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

D 4.2 - AIDOaRt AI-Augmented toolkit - Intermediate 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:	March 31, 2023
Authors:	Overall document structure and content: Andreas Hametner (DT), Romina Eramo (UNIVAQ), Bilal Said (SOFT)
Contributing Partners:	ABI, ABO, ACO, AIT, AND, AST, AVL, BT, CAMEA, CSY, DT, HIB, IMTA, INT, ITI, JKU, MDH, PRO, QEN, RISE, ROTECH, SOFT, TEK, TUG, UCAN, UNISS, UNIVAQ, UOC, VCE, WESTMO
Date:	March 31, 2023
Version	V1
Revision:	08
Abstract:	This deliverable provides the intermediate version of the AI-augmented tool set as developed in the context of WP4. Relying on the propositions from WP2 and WP4, the support offered by the provided tools will notably cover the features expected by the AIDOaRt use cases providers in terms of AIOPS for requirements, monitoring, modelling, coding, and testing in the context of DevOps.
Status:	Type: other, Dissemination Level: PU

DOCUMENT REVISION LOG

VERSION	REVISION	DATE	DESCRIPTION	AUTHOR
V1	01	06/12/2022	First Doc Generation	Bilal Said (SOFT)
	02	20/12/2022	Structure Update	Andreas Hametner (DT)
	03	20/01/2023	Content Update / Micro Tasks	Andreas Hametner, AIDOaRt Partners
	04	14/02/2023	Micro Tasks	AIDOaRt Partners
	05	14/3/2023	Micro Tasks	AIDOaRt Partners
	06	15/3/2023	Update from Modelio, prep review	Andreas Hametner (DT), Bilal Said (SOFT)
	07	27/3/2023	Integrate feedback from internal Review	Andreas Hametner (DT)
	08	31/3/23	Final version	Andreas Hametner (DT)

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 WP4 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).

In order to realise this in practice, the AIDOaRt AI-augmented Tool Set is expected to be designed, developed, deployed and experimented within the context of the project use cases. It will comprise several tools, mostly coming from our project partners (and possibly supplemented by a few others according to further needs), that will be integrated in order to achieve the above-mentioned WP4 objectives. In order to make the glue between WP2 and WP3, this AI-augmented Tool Set will have to work in good synchronisation with its siblings: AIDOaRt Data Engineering Tool Set and AIDOaRt Core Tool Set (cf. the corresponding deliverables D2.2 (AIDOART-D2.2) and D3.3 (AIDOART-D3.3), respectively, for more details on these). Finally, this deliverable fully refers to the AIDOaRt global architecture delivered in D1.4.

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 (eventually) properly integrated with each other. This deliverable D4.2 gives an update to D4.1 of the AI-augmented Tool Set and provides insights on its main components (i.e., achieved capabilities, adopted AI/ML techniques, ...), corresponding supporting tools and regarding use cases/requirements, as well as on how they all relate. Also, we performed a "gap analysis" with the aim to analyse current mapping between requirements, architectural component and AI-augmented tools and to derive a tentative roadmap including mitigation actions needed in the following months of the projects.

As far as WP4 is concerned (and still in close relation with similar work to be carried out in WP2 and WP3), the next months will be devoted to actual development and integration steps (in collaboration with WP5 - Integration) of the AI-augmented Tool Set as introduced in this deliverable. This work will be reported in future deliverables D2.3, D3.4 and D4.3.

In this deliverable [AIDOART-D4.2], the initial mapping at component level (provided by D4.1 [AIDOART-D4.1]) is refined by mapping the requirements of use cases to specific interfaces of the AI-augmented Tool Set. A similar mapping is provided for the solution components. This helps to better understand requirements and to identify the gaps in the solutions.

We provide an overview of the current application of one or several solutions (a.k.a. tools) from the AIDOaRt Core Tool Set in the context of specific use case challenges.

Finally, this deliverable also provides an updated description of the tool components of WP4, including their capabilities, interfaces, deployment details, technical constraints and licences, and sheds more light on the cyber-physical system aspects of the individual solution components.

Table of Contents

Key Terminology Abbreviations

Abbreviations	Terminology
AI	Artificial Intelligence
AIOps	AI Operations
CPS	Cyber-Physical Systems
CPSoS	Cyber-Physical Systems of Systems
DevOps	Development Operations
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 Modeling Language
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 University
ACO	ACORDE Technologies S.A.
AIT	AIT Austrian Institute of Technology GmbH
AND	Anders Innovations Oy
AST	Automated Software Testing 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)
PRO	Prodevelop SL
QEN	Qentinel Oy
RISE	RISE Research Institutes of Sweden
ROTECH	Ro Technology srl
SOFT	Softeam
TEK	Tekne SRL
TUG	Technische Universitaet Graz
UCAN	Universidad de Cantabria
UNISS	Università degli Studi di Sassari
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 described in the AIDOaRt project proposal and in the already submitted AIDOaRt deliverables, AIDOaRt aims at combining Model Driven Engineering (MDE) and Artificial Intelligence (AI) principles and techniques to improve the support for the continuous development (a.k.a. DevOps process) of large and complex Cyber-Physical Systems (CPS).

The WP4 represents a key work package that, exploiting the capabilities provided by the WP2 and WP3 (see D2.2 "Data collection and representation" and D3.3 "AIDOaRt Core Infrastructure and Framework"), can efficiently employ the various kinds of data artefacts (mostly data models) along with the other kinds of available software and system engineering models, and provide AI-augmented tools to support the various phases of the DevOps process.

More particularly, the objective of WP4 is to support the development of the AI-augmented Tool Set that will extend the AIDOaRT framework developed in WP3, according to the needs of the various kinds of CPSs under development (i.e., the use cases requirements defined in WP1). Additional capabilities related to different CPS development tasks will include the application of AI for requirements, monitoring, modelling, coding and testing (i.e., a combination of AIOPS and DevOps in MDE settings). According to the AIOPS methodology, the toolkit will support the ingestion of data, events and metrics from many different sources, engagement and analysis by employing ML and AI, and the automation of operations belonging to the development process. This Tool Set is applied in use case challenges and will further used in the context of various project use cases (cf. WP1 and WP5).

Based on the work already performed and reported in already released deliverables, especially on D4.1, and in parallel with similar deliverables D2.2 and D3.3 (already delivered), the present deliverable D4.2 intends to further refine the capabilities and relationships between use cases and solutions. Its main objective is to provide a current view to a realistic vision over the AIDOaRt AI-augmented Tool Set to be further developed and enriched during the next year in the context of AIDOaRt. At this stage, we update the initial list of tools developed and/or provided by the project partners that we use in the context of this AIDOaRt AI-Augmented Tool Set and related WP4 activities.

The document is structured as follows. Section 2 starts by giving an overview of the AIDOaRt AI-Augmented Tool Set. Section 3 provides the progress status of the AIDOaRt AI-Augmented Tool Set. Section 4 presents a refined version of the mapping between these solutions and the components of the AIDOaRt AI-Augmented Tool Set, at the interface-level. Section 5 provides the refined mapping of the solution to the AIDOaRt AI-Augmented Tool Set components as well on interface level. Section 6 provides the mapping of Use Case Requirements and the AIDOaRt AI-Augmented Tool Set components and their interfaces. In section 7 we conclude the deliverable including identified gaps and roadmap.

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.
Recall:
MS1 (M6) to MS3 (M16) = Baseline
MS4 (M20) to MS5 (M24) = Intermediate
MS6 (M28) to MS8 (M36) = Final
For this deliverable, we include all solutions having features already available as baseline (M0) and/or released from MS1 (M6) until MS5 (M24), included.

Solution Name	Baseline	Intermediate	Final	Inc in D4.2 ?
STGEM (ABO)			Test generation and prioritisation **(MS7 (M32))**	
ESDE (ACO)		Multi-level functional and performance logs and traces **(MS4 (M20))**, Automated (or semiautomated) bug detection/prediction **(MS5 (M24))**		Yes
Position Monitoring for Industrial Environment (ACO)		Monitoring System Design and Development **(MS4 (M20))**, AI/ML based analysis of monitored data **(MS5 (M24))**, Provide Resilient and Accurate Position **(MS4 (M20))**		Yes
DTsynth (AIT)			Digital Twin Learning **(MS6 (M28))**, Digital Twin Learning-Data Derivation **(MS6 (M28))**	
devmate (AST)		Code Parser **(MS5 (M24))**, Testcode Generator **(MS4 (M20))**	Testmodel Editor **(MS7 (M32))**, Testcase Generator **(MS6 (M28))**, Testcase Evaluation **(MS7 (M32))**, Testdata prediction system	Yes

			(MS8 (M36)), Model Parser **(MS8 (M36))**	
Active DoE (AVL)			Active DoE **(MS7 (M32))**	
Keptn (DT)			Keptn CloudEvents **(MS6 (M28))**, Keptn Control-Plane **(MS6 (M28))**, Keptn Quality Gates **(MS6 (M28))**	
HIB_logAnalyzer (HIB)	HIB-LA **(MS2 (M12))**			Yes
EMF Views (IMTA)		Model Viewpoint and View specification **(MS4 (M20))**, Model Viewpoint and View navigation and query **(MS5 (M24))**	Model Viewpoint and View computation **(MS6 (M28))**, Model Viewpoint and View update **(MS7 (M32))**	Yes
INT-DET (INT)			Real-Time Object Detection **(MS6 (M28))**	
INT-DEPTH (INT)			Depth Estimation **(MS7 (M32))**	
a2k-runman (ITI)			a2k/tunning **(MS7 (M32))**	
a2k-depman (ITI)			a2k/optimiser **(MS7 (M32))**	
MOMoT (JKU)	MOMoT **(MS1 (M6))**		AI-augmented MOMoT **(MS7 (M32))**	Yes
GAN-Based Instance Model Generator (JKU)			Instance_Generator **(MS7 (M32))**	
Requirements Ambiguity Checker (MDU)			checkRequirementsAmbiguity **(MS8 (M36))**	
TATAT (PRO)		TestAutomationlink **(MS4 (M20))**		Yes
CRT (QEN)			Cloud test execution platform **(MS7 (M32))**	
CRTQI (QEN)		QI for DevOps **(MS4 (M20))**		Yes
QEDITOR (QEN)		QEditor - AI-assisted test authoring **(MS5 (M24))**		Yes
VARA (RISE)	Requirements-based reuse analysis and feature reuse			Yes

	recommendation **(MS1 (M6))**			
DataAggregator (ROTECH)			Data aggregation **(MS6 (M28))**	
Modelio (SOFT)	Modelling **(MS1 (M6))**, Meta-Modeling **(MS1 (M6))**, Model Exchange (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))**	Automated Requirements Identification, Extraction & Classification **(MS5 (M24))**	CPS Simulation **(MS6 (M28))**, Process Execution **(MS6 (M28))**, DevOps Connector **(MS6 (M28))**, AI-Assisted Modelling **(MS6 (M28))**, AI-Optimized Model Checking **(MS6 (M28))**, AI-Enhanced Test Generation **(MS6 (M28))**	Yes
Constellation (SOFT)	Model Distributivity **(MS1 (M6))**, Version Control **(MS1 (M6))**, Collaboration (access control, parallel editing, synchronization...) **(MS1 (M6))**	Collaboration Workflow **(MS5 (M24))**		Yes
AALpy (TUG)			Active automata learning of deterministic systems **(MS7 (M32))**, Active automata learning of non-deterministic systems **(MS7 (M32))**, Active automata learning of stochastic systems **(MS7 (M32))**	
S3D (UCAN)			S3D **(MS6 (M28))**	
SoSIM (UCAN)			SoSIM **(MS6 (M28))**	

UNISS_SOL_01 (UNISS)			Requirements consistency verification **(MS7 (M32))**	
UNISS_SOL_02 (UNISS)			NN verification **(MS7 (M32))**	
UNISS_SOL_03 (UNISS)			Test generation **(MS7 (M32))**	
UNISS_SOL_04 (UNISS)			Property verification **(MS7 (M32))**	
UNISS_SOL_05 (UNISS)			Consistency verification wrt a guideline **(MS7 (M32))**	
HEPSYCODE (UNIVAQ)			Performance Simulation and Predictions **(MS7 (M32))**, Design Space Alternatives Exploration **(MS7 (M32))**, Model-Driven DevOps approach for HW/SW Co-Design **(MS7 (M32))**	
MORGAN (UNIVAQ)			Recommender of model and metamodel elements **(MS7 (M32))**	
TWIMO (UNIVAQ)			Domain-specific modeling **(MS7 (M32))**, MLAnalysis **(MS7 (M32))**	

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 gives an overview of the implementation achievements and development status of the AIDOaRt AI-Augmented Tool Set solutions with regards to the Intermediate milestone, as stated in deliverable D4.1, [AIDOART-D4.1]. The achievements of each tool are thus presented, and commented, to make explicit the evolution of the tool requirements (also called purposes) both in the Baseline and the Intermediate version of the framework release.
To briefly summarise the reported status, it should be noted that:

- 12 tools are related to the AIDOaRt AI-Augmented Tool Set,

22 tool capabilities are related to the solutions participating in core issues:

- 3 concern the release at the baseline milestone up to M16.
- 14 concern the release at the Intermediate milestone up to MS5 (M24)
- 5 concern the release at the Intermediate milestone up to MS6 (M28)

It should also be noted that this deliverable focuses on the intermediate milestones.

3.1 Solution - ESDE (ACO)

3.1.1 Overview

The Electronic System-Level (ESL) embedded Software Development Environment, or ESDE in short, is developed by ACO to improve embedded software design productivity. The basic architecture of this tooling was introduced in section 3.1.1 of D2.2, [AIDOART-D2.2]. A main advantage of this environment is to facilitate the validation of embedded system software on top of a virtual environment. The idea is to specify system functionalities either directly in C or C++, or by starting from a specification captured on a host standard language (SystemC) which is later translated to embedded SW (eSW) via specific libraries which help to automate the process. Then, such eSW can be validated in a virtual environment (VIL or Virtual environment In the Loop). Such a virtual environment is able to integrate both the physical environment and the HW platform. This way the eSW is integrated in a holistic model, able to capture closed-loop interactions between the CPS and the physical environment, and moreover, the limitations and impact of the HW platform. While some works may consider the impact of sensors/actuators, actually I/O of the CPS, this research focuses on a complementary and also relevant aspect, namely the computation and memory resources of the IoT platforms at the very edge of the computing continuum platforms of modern monitoring systems. The computing continuum refers to the *progressive convergence among Cloud Computing (CC) and the Internet of Things (IoT) through Edge Computing* that is *the base of future digital solutions* [RiFa21] .

A good example of it is the type of CPS targeted by ACO in the AIDOaRt smart port use case, sketched in a figure in section 3.2.1 of D2.2. Figure 1 provides an enriched version of that figure, which sketches the ESDE extension investigated in AIDoARt.

