVE-ANALYSIS is expected to provide expected behaviour of a system in order to provide plausibility checking of a learned model.

Requirement ID	Requirement Description	Rationale
AVL_TCV_R03	The new parameter values given by the SCENIUS parameter recommender must lead to critical situations which are not covered by the generated Tests from the SCENIUS test case generator.	The IF-PREDICTIVE-ANALYSIS component is expected to identify critical test case parameters not included in the training set.
AVL_ODP_R02	Implement approaches of data-driven models based on project data (especially Key Performance Index (KPI) and parameter value evolution) provided by AVL. The models are utilised to forecast the KPI (and parameter) value evolution in the project.	AVL will use the IF-PREDICTIVE-ANALYSIS to predict the KPI (and parameter) value evolution in the project.

Table 123 Use Case Requirements Mapping to the "IF-PREDICTIVE-ANALYSIS" Interface

Data Requirement ID	Data Requirement Description	Rationale
PRO_Monitoring	The monitoring platform with collaboration with some AI algorithms will detect problems in the platform. Every time that a problem is found an alarm/notification will be generated.	Uses IF-PREDICTIVE-ANALYSIS, Anomaly Detection and Performance and Resources Utilisation interfaces to detect and predict future problems with SPMP platform.
PRO_IoT	IoT devices periodically send data collected by the different sensors they contain. This data is sent via JSON messages. Regarding trucks and cranes, they send one message per second with information about the vehicle's operation/status. The important thing about this data is to verify that it is sent and that the messages are not lost. The content of the messages is not relevant to the purpose of the use case.	IF-PREDICTIVE-ANALYSIS and Anomaly Detection to detect and predict future problems with IoT.

Table 124 Use Case Data Requirements Mapping to the "IF-PREDICTIVE-ANALYSIS" Interface

Requirement ID	Requirement Description	Rationale
W_R_1	AI/ML-powered monitoring/automation of DevOps process	W_R_1 is about monitoring and automation in the DevOps process. Using AI/ML for anomaly detection would be one way of doing this, in addition to static limits or rules of some kind.

W_R_2	Quality monitoring and predictions in DevOps process	W_R_2 is about quality monitoring and prediction in the DevOps process. Using AI/ML for anomaly detection would be one way of doing this, in addition to static limits or rules of some kind.
W_R_3	Extract data from steps in the DevOps process.	We wish to monitor the DevOps process, in addition to doing this with static rules, an AI-augmentation could probably also bring benefits.
W_R_4	Log file storing, indexing, searching, clustering and comparing	W_R_4 is about using log files. Here text analytics is one promising technology we wish to explore.
AVL_SEC_R05	Use AI (ML) methods to learn on the normal behaviour on a powertrain CAN	AI/ML for anomaly detection is expected to provide solutions for identifying particular spots in the data to facilitate anomaly detection.
AVL_SEC_R06	Use AI (ML) methods to learn detect abnormal behaviour on a CAN	
AVL_SEC_R07	Use intelligent fuzzing techniques on a CAN bus	AI/ML for anomaly detection is expected to provide feedback from learned models to a fuzzer in the form of rewards to the learner on successfully going towards abnormal behaviour when fuzzing.

Table 125 Use Case Requirements Mapping to the "AI/ML for Anomaly Detection" Interface

Data Requirement ID	Data Requirement Description	Rationale
PRO_IoT	IoT devices periodically send data collected by the different sensors they contain. This data is sent via JSON messages. Regarding trucks and cranes, they send one message per second with information about the vehicle's operation/status. The important thing about this data is to verify that it is sent and that the messages are not lost. The content of the messages is not relevant to the purpose of the use case.	Uses Anomaly Detection to detect and predict future problems with IoT.
PRO_Monitoring	The monitoring platform with collaboration with some AI algorithms will detect problems in the platform. Every time that a problem is found an alarm/notification will be generated.	Uses IF-PREDICTIVE-ANALYSIS, Anomaly Detection and Performance and Resources Utilisation interfaces to detect and predict future problems with SPMP platform.
W_DR_02	To identify non-trivial indicators for quality shortcomings, the test cases could be parsed with NLP.	Westermo test cases could have their documentation parsed with AI techniques in order to monitor for

		e.g. copy/paste text, or to support a developer while he or she is writing the documentation.
--	--	---

Table 126 Use Case Data Requirements Mapping to the "AI/ML for Anomaly Detection" Interface

Requirement ID	Requirement Description	Rationale
ABI_R12	The system shall estimate the vehicles approaching speed.	ABI_R12 specifies that the system shall estimate the vehicles approaching speed. This is strictly related to the ML-based analysis given by ML-based Object Detection.
ABI_R11	The system shall detect relevant objects.	ABI_R011 specifies that the system shall detect relevant objects while backing up. This is strictly related to the ML-based analysis given by ML-based Object Detection.
ABI_R04	Use ML for video elaboration.	ABI_R04 is related to the use of ML for video elaboration. Therefore, it is strictly related to the ML-based analysis given by ML-based Object Detection.

Table 127 Use Case Requirements Mapping to the "ML-based Object Detection" Interface

Requirement ID	Requirement Description	Rationale
VCE_R03	Use ML for predicting values which are actually not measurable	The interface can enable the required predictions of performance related aspects.
TEK_R_102	The AIDoArT Framework verifies in a semi-automatic manner, at design time, with respect to the requirements, the adequacy (the response versus the resources) of the real components on-which/with-which the system architect has in mind to map/realise the architecture.	The interface supports the ML-based analysis and estimation for HW/SW platform timing performance (e.g., response time, execution time, HW latency) and platform resource utilisation (e.g., FPGA area utilisation, memory allocation) and can satisfy the TEK_R_102 requirement related to the AIDoArT Framework that verifies in a semi-automatic manner, at design time, with respect to the requirements, the adequacy (the response versus the resources) of the real components on-which/with-which the system architect has in mind to map/realise the architecture.

AVL_MBT_R01	tool set to generate test cases based on an Unified Modeling Language (UML) model in order to make initial measurements to generate a surrogate model for large (dynamic and or combinatorial) systems. For example, uniform or Sobol sampling for continuous regression models or Amplitude Modulated Pseudo-Random Bit Sequences (APRBS) for dynamic models.	The Engagement & Analysis component should utilise test space reduction resulting in increased performance and reduced resources.
--------------------	--	---

Table 128 Use Case Requirements Mapping to the "ML-based Prediction For Performance and Resource Utilisation" Interface

Data Requirement ID	Data Requirement Description	Rationale
PRO_Monitoring	The monitoring platform with collaboration with some AI algorithms will detect problems in the platform. Every time that a problem is found an alarm/notification will be generated.	Uses IF-PREDICTIVE-ANALYSIS, Anomaly Detection and Performance and Resources Utilisation interfaces to detect and predict future problems with SPMP platform.

Table 129 Use Case Data Requirements Mapping to the "ML-based Prediction For Performance and Resource Utilisation" Interface

xi. Mapping to Automation

In this section, we present the mapping of the use case requirements and data requirements to the "Automation" component of the AI-Augmented tool set.

Requirement ID	Requirement Description	Rationale
VCE_R01	Use automated reasoning and ML techniques for verification of specifications and high-level models	AIOPS for automation component is expected to ensure that user defined evaluation criteria can be used to perform automated evaluation analysis on models and generate reports with the results. VCE would use automation to improve modelling workflows and is vital to VCE_UCS_2 regarding architecture verification and validation.
VCE_R02	AI/ML method for auto-adjusting model parameters w.r.t. similarity of execution traces of a Digital Twin with a CPS	AIOPS for automation components is expected to realise the auto-adjusting capabilities required to meet the use case requirement. VCE would use automation to provide the capability of auto adjustment during run-time.

Requirement ID	Requirement Description	Rationale
VCE_R04	Use of automated tools for compliance verification	AIOPS for automation components is expected to ensure that compliance verification is performed manually. VCE would use automation to improve the involved workflow of compliance verification.
CSY_R02	Use reinforcement and deep learning techniques on proof theory and solving	CSY can use Automation capabilities to automatically solve proof obligation as soon as the B-model is created. This can also lead to early detection of errors (unprovable PO).
TEK_R_104	The AIDOaRt Framework interprets in a semi-automatic manner the results of the design time verification.	TEK_R_104 asks for the following functionality: semi-automatic interpretation of the results of the design-time verification. The functionality is supposed to be provided (and TEK_R_102 to be satisfied) conjointly by the components of the AIDOaRt Framework "Automation" and "AI for Testing". Between "Automation" and other components, there can be interfaces that the former needs in order to provide such a functionality. Other than the interface with "AI for Testing", which is explicitly pointed out, those with the components "AI for Modelling", "AI for Requirements", and "Model-Based Capabilities" could be the most likely ones. Notes: <ul style="list-style-type: none"> • It could be worth reading, as an introduction, the flow recap in the comment to the relationship «satisfy» between "AI for Modelling" and TEK_R_103. • The Use Case Scenario TEK_UCS_01 "Design choices verification" uses this functionality in the step № 7 "Interpret the results of the design-time tests" (Case Story TEK_CS_06 "InterpretDesignTimeResults").
W_R_1	AI/ML-powered monitoring/automation of DevOps process	An automated response to an issue in the DevOps process, like restarting a service, raising an alarm, etc., would be suitable ways to achieve W_R_1.
TEK_R_203	The AIDOaRt Framework interprets, in a semi-automatic manner, the results of the runtime verification.	TEK_R_203 asks for the following functionality: semi-automatic interpretation of the results of the run-time verification. The functionality is supposed to be provided (and TEK_R_203 to be satisfied) conjointly by the AIDOaRt Framework components "Automation" and "AI for Testing". Between "AI for Testing" and other components,

Requirement ID	Requirement Description	Rationale
		<p>there can be interfaces that the former needs in order to provide such a functionality. Other than the interface with " AI for Testing", which is explicitly pointed out, those with the components "AI for Modelling", "AI for Requirements", and "Model-Based Capabilities" could be the most likely ones.</p> <p>Notes:</p> <ul style="list-style-type: none"> • It could be worth reading, as an introduction, the flow recap in the comment to the relationship «satisfy» between "AI for Modelling" and TEK_R_202. • The Use Case Scenario TEK_UCS_02 "Run-time verification" uses this functionality in the step № 6 "Interpret the results of the run time tests" (Case Story TEK_CS_11 "InterpretRunTimeResults").
TEK_R_301	<p>The AIDOaRt Framework interprets, in a semi-automatic manner, the state of health of the software system on the basis of the data that this produces.</p>	<p>TEK_R_301 asks for the following functionality: semi-automatic interpretation of the state of health of the system based on the data that the latter produces.</p> <p>The functionality is supposed to be provided (and TEK_R_301 to be satisfied) conjointly by the components of the AIDOaRt Framework "Automation" and "AI for Monitoring".</p> <p>Between " Automation" and other components, there can be interfaces that the former needs in order to provide such a functionality. Other than the interface with " AI for Monitoring", which is explicitly pointed out, the most likely ones could be those with the data processing components, such as "Engagement & Analysis", "Ingestion & Handling", and "Data Management".</p> <p>Notes:</p> <ul style="list-style-type: none"> • The Use Case Scenario TEK_UCS_03 "Operating life monitoring" uses this functionality.
HIB_R03	<p>The AIDOaRt solution must be able to analyse the continuous integration process and detect anomalies.</p>	<p>Automation of the update process is the key of the HIB_R03. The AI system will need to decide on the relevance of an update operation on any of the modules in the Case Study.</p>
HIB_R04	<p>The AIDOaRt AI algorithms will enable analysing the success of deploying a new version of the POS application.</p>	<p>In combination with the Update, the AI system needs to decide on the pertinence of deployment of parts of the Case Study (module versions, data) if necessary.</p>

Requirement ID	Requirement Description	Rationale
HIB_R02	The AIDoArt solution will enable to process requirements expressed in natural language in Trello boards	The requirements management takes inputs from the available requirements repository and uses AI to assign tasks to developers.
W_R_2	Quality monitoring and predictions in DevOps process	Quality monitoring ought to be automated.

Table 130 Use Case Requirements Mapping to the "Automation" Component

Table 139

Requirement ID	Requirement Description	Rationale
VCE_R01	Use automated reasoning and ML techniques for verification of specifications and high-level models	The interface can enable reasoning about the user models to provide proactive feedback and refinement from verification activities.
CSY_R02	Use reinforcement and deep learning techniques on proof theory and solving	The interface can ease the interactive proof process and contribute to reducing the V&V activity.

Table 140 Use Case Requirements Mapping to the "IF-REMEDIATION" Interface

Requirement ID	Requirement Description	Rationale
VCE_R02	AI/ML method for auto-adjusting model parameters w.r.t. similarity of execution traces of a Digital Twin with a CPS	The interface can enable the automation of validation activities at design time which utilises simulation as the main activity by automatically responding to results of validation and suggesting the next steps, which could be a new simulation with altered parameters.
VCE_R04	Use of automated tools for compliance verification	The interface can enable the automation of certain activities related to compliance verification.
HIB_R03	The AIDoArt solution must be able to analyse the continuous integration process and detect anomalies.	In the proposed CI/CD pipeline for TAMUS, automation (of testing, packaging of assets, etc) is one of the key features required to improve the workflow.
HIB_R04	The AIDoArt AI algorithms will enable analysing the success of deploying a new version of the POS application.	
W_R_1	AI/ML-powered monitoring/automation of DevOps process	An automated response to an issue in the DevOps process, like restarting a service, raising an alarm, etc., would be suitable ways to achieve W_R_1.
W_R_2	Quality monitoring and predictions in DevOps process	If quality shortcomings are identified, an automated response (like an alarm)

		would be a relevant way of notifying a human.
HIB_R02	The AIDoArT solution will enable to process requirements expressed in natural language in Trello boards	The requirement analysis uses automation to assign developers to tasks.
TEK_R_104	The AIDoArT Framework interprets in a semi-automatic manner the results of the design time verification.	Design time verification.
TEK_R_203	The AIDoArT Framework interprets, in a semi-automatic manner, the results of the runtime verification.	Runtime verification.
TEK_R_301	The AIDoArT Framework interprets, in a semi-automatic manner, the state of health of the software system on the basis of the data that this produces.	Operating life monitoring for diagnostics and prognostics.

Table 131 Use Case Requirements Mapping to the "IF-RESPONSE-AUTOMATION" Interface

xii. Mapping to AI for Requirements Engineering

In this section, we present the mapping of the use case requirements and data requirements to the "AI for Requirements Engineering" component of the AI-Augmented tool set.

Requirement ID	Requirement Description	Rationale
VCE_R04	Use of automated tools for compliance verification	AI for requirements component is expected to ensure consistency of user requirements. VCE would use AI for requirements to verify consistency of requirements so that further automated verification and analysis is based on consistent requirements.
BT_R01	NLP contextual analysis of requirements and match against database of responses/solutions	The AI for Requirements component is expected to analyse requirements, evaluate them, and recommend actions that should be done by the requirements engineer. The component should be interoperable with third-party tools (for example, IBM Rational DoorsOORS) and provide data-exchange capabilities. Alstom would use AI for Requirements to enhance the capabilities of requirements engineers and automate the requirements engineering process.
ABI_R01	Use automated reasoning and ML techniques for verification of specifications and high-level models.	The AI for Requirements component is expected to ensure the consistency check of technical specifications and of a set of properties' system design, by using AI/ML techniques.