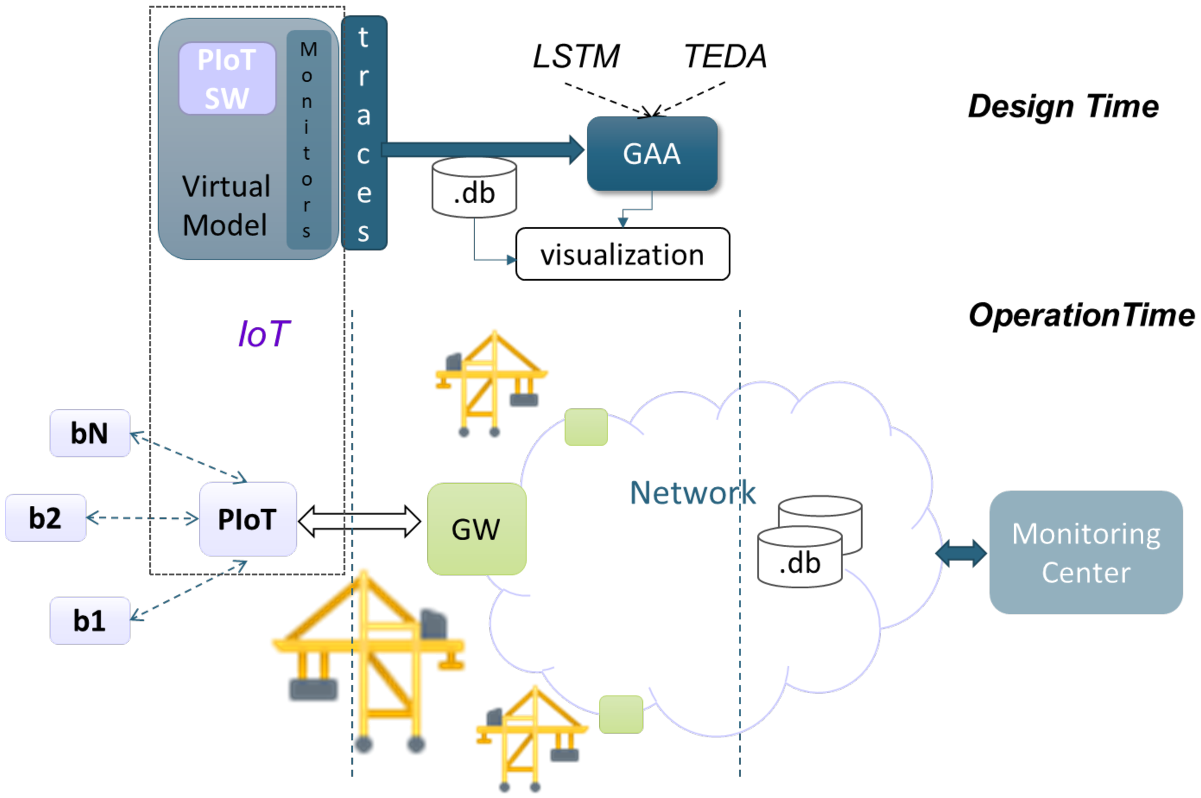


Figure 1 ESDE extension investigated and developed in AIDOaRt.

In AIDOaRt, ACO focuses on investigating the instrumentation of virtual models of embedded systems in order to obtain traces that enable an intelligent analysis. Specifically, ACO focuses also on anomalies analysis out of these traces. Therefore, this addresses valuable data collection and analysis at design time.

The virtual models referred on this investigation integrate the actual implementation of several software layers (application, RTOS), and a virtual HW model in the loop (VHIL), which provides a unique opportunity to obtain information, e.g., traces, from both the virtual HW model and the SW running. The virtual platform approach provides unique advantages wrt instrumented prototypes, like enabling an instrumentation without Heisenberg effect (very important in limited resources IoT devices), and better chance for embedding targeted software tests on DevOps cycles.

In AIDOaRt, ESDE is applied in the development of the Position monitoring IoT (PIoT) system, which is a component of the Position Monitoring solution posed by ACO. This solution relies on a computing continuum architecture, with different services run at different devices on the IoT, Edge and Cloud sides. Specifically, the Positioning IoT (PIoT) device is in charge of sensing and estimated geo-referenced positioning data at the cranes. This PIoT computes and sends this data from the edge to the cloud via gateways (GW). GWs gather data from the PIoT and eventually from other sensors and send them to the cloud. Moreover, gateways are able to run services which, for instance, can feed useful data back to the IoTs (e.g. configuration, etc). This means that, in reality, there is a richer, bidirectional communication between the PIoT and the GW. Moreover, in the posed solution, the PIoT shall be able to consider data from auxiliary devices (beacons). Therefore, the PIoT has to compute a non-straightforward position estimation functionality, that has to fuse different data sources, and yield an optimised design according to the embedded HW platform posed for the PIoT device, able to fulfil the cost constraints.

As the figure sketches, ACO is developing specific monitoring techniques to extract a set of traces, at different levels of the system (SW, HW) and at different granularity levels (to enable a trade off between simulation speed and accuracy). Traces are collected during the simulation. Then the anomalies analysis is applied to enable the embedded designer to detect potential issues, or optimization chances. Therefore, the type of anomalies analysis that is proposed in this context is therefore a design time analysis.

At this point, advance has been done to enable the exporting of several types of traces, which can be summarised as:

- start and end of software functions
- amount of executed instructions
- access (and its type, i.e. read/write) to memory regions of the HW platform

All this provides valuable information to trigger different types of analysis on the execution of a given eSW configuration on a specific scenario, to understand if it fulfils performance constraints, if such fulfilment is at risk, or if the eSW or HW design can be improved. This detection at design time enables eSW bug fixing or redesign.

As mentioned, this research focuses on developing, integrating and assessing anomalies analysis. This type of analysis requires in general the build or handling of a predictor or a knowledge base, which enables detection of an anomalous trace on the monitored and analysed signals. It can be interpreted as an execution state with the potential to lead to performance or functional issues or constraints violations.

Following, several aspects of the anomalies analysis research done by ACO in AIDOaRt are pointed out. These aspects have the potential to be common and thus exploitable at different sides (IoT, Edge, Cloud) of the computing continuum and in different DevOps stages. In this section, they are applied to a design-time analysis of a (Positioning) IoT device. In the next section (3.2) of this deliverable, they are applied to an online, real-time analysis at operation time, on the edge and cloud side of the computing continuum.

Anomalies Analysis:

The research on anomalies analysis (AA in Figure 1) performed by ACO in AIDOaRt targets the analysis of a set of multiple monitored signals (time series). A specific focus of the research is the look for techniques that enable a Generic Anomalies Analysis (GAA), which consist in the analysis of the "no-pattern", i.e., of anomalies as signal combinations that get out of "normal" patterns. This is a complementary approach to the one consisting in looking for specific patterns known in advance by the domain expert, and considered anomalous, so that it can lead to an issue or to a potentially catastrophic problem. This approach enables the automated search of anomalies, a big benefit to look into long records. However, it requires some deep domain expertise. As mentioned, this research focuses on GAA techniques which, while can be complemented with pattern-based searches, enable more domain agnostic techniques to detect anomalies. In this case, anomalies are understood and defined as signal patterns that get out of the combinations or records monitored and experienced in the past.

The expected outcome is improving the capability to detect anomalies, by detecting nuances and patterns beyond the expert eye, and also advancing, or predicting earlier the issues, especially catastrophic ones.

The current research is assessing two approaches based on:

- deep learning, specifically, based on long-short term memory (LSTM) neural networks
- evolutive algorithms. i.e. based on typicality and eccentricity data analysis (TEDA)

The assessment targets comparing metrics and aspects like the capabilities for detecting anomalies, training requirements and capability for adaptation/evolution. In theory, the nature of the explored approaches suggests certain trade-offs. For instance, TEDA approach should provide better adaptation and be more efficient for an edge implementation, while LSTM based analysis is expected to provide better capability for anomalies detection, although requiring more resources, and suffering more for enabling a real-time implementation. Therefore, taking into account the capabilities for validation and testing with the monitored data at design time, it can be claimed that ESDE extension implements the following AI-Augmented Tool Set components:

- AI for Testing (AI DevOps Engineering)

- AI for Monitoring (AI DevOps Engineering)

3.1.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
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)	**Implementation Level:** Former version Implemented **Estimated Delivery Date:** MS4 (M20) **License:** Proprietary – ACO. (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	At software level, traces, reporting function entry/exit events, are available. At hardware level, former versions of traces of accesses (write/read) to specific memory regions and amount of executed instructions are available. Yet work to

	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).	support accuracy-simulation speed trade-offs is on-going.
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 to use the collected metrics for an on-the-fly (real-time, or at least causal) computation for detecting/predicting anomalies.	**Implementation Level:** Partially Implemented **Estimated Delivery Date:** MS5 (M24) **License:** Proprietary – ACO. (It refers to specific extensions performed on ESDE to detect and/or predict functional and performance bugs 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).	A preliminary study has been done with LSTM neural networks for detecting anomalies on individual signals. Work is on going to test on multiple time series. A former implementation of TEDA metrics computation, required for TEDA based AA is in place. Once AA modules with these techniques are ready, they will be applied to specific traces from ESDE will be done.

Table 2 Capabilities Implementation Status of the ESDE Solution

3.1.3 Useful Resources

Related publications:

- https://towardsdatascience.com/lstm-autoencoder-for-anomaly-detection-e1f4f2ee7ccf
- G. Signoretti et al. "An Evolving TinyML Compression Algorithm for IoT Environments Based on Data Eccentricity" In Sensors 2021, 21.

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

3.2 Solution - Position Monitoring for Industrial Environment (ACO)

3.2.1 Overview

ACO is developing an innovative industrial positioning monitoring solution which, as reported in D2.2 [AIDOART-D2.2], relies on the combination of three main aspects: (1) a smooth and efficient integration over a computing continuum architecture, (2) an improved positioning performance relying on such architecture, and (3) AI/ML extensions for analysis and detection of anomalies that could represent potential issues.

This solution provides a modernised alternative in complex monitoring scenarios as the ones of the smart port use case. The port can be seen in this case as a complex cyber-physical system (CPS), where the "cyber" part relies on a computing continuum architecture composed of many different types of sensing devices (positioning IoT among them), and which requires an efficient exploitation of the sensing and computing resources in the edge (gateways) and in the cloud. D2.2 provided a former sketch of the computing continuum based architecture of the monitoring solution posed by ACO and suited to the smart port use case. Figure 2 provides an update of that picture according to the advance done so far.

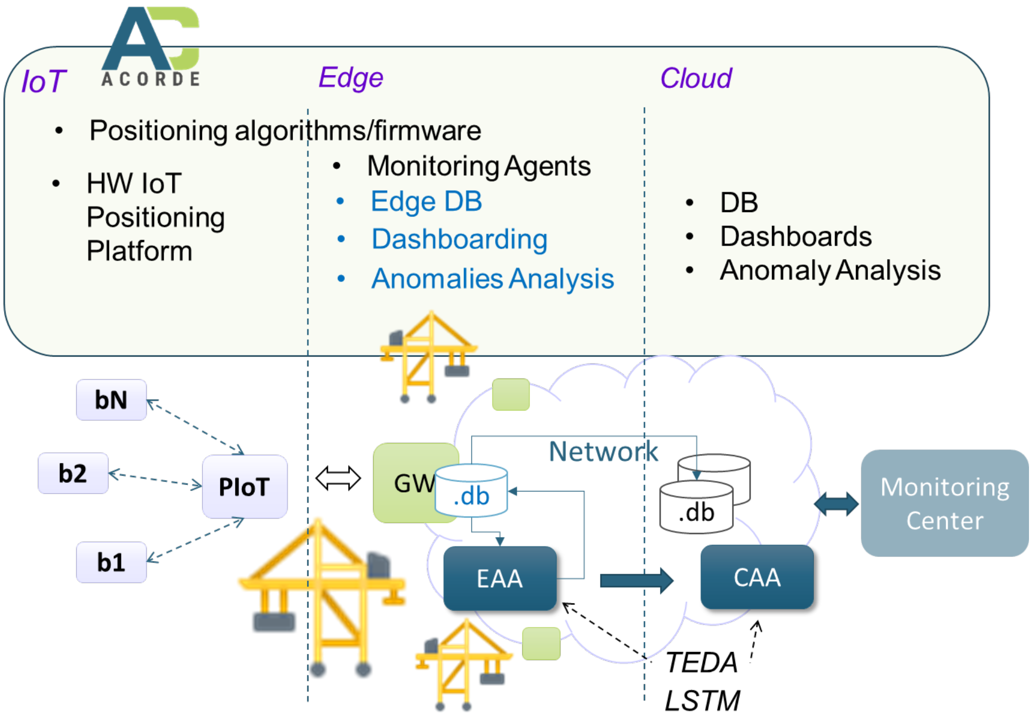


Figure 2 Position Monitoring Architecture

With respect to D2.2 [AIDOART-D2.2], a main novelty has been an assessment, done by ACO, on the services capabilities enabled by the computational resources of the gateway device under development. It has been checked that the new design, while holding a cost similar to existing reference industrial monitoring gateways, is not only able to run monitoring agents, but it is capable to run smoothly other edge services, like:

- local database services
- dashboarding related to edge data

- Custom services, e.g. related to position monitoring
- local data analysis services.

More specific to this context, this opens the door to perform anomaly analysis at the edge. From a global perspective, the assessed capability of the gateway should enable the federation of database services collaborating on a heterogeneous edge-cloud architecture. In a similar way, there is the possibility for the federation of analysis services. Specifically, running at least some anomalies analysis on the edge will make sense in many scenarios.

The previous section 3.1 reports the extension of ESDE to enable anomalies analysis at design time from traces obtained from the execution of embedded software on the virtual platform model. There, a set of transversal aspects of Generic Anomalies Analysis (GAA) research performed by ACO was presented. Those aspects are equally applicable here, but in an operational (AIOPS) scenario.. Therefore, this solution implements the following AI-Augmented Tool Set components:

- Engagement & Analysis (AIOPS Engineering)

The focus is the analysis done on the edge+cloud part of the positioning monitoring solution developed by ACO. In this sense, the architecture Figure 2 provided above shows a distinctive aspect, i.e., the infrastructure now enables the federation of the anomalies analysis (AA), i.e. the cooperation of an edge-side anomalies analysis (EAA) with the analysis performed at the cloud (CAA).

The conducted GAA research is looking into the feasibility of the assessed AA techniques (LSTM and TEDA based) for enabling an online analysis, over the real-time monitored data (in contrast to the design time analysis explored with ESDE).