		ABI would use AI/ML-based solutions for the verification of specifications and high-level models.
ABI_R05	Use of automatic tools for compliance verification.	The AI for Requirement component is expected to enable AI/ML based capabilities to support the requirements phase of the system development. ABI would use AI/ML-based solutions for consistency verification of technical specifications.
HIB_R02	The AIDoArt solution will enable to process requirements expressed in natural language in Trello boards	The proposed AI extension of the HIB requirements management process requires that AI is used during the whole requirements phase of DevOps (i.e., detecting relevant topics from inserted requirements and assigning tasks to relevant developers).

Table 132 Use Case Requirements Mapping to the "AI for Requirements Engineering" Component

Requirement ID	Requirement Description	Rationale
ABI_R01	Use automated reasoning and ML techniques for verification of specifications and high-level models.	ABI_R01 is related to the use of automated reasoning and ML techniques for verification of specifications and high-level models. This requires rigorous procedures in the definition and analysis of requirements, as well as in their descriptions.
ABI_R05	Use of automatic tools for compliance verification.	ABI_R05 is related to the use of automatic tools for compliance verification. This requires rigorous procedures in the definition and analysis of requirements, as well as in their descriptions.
HIB_R02	The AIDoArt solution will enable to process requirements expressed in natural language in Trello boards	The main purpose of this requirement is to enable requirements engineering for the production of the TAMUS assets.

Table 133 Use Case Requirements Mapping to the "IF-AI-FOR-REQUIREMENTS-ENGINEERING" Interface

Requirement ID	Requirement Description	Rationale
HIB_R02	The AIDoArt solution will enable to process requirements expressed in natural language in Trello boards	In the AI enabled requirements engineering, one of the main features is the classification of new requirements so that they can be assigned to developers. This is done by means of similarity check with regards to past requirements and the developers who have worked on them.

Table 134 Use Case Requirements Mapping to the "AI for Requirements Similarity Check" Interface

Requirement ID	Requirement Description	Rationale
VCE_R04	Use of automated tools for compliance verification	The interface can enable the analysis of whether a model is compliant with various standards or patterns of development during design time. In particular aspects relating to functional safety are important to be handled at this stage across several artefacts in addition to the system itself.
ABI_R01	Use automated reasoning and ML techniques for verification of specifications and high-level models.	ABI_R01 is related to the use of automated reasoning and ML techniques for verification of specifications and high-level models. Therefore, it is strictly related to the methods for automated formal consistency verification given by AI for Model Consistency Verification.

Table 135 Use Case Requirements Mapping to the "AI for Model Consistency Verification" Interface

Requirement ID	Requirement Description	Rationale
ABI_R05	Use of automatic tools for compliance verification.	ABI_R05 is related to the use of automatic tools for compliance verification. Therefore, it is strictly related to the methods for automated formal consistency verification of specifications given by AI for Model Consistency Verification.

Table 136 Use Case Requirements Mapping to the "AI for Specifications Consistency Verification" Interface

Requirement ID	Requirement Description	Rationale
HIB_R02	The AIDOaRt solution will enable to process requirements expressed in natural language in Trello boards	In the AI empowered requirements engineering, we use AI to allocate requirements to particular developers of the software assets. This is done using similarity checks and suggesting to the product owner that related developers take similar requirements.

Table 137 Use Case Requirements Mapping to the "AI for Requirements Allocation" Interface

Requirement ID	Requirement Description	Rationale
HIB_R02	The AIDOaRt solution will enable to process requirements expressed in natural language in Trello boards	The recommendation is used to let the TAMUS product owner make better decisions assigning tasks to developers coming from the Trello boards.
BT_R01	NLP contextual analysis of requirements and match against database of responses/solutions	This interface will be used to identify similar requirements across projects and recommend ones that can be reused for a new project.

Table 138 Use Case Requirements Mapping to the "AI for Reuse Analysis and Recommendation" Interface

xiii. Mapping to AI for Modeling

In this section, we present the mapping of the use case requirements and data requirements to the "AI for Modeling" component of the AI-Augmented tool set.

Requirement ID	Requirement Description	Rationale
VCE_R01	Use automated reasoning and ML techniques for verification of specifications and high-level models	The AI for Modeling component is expected to capture and describe the necessary details of the architecture model which allows for verification by applying AI techniques. VCE would use AI for modelling to verify high level models from specific evaluation criteria.
VCE_R02	AI/ML method for auto-adjusting model parameters w.r.t. similarity of execution traces of a Digital Twin with a CPS	The AI for Modeling component is expected to ensure model configuration during run-time. VCE would use AI for modelling to configure the digital twin models based on the AI/ML methods analysis from data gathered and analysed.
VCE_R05	Customise standards based modelling frameworks (e.g. UAF, SysML, UML) and metamodels to develop system, software, data architecture models	The AI for Modeling component is expected to ensure instantiation of templates based on customised standards with different views/traceability/descriptions. All models should be interoperable with third party-tools and provide data-exchange capabilities. VCE would use AI for modelling to develop models meeting the traceability and interoperability demands of the internal process and existing standards.
VCE_R06	Integration of DevOps workflows and continuous integration/ configuration of models and corresponding technical solutions	The AI for Modeling component is expected to enable modelling of DevOps workflows and continuous integration/continuous configuration of models. VCE would use AI for modelling to harmonise the digital twin solution(s) with DevOps workflows.
VCE_R07	Development of standard data classification, reusable definition, representation, usage	The AI for Modeling is expected to ensure a model-based approach can capture the AI/ML methods and processes represented in the AIDOaRt project or external sources. VCE would use AI for modelling to facilitate reuse and enable interoperable artefacts that can interact with different environments/tools.
AVL_RDE_R01	Based on the real driving recordings (time based data on	The AI for Modeling component is expected to ensure that a human behaviour while

Requirement ID	Requirement Description	Rationale
	<p>vehicle speed, throttle/brake pedals, curvature, road gradient, GPS coordinates...) the ML model is trained to simulate human-like driving given a target route, vehicle and traffic conditions. During the training of the ML model vehicle characteristics are known. Traffic conditions have to be extracted from the recorded data based on the speed profile. Additionally, traffic conditions can be estimated based on traffic data provided by AVL partners (digital map service). Thus, driver behaviour in constant speed limit areas can be simulated by augmenting the AI based model on top of a dynamics simulator.</p>	<p>driving can be modelled from data records of driving cycles using AI techniques. AVL would use AI/ML solution to simulate human behaviour while driving to create a driving profile on an arbitrarily defined driving route.</p>
AVL_RDE_R02	<p>AI method that will provide better statistics of the environment based on the statistics of the real driving recording and data from digital map service</p>	<p>The AI for Modeling component is expected to ensure that a traffic/environmental condition of the road can be extracted and then modelled from data records of driving cycles using AI techniques. AVL would use AI/ML solution to simulate traffic/environmental condition an arbitrarily defined driving route.</p>
AVL_TCR_R01	<p>Implement approaches of data driven models based on AVL provided testbed data with the aim to do online diagnostics and anomaly detection. Use the generated model when running future tests as a reference model for identifying error prone behaviours.</p>	<p>The AI for Modeling component is expected to capture the behaviour of the unit under the test (UUT) during run-time using AI/ML techniques to create a digital twin. AVL would use AI/ML for Modeling to identify forbidden working conditions.</p>
AVL_TCR_R02	<p>Implement approaches of data driven models based on AVL provided testbed data with the aim to do simulation.</p>	<p>The AI for Modeling component is expected to capture and describe the necessary details of the system component or the entire system using AI / ML techniques. AVL would use AI/ML for Modeling to create digital tween of a system component or the</p>

Requirement ID	Requirement Description	Rationale
		entire unit under the test (UUT) that can be used in the simulation.
AVL_TCR_R03	Finding new AI/ML technologies to increase and extend the model efficiency and capabilities of the current modelling solutions.	The AI for Modeling component is expected to improve upon physical-based models by extending them with data-driven technologies. AVL would use AI/ML for Modeling to create a (hybrid) digital tween of a component used in the simulation.
AVL_MBT_R02	An ML tool set to generate surrogate models for system structures with a lot of binary or combinatorial inputs. Particularly we want to focus on large system structures that can be partitioned into several sub-systems which can be modelled independently.	The AI for Modeling component is expected to capture and describe the necessary details of the complex system using the data-driven methods that require only a limited amount of experimental data. AVL would use AI/ML for Modeling to generate models of the complex unit under the test systems.
AVL_ODP_R01	Implement approaches of learning data-driven models based on project data (especially Key Performance Index (KPI) and parameter value evolution) provided by AVL. The models are utilised to compute the maturity of a specific KPI result in a specific project at a specific point in time in a standardised and objective way.	The AI for Modeling component is expected to capture the details of the project necessary to define and estimate Key Performance Index (KPI) value using the AI/ML approaches. AVL would use AI/ML for Modeling to use to standardise and objectively estimate and compute the maturity of a specific KPI result of a project in a specific point in time.
AVL_ODP_R02	Implement approaches of data-driven models based on project data (especially Key Performance Index (KPI) and parameter value evolution) provided by AVL. The models are utilised to forecast the KPI (and parameter) value evolution in the project.	The AI for Modeling component is expected to capture the details of the project necessary to estimate Key Performance Index (KPI) parameters using the AI/ML approaches. AVL would use AI/ML for Modeling to forecast the KPI value evaluation in the projects.
BT_R02	ML aided control model parameterization during propulsion system testing	Alstom will use the AI for Modeling Component to support the modelling of the temperature model.
CSY_R03	Use classification on project PO to predict the best tool for automatic proving	CSY can use AI-Modelling for classification of models of proof obligations.

Requirement ID	Requirement Description	Rationale
CSY_R04	Use deep learning to write abstraction of implementation	CSY can use AI-Modelling to transform models, especially implementations into their abstractions
AVL_SEC_R04	Use formal model checking methods to derive test cases out of a system model	The AI for Modeling component should deliver a proper model that could be used for formal model checking. This could be used to assure correctness during the modelling phase.
TEK_R_103	The AIDoArT Framework synthesises in a semi-automatic manner the models needed for the verification at design time (the models that define both the tests and the results, i.e. the pass/fail criteria).	<p>TEK_R_103 asks for the following functionality: semi-automatic synthesis of extended models that define the design-time verification tests, including the pass/fail criteria.</p> <p>The functionality is supposed to be provided (and TEK_R_103 to be satisfied) by the component "AI for Modelling" of the AIDoArT Framework.</p> <p>Between "AI for Modelling" and other components, there can be interfaces that the former needs in order to provide such a functionality: those with the components "AI for Requirements", "AI for Testing", "Model-Based Capabilities", could be the most likely ones.</p> <p>Notes:</p> <ul style="list-style-type: none"> • The Use Case Scenario TEK_UCS_01 "Design choices verification" uses this functionality in the step № 4 "Extend the models" (Case Story TEK_CS_04 "ExtendModels"). • We consider two types of design choices. These can be described through their global results: (1) the models and (2) the selection of target components on-which/with-which to map/realise the models and that, at design-time, can be simulated or obtained by rapid prototyping. "Design choices verification" considers both: it verifies that the models' "COVERAGE" (whether the models work and comply with the requirements), as well as the "RESPONSE" of the target components (performance and the resources demand). The verification of the selected target components (prototypes, simulated) is among the design activities even if it has many parts in common with the run-time verification (Use Case Scenario TEK_UCS_02 "Run-time

Requirement ID	Requirement Description	Rationale
		<p>verification"), parts that can be exploited during the project.</p> <ul style="list-style-type: none"> Flow recap: <ol style="list-style-type: none"> TEK_R_103 — TEK_R_103 asks for a semi-automatic synthesis of the extended models. TEK_R_101 and TEK_R_102 — The extended models are used by two functionalities: (i) semi-automatic test of the COVERAGE of the models as requested by TEK_R_101, and (ii) semi-automatic test of RESPONSE of the target components as requested by TEK_R_102. <p>Please, note that both tests are carried out in the single step № 6 (Case Story TEK_CS_05).</p> <ol style="list-style-type: none"> TEK_R_104 — The results of both tests are analysed in a semi-automatic manner by the functionality requested by TEK_R_04 in the single step № 7 (Case Story TEK_CS_06). TEK_UCS_01 — The Use Case Scenario TEK_UCS_01 ends successfully when no error is detected.
TEK_R_202	<p>The AIDOaRt Framework synthesises, in a semi-automatic manner, the models needed for the verification at run time (the models that define the tests and the tests results).</p>	<p>TEK_R_202 asks for the following functionality: semi-automatic synthesis of extended models that define the run-time verification tests, including the pass/fail criteria.</p> <p>The functionality is supposed to be provided (and TEK_R_202 to be satisfied) by the component "AI for Modelling" of the AIDOaRt Framework.</p> <p>Between "AI for Modelling" and other components, there can be interfaces that the former needs in order to provide such a functionality: the components "AI for Requirements", "AI for Testing", "Model-Based Capabilities", could be the most likely ones.</p> <p>Notes:</p> <ul style="list-style-type: none"> The Use Case Scenario TEK_UCS_02 "Run-time verification" uses this functionality in the step № 3 "Design the test models" (Case Story TEK_CS_09 "DesignTestModels"). Flow recap: <ol style="list-style-type: none"> TEK_R_202 — TEK_R_202 asks for a semi-automatic synthesis of the extended

Requirement ID	Requirement Description	Rationale
		<p>models.</p> <p>(2) TEK_R_201 — The extended models are used by the semi-automatic run-time test that is a functionality requested by TEK_R_201.</p> <p>(3) TEK_R_203 — The results of the test are analysed in a semi-automatic manner by the functionality requested by TEK_R_203.</p> <p>(4) TEK_UCS_02 — The Use Case Scenario TEK_UCS_02 ends successfully when no error is detected.</p>
AVL_SEC_R01	Use automata learning and ML techniques to derive SUT models	The AI for Modelling component should deliver a valid model of the SUT to generate test cases.

Table 139 Use Case Requirements Mapping to the "AI for Modeling" Component

Requirement ID	Requirement Description	Rationale
AVL_SEC_R04	Use formal model checking methods to derive test cases out of a system model	The IF-AI-FOR-MODELLING component should deliver a proper model that could be used for formal model checking. This could be used to assure correctness during the modelling phase.
CSY_R04	Use deep learning to write abstraction of implementation	The module should help generating partial frames of specifications for existing components. In particular cases, where the abstract variables of the model are created and the developer has a clear idea of the implementation that should be done, it can be an advantage to automatically generate the specification of the operation based on the abstract variables. This abstraction could help the developer to decide if its implementation is conform to what he intended.