In a former stage, the research is conducted to assess that the analysis can be implemented and run in the cloud by a server or PC platform. Then, it will be assessed whether it can be run and in which terms on the GW platform. The objective is that the GW can release the cloud of some analysis tasks which, under consideration of aspects like data volume, data rate, and which data need global awareness, make a more efficient use of the overall computational, memory and network resources of the computing continuum architecture.

3.2.2 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:** Partially Implemented **Estimated Delivery Date:** MS4 (M20) **Licence:** Proprietary – ACO. (It refers to a specific monitoring solution on the Cloud-Edge-IoT architecture developed by ACO and considered key for protecting	For the integration on the monitoring infrastructure, ACO has covered so far: i) a preliminary definition of application and system health metrics related to the positioning system (to

	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).	be covered on a generic AIDOaRt meta-modelling); ii) the setup of data aggregators, a Mock Up infrastructure, and preliminary version of monitoring agents, iii) a first version of a visualisation dashboard. iV) a former version of the demonstrator with the database set up on the gateway. It has been also assessed that the gateway has capacity for further functionality, typically to be encapsulated in a docker container. The plan is to encapsulate the AI/ML based EAA in a docker container in the gateway.
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: Partially Implemented Estimated Delivery Date: MS5 (M24) Licence: Proprietary – ACO. (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).	A preliminary study has been done with LSTM neural networks for detecting anomalies on individual signals. Work is on going to test on multiple time series. A former implementation of TEDA metrics computation, required for TEDA based AA is in place. Once AA modules with these techniques are ready, they will be applied to specific traces from the positioning monitoring system.

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:** Partially Implemented **Estimated Delivery Date:** MS4 (M20) **Licence:** Proprietary – ACO. (It refers to all the embedded software, and source code in general related to the industrial positioning solution 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).	At this stage, ACO has already designed the architecture of the proof-of-concept of the edge+IoT side of the solution. A first version of the gateway (edge side) is ready, and it is currently implementing a second version. The IoT side relies on a Positioning IoT (PIoT) device based on an embedded processor, a GNSS receiver, and at least a wired industrial interface to be plugged to the gateway (edge). ACO, together with Prodevelop, have already defined the PIoT requirements. ACO has already designed a PIoT platform and it is implementing it.

Table 3 Capabilities Implementation Status of the Position Monitoring for Industrial Environment Solution

3.2.3 Useful Resources

none

3.3 Solution - devmate (AST)

3.3.1 Overview

devmate is a software testing solution suite that provides semi-automatic unit testing in various development environments and programming languages. Design goals include the black box testing approach, 100% International Software Testing Qualifications Board (ISTQB) test techniques, model based testing and test driven development and openness through its UI. The test engineer, which does not need to be an expert in the given language, can set and generate test cases through the UI, whereas a developer will find the generated unit test code easy to read and modify.

The novelty lies in our black box approach using equivalence class partitioning and boundary value analysis. Only interfaces are being parsed from code while the user sets and modifies test cases through a grid based UI. This process ensures consideration of all necessary test cases to reach the desired code coverage.

Figure 3 hereafter shows devmate's grey-box approach compared to others. Since code is still scanned in by reading the definitions and interfaces, and because Unit Test Code can be generated it shares some overlap with white-box testing. The generation of the test cases however follow the black-box principle and further functionalities allow for integration testing as well.

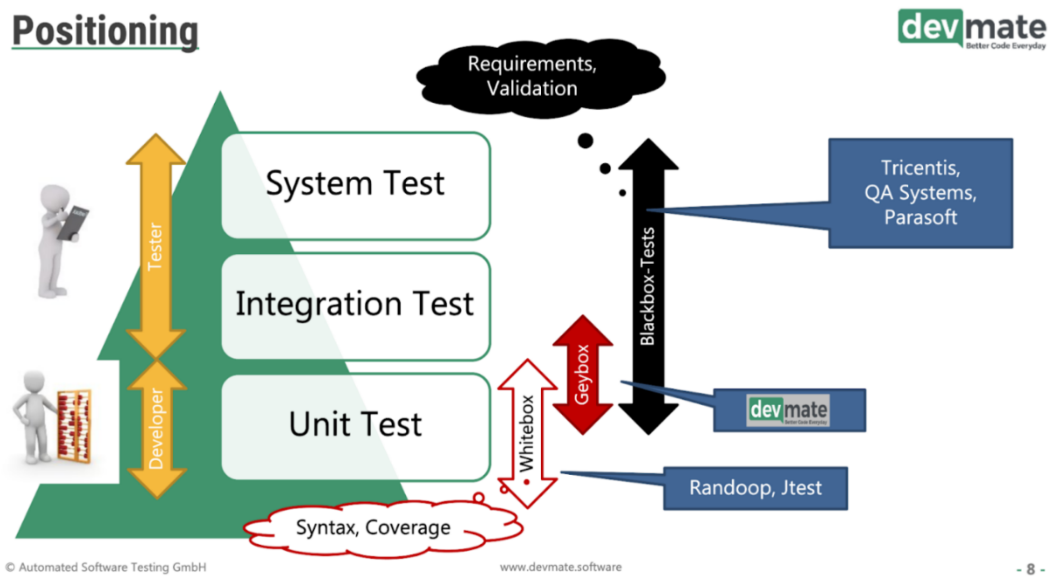


Figure 3 devmate approach vs. other approaches

devmate can assist with testing CPSs at design time by parsing their source code and producing tests based on black box testing enriched with AI techniques, to help finding test cases and test values.The possibility of runtime testing can be analysed, e.g. receiving error messages as input and semi-automatically generating test cases, which would be particularly relevant in the context of CPSs.

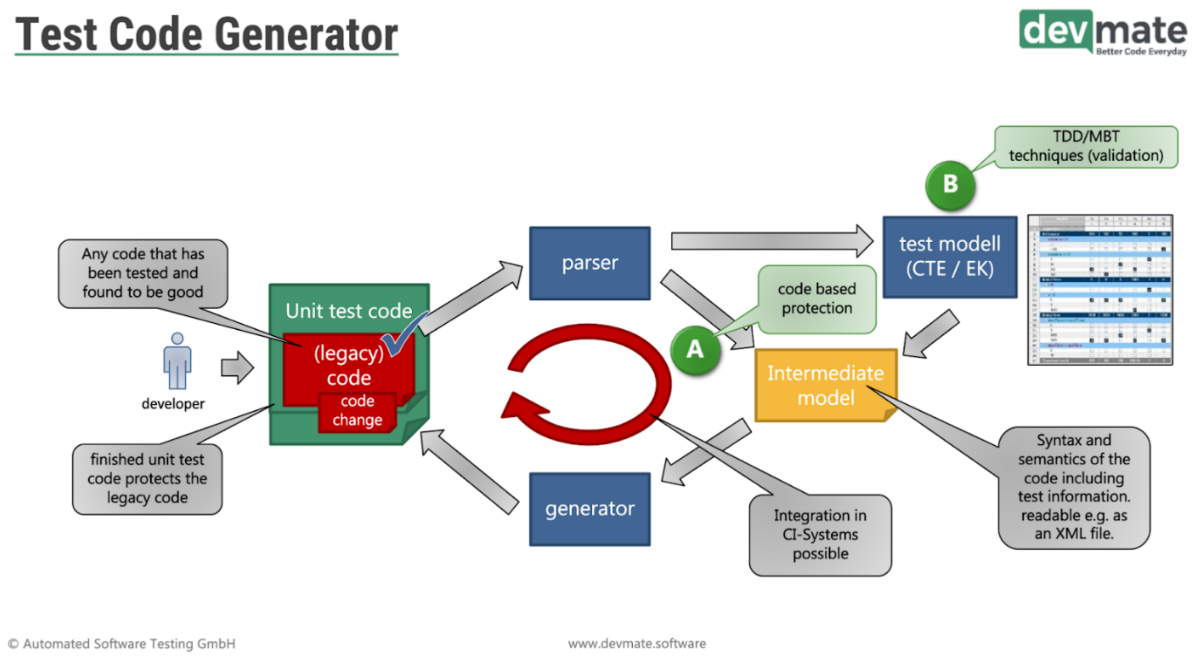

Test Code Generator
devmate
TDD/MBT techniques (validation)
B
Any code that has been tested and found to be good
Unit test code
(legacy) code
code change
developer
finished unit test code protects the legacy code
parser
generator
test modell (CTE / EK)
code based protection
A
Intermediate model
Integration in CI-Systems possible
Syntax and semantics of the code including test information. readable e.g. as an XML file.
© Automated Software Testing GmbH
www.devmate.software

Figure 4 Overview of the devmate architecture and process

Figure 4 shows a high-level view of the architecture and process of devmate. Code is being parsed and a test model generated. The intermediate model is used internally by different components like the code generator as shown in this example. devmate continues to monitor the generated code and modifies it in accordance with new user input from the UI.

The microservice architecture allows us to easily introduce new modules like mock-code generation and enhance the test-generator through AI methods. This currently includes a supervised learning agent for user input and a research project on automatic suggestion of test data values/ranges from parsed interfaces.

This solution implements the following AI-Augmented Tool Set components:

- AI for Testing (AI DevOps Engineering)
- Engagement & Analysis (AIOPS Engineering)

3.3.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
Code Parser: **Parse input-/output-parameters from existing code or unit-/system-models (eg. XML / DSL specified models)**	**Implementation Level:** Partially Implemented **Estimated Delivery Date:** MS5 (M24) **License:** Proprietary license (AST)	Implemented for supported languages (C#, Java). We have a working parser prototype for the C language. A C++ parser is in the design phase. Future plans include other popular or requested languages like Python.
Testcode Generator: **Generating ready to run testcode for several unit-testing frameworks (well structured, datadriven, executable, automatic mocking)**	**Implementation Level:** Partially Implemented **Estimated Delivery Date:** MS4 (M20) **License:** Proprietary license (AST)	Implemented for supported languages (C#, Java). Generator for the C language is in development. Future plans follow directly from parser development.

Table 4 Capabilities Implementation Status of the devmate Solution

3.3.3 Useful Resources

Main website: https://www.devmate.software/

Other relevant resources: https://ebook.devmate.software/

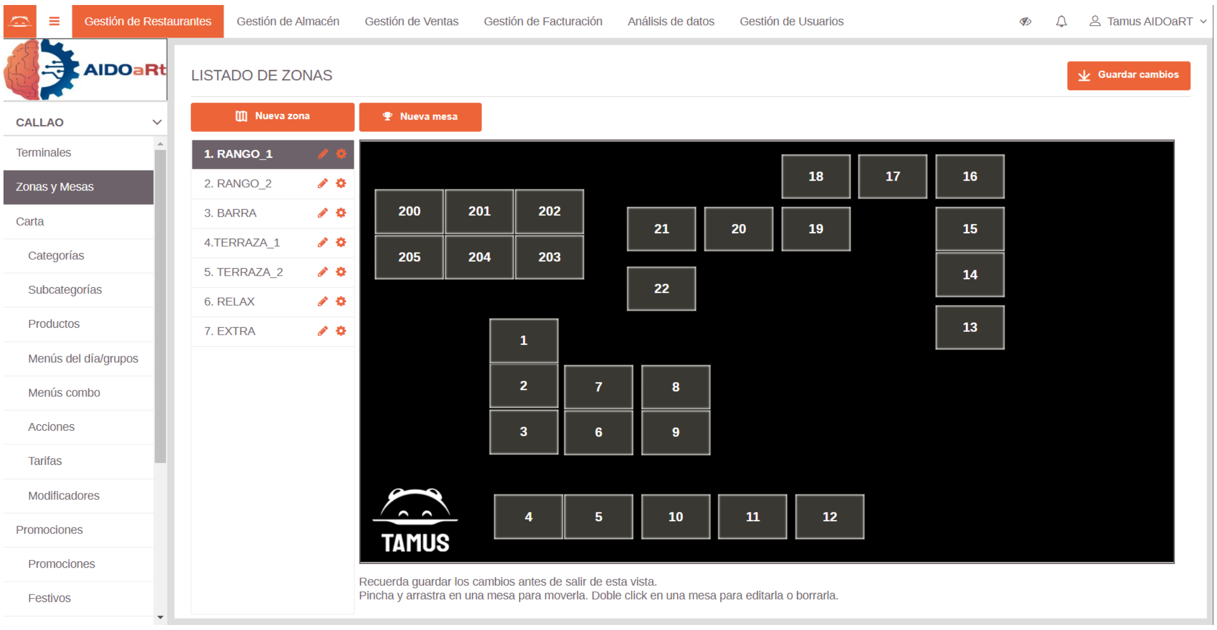
3.4 Solution - HIB_logAnalyzer (HIB)

3.4.1 Overview

HIB_logAnalyzer (or HIBLA from now on) is a tool to automate the analysis of log files captured during the runtime of the TAMUS application. It is not used during runtime but afterwards to provide a post-mortem analysis of the execution of the system. This is done in batches, typically weekly and then a report is generated for the developers of the TAMUS system.

The logs a provided by a large set or software and hardware components in the restaurant system:

- The restaurant system front-end, used by waiters and managers to configure the different aspects of the restaurant (e.g., prices) and ensure that the configuration reflects the physicality of the restaurant (e.g., table layouts and numbering, using in the description of table orders).
- The underlying Operating System logs of the system where TAMUS is run. This is typically a single server per restaurant running Windows plus a cloud server running Linux (see deliverable D1.3, [AIDOART-D1.3] for a description and figure of the entire architecture).
- The different hardware devices used in each restaurant: mobile phones used by waiters as order taking devices, the payment devices (card readers and cash drawers) and other ancillary items such as the wireless access points providing customer Wi-Fi.



Recuerda guardar los cambios antes de salir de esta vista.
Pincha y arrastra en una mesa para moverla. Doble click en una mesa para editarla o borrarla.

FIgure 5: example of table management in TAMUS for a fictional restaurant in the AIDOaRt prototype. Rectangles in the black box correspond to different seating positions for customers.

The result of the use of HIBLA 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). 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.

For implementing these capabilities, this solution implements the following AI-Augmented Tool Set components from AIDOaRt.

- AI for Monitoring (AI DevOps Engineering)
- Ingestion & Handling (AIOPS Engineering)

In the following sections we will detail its operation, status and plans for development during AIDOaRt.

3.4.2 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:** MS5 (M24) **License:** Proprietary HIB	The system is partially implemented in a barebones version. The text ingestion and first stage analysis is mostly solved using command line tools (sed, awk) along with the use of Graylog Open. For the analysis of natural text comments we are integrating the Stanford NLP pipeline for Spanish (many logs contain comments in Spanish) and for the time series analysis we're doing an analysis of introducing a custom LSTM network using the Keras framework and Python libraries.

Table 6 Capabilities Implementation Status of the HIB_logAnalyzer Solution

3.4.3 Useful Resources

Main website: current TAMUS overall prototype: https://aidoart.tamus.io/

Other relevant resources: TAMUS commercial website: https://tamus.io/

3.5 Solution - EMF Views (IMTA)

3.5.1 Overview

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. 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 CPSs.