Table 140 Use Case Requirements Mapping to the "IF-AI-FOR-MODELING" Interface

Requirement ID	Requirement Description	Rationale
VCE_R01	Use automated reasoning and ML techniques for verification of specifications and high-level models	The interface can enable the user to correctly perform modelling activities in case of doubt regarding patterns or standard procedure.
VCE_R05	Customise standards based modelling frameworks (e.g. UAF, SysML, UML) and metamodels to develop system, software, data architecture models	The interface can enable the user to implement the customised standards developed towards the use case requirements.
VCE_R07	Development of standard data classification, reusable definition, representation, usage	The interface can enable the user to adhere to proper

		guidelines when performing modelling activities to enable standard classification and re-use.
AVL_ODP_R01	Implement approaches of learning data-driven models based on project data (especially Key Performance Index (KPI) and parameter value evolution) provided by AVL. The models are utilised to compute the maturity of a specific KPI result in a specific project at a specific point in time in a standardised and objective way.	AVL will use AI-based Modeling Assistant to recommend a standardised and objective maturity estimate for a specific KPI result of a project at a specific point in time.
CSY_R03	Use classification on project PO to predict the best tool for automatic proving	Modeling Assistant can be used by CSY to help reducing the time required to prove a project, i.e. to demonstrate that claim properties are true, and the implementation respects specification.

Table 141 Use Case Requirements Mapping to the "AI-based Modeling Assistant" Interface

Requirement ID	Requirement Description	Rationale
TEK_R_103	The AIDoArt Framework synthesises in a semi-automatic manner the models needed for the verification at design time (the models that define both the tests and the results, i.e. the pass/fail criteria).	Design time verification pass/fail criteria.
TEK_R_202	The AIDoArt Framework synthesises, in a semi-automatic manner, the models needed for the verification at run time (the models that define the tests and the tests results).	Design time verification results interpretation.

Table 142 Use Case Requirements Mapping to the "Design Space Explorer" Interface

Requirement ID	Requirement Description	Rationale
VCE_R01	Use automated reasoning and ML techniques for verification of specifications and high-level models	The interface can enable different models to be combined for more advanced analysis.
VCE_R05	Customise standards based modelling frameworks (e.g. UAF, SysML, UML) and metamodels to develop system, software, data architecture models	The interface can enable the user to follow the customised frameworks developed in the project.
VCE_R06	Integration of DevOps workflows and continuous integration/ configuration of models and corresponding technical solutions	The interface can enable the necessary bridge between the Dev and Ops contexts with their corresponding artefacts.

Table 143 Use Case Requirements Mapping to the "AI for View-Model Synchronisation" Interface

Requirement ID	Requirement Description	Rationale
VCE_R02	AI/ML method for auto-adjusting model parameters w.r.t. similarity of execution traces of a Digital Twin with a CPS	The interface can enable the refinement of the model based on the continuous measurements of a digital twin.
BT_R02	ML aided control model parameterization during propulsion system testing	The interface can enable the returning of the model based on the new measurements.
AVL_RDE_R01	Based on the real driving recordings (time based data on vehicle speed, throttle/brake pedals, curvature, road gradient, GPS coordinates...) the ML model is trained to simulate human-like driving given a target route, vehicle and traffic conditions. During the training of the ML model vehicle characteristics are known. Traffic conditions have to be extracted from the recorded data based on the speed profile. Additionally, traffic conditions can be estimated based on traffic data provided by AVL partners (digital map service). Thus, driver behaviour in constant speed limit areas can be simulated by augmenting the AI based model on top of a dynamics simulator.	The AI for Modeling component should allow learning of a model which is capable of generating new, highly realistic real driving recordings.
AVL_RDE_R02	AI method that will provide better statistics of the environment based on the statistics of the real driving recording and data from digital map service	The AI for Modeling should allow for generating a model that captures the statistics of the data.
AVL_TCR_R01	Implement approaches of data driven models based on AVL provided testbed data with the aim to do online diagnostics and anomaly detection. Use the generated model when running future tests as a reference model for identifying error prone behaviours.	The AI for Modeling component should allow for real-time modelling of the unit under test.
AVL_TCR_R02	Implement approaches of data driven models based on AVL provided testbed data with the aim to do simulation.	
AVL_TCR_R03	Finding new AI/ML technologies to increase and extend the model efficiency and capabilities of the current modelling solutions.	
AVL_MBT_R02	An ML tool set to generate surrogate models for system structures with a lot of binary or combinatorial inputs. Particularly we want to focus on large system structures that can be	The AI for Modeling component should allow for modelling of the unit under test with limited training data.

	partitioned into several sub-systems which can be modelled independently.	
AVL_SEC_R01	Use automata learning and ML techniques to derive SUT models	The Model Learning component should deliver a model of the system under test (SUT) to enable automatic test case generation. Core method for learning will be an Automata Learning approach.
AVL_ODP_R01	Implement approaches of learning data-driven models based on project data (especially Key Performance Index (KPI) and parameter value evolution) provided by AVL. The models are utilised to compute the maturity of a specific KPI result in a specific project at a specific point in time in a standardised and objective way.	AVL will use Model Learning to learn a KPI maturity model from data.
AVL_ODP_R02	Implement approaches of data-driven models based on project data (especially Key Performance Index (KPI) and parameter value evolution) provided by AVL. The models are utilised to forecast the KPI (and parameter) value evolution in the project.	AVL would use Model Learning for creating models to forecast the KPI value evaluation in projects.

Table 144 Use Case Requirements Mapping to the "Model Learning" Interface

Requirement ID	Requirement Description	Rationale
VCE_R01	Use automated reasoning and ML techniques for verification of specifications and high-level models	The interface can enable the creation of instances of models based on high-level descriptions that in turn can be analysed as part of the use case requirement.
VCE_R05	Customise standards based modelling frameworks (e.g. UAF, SysML, UML) and metamodels to develop system, software, data architecture models	The interface can enable the, at least partially, implementation of models adhering to the frameworks developed via the project activities.

Table 145 Use Case Requirements Mapping to the "AI for Instance Model Generation" Interface

xiv. Mapping to AI for Code

In this section, we present the mapping of the use case requirements and data requirements to the "AI for Code" component of the AI-Augmented tool set.

Requirement ID	Requirement Description	Rationale
CSY_R05	Use deep learning to write refinement of specification	CSY can use AI for code to transform models into code, B-specification into B-implementation.

CSY_R04	Use deep learning to write abstraction of implementation	CSY can use AI for code to transform models into code, B-implementation implementation into B-implementation.
----------------	--	---

Table 146 Use Case Requirements Mapping to the "AI for Code" Component

Requirement ID	Requirement Description	Rationale
CSY_R04	Use deep learning to write abstraction of implementation	This interface would be used to generate implementation from abstractions or the opposite. This phase is part of the coding in the B method development.
CSY_R05	Use deep learning to write refinement of specification	This interface would be used to generate implementation from abstractions or the opposite. This phase is part of the coding in the B method development.

Table 147 Use Case Requirements Mapping to the "IF-AI-FOR-CODE" Interface

Requirement ID	Requirement Description	Rationale
CSY_R05	Use deep learning to write refinement of specification	This interface is a direct expression of the need, generating code from formal specification.

Table 148 Use Case Requirements Mapping to the "AI/ML Techniques for Functional Code Generation" Interface

xv. Mapping to AI for Testing

In this section, we present the mapping of the use case requirements and data requirements to the "AI for Testing" component of the AI-Augmented tool set.