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.

Figure 6 shows the overall architecture of the EMF Views solution as well as the different artefacts that are involved in a given viewpoint/view creation and manipulation process..

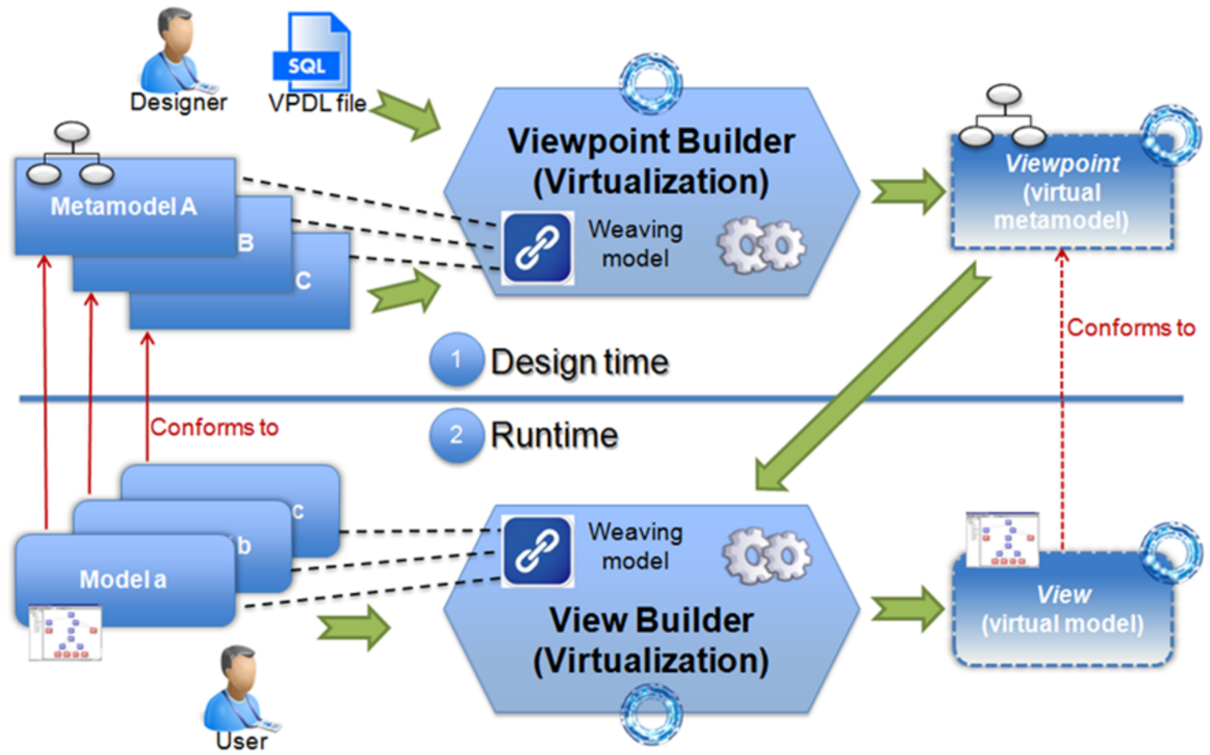


Figure 6 Overview of EMF Views architecture

The Viewpoint Builder provides the required virtualization API at metamodel-level in order to create and handle viewpoints. As said before, EMF Views relies on the Eclipse Modeling Framework (EMF) for the generic manipulation and inter-connection of any metamodels defined in Ecore. Similarly, the View Builder provides the needed virtualization API at model-level in order to create and handle corresponding views. It also relies on EMF for the handling and inter-connection of any models that conform to the metamodels used in the corresponding viewpoint. EMF Views comes with a ViewPoint Definition Language (VPDL) that allows users to define more easily, in a textual way, their viewpoints (and initiating corresponding views to a certain extent). Based on the feedback on its usage, this language could be further refined and extended. Moreover, other languages (textual and/or graphical) could be designed and plugged into the EMF Views framework if needed.

The Viewpoint Builder and View Builder, via their virtualization API, provide the support for automatically computing viewpoints and initialising (at least partially) the corresponding views. Scalability is already not an issue at metamodel/viewpoint-level, due to the reasonable size of such (meta)models. Experiments have also been performed at model/view-level accordingly and they showed promising results in terms of model-level scalability. Additional experiments to be conducted could also involve the use of Machine Learning techniques in order to improve such a scalable computation. This is particularly important in the context of the model-based engineering of large and complex CPSs, as such engineering processes require the use of possibly very large and heterogeneous models.

Currently, the Viewpoint Builder and View Builder provide a basic support for Verification and Validation based on metamodel conformance. The provided VPDL language also comes with base syntactic validation. More advanced Verification and Validation (V&V) capabilities could be added to EMF Views depending on the needs of the use case partners in the project, for instance to facilitate the verification and/or validation of some CPS-related properties. The Viewpoint Builder and View Builder natively stores the viewpoints and views (respectively) by relying on a third-party weaving model. Generated views are already partially editable by default (e.g., attribute values can be modified and reflected back to the base models). However, the integration of different view update strategies, possibly relying on the use of Machine Learning techniques, is still a challenge to be fully addressed. This is notably important in the context of a CPS model-based engineering context, since numerous models can be used and updated at different steps/iterations of the CPS model-based engineering process.

This solution implements the following AI-Augmented Tool Set components:

- AI for Modeling (AI DevOps Engineering)

3.5.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
Model Viewpoint and View specification: Be able to easily specific the viewpoint / view by defining the concerned elements from the different metamodels / models, their interrelations, etc. In the context of AIDOaRt, we plan to consider such viewpoints and views to interconnect different models of the target systems (e.g. requirement models, design models, code models, data models) and then to use these viewpoints/views as a way to better support various activities of the continuous engineering process (e.g. modelling of course, but also requirements or monitoring).	**Implementation Level:** Partially Implemented **Estimated Delivery Date:** MS4 (M20) **License:** EPL 2.0 / GPL 3.0	As introduced earlier, this tool capability is already available in the current version of the EMF Views. However, some updates are still being performed on the VPDL language for facilitating the specification of viewpoints and views (as some bugs have been detected on particular view operations). In the next phase of the project, other ways to specify viewpoints and views (e.g. add-ons to VPDL, new language) may be added to EMF Views, notably as long as it relates to the internal use of Machine Learning techniques, in order to complement the

		already available automation support.
Model Viewpoint and View navigation and query: **Be able to efficiently navigate and query an already computed view.**	**Implementation Level:** Partially Implemented **Estimated Delivery Date:** MS5 (M24) **License:** EPL 2.0 / GPL 3.0	As introduced earlier, this tool capability is already available in the current version of the EMF Views. However, we are still fixing some bugs regarding the handling of already built views. This notably concerns views over contributing models including one or several profiled UML models (e.g. SysML models), as of particular interest in the particular context of CPSs and the AIDOaRt project.

Table 7 Capabilities Implementation Status of the EMF Views Solution

3.5.3 Useful Resources

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

Source code repository (if open source): https://github.com/atlanmod/emfviews

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

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

Related publications:

- 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, Springer Verlag, 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. ACM/IEEE 21th International Conference on Model Driven Engineering Languages and Systems (MODELS '18), Oct 2018, Copenhagen, Denmark. pp.211-211, ⟨10.1145/3239372.3242895⟩.

3.6 Solution - MOMoT (JKU)

3.6.1 Overview

MOMoT is a framework that combines model-driven engineering (MDE) techniques with search-based optimization (population-based search and local search) to solve highly complex problems on the model level. This framework represents problems through Ecore metamodels and their respective model instances. These problem instances can be manipulated through dedicated model transformations modelled as graph transformation rules in Henshin language[1]. The output model's desired and prohibited characteristics (objectives and constraints) can be specified using OCL[2] or a Java-like expression language (Xbase[3]). Search-based optimization techniques can then be used to search for a Pareto-optimal set of transformation orchestrations, i.e., an ordered sequence of transformations and their parameters, to produce models with these characteristics.

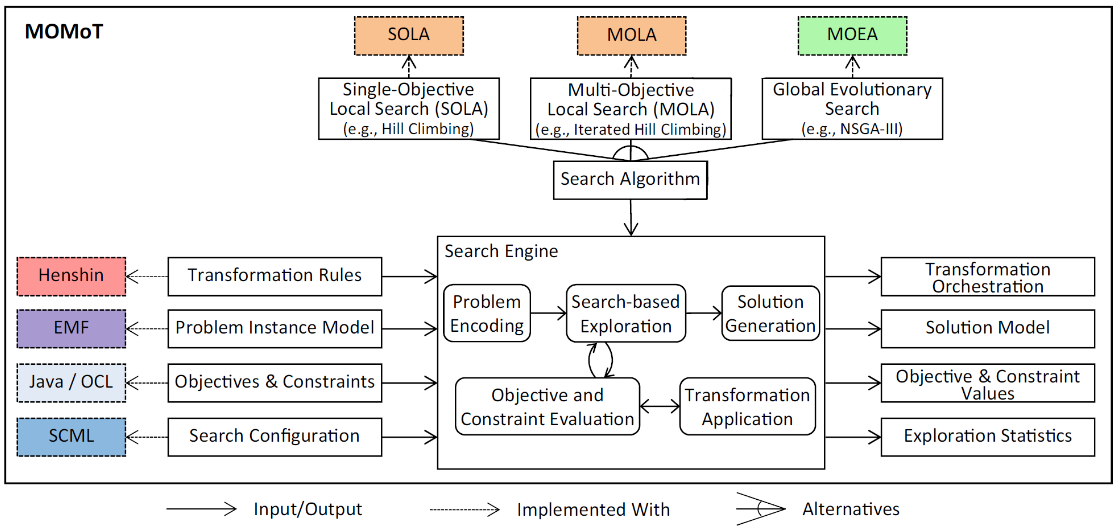


Figure 7 Overview of MOMoT

An overview of MOMoT is depicted in Figure 7. The studied problem domain is defined with a metamodel using Ecore. The concrete problem can be defined by providing an instance model of this metamodel. The model transformation, whichtransform the problem instance model into a solution model that fulfills certain quality criteria, is defined with the model transformation language Henshin. The quality criteria of the desired solution model comprising objectives and constraints are defined using Java or OCL. MOMoT's search engine is responsible for investigating the search space to find an orchestration of the model transformation rules that produces a good solution model concerning the defined quality criteria. MOMoT provides different search algorithms for investigating the search space, including multi-objective evolutionary algorithms (MOEA), single-objective local-search algorithms (SOLA), multi-objective local-search approaches (MOLA), and Reinforcement Learning (RL) based.

[1] https://wiki.eclipse.org/Henshin
[2] https://wiki.eclipse.org/OCL
[3] https://wiki.eclipse.org/Xbase

In order to employ reinforcement learning (RL) methods, the task of finding valuable model transformations (MTs) can be formulated by means of a Markov Decision Process (MDP). Generally, the MDP is defined by a set of states *S* in an environment, a set of actions *A(s)* executable in a given state *s*, a probability distribution *P (s'|s, a)* over all possible successor states when performing action *a* in state *s*, and a reward function *R(s, a, s')* to assess the benefit of performing action *a* in state s and ending up in state *s'*. Figure 8 depicts the MDP adopted for rule-based in-place MTs. The agent interacts with the environment as it selects a rule *a* to which the environment responds with the transformed model *s'* as a result of applying *a* on the previous model state. Furthermore, a reward *r* hints at the value of selecting rule *a*. In order to find an optimised version of a model, the agent needs to choose rules successively that, when executed on the initial model, produce the result models that optimise the target objectives. In this sense, the states are models in their current composition, the actions are rule applications that modify a model's composition. Such modifications may be feature changes or adding/removing model elements based on a predefined set of rules. The reward can be assessed as the change in the objective value after applying a rule on a model. Rewards are to be maximised, hence, in case of a minimization target, the additive inverse of the fitness value needs to be maximised. Since no uncertainty is involved in the MT process, i.e., the successor model *s'* after executing a rule *a* on a current model *s* is always identical, the MDP is deterministic. Hence, there is no probability distribution over possible successor models. The RL-augmentation of MOMoT is under development. We expect further empirical studies to be applied to challenges posed by use case providers in the challenges faced during the periodic hackathons.

The output provided by MOMoT does not only comprise the found model transformation rule orchestration and solution model but also the values computed for the defined objectives and constraints for the found solution, as well as statistical information about the performed search.

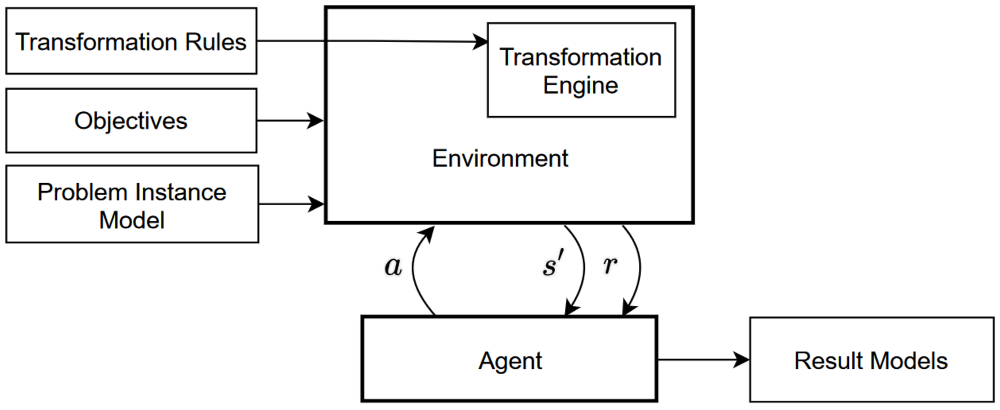


Figure 8 Integration of RL in MOMoT

MOMoT is developed for the Eclipse platform and provides a dedicated configuration language that supports model engineers in configuring the search process. MOMoT has been successfully applied to several case studies which showcase how to use MOMoT in a specific scenario. These case studies are available on the main website. This solution implements the following AI-Augmented Tool Set components:

- AI for Modeling (AI DevOps Engineering)

3.6.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
MOMoT is a framework that combines model-driven engineering (MDE) techniques with search-based optimization (population-based search and local search) to solve highly complex problems on model level.	**Implementation Level:** Fulfilled **Estimated Delivery Date:** MS1 (M6) **License:** Eclipse Public License - V2.0	The stable version of MOMoT is available on the main website. The extended version includes a Reinforcement Learning method, and it is currently under development.