Requirement ID	Requirement Description	Rationale
CAM_R02	Use AI-based methods for tuning of parameters of system's configuration (e.g. for lower power consumption)	The AI for Testing component is expected to generate a wide set of potentially suitable configurations for reduction of power consumption of embedded radar devices. As well, it will evaluate and help find the best candidate.
VCE_R04	Use of automated tools for compliance verification	The AI for Testing component is expected to ensure automatic verification of system architecture is applicable for system architecture regarding the various customizable standards involved. VCE would use AI for requirements to verify compliance to the applied standards.
AVL_TCV_R01	The SCENIUS Test Case Selection Validator must determine with a given	The AI for Test component should allow determination of whether the generated test case is adequate for testing.

Requirement ID	Requirement Description	Rationale
	accuracy, if a given test case selection is adequate.	
ABI_R03	Use automated reasoning and ML techniques for generation of optimal test suites.	The AI for Testing component is expected to provide support for the testing phase by using AI/ML techniques. ABI would use AI/ML-based solutions for generation of optimal test suites.
AVL_SEC_R02	Use an ANN to perform plausibility checks on models	The AI for Testing component should be able to generate plausibility checking for models to be used for generating test cases.
AVL_SEC_R01	Use automata learning and ML techniques to derive SUT models	The AI for Testing component should be able to generate suitable models to be used for generating test cases.
AVL_SEC_R03	Train ANN on SUT topology discovery using test observation	The AI for Testing component should be able to generate suitable models to be used for generating test cases of the whole-system model.
AVL_SEC_R04	Use formal model checking methods to derive test cases out of a system model	The AI for Testing component should be able to generate suitable models to be used for model checking, where the results will be used for generating test cases.
AVL_SEC_R07	Use intelligent fuzzing techniques on a CAN bus	The AI for Testing component should be able to generate suitable models to be used for model checking, where the results will be used for guiding fuzz tests.
TEK_R_104	The AIDoArt Framework interprets in a semi-automatic manner the results of the design time verification.	TEK_R_104 asks for the following functionality: semi-automatic interpretation the results of the design-time verification. The functionality is supposed to be provided (and TEK_R_104 to be satisfied) conjointly by the AIDoArt Framework components "AI for Testing" and "Automation". Between "AI for Testing" and other components, there can be interfaces that the former needs in order to provide such a functionality. Other than the interface with "Automation", which is explicitly pointed out, those with the components "AI for Modelling", "AI for Requirements", and "Model-Based Capabilities" could be the most likely ones. Notes: <ul style="list-style-type: none"> • It could be worth to read, as introduction, the flow recap in the comment to the relationship «satisfy» between "AI for Modelling" and TEK_R_103. • The Use Case Scenario TEK_UCS_01 "Design choices verification" uses this functionality in the

Requirement ID	Requirement Description	Rationale
		step № 7 "Interpret the results of the design-time tests" (Case Story TEK_CS_06 "InterpretDesignTimeResults").
TEK_R_201	The AIDOaRt Framework verifies in a semi-automatic manner the implemented software artefact (system, sub-system, component) with respect to the requirements as well as with respects to the architectural and detailed models.	<p>TEK_R_201 asks for the following functionality: semi-automatic run-time verification.</p> <p>The functionality is supposed to be provided (and TEK_R_201 to be satisfied) by the component "AI for Testing" of the AIDOaRt Framework.</p> <p>Between "AI for Testing" and other components, there can be interfaces that the former needs in order to provide such a functionality: those with the components "AI for Modelling", "AI for Monitoring", and "Model-Based Capabilities" could be the most likely ones.</p> <p>Notes:</p> <ul style="list-style-type: none"> • It could be worth to read, as introduction, the flow recap in the comment to the relationship «satisfy» between "AI for Modelling" and TEK_R_202. • The Use Case Scenario TEK_UCS_02 "Run-time verification" uses this functionality in the step № 5 "Execute the run-time tests" (Case Story TEK_CS_10 "TestRunTimeSystem").
TEK_R_203	The AIDOaRt Framework interprets, in a semi-automatic manner, the results of the runtime verification.	<p>TEK_R_203 asks for the following functionality: semi-automatic interpretation the results of the run-time verification.</p> <p>The functionality is supposed to be provided (and TEK_R_203 to be satisfied) conjointly by the AIDOaRt Framework components "AI for Testing" and "Automation".</p> <p>Between "AI for Testing" and other components, there can be interfaces that the former needs in order to provide such a functionality. Other than the interface with "Automation", which is explicitly pointed out, those with the components "AI for Modelling", "AI for Requirements", and "Model-Based Capabilities" could be the most likely ones.</p> <p>Notes:</p> <ul style="list-style-type: none"> • It could be worth to read, as introduction, the flow recap in the comment to the relationship «satisfy» between "AI for Modelling" and TEK_R_202. • The Use Case Scenario TEK_UCS_02 "Run-time verification" uses this functionality in the step № 6 "Interpret the results of the run time tests" (Case Story TEK_CS_11 "InterpretRunTimeResults").

Table 149 Use Case Requirements Mapping to the "AI for Testing" Component

Requirement ID	Requirement Description	Rationale
CAM_R02	Use AI-based methods for tuning of parameters of system's configuration (e.g. for lower power consumption)	The functional interface would offer our use case the capability to use AI techniques for generation of suitable candidates for improved radar configurations. It will allow to optimize configuration parameters that affect the performance of low-power devices. The AI algorithms from this functional interface can explore the space of possible configurations efficiently and find configurations that minimise power consumption while maintaining the required performance of the device.
ABI_R03	Use automated reasoning and ML techniques for generation of optimal test suites.	ABI_R03 is related to the use of automated reasoning and ML techniques for generation of optimal test suites. Therefore it is strictly related to the AI/ML based capabilities to support the testing phase given by IF-AI-FOR-TESTING.
TEK_R_104	The AIDoArT Framework interprets in a semi-automatic manner the results of the design time verification.	The interface supports the AI/ML based capabilities to support the testing phase of the system development and can satisfy the TEK_R_104 requirement related to the AIDoArT Framework that interprets in a semi-automatic manner the results of the design time verification.
TEK_R_203	The AIDoArT Framework interprets, in a semi-automatic manner, the results of the runtime verification.	The interface supports the AI/ML based capabilities to support the testing phase of the system development and can satisfy the TEK_R_203 requirement related to the AIDoArT Framework that interprets, in a semi-automatic manner, the results of the runtime verification.
TEK_R_201	The AIDoArT Framework verifies in a semi-automatic manner the implemented software artefact (system, sub-system, component) with respect to the requirements as well as with respects to the architectural and detailed models.	The interface supports the AI/ML based capabilities to support the testing phase of the system development and can satisfy the TEK_R_201 requirement related to the AIDoArT Framework that verifies in a semi-automatic manner the implemented software artefact (system, sub-system, component) with respect to the requirements as well as with respects to the architectural and detailed models.

Table 150 Use Case Requirements Mapping to the "IF-AI-FOR-TESTING" Interface

Requirement ID	Requirement Description	Rationale
CAM_R02	Use AI-based methods for tuning of parameters of system's configuration (e.g. for lower power consumption)	The functional interface would offer our use case the capability to automate test suite generation (test input selection, test scheduling, and oracle synthesis). The AI algorithms from this functional interface use machine learning algorithms to train both a test generator and a surrogate model of the system under test.
ABI_R03	Use automated reasoning and ML techniques for generation of optimal test suites.	ABI_R03 is related to the use of automated reasoning and ML techniques for generation of optimal test suites. Therefore it is strictly related to the AI/ML based techniques for automatic test suite generation given by AI for Test Suite Generation.