Table 10 Capabilities Implementation Status of the MOMoT Solution

3.6.3 Useful Resources

Main website: http://martin-fleck.github.io/momot/

Source code repository (if open source):

- https://github.com/martin-fleck/momot
- https://github.com/RL4MT/RL4MT

Installation instructions: https://github.com/RL4MT/RL4MT/blob/main/INSTALL.pdf

Related publications:

- Martin Fleck, Javier Troya, and Manuel Wimmer: Marrying Search-based Optimization and Model Transformation Technology. In Proceedings of the 1st North American Symposium on Search Based Software Engineering (NasBASE), 2015.
- Martin Fleck, Javier Troya, and Manuel Wimmer: Search-Based Model Transformations. 2016. In Journal of Software: Evolution and Process.
- Martin Fleck, Javier Troya, and Manuel Wimmer: Towards Generic Modularization Transformations. In Companion Proceedings of the 15th International Conference on Modularity, 1st International Workshop on Modularity in Modelling (MODULARITY), pages 190–195, 2016.
- Martin Fleck, Javier Troya, Manuel Wimmer: Search-Based Model Transformations with MOMoT. In Proceedings of the 9th International Conference on Model Transformations (ICMT), pages 79-87, 2016.
- Martin Fleck, Javier Troya, and Manuel Wimmer: The Class Responsibility Assignment Case. In Proceedings of the 9th Transformation Tool Contest (TTC), 2016.
- Robert Bill, Martin Fleck, Javier Troya, Tanja Mayerhofer, and Manuel Wimmer: A local and global tour on MOMoT. Software and Systems Modeling, pages 1017–1046, 2019.

- Martin Eisenberg, Hans-Peter Pichler, Antonio Garmendia, and Manuel Wimmer: Towards Reinforcement Learning for In-Place Model Transformations. ACM/IEEE 24th International Conference on Model Driven Engineering, pages 82-88, 2021.

3.7 Solution - TATAT (PRO)

3.7.1 Overview

TATAT (Traceability and Test Automation Tools) is a set of utilities and developments made by Prodevelop that was initially designed to support the traceability of errors, through the execution of integration and acceptance tests, for web applications deployed on premise. This toolset helps automate functional and non-functional tests. Its main objective is to automate the validation of infrastructure deployment by automatically running tests. This ensures that the infrastructure is automatically evaluated after each deployment.

The main advantage is that the tool can unify the automation of the execution of tests implemented with different testing tools because it provides the capability to launch multiple tests in an automatic way. For example, Cucumber and Selenium scripts have been used to partially test the behaviour and availability of the different web interfaces of the platform.

Throughout the execution of AIDOaRT, the application is being improved so that it is capable of incorporating new functionalities, which allow testing deployments of infrastructures carried out through Infrastructure as code techniques.

In AIDOaRt, the application is improved to test deployment of infrastructure through infrastructure as code techniques. The functionalities that have been incorporated are:

- Cloud deployment testing.
- Testing of deployments of nodes and computational services.
- Testing of Dockers deployments.
- Service operation testing (access by several protocols).
- Testing of security and access to different services.

This set of utilities is capable of orchestrating the execution of the different test cases that have been automated through the definition of tests in the different testing tools used for this purpose. For this, TATAT includes the development of different plugins, that allow to interact with both, (1) the testing tool and the test case management tool (TestLink), (2) as well as with the project management tool (JIRA).

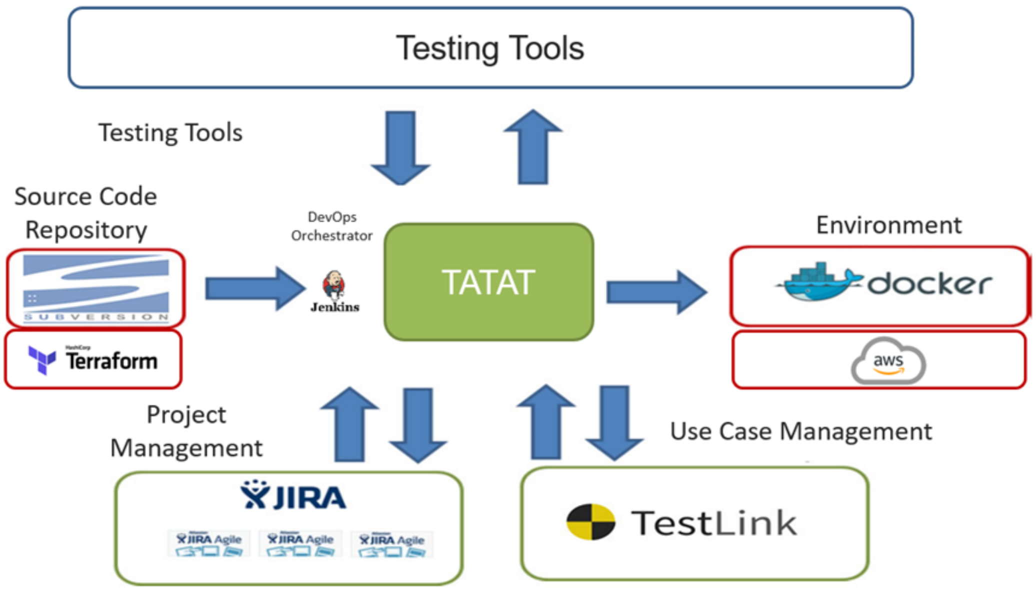


Figure 9 Overview of TATA integration

This solution implements the following AI-Augmented Tool Set components:

- Automation (AIOPS Engineering)

3.7.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
TestAutomationlink: Service that unifies the automation of the execution of tests implemented with different testing tools.	**Implementation Level:** Partially Implemented **Estimated Delivery Date:** MS4 (M20) **License: Open Source (GPL).**	The tool is partially developed and the first validation tests have begun in the PRO_UCS2.

Table 12 Capabilities Implementation Status of the TATAT Solution

3.7.3 Useful Resources

As TATAT is a tool for internal use by PRO, it is not available to other users.

3.8 Solution - CRTQI (QEN)

3.8.1 Overview

Copado provides DevOps analytics (CRTQI) SaaS products consisting of:

- Data Warehouse built of AWS RDS to store DevOps data.

- Pull Services to pull DevOps data from external DevOps tools such as Azure DevOps or various Git version control systems, as well as from Copado's own Copado CI/CD product.
- OpenSearch based, hyper scalable Metrics service to provide highly customizable DevOps metrics for Analytics applications.
- Analytics applications built on the Analytics platform: 1) Test Analytics for Copado Robotic Testing, 2) Copado Analytics for Copado CI/CD (Salesforce), and 3) Enterprise Value Stream Management (EVSM) Analytics.

CRTQI is an analytics application revolving around DevOps metrics, its main objective is to show the development process from planning to deployment, also to show high level DORA metrics (DORA metrics are used by DevOps teams to measure their performance and find out whether they are "low performers" to "elite performers", through metrics such as deployment frequency (DF), lead time for changes (MLT), mean time to recovery (MTTR), and change failure rate (CFR)) to measure the performance. The Copado Analytics application is built to improve the operational (DevOps) efficiency by transforming the customer's DevOps related data generated by Copado CI/CD solution only, and converting the data into meaningful insights. It exposes the current performance of the customers' development process by using customers' own DevOps related data in a form of data visualisation. Its purpose is to help the teams fill the gaps indicated by these DevOps metrics.

This solution implements the following AI-Augmented Tool Set components:

- AI for Testing (AI DevOps Engineering)
- AI for Monitoring

3.8.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
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**	**Implementation Level:** Partially Implemented **Estimated Delivery Date:** MS4 (M20) **License:** https://www.copado.com/company-legal-agreements/	The biggest technical change within H2 2022 was the introduction of OpenSearch technology as the 'search engine' for the data. That fundamental platform change has improved the scalability, responsiveness and reliability to the desired levels. The implementation of Copado Analytics that was also embedded into Salesforce based Copado CI/CD required the team to get familiar with

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		Salesforce integration. A new pull service, authentication mechanism and a solution to embed the dashboard into Salesforce application were developed.

Table 13 Capabilities Implementation Status of the CRTQI Solution

3.8.3 Useful Resources

Main website: https://docs.copado.com/articles/#!copado-ci-cd-publication/copado-analytics1

User manual, tutorials, etc.: https://docs.copado.com/articles/#!copado-ci-cd-publication/copado-analytics1

Other relevant resources: https://www.copado.com/devops-hub/value-stream-management

3.9 Solution - QEDITOR (QEN)

3.9.1 Overview

Copado's QEditor is the next-generation highly advanced test development environment for Robot Framework (an open-source, keyword-driven test automation framework [https://robotframework.org/]) and QWord (Copado's Robot Framework library mainly for for browser automation, enabling easy testing and interaction with web applications) test scripts. By leveraging a massive amount of test case data, it guides you through the scripting process and helps you to create scripts by constantly monitoring and analysing your actions. The guidance is then provided from various directions. It supports numerous features to aid in your test development and make it extremely efficient.

Here are a few benefits of doing test development with QEditor:

- IntelliSense features for Robot FrameWork and QWords scripts: code completion, parameter info, quick info, colour coding, etc.
- Develop locally run in Copado Robotic Testing cloud.
- Prediction of next possible steps based on your test cases and flows so far. Thus ensuring the overall goals of test case scripting which often include completeness and quality
- Automatic detection of deviations in the test scripts' logic from what was expected.
- Predictive estimations on a test case's probability of success, execution time, correlation with other test cases, and more information on test generation. All this way before the test cases are ready for test execution.

There are two ways of using QEditor: directly within Copado Robotic Testing or as a Visual Studio Code (VS Code) extension. For the former, you need to have a Copado Robotic Testing account. If you don’t have a Copado Robotic Testing account, you can quickly create one at https://robotic.copado.com/signup in US and http://eu-robotic.copado.com/signup in EU.

For the latter, QEditor can be installed to Visual Studio Code. All the predictive features and cloud execution capabilities are made available to you when you integrate your VS Code with your Copado Robotic Testing account. If you do not wish to sign up for Copado Robotic Testing and wish to use this extension as a standalone for writing Robot framework test scripts that is possible too.

This solution implements the following AI-Augmented Tool Set components:

- AI for Testing (AI DevOps Engineering)

3.9.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
QEditor - AI-assisted test authoring: **QEditor is 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**	**Implementation Level:** Partially Implemented **Estimated Delivery Date:** MS5 (M24) **License:** https://www.copado.com/company-legal-agreements/	The biggest change within H2 is the introduction of guided authoring with Salesforce metadata which allows the editor to track users' interactions and to (1) check their validity against the implementation on the fly and to (2) predict and auto-complete users test scripts..

Table 14 Capabilities Implementation Status of the QEDITOR Solution

3.9.3 Useful Resources

Main website: https://marketplace.visualstudio.com/items?itemName=qentinelqi.paceeditor

Installation instructions: https://marketplace.visualstudio.com/items?itemName=qentinelqi.paceeditor

User manual, tutorials, etc.: https://help.pace.qentinel.com/docs/current/ui/editor/index.html

Other relevant resources: https://docs.copado.com/articles/#!copado-robotic-testing-publication/qeditor

3.10 Solution - VARA (RISE)

3.10.1 Overview

RISE Research Institutes of Sweden developed an automated solution VARA based on Natural Language Processing (NLP) and Machine Learning (ML) to support the similarity analysis and feature reuse recommendation. VARA enables the automated analysis of textual requirements for a new project to identify reusable artefacts and components from a previous project based on similarity techniques for implementing new requirements.

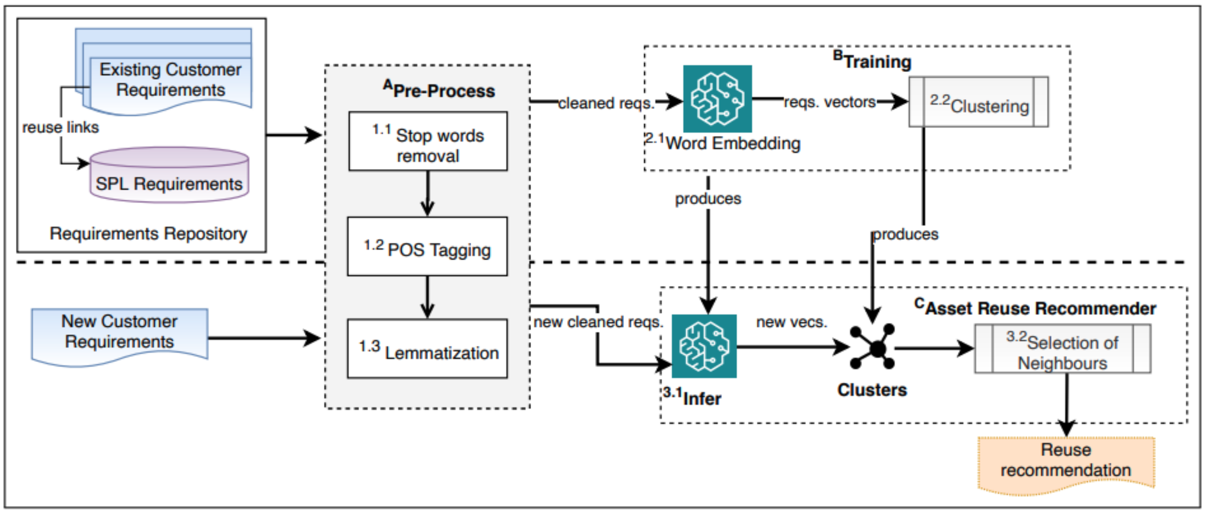


Figure 10 Overview of approach for reuse analysis and recommendation

The overview of the VARA approach is depicted in Figure 10. The existing Software product Line (SPL) requirements are preprocessed to prepare the input for training the ML model with the aim of clustering the requirements based on their similar feature vectors. The generated feature vectors are compared with the new customer requirements for reuse recommendation of requirements from a previous project. This solution is also helpful for the development of new CPSs to reuse the components of existing systems based on similar features.

This solution implements the following AI-Augmented Tool Set components:

- AI for Requirements Engineering (AI DevOps Engineering)

3.10.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
Requirements- based reuse analysis and feature reuse recommendation: Variability-Aware requirements Reuse Analysis (VARA) is a solution based on natural language processing (NLP) and machine learning that can automatically analyse existing projects and recommend implemented features that can be candidates for reuse to realise new products.	**Implementation Level:** Fully Implemented **Estimated Delivery Date:** MS1 (M6) **License:** Proprietary RISE	The VARA reuse solution of requirements for development of new products is completed and available for integration. However, the extension of VARA toolchain is under development. The extended version will include the automated allocation of requirements to different teams and identification of requirements from large tender documents.

Table 15 Capabilities Implementation Status of the VARA Solution

3.10.3 Useful Resources

Related publications:

- Abbas, Muhammad, Mehrdad Saadatmand, Eduard Enoiu, Daniel Sundamark, and Claes Lindskog. "Automated reuse recommendation of product line assets based on natural language requirements." In *Reuse in Emerging Software Engineering Practices: 19th International Conference on Software and Systems Reuse, ICSR 2020, Hammamet, Tunisia, December 2–4, 2020, Proceedings*, pp. 173-189. Cham: Springer International Publishing, 2020.

- Abbas, Muhammad, Alessio Ferrari, Anas Shatnawi, Eduard Enoiu, Mehrdad Saadatmand, and Daniel Sundmark. "On the relationship between similar requirements and similar software: A case study in the railway domain." *Requirements Engineering* (2022): 1-25.

3.11 Solution - Modelio (SOFT)

3.11.1 Overview

Modelio is an open-source modelling tool that supports industry standards such as UML and BPMN. It provides a central repository for storing models, which allows users to combine various modelling languages, such as UML2 profiles like SysML and MARTE, in the same model. This allows for the management of abstraction layers and the creation of traceability links between different model elements. Modelio offers various extension modules and can be used as a platform for developing new Model-Driven Engineering (MDE) features, including code generation and reverse engineering for Java and C++. It also enables users to build UML2 profiles and use a rich graphical interface (cf. Figure 11) for drawing specialised diagrams, editing model element properties, and issuing custom action commands.

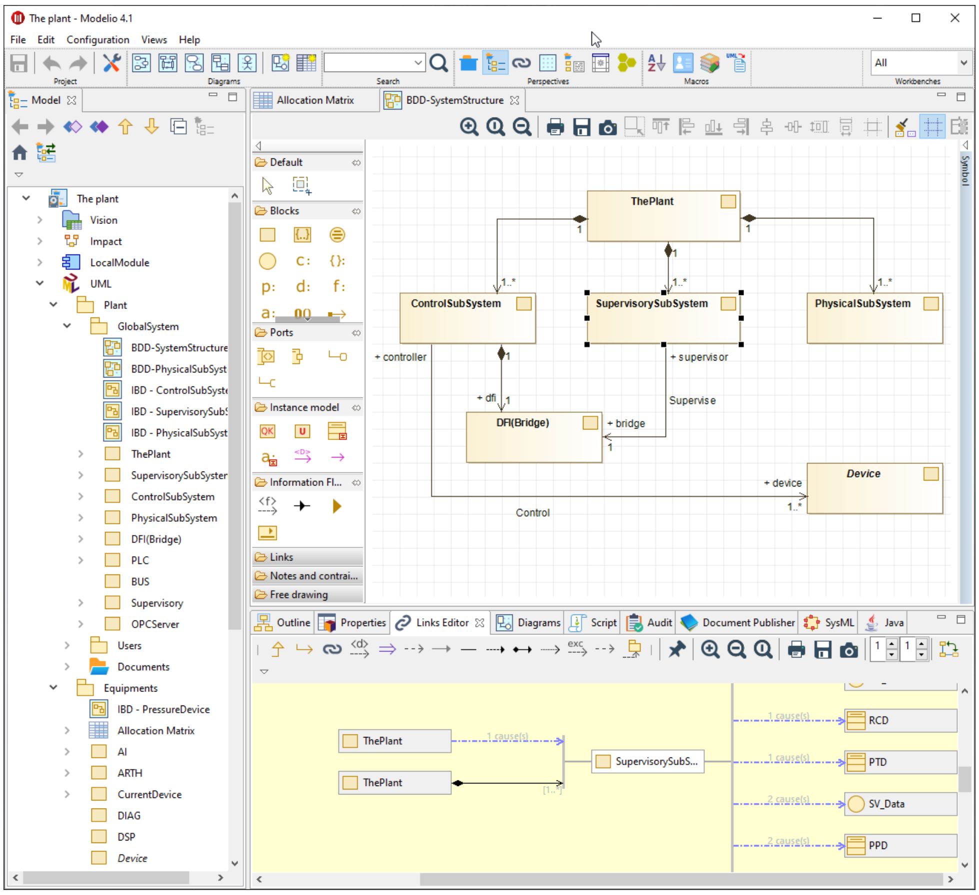


Figure 11 Modelio's GUI workbench

In the context of the AIDOaRT project, SOFT is offering Modelio to capture the various use case requirements, using a Model-based Requirements Engineering approach (MBRE), to specify the research and industrial solutions, to identify potential collaborations between solution and use case providers, and to trace the progress of the use cases and solutions development.

Furthermore, we have identified potential AI/ML extensions of Modelio's capabilities, such as requirements engineering, modelling, code generation and reverse engineering, test case generation from models and specifications, and automation of CI/CD DevOps. For instance, AI assisted modelling may generate model completion hints and suggestions to analysts based on knowledge extracted from previous models.

Based on the use case needs and our current expanding expertise in Natural Language Processing for Requirements Engineering (NLP4RE), we focus in AIDOaRt on applying AI/ML techniques, particularly NLP, on identifying, extracting, classifying and analysing textual requirements. This is helpful for assisting CPS/CPSoS system engineers in managing large unstructured text documents of requirements with rigour and insight based on prior knowledge extracted from previous projects. In AIDOaRt, we created an NLP4RE prototype for requirements similarity check based on real case study datasets from the railway industry. To do so, we use state-of-the-art sentence transformers, such as the MPNet language model, which is based on both BERT and XLNet. Further refinement of the proposed solution is planned for the next project milestones.

The Modelio solution intends to implement the following AI-Augmented Tool Set components:

- AI for Requirements Engineering (AI DevOps Engineering)
- Automation (AIOPS Engineering)
- AI for Code (AI DevOps Engineering)
- AI for Testing (AI DevOps Engineering)
- AI for Modeling (AI DevOps Engineering)

3.11.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
Automated Requirements Identification, Extraction & Classification: Identify, extract and classify requirements from textual documents with NLP & AI/ML techniques	**Implementation Level:** Partially Implemented **Estimated Delivery Date:** MS5 (M24) **Licence:** Apache Public License (APL)	SOFT plans to augment Modelio with NLP4RE capabilities to allow analysts to gain insights on large requirement specifications based on prior knowledge from previous projects. An ongoing collaboration on a case study for Alstom is used to validate our results.
AI-Assisted Modelling: Generate model completion hints and suggestions based to help analysts complete their models based on knowledge extracted from previous models.	**Implementation Level:** Not Implemented Yet **Estimated Delivery Date:** MS6 (M28) **Licence:** Apache Public License (APL)	Modelio already provides a wide range of modelling standards. Inside the AIDOaRT project, SOFT plans to investigate AI techniques for modelling to increase Modelio usability. We did not find a related case study in AIDOaRt, and we may skip this research activity to focus more on the NLP4RE case study.
AI-Optimised Model Checking: Generate more complete and efficient consistency reports by analysing a model based on knowledge extracted from previous models.	**Implementation Level:** Not Implemented Yet **Estimated Delivery Date:** MS6 (M28) **Licence:** Apache Public License (APL)	Modelio already provides a set of predefined rules for model checking. SOFT plans to provide means to allow the end-users to define their own set of rules and integrate them inside the tool. It also plans to investigate AI/ML techniques for data-driven model checking. Having no particular case study within AIDOaRt, this

		research activity may be skipped in order to focus more on the NLP4RE case study.
AI-Enhanced Test Generation: Generate test cases from a model and a requirements specification based on knowledge extracted from prior development projects.	**Implementation Level:** Not Implemented Yet **Estimated Delivery Date:** MS6 (M28) **Licence:** Apache Public License (APL)	Modelio already offers a set of code generators for various programming languages, namely for Java, C++, C#. These can be used, with eventual AI/ML extensions, to automate test case generation from NL requirements with links to source code and test coverage criteria provided as input. In the absence of an industrial case study collaboration in AIDOaRt, SOFT would skip this research activity in favour of the ongoing work on the NLP4RE case study.
Process Execution and CPS Simulation: Simulate the execution of BPM processes and Functional Mockup Interface (FMI)	**Implementation Level:** Not Implemented Yet **Estimated Delivery Date:** MS6 (M28) **Licence:** Apache Public License (APL)	Modelio already allows defining BPMN processes and has a basic implementation of FMIs. We originally planned to extend these capabilities in AIDOaRt to use AI for process mining and for the generation of process execution traces from a process model and a described scenario, or alternatively from a model and an FMU. Without a case study collaboration within AIDOaRt, SOFT would skip this research activity in favour of the active work on the NLP4RE case study.

Table 18 Capabilities Implementation Status of the Modelio Solution

3.11.3 Useful Resources

Main website: https://www.modeliosoft.com/fr/

Source code repository (if open source): https://github.com/ModelioOpenSource/Modelio

Installation instructions: https://github.com/ModelioOpenSource/Modelio or https://github.com/ModelioOpenSource/Modelio/wiki/Build-Modelio-Index for instructions on building Modelio from sources

User manual, tutorials, etc.: https://github.com/ModelioOpenSource/Modelio/wiki as well as the https://www.youtube.com/user/modelioUML and https://www.youtube.com/user/ModelioCommunity YouTube channels.

Related publications:

- V. Ivanov, A. Sadovykh, A. Naumchev, A. Bagnato, and K. Yakovlev, "Extracting Software Requirements from Unstructured Documents," in Recent Trends in Analysis of Images, Social Networks and Texts: 10th International Conference, AIST 2021, Tbilisi, Georgia, December 16–18, 2021, Revised Selected Papers, 2022, pp. 17–29.
- A. Sadovykh et al., "NLP-based Testing and Monitoring for Security Checking," in 33rd IFIP WG 6.1 International Conference on Testing Software and Systems (ICTSS), NOV 10-12, 2021, Univ Coll London, ELECTR NETWORK, 2022, vol. 13045, pp. 235–237.
- I. Nigmatullin, A. Sadovykh, N. Messe, S. Ebersold, and J.-M. Bruel, "RQCODE–Towards Object-Oriented Requirements in the Software Security Domain," in 2022 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW), 2022, pp. 2–6.
- B. Said, A. Sadovykh, E. Brosse, A. Bagnato, *Towards AIDOaRt Objectives via Joint Model-based Architectural Effort*. RCIS 2022 Workshops and Research Projects Track 2022
- A. Sadovykh, D. Truscan, and H. Bruneliere, "Applying Model-based Requirements Engineering in Three Large European Collaborative Projects: An Experience Report," in *2021 IEEE 29th International Requirements Engineering Conference (RE)*, Sep. 2021, pp. 367–377. doi: 10.1109/RE51729.2021.00040.
- A. Sadovykh *et al.*, "Model-Based System Engineering in Practice: Document Generation-MegaM@ Rt2 Project Experience," in *Proceedings of the 14th Central and Eastern European Software Engineering Conference Russia*, 2018, p.

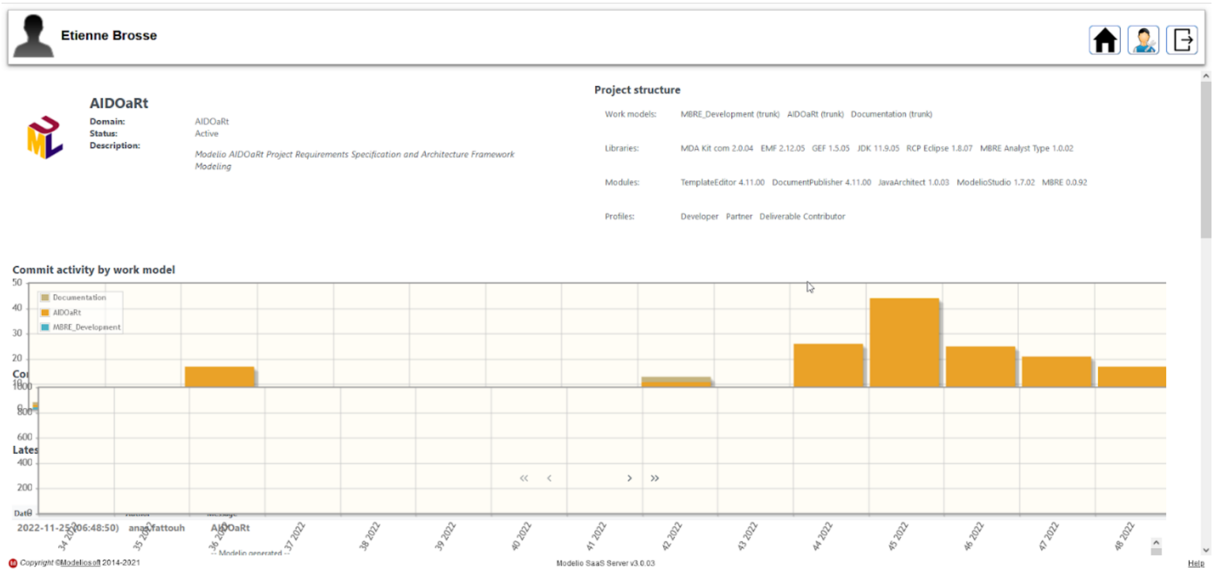
Other relevant resources: https://www.modelio.org/ for Modelio Open Source

3.12 Solution - Constellation (SOFT)

3.12.1 Overview

Complementary to Modelio, Modelio Constellation enables models/projects governance and centralised administration, the specification of indicators and the definition of automated procedures - reports, audit, code generation and continuous integration, document portal update, etc. Constellation also supports enterprise federations and repositories federation, whatever their heterogeneity. Overall, it allows leading architects and engineers to more easily tackle the engineering process of large and complex systems, and notably CPSs.

Moreover, as shown in Figure 12, the web administration interface provides a convenient management tool to project managers or repository administrators.



Etienne Brosse

AIDOaRt

Domain: AIDOaRt
Status: Active
Description: *Modelio AIDOaRt Project Requirements Specification and Architecture Framework Modeling*

Project structure

Work models:	MBRE_Development (trunk) AIDOaRt (trunk) Documentation (trunk)
Libraries:	MDA Kit com 2.0.04 EMF 2.12.05 GEF 1.5.05 JDK 11.9.05 RCP Eclipse 1.8.07 MBRE Analyst Type 1.0.02
Modules:	TemplateEditor 4.11.00 DocumentPublisher 4.11.00 JavaArchitect 1.0.03 ModelioStudio 1.7.02 MBRE 0.0.92
Profiles:	Developer Partner Deliverable Contributor

Figure 12 Constellation web administration interface

Constellation allows to automate the execution of engineers' predefined custom tasks based on the status of a Modelio model and its performed model change or transformation. This can be used for the automation of the CPS/CPSoS system engineering and DevOps processes. In the context of AIDOaRt, this can be further enhanced when coupled with AI/ML augmented capabilities, based on the needs of use cases.