Table 151 Use Case Requirements Mapping to the "AI for Test Suite Generation" Interface

Requirement ID	Requirement Description	Rationale
CAM_R02	Use AI-based methods for tuning of parameters of system's configuration (e.g. for lower power consumption)	The functional interface would offer our use case the capability to create machine learning models for both online and offline testing scenarios by using adaptive learning and supervised learning algorithms.
AVL_SEC_R01	Use automata learning and ML techniques to derive SUT models	The AI for Testing component should be able to generate suitable models to be used for generating test cases.
AVL_SEC_R02	Use an ANN to perform plausibility checks on models	The AI for Testing component should be able to generate plausibility checking for learned models to be used for generating test cases.
AVL_SEC_R03	Train ANN on SUT topology discovery using test observation	The Learning-based Testing component should be able to generate suitable models to be used for generating test cases of the whole-system model. Components of the complete systems should be fingerprinted and identified to facilitate the creation of an attack tree based on matching vulnerabilities to these components.
AVL_SEC_R07	Use intelligent fuzzing techniques on a CAN bus	The AI for Testing component should be able to generate models that can be used to control and steer model-based fuzz testing.

Table 152 Use Case Requirements Mapping to the "Learning Based Testing" Interface

Requirement ID	Requirement Description	Rationale
VCE_R04	Use of automated tools for compliance verification	The interface can enable the required testing of models to satisfy the use case requirement.

AVL_SEC_R04	Use formal model checking methods to derive test cases out of a system model	The AI for Unit Test Generation component should be able to match known vulnerabilities and exploits to a given system model in order to generate test cases.
--------------------	--	---

Table 153 Use Case Requirements Mapping to the "AI for Unit Test Generation" Interface

Requirement ID	Requirement Description	Rationale
CAM_R02	Use AI-based methods for tuning of parameters of system's configuration (e.g. for lower power consumption)	The functional interface would offer our use case the capability to use AI techniques for test case selection, prioritisation, and reduction.
TEK_R_201	The AIDoArt Framework verifies in a semi-automatic manner the implemented software artefact (system, sub-system, component) with respect to the requirements as well as with respects to the architectural and detailed models.	The interface supports the AI techniques for test case reduction of an equivalence set based test model and can satisfy the TEK_R_201 requirement related to the AIDoArt Framework that verifies in a semi-automatic manner the implemented software artefact (system, sub-system, component) with respect to the requirements as well as with respects to the architectural and detailed model.
AVL_TCV_R01	The SCENIUS Test Case Selection Validator must determine with a given accuracy, if a given test case selection is adequate.	The AI for Test component should allow determination of whether the generated test case is adequate for testing.

Table 154 Use Case Requirements Mapping to the "AI for Test Case Reduction" Interface

xvi. Mapping to AI for Monitoring

In this section, we present the mapping of the use case requirements and data requirements to the "AI for Monitoring" component of the AI-Augmented tool set.

Requirement ID	Requirement Description	Rationale
VCE_R02	AI/ML method for auto-adjusting model parameters w.r.t. similarity of execution traces of a Digital Twin with a CPS	The AI for Monitoring component is expected to ensure monitoring of the system is performed correctly on all involved sources and identifies potential deviations from expected behaviour. VCE would use AI for monitoring to provide other components necessary input to perform auto-adjustment of involved models.
VCE_R03	Use ML for predicting values which are actually not measurable	The AI for Monitoring component is expected to provide predictions of future unwanted behaviours

Requirement ID	Requirement Description	Rationale
		based on other observed features using AI/ML techniques. VCE would use AI for monitoring to guide the user with automated predictions or use automation to update models if required.
AVL_MBT_R03	An AI method to generate new test cases based on 1) calibration demands 2) information from previous measurements 3) the ML model from MBT_R_2. New test cases should be generated in order to improve model quality of the ML model, while avoiding similar measurements throughout the procedure and fulfilling calibration demands, for example given by constraints.	The AI for monitoring components should allow monitoring of the system under test.
AVL_SEC_R05	Use AI (ML) methods to learn on the normal behaviour on a powertrain CAN	The monitoring component should be suitable to allow for anomaly detection.
AVL_SEC_R06	Use AI (ML) methods to learn detect abnormal behaviour on a CAN	
W_R_1	AI/ML-powered monitoring/automation of DevOps process	W_R_1 is about monitoring and automation in the DevOps process. Using AI/ML for anomaly detection would be one way of doing this, in addition to static limits or rules of some kind.
W_R_2	Quality monitoring and predictions in DevOps process	W_R_2 is about quality monitoring and prediction in the DevOps process. Using AI/ML for anomaly detection would be one way of doing this, in addition to static limits or rules of some kind.
W_R_3	Extract data from steps in the DevOps process.	Automation and/or AI for monitoring will be central for implementing Westermo's use case.
W_R_4	Log file storing, indexing, searching, clustering and comparing	
TEK_R_301	The AIDoArT Framework interprets, in a semi-automatic manner, the state of health of the software system on the basis of the data that this produces.	TEK_R_301 asks for the following functionality: semi-automatic interpretation of the state of health of the system based on the data that the latter produces. The functionality is supposed to be provided (and TEK_R_301 to be satisfied) conjointly by the components of the AIDoArT

Requirement ID	Requirement Description	Rationale
		<p>Framework "Automation" and "AI for Monitoring".</p> <p>Between " Automation" and other components, there can be interfaces that the former needs in order to provide such a functionality. Other than the interface with " AI for Monitoring", which is explicitly pointed out, the most likely ones could be those with the data processing components, such as "Engagement & Analysis", "Ingestion & Handling", and "Data Management".</p> <p>Notes:</p> <ul style="list-style-type: none"> • The Use Case Scenario TEK_UCS_03 "Operating life monitoring" uses this functionality.
HIB_R01	The AIDoArt AI algorithms must be able to analyse log files (text) from the restaurant application.	The analysis of logs is the cornerstone of the proposed AIOps extension of the Case Study restaurants. It requires that AI is used to analyse the logs in search of anomalies and that reports are generated for developers and the management team.
HIB_R03	The AIDoArt solution must be able to analyse the continuous integration process and detect anomalies.	The Updating that is presented in HIB_R03 is based on the analysis of the Monitoring phase of DevOps for the Case Study restaurants system.
HIB_R04	The AIDoArt AI algorithms will enable analysing the success of deploying a new version of the POS application.	The Deployment of assets that is presented in HIB_R04 is based on the analysis of the Monitoring phase of DevOps for the Case Study restaurants system.
PRO_R05	Detect automatically Anomalies in the solution during the execution based on AI	Use AI for detecting automatically Anomalies in the solution during the execution
PRO_R07	Monitor the platform in real time to reduce the downtime and the data lost	<p>Monitor the platform in real time to reduce the downtime and the data lost based on the anomalies and predictions.</p> <p>Uses IF-AI-FOR-MONITORING to monitor the platform.</p>

Requirement ID	Requirement Description	Rationale
PRO_R08	Self-healing and self learning solution to minimise the downtime of the platform by detecting and correcting the problems automatically. Avoiding the manual recovery of the problems	Apply Self-healing and self learning techniques to minimise the downtime of the platform by detecting and correcting the problems automatically.

Table 155 Use Case Requirements Mapping to the "AI for Monitoring" Component

Data Requirement ID	Data Requirement Description	Rationale
W_DR_02	To identify non-trivial indicators for quality shortcomings, the test cases could be parsed with NLP.	For some artefacts it could be beneficial to ingest them with natural language processing. E.g. Westermo test cases could have their documentation parsed with AI techniques in order to monitor for e.g. copy/paste text, or to support a developer while he or she is writing the documentation.
PRO_Monitoring	The monitoring platform with collaboration with some AI algorithms will detect problems in the platform. Every time that a problem is found an alarm/notification will be generated.	Uses IF-AI-FOR-MONITORING and AI/ML for Anomaly Detection for detecting anomalies in the SPMP.
PRO_IoT	IoT devices periodically send data collected by the different sensors they contain. This data is sent via JSON messages. Regarding trucks and cranes, they send one message per second with information about the vehicle's operation/status. The important thing about this data is to verify that it is sent and that the messages are not lost. The content of the messages is not relevant to the purpose of the use case.	