This solution implements the following AI-Augmented Tool Set components:

- Automation (AIOPS Engineering)

3.12.2 Capabilities Implementation Status

Capability Name & Description	Implementation Status	Comments / Release notes
Collaboration Workflow : Automate workflow management of collaborative modelling projects based on the execution of a state diagram across the effective project events	**Implementation Level:** Partially Implemented **Estimated Delivery Date:** MS5 (M24) **Licence:** Proprietary	Constellation workflow allows to associate a state called "workflow state" with model elements and to control the evolution of this state. This allows to automate the life cycle management of system engineering elements, in particular high level ones, such as documents, specifications, API, and technical components. The work on this extension is ongoing, even if there is no case study to validate in AIDOaRt, and a first

		working version is expected for the end of 2023.
DevOps Connector: Connection between models and other DevOps Tools, s.a. ticketing systems (e.g. Jira), and CI (e.g. Jenkins), etc.	**Implementation Level:** Not Implemented Yet **Estimated Delivery Date:** MS6 (M28) **Licence:** Apache Public License (APL)	In Constellation, we are able to automate code and document generation based on users' updates to the Modelio models. In AIDOaRt, we planned to further support DevOps with more automation actions that could be relevant to rigger based on modelling activities. There is no relevant case study in AIDOaRt, and accordingly, the development of such features has been suspended.

Table 19 Capabilities Implementation Status of the Constellation Solution

3.12.3 Useful Resources

Main website: https://www.modeliosoft.com/fr/produits/modelio-constellation.html

Related publications:

- P. Desfray, “Model repositories at the enterprises and systems scale: The Modelio Constellation solution,” in *2015 International Conference on Information Systems Security and Privacy (ICISSP)*, Feb. 2015, p. IS-17-IS-17.

4 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.

A detailed description of the component is provided in section 1 of [AIDOART-D4.1].

4.1 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 17 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 18 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 19 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 20 Solutions Mapping to the "ML-based Dataset Balancing" Interface

4.2 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)	ACO 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 reaction, and thus improvements in the control loop of the monitored system.
devmate (AST)	Devmate could contribute to this component through its planned functionality "Testdata Prediction System"
INT-DEPTH (INT)	This component supports the deduction capabilities based on deep learning architecture. Particularly, it allows to retrieve 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 to detect objects (vehicles, pedestrians, etc.) from images coming from cameras installed in intelligent vehicles.
a2k-depman (ITI)	Multiobjective optimisation 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.
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 (UNIVAQ)	It provides ML-based analysis and prediction capability.

Table 21 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 22 Solutions Mapping to the "IF-INSIGHT-ANALYSIS" Interface

Solution Name	Rationale
devmate (AST)	devmate has methods for checking the similarity of test cases to find duplicates and a supervised learning agent predicting user input based on previous input for custom assertions (a form of user defined test cases in the UI). A planned feature utilises a test case rating to determine how many and which test cases are needed to reach a specific code coverage.
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 (UNIVAQ)	TWIMO provides ML-based analysis for the prediction of specific behaviour.

Table 23 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 24 Solutions Mapping to the "IF-COLLABORATION-SERVICE" Interface

Solution Name	Rationale
Position Monitoring for Industrial Environment (ACO)	ACO 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 25 Solutions Mapping to the "AI/ML for Anomaly Detection" Interface

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

Table 26 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 27 Solutions Mapping to the "ML-based Object Detection" Interface

Solution Name	Rationale

devmate (AST)	devmate's equivalence class prediction module is a supervised learning agent, predicting user input based on previous entered data to generate custom assertion test cases.

Table 28 Solutions Mapping to the "AI for Equivalence Class Prediction" 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)	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. 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.
SoSIM (UCAN)	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.
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 Regressor, 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 29 Solutions Mapping to the "ML-based Prediction For Performance and Resource Utilisation" Interface

4.3 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
Keptn (DT)	Keptn – pronounced captain – is a control-plane for DevOps automation of cloud-native applications. Keptn uses a declarative approach to build scalable automation for delivery and operations which can be scaled to a large number of services. Scriptless delivery for maintenance 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. Separation of concerns for higher compliance Keptn separates the process defined by SREs from the actual tooling defined by DevOps engineers and the information about the artefacts. 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. Event-based automation for easy extensibility 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. Micro-service based tool integrations avoiding lock-in 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. See all details at https://keptn.sh/ .
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)	TATAT automates the validation of a deployment. For each deployment a set of tests 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 30 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 31 Solutions Mapping to the "IF-REMEDIATON" Interface

Solution Name	Rationale
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 allow 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 pre-programmed tasks.

Table 32 Solutions Mapping to the "IF-RESPONSE-AUTOMATION" Interface

4.4 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 ambiguity checker is an NLP tool that receives as input a given textual requirement and provides as output whether the requirement is ambiguous or not, supporting the end-user's decision-making process.
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 supports the requirement phase by checking the consistency of technical specifications using AI techniques.

UNISS_SOL_04 (UNISS)	This solution verifies the consistency of a set of properties' system design by using AI techniques with the purpose of identifying inconsistency to pre-defined specifications.
UNISS_SOL_05 (UNISS)	This solution enables AI/ML based capabilities to support the requirements phase of the system development.

Table 33 Solutions Mapping to the "AI for Requirements Engineering" Component

Solution Name	Rationale
Requirements Ambiguity Checker (MDU)	Support for requirement engineering during the requirement analysis phase using Natural Language Processing (NLP) and deep learning methods for analysis and classification.
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 34 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)	SOFT uses AI/ML NLP techniques to evaluate the similarity of requirements in large specification documents. Within AIDOaRt, SOFT has developed a proof-of-concept tool for checking requirements similarities, using state-of-the-art sentence transformers.

Table 35 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 36 Solutions Mapping to the "AI for Model Consistency Verification" Interface

Solution Name	Rationale
UNISS_SOL_05 (UNISS)	UNISS_SOL_05 provides AI/ML based capabilities to support automated consistency verification of technical specifications.

Table 37 Solutions Mapping to the "AI for Specifications Consistency Verification" Interface

Solution Name	Rationale
Modelio (SOFT)	SOFT 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, SOFT 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 38 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 39 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 40 Solutions Mapping to the "AI for Reuse Analysis and Recommendation" Interface

4.5 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
DTsynth (AIT)	DTsynth uses AI learning techniques to derive a digital twin of device under test.
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-Optimised 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 to extract 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 (UNIVAQ)	It offers advanced modelling and AI/ML analysis .

Table 41 Solutions Mapping to the "AI for Modeling" Component

Solution Name	Rationale
DTsynth (AIT)	DTsynth provides the possibility to learn a model of CPS device, i. e., the finite state automata of a device.
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 algorithm. In such a way, MORGAN can support automation during the design of the system.

TWIMO (UNIVAQ)	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 42 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 43 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 44 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 45 Solutions Mapping to the "AI for View-Model Synchronization" 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 46 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.

Table 47 Solutions Mapping to the "AI for Instance Model Generation" Interface

4.6 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 48 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 49 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 50 Solutions Mapping to the "AI/ML Techniques for Functional Code Generation" Interface

4.7 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)	Builds a digital twin of a CPS using AI/ML which in turn can be used for checking whether a CPS conforms to its description (model checking) or to detect unwanted behaviour without having to put the real device under test.
devmate (AST)	Devmate contributes to this component through its core functionality "Testcase Evaluation" by providing evaluation and reduction of test cases.
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)	-- Test execution & test execution libraries -- QEN offers test execution libraries for various types of application interfaces (web (open sourced), mobile and native application testing). 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 -- Test case maintenance through self-healing tests -- 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 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 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
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. Quality Intelligence for DevOps is a new cloud-based analytics solution that allows CRT users to measure and optimise DevOps

	- Enables generation of the essential metrics based on Agile, DevOps and Flow Framework best practices with one click - Copado and QEN 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)	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 Using normative models of test cases, editor identifies likely/unlikely test steps guiding test cases towards test automation best practices 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
DataAggregator (ROTECH)	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. 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.
Modelio (SOFT)	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.
AALpy (TUG)	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.
UNISS_SOL_03 (UNISS)	This solution supports the testing phase by using AI/ML techniques.
TWIMO (UNIVAQ)	It offers validation capabilities.

Table 51 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 allows to actively/passively learn the finite state automata of a device under test and, therefore, generates a digital twin. The learned digital twin shall provide the possibility to reduce the testing effort. It further allows to learn and generate test cases, i.e., for the ADAS use cases.
devmate (AST)	devmate uses AI methods to support test development. This includes a similarity check on test cases, prediction of user defined test cases, a prototype on generating test data from parsed interfaces and more planned features.
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
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.
UNISS_SOL_03 (UNISS)	UNISS_SOL_03 supports the testing phase by using AI/ML based techniques.
TWIMO (UNIVAQ)	TWIMO supports the phase of testing, validation and verification of the system development, by providing ML-based analysis and prediction capabilities.

Table 52 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 53 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 54 Solutions Mapping to the "Learning Based Testing" Interface

Solution Name	Rationale
devmate (AST)	devmate uses advanced methods for reducing and generating test cases based on interface models. More features for this are at design or prototype phase to enhance this process.

Table 55 Solutions Mapping to the "AI for Test Model Generation" 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 56 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.
devmate (AST)	devmate ranks test cases through multiple parameters to find duplicates and reduces the set through equivalence classes. A supervised learning prototype enhances this functionality through predictive analysis.
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 57 Solutions Mapping to the "AI for Test Case Reduction" Interface

4.8 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 logs and traces associated with 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 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.

Table 58 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 the 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.

Table 59 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 60 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, ACO expects to derive patterns that could be helpful for earlier detection of the aforementioned 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 61 Solutions Mapping to the "AI/ML based Functional / Performance Bug Detection" Interface

Solution Name	Rationale
ESDE (ACO)	ACO 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 62 Solutions Mapping to the "AI/ML for Anomaly Detection" Interface

The given solutions show a good coverage for the AI-Augmented toolkit solutions. We have

- 4 solutions supporting *Ingestion & Handling* having:
  - 2 solutions for *IF-CONTINUOUS-MONITORING*,
  - 2 solutions for *IF-DATA-SELECTION* and
  - 1 solution for *ML-based Dataset Balancing*.

- 13 solutions supporting *Engagement & Analysis* having:
  - 5 solutions for *IF-INSIGHT-ANALYSIS*,
  - 5 solutions for *IF-PREDICTIVE-ANALYSIS*,
  - 1 solution for *IF-COLLABORATION-SERVICE*,
  - 2 solutions for *AI/ML for Anomaly Detection*,
  - 1 solution for *ML-based Prediction for Human–Machine Interaction (HMI)*,
  - 2 solutions for *ML-based Object Detection*,
  - 1 solution for *AI for Equivalence Class Prediction* and
  - 4 solutions for *ML-based Prediction For Performance and Resource Utilization*.

- 5 solutions supporting *Automation* having:
  - 1 solution supporting *IF-REMEDIATION* and
  - 5 solutions supporting *IF-RESPONSE-AUTOMATION*.

- 6 solutions supporting *AI for Requirements Engineering* having:
  - 3 solutions for *IF-AI-FOR-REQUIREMENTS-ENGINEERING*,
  - 2 solutions for *AI for Requirements Similarity Check*,
  - 1 solution for *AI for Model Consistency Verification*,
  - 1 solutions for *AI for Specifications Consistency Verification*,
  - 1 solution for *AI for Requirements Allocation*,
  - 1 solution for *AI for Requirements Ambiguity Check* and
  - 1 solution for *AI for Reuse Analysis and Recommendation*.

- 11 solutions supporting *AI for Modeling* having:
  - 8 solutions for *IF-AI-FOR-MODELING*,
  - 1 solution for *AI-based Modeling Assistant*,
  - 3 solutions for *Design Space Explorer*,
  - 1 solution for *AI for View-Model Synchronization*,
  - 2 solutions for *Model Learning* and
  - 1 solution for *AI for Instance Model Generation*.

- 2 solutions supporting *AI for Code* having:
  - 2 solutions for *IF-AI-FOR-CODE* and
  - 1 solution for *AI/ML Techniques for Functional Code Generation*.

- 13 solutions supporting *AI for Testing* having:
  - 11 solutions for *IF-AI-FOR-TESTING*,
  - 5 solutions for *AI for Test Suite Generation*,
  - 4 solutions for *Learning Based Testing*,
  - 1 solutions for *AI for Test Model Generation*,
  - 3 solutions for *AI for Unit Test Generation* and
  - 3 solutions for *AI for Test Case Reduction*.

- 5 solutions for *AI for Monitoring* having:
  - 4 solutions for *IF-AI-FOR-MONITORING*,
  - 1 solution for *AI for Text Analytics*,
  - 2 solutions for *AI/ML based Functional / Performance Bug Detection* and
  - 4 solutions for *AI/ML for Anomaly Detection.*

5 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.

5.1 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: - acceleration / deceleration histogram - braking 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.

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 dataset. 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 dataset. 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 63 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 64 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 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 65 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 66 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 - braking behaviour (preference of	The IF-DATA-SELECTION is expected to allow user friendly data analysis and visualisation.

	coast down vs active braking) - cornering behaviour (speed per curvature) - max speed preference	

Table 67 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 extract meaningful relationships in proof obligations.
CSY_R03	Use classification on project PO to predict the best tool for automatic proving	This interface could 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 68 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 69 Use Case Requirements Mapping to the "Bug Pattern Discovery" Interface

5.2 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 Engagement & Analysis solution to identify parameters of the worst-case driving profile in terms of emission.
AVL_MBT_R01	Toolkit 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 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 the 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 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.	The Engagement & Analysis component is expected to ensure analysis capabilities including root cause analysis, inference and deduction based on ML techniques. AVL would use the 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.	The Engagement & Analysis component is expected to provide solutions for soiling detection, by using AI/ML techniques. ABI would use AI/ML-based solutions for video elaboration in the context of intelligent driving assistance.
ABI_R10	The system shall adapt when driving into/out of a tunnel.	The Engagement & Analysis component is expected to provide solutions for intelligent driving assistance, as light adaptation, by using AI/ML techniques. ABI would use AI/ML-based solutions for video elaboration in the context of intelligent driving assistance.
ABI_R11	The system shall detect relevant objects while backing up.	The Engagement & Analysis component is expected to provide solutions for intelligent driving assistance, such as an explainable object detection module. ABI would use AI/ML-based solutions for video elaboration in the context of intelligent driving assistance.
ABI_R12	The system shall estimate the vehicles approaching speed.	The Engagement & Analysis component is expected to provide solutions for intelligent driving assistance, such as an explainable object detection module and depth estimation. ABI would use AI/ML-based solutions for video elaboration in

		the context of intelligent driving assistance.
AVL_SEC_R01	Use automata learning and ML techniques to derive SUT models	Engagement & 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	Engagement & Analysis is expected to provide solutions for identifying particular spots in a model for plausibility and formal model checking.
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	Engagement & Analysis is expected to provide solutions for identifying particular spots in a model for plausibility and formal model checking.
AVL_SEC_R05	Use AI (ML) methods to learn on the normal behaviour 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 behaviour on a CAN	Engagement & Analysis is expected to provide solutions for identifying particular spots in the data to facilitate anomaly detection
AVL_SEC_R07	Use intelligent fuzzing techniques on a CAN bus	Engagement & Analysis 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.
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	In order to support Westermo's use cases, some inferences, deductions or predictions are required.
W_R_3	Extract data from steps in DevOps process.	In order to support Westermo's use cases, some inferences,

		deductions or predictions are required.
W_R_4	Log file storing, indexing, searching, clustering and comparing	In order to support Westermo's use cases, some inferences, deductions or predictions are required.
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/realise 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 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 utilised 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 70 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 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.	Verification of sending and receiving messages to ensure correct operation

Table 71 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	IF-INSIGHT-ANALYSIS is expected to provide solutions for identifying particular spots in a model for plausibility and formal model checking.