Table 156 Use Case Data Requirements Mapping to the "AI for Monitoring" Component

Requirement ID	Requirement Description	Rationale
VCE_R02	AI/ML method for auto-adjusting model parameters w.r.t. similarity of execution traces of a Digital Twin with a CPS	The interface can enable more advanced capabilities when performing the measurements on the digital

		twin so that correct parameterization is conducted.
VCE_R03	Use ML for predicting values which are actually not measurable	The interface can enable the monitoring necessary for the required predictions as per the requirement.
HIB_R01	The AIDoArt AI algorithms must be able to analyse log files (text) from the restaurant application.	HIB_logAnalyzer uses the collected data to monitor the operation of the POS system and detect events of interest that might occur during the operation.
W_R_1	AI/ML-powered monitoring/automation of DevOps process	W_R_1 is about monitoring and automation in the DevOps process. Using AI/ML for monitoring would be one way of doing this, in addition to static limits or rules of some kind.
W_R_2	Quality monitoring and predictions in DevOps process	W_R_2 is about quality monitoring and prediction in the DevOps process. Using AI/ML for monitoring would be one way of doing this, in addition to static limits or rules of some kind.
PRO_R07	Monitor the platform in real time to reduce the downtime and the data lost	Uses IF-AI-FOR-MONITORING to monitor the platform.
W_R_3	Extract data from steps in DevOps process.	We wish to monitor the DevOps process, in addition to doing this with static rules, an AI-augmentation could probably also bring benefits.
W_R_4	Log file storing, indexing, searching, clustering and comparing	W_R_4 is about using log files, e.g. for clustering and searching in them. A key purpose here is to enable monitor embedded systems as well as the DevOps process.
AVL_MBT_R03	An AI method to generate new test cases based on 1) calibration demands 2) information from previous measurements 3) the ML model from MBT_R_2. New test cases should be generated in order to improve model quality of the ML model, while avoiding similar measurements	The AI for monitoring component should allow monitoring of the system under test.

	throughout the procedure and fulfilling calibration demands, for example given by constraints.	
--	--	--

Table 157 Use Case Requirements Mapping to the "IF-AI-FOR-MONITORING" Interface

Data Requirement ID	Data Requirement Description	Rationale
PRO_Monitoring	The monitoring platform with collaboration with some AI algorithms will detect problems in the platform. Every time that a problem is found an alarm/notification will be generated.	Uses IF-AI-FOR-MONITORING and AI/ML for Anomaly Detection for detecting anomalies in the SPMP.
PRO_IoT	IoT devices periodically send data collected by the different sensors they contain. This data is sent via JSON messages. Regarding trucks and cranes, they send one message per second with information about the vehicle's operation/status. The important thing about this data is to verify that it is sent and that the messages are not lost. The content of the messages is not relevant to the purpose of the use case.	Uses IF-AI-FOR-MONITORING to ensure data quality of IoT devices.

Table 158 Use Case Data Requirements Mapping to the "IF-AI-FOR-MONITORING" Interface

Requirement ID	Requirement Description	Rationale
HIB_R01	The AIDOaRt AI algorithms must be able to analyse log files (text) from the restaurant application.	HIB_logAnalyzer collects the information from the system operation from logs that are text-based and then AI is applied to this text to extract events of interest.
W_R_4	Log file storing, indexing, searching, clustering and comparing	W_R_4 is about using log files. Here text analytics is one promising technology we wish to explore.

Table 159 Use Case Requirements Mapping to the "AI for Text Analytics" Interface

Data Requirement ID	Data Requirement Description	Rationale
W_DR_02	To identify non-trivial indicators for quality shortcomings, the test cases could be parsed with NLP.	Westermo test cases could have their documentation parsed with AI techniques in order to monitor for e.g. copy/paste text, or to support a developer while he or she is writing the documentation.

Table 160 Use Case Data Requirements Mapping to the "AI for Text Analytics" Interface

Requirement ID	Requirement Description	Rationale
W_R_1	AI/ML-powered monitoring/automation of DevOps process	W_R_1 is about monitoring and automation in the DevOps process. Using AI/ML for identifying functional or performance issues, would be one way of doing this, in addition to static limits or rules of some kind.
W_R_2	Quality monitoring and predictions in DevOps process	W_R_2 is about quality monitoring and prediction in the DevOps process. Using AI/ML for identifying functional or performance issues, would be one way of doing this, in addition to static limits or rules of some kind.
PRO_R08	Self-healing and self-learning solution to minimise the downtime of the platform by detecting and correcting the problems automatically. Avoiding the manual recovery of the problems	Uses AI/ML based Functional/Performance Bug Detection to find performance problems in the platform.

Table 161 Use Case Requirements Mapping to the "AI/ML based Functional / Performance Bug Detection" Interface

Requirement ID	Requirement Description	Rationale
HIB_R01	The AIDOaRt AI algorithms must be able to analyse log files (text) from the restaurant application.	HIB_logAnalyzer recognizes events from the analysis of logs. Some of these events are anomalies (e.g., statistical outliers in the operation of the system) that are recognized using AI methods.
W_R_1	AI/ML-powered monitoring/automation of DevOps process	W_R_1 is about monitoring and automation in the DevOps process. Using AI/ML for anomaly detection would be one way of doing this, in addition to static limits or rules of some kind.
W_R_2	Quality monitoring and predictions in DevOps process	W_R_2 is about quality monitoring and prediction in the DevOps process. Using AI/ML for anomaly detection would be one way of doing this, in addition to static limits or rules of some kind.
TEK_R_301	The AIDOaRt Framework interprets, in a semi-automatic manner, the state of health of the software system on the basis of the data that this produces.	The interface supports the AI/ML algorithms for the detection of anomalies in time series monitoring data (offline or online) related to bugs or malfunctions of the system or the network (for involvement in the development process) and can satisfy the TEK_R_301 requirement related to the AIDOaRt Framework that interprets, in a semi-automatic manner, the state of health

		of the software system on the basis of the data that this produces.
PRO_R05	Detect automatically Anomalies in the solution during the execution based on AI	Uses AI/ML for Anomaly Detection to detect problems in the SPMP.
HIB_R03	The AIDoArT solution must be able to analyse the continuous integration process and detect anomalies.	We use anomaly detection to find issues in the newly generated versions of assets in TAMUS.
HIB_R04	The AIDoArT AI algorithms will enable analysing the success of deploying a new version of the POS application.	
AVL_SEC_R05	Use AI (ML) methods to learn on the normal behaviour on a powertrain CAN	The monitoring component should be suitable to allow for anomaly detection.
AVL_SEC_R06	Use AI (ML) methods to learn detect abnormal behaviour on a CAN	

Table 162 Use Case Requirements Mapping to the "AI/ML for Anomaly Detection" Interface

Data Requirement ID	Data Requirement Description	Rationale
PRO_Monitoring	The monitoring platform with collaboration with some AI algorithms will detect problems in the platform. Every time that a problem is found an alarm/notification will be generated.	Uses AI/ML for Anomaly Detection to detect problems in the SPMP.

Table 163 Use Case Data Requirements Mapping to the "AI/ML for Anomaly Detection" Interface