AVL_SEC_R03	Train ANN on SUT topology discovery using test observation	IF-INSIGHT-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	IF-INSIGHT-ANALYSIS is expected to provide solutions for identifying particular spots in a model for plausibility and formal model checking
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 72 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. 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-INSIGHT-ANALYSIS to verify the service level agreements of the different sensors.

Table 73 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	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

	real components on-which/with-which the system architect has in mind to map/realise the architecture.	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.
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 while backing up.	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.
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 74 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 75 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 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	AI/ML for anomaly detection is expected to provide solutions for identifying particular spots in the data to facilitate anomaly detection.
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 76 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 77 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 while backing up.	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 78 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	Toolkit 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 79 Use Case Requirements Mapping to the "ML-based Prediction For Performance and Resource Utilization" 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 80 Use Case Data Requirements Mapping to the "ML-based Prediction For Performance and Resource Utilization" Interface

5.3 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	The component AIOPS for automation 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	The component AIOPS for automation 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.
VCE_R04	Use of automated tools for compliance verification	The component AIOPS for automation 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 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: • 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 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, 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: • 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	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 Framework "Automation" and "AI for Monitoring". 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". Notes: • The Use Case Scenario TEK_UCS_03 "Operating life monitoring" uses this functionality.
HIB_R03	The AIDOaRT solution must be able to analyse the continuous integration process and detect anomalies.	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.
HIB_R04	The AIDOaRT AI algorithms will enable analysing the success of deploying a new version of the POS application.	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.
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 81 Use Case Requirements Mapping to the "Automation" Component

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 82 Use Case Requirements Mapping to the "IF-REMEDIATON" 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.	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.
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 83 Use Case Requirements Mapping to the "IF-RESPONSE-AUTOMATION" Interface

5.4 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 are to 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 84 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 85 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 86 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 87 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 88 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 89 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 90 Use Case Requirements Mapping to the "AI for Reuse Analysis and Recommendation" Interface

5.5 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 modeling* 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 modeling* to configure the digital twin models based on the AI/ML methods analysis from data gathered and analysed.
VCE_R05	Customise standards based modeling 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 modeling* 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 modeling* 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 modeling* 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 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 is expected to ensure that a human behaviour while 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.
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* 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 conditions on an arbitrarily defined driving route.
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 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.

AVL_TCR_R02	Implement approaches of data driven models based on AVL provided testbed data with the aim to do simulation.	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 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 toolkit 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 subsystems 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 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 standardise and objectively estimate and compute the maturity of a specific KPI result of a project at 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.
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).	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. The functionality is supposed to be provided (and TEK_R_103 to be satisfied) by the component "AI for Modelling" of the AIDOaRt Framework. 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. Notes: • 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

		verification"), parts that can be exploited during the project. • Flow recap: (1) TEK_R_103 — TEK_R_103 asks for a semi-automatic synthesis of the extended models. (2) 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. Please, note that both tests are carried out in the single step № 6 (Case Story TEK_CS_05). (4) 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). (5) TEK_UCS_01 — The Use Case Scenario TEK_UCS_01 ends successfully when no error is detected.
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).	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. The functionality is supposed to be provided (and TEK_R_202 to be satisfied) by the component "AI for Modelling" of the AIDOaRt Framework. 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. Notes: • 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: (1) TEK_R_202 — TEK_R_202 asks for a semi-automatic synthesis of the extended models. (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. (3) TEK_R_203 — The results of the test are analysed in a semi-automatic manner by the functionality requested by TEK_R_203. (4) TEK_UCS_02 — The Use Case Scenario TEK_UCS_02 ends successfully when no error is detected.
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 91 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 generate partial frames of specifications for existing components. In practical 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 92 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	AVL will use AI-based Modeling Assistant to recommend a standardised and objective maturity estimate for a specific

	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.	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 reduce the time required to prove a project, i.e. to demonstrate that claim properties are true, and that implementation respects specification.

Table 93 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 94 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 95 Use Case Requirements Mapping to the "AI for View-Model Synchronization" 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 re-tuning 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.	The *AI for Modeling* component should allow for real-time modelling of the unit under test.
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 should allow for hybrid modelling of the unit under test in.
AVL_MBT_R02	An ML toolkit 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 subsystems which can be modelled independently.	The *AI for Modeling* component should allow for modelling of the unit under test with limited training data.
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 96 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 97 Use Case Requirements Mapping to the "AI for Instance Model Generation" Interface

5.6 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 98 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 99 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 100 Use Case Requirements Mapping to the "AI/ML Techniques for Functional Code Generation" Interface

5.7 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 wide set of potentially suitable configurations for reduction of power consumption of embedded radar device. As well, it will evaluate and helps find the best candidate.
VCE_R04	Use of automated tools for compliance verification	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 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.
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 of 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: • 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").
TEK_R_201	The AIDOaRt Framework verifies in a semi-automatic manner the implemented software artefact (system, subsystem, component) with respect to the requirements as well as with respects to the	TEK_R_201 asks for the following functionality: semi-automatic run-time verification. The functionality is supposed to be provided (and TEK_R_201 to be satisfied) by the component "AI for Testing" of the AIDOaRt Framework. Between "AI for Testing" and other components, there can be interfaces that the former needs in order to provide such a functionality: those with

	architectural and detailed models.	the components "AI for Modelling", "AI for Monitoring", and "Model-Based Capabilities" could be the most likely ones. Notes: • 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 № 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.	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 "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: • 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").

Table 101 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 optimise 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, subsystem, 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, subsystem, component) with respect to the requirements as well as with respects to the architectural and detailed models.

Table 102 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	ABI_R03 is related to the use of automated reasoning and ML techniques for generation of

	generation of optimal test suites.	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 103 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 104 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 105 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	The interface supports the AI techniques for test case reduction of an equivalence set based test model and can satisfy the

	(system, subsystem, component) with respect to the requirements as well as with respects to the architectural and detailed models.	TEK_R_201 requirement related to the AIDOaRt Framework that verifies in a semi-automatic manner the implemented software artefact (system, subsystem, 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 106 Use Case Requirements Mapping to the "AI for Test Case Reduction" Interface

5.8 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 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 monitoring component is expected to provide predictions of future unwanted behaviours 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 monitoring component 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	The monitoring component should be suitable to allow for anomaly detection.
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 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	Automation and/or AI for monitoring will be central for implementing Westermo's use case.
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 Framework "Automation" and "AI for Monitoring". 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". Notes: • 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	Monitor the platform in real time to reduce the downtime and the data lost based on the anomalies and predictions. Uses IF-AI-FOR-MONITORING to monitor the platform.
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 107 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	Verification of sending and receiving messages to ensure correct operation

	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 108 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 analyze 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	The AI for monitoring component should allow monitoring of the system under test.

	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.	

Table 109 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 110 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 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 111 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 112 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 113 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.	We use anomaly detection to find issues in the newly deployed versions of assets in TAMUS.
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	The monitoring component should be suitable to allow for anomaly detection

Table 114 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 115 Use Case Data Requirements Mapping to the "AI/ML for Anomaly Detection" Interface

In the following overview one can see how the components of the AI-Augmented toolkit, respectively their interfaces, are used in the requirements, respectively data requirements:

- 10 use case requirements are mapped to *Ingestion & Handling* whereof:
  - 4 are mapped to *IF-CONTINUOUS-MONITORING*,
  - 3 are mapped to *IF-DATA-SELECTION*,
  - 4 are mapped to *IF-PATTERN-DISCOVERY* and
  - 1 is mapped to *Bug Pattern Discovery*.
- 3 use case data requirements are mapped to *Ingestion & Handling*, all 3 are mapped to *IF-CONTINUOUS-MONITORING*.

- 27 use case requirements are mapped to *Engagement & Analysis* , whereof:
  - 7 are mapped to *IF-INSIGHT-ANALYSIS*,
  - 14 are mapped to *IF-PREDICTIVE-ANALYSIS*,
  - 7 are mapped to *AI/ML for Anomaly Detection*,
  - 3 are mapped to *ML-based Object Detection* and

    - 3 are mapped to *ML-based Prediction For Performance and Resource Utilization*.
- 3 use case data requirements are mapped to *Engagement & Analysis*, whereof:
    - 1 is mapped to *IF-INSIGHT-ANALYSIS*,
    - 2 are mapped to *IF-PREDICTIVE-ANALYSIS*,
    - 3 are mapped to *AI/ML for Anomaly Detection* and
    - 1 is mapped to *ML-based Prediction For Performance and Resource Utilization*.

- 12 use case requirements are mapped to *Automation*, whereof:
    - 2 are mapped to *IF-REMEDIATON* and
    - 10 are mapped to *IF-RESPONSE-AUTOMATION*.

- 5 use case requirements are mapped to *AI for Requirements Engineering*, whereof:
    - 3 are mapped to *IF-AI-FOR-REQUIREMENTS-ENGINEERING*,
    - 1 is mapped to *AI for Requirements Similarity Check*,
    - 2 are mapped to *AI for Model Consistency Verification*,
    - 1 is mapped to *AI for Specifications Consistency Verification*,
    - 1 is mapped to *AI for Requirements Allocation* and
    - 2 are mapped to *AI for Reuse Analysis and Recommendation*.

- 20 use case requirements are mapped to *AI for Modeling*, whereof:
    - 2 are mapped to *IF-AI-FOR-MODELING*,
    - 5 are mapped to *AI-based Modeling Assistant*,
    - 2 are mapped to *Design Space Explorer*,
    - 3 are mapped to *AI for View-Model Synchronization*,
    - 11 are mapped to *Model Learning* and
    - 2 are mapped to *AI for Instance Model Generation*.

- 2 use case requirements are mapped to *AI for Code*, whereof:
    - 2 are mapped to *IF-AI-FOR-CODE* and
    - 1 is mapped to AI/ML Techniques for Functional Code Generation.

- 12 use case requirements are mapped to *AI for Testing*, whereof:
    - 5 are mapped to *IF-AI-FOR-TESTING*,
    - 2 are mapped to *AI for Test Suite Generation*,
    - 5 are mapped to *Learning Based Testing*,
    - 2 are mapped to *AI for Unit Test Generation* and

    - 3 are mapped to *AI for Test Case Reduction*.

- 16 use case requirements are mapped to *AI for Monitoring*, whereof:
    - 9 are mapped to *IF-AI-FOR-MONITORING*,
    - 2 are mapped to *AI for Text Analytics*,
    - 3 are mapped to *AI/ML based Functional / Performance Bug Detection*,
    - 9 are mapped to *AI/ML for Anomaly Detection*,
- 3 use case data requirements are mapped to *AI for Monitoring*, whereof:
    - 2 are mapped to *IF-AI-FOR-MONITORING*,
    - 1 is mapped to *AI for Text Analytics* and
    - 1 is mapped to AI/ML for Anomaly Detection.

6 AIDOaRt AI-Augmented Toolkit Solutions in Use Cases

6.1 Application in AVL Challenge - Real Driver Emission - TUG

In the automotive domain, there is interest in computing KPIs derived from the performance of the car driving along a specific route (e.g. energy consumption or emissions). However, this depends greatly on the driving behaviour. The goal of this challenge is to model the behaviour of human drivers from a given set of observations. More precisely, the aim is to model longitudinal behaviour (i.e. vehicle speed) without knowledge of the exact dynamic situation on the road such as weather conditions and other traffic participants.

A potential method of generating such models is passive automata learning which can be used to infer automata models from a pre-recorded set of traces. This is provided by TUG in the form of the automata learning library AALpy. In this setting, there is non-determinism caused by the unknown circumstances on the road and the variability inherent to human behaviour. This can be accounted for by using probabilistic models such as Markov Decision Processes (MDPs). In this type of automaton model, discrete actions trigger probabilistic transitions between finitely many states, where each state emits a specific output.

By using the known environment stimuli (e.g. speed limit, road curvature, etc.) as inputs and the velocity as output, such a model can be used to describe human driver behaviour in terms of an MDP and can be extracted from existing data using AALpy. This model can then be used to generate possible instances of velocity profiles for different routes by feeding a sequence of inputs corresponding to the new route to the model and observing the outputs.

Since MDPs are finite regarding their inputs and outputs, some form of discretization or abstraction is needed in order for the approach to be applicable. The granularity of these abstractions also influences the level of detail that the model can provide. Finer discretization will lead to a more detailed model at the expense of requiring more data.

This approach results in models that are interpretable by human experts as long as the abstractions are not too complex and can be subjected to automatic model checking which potentially could be used to establish worst case scenarios for a route given some criterion such as energy consumption.

6.2 Application in Smart Port Monitoring (PRO) Challenge - Anomalies Detection on CPS - ACO

As exposed in section 3.1 and 3.2, the Anomalies Analysis (AA) techniques developed and assessed in AIDOaRt by ACO are expected to be applicable to both: the embedded and distributed domain, i.e, at all the sides (IoT, edge, cloud) of modern computing continuum architectures. AA for simulatable models of the IoTs is proposed for anomalies detection on embedded software running on top of virtual hardware models, and federated AA, supporting AA on gateway platforms is proposed for an efficient, yet effective, online anomalies detection on the computing continuum architecture.

These solutions are expected to be applicable to the corresponding dimensions on the smart port monitoring use case:

- in the automatic detection of anomalies and potential functional and/or performance bugs on the solution of the Positioning IoT when this is designed and validated and
- In the detection of anomalies and potential issues on port monitored signals, which can include position and positioning system status data served by ACO, and other port sensed data.

6.3 Application in ABI Challenge - Formal verification of Neural Networks - UNISS

Modern cars are connected systems that acquire inputs from the environment, thus they can be considered as Cyber-Physical Systems. With the increment of sources of information and data, safety represents even more a critical objective and new challenges in the development process are arising. This is especially true where several stakeholders, such as hardware specialists, software developers and system designers have to work together with safety engineers to ensure a reliable and safe system. In this context, the emergence of standards, such as ISO 26262 and ISO 16505, has helped the automotive industry to focus on practices to address safety in a systematic and consistent way.

This opened the path to new research for guaranteeing safety in the automotive context. The use of AI and ML is on the rise in order to enhance the automated verification of systems applied in real safety-critical applications. However, if it is true that AI is now recognized as innovative technology, it is far from being applied in real safety-critical applications due to the lack of methodologies, for example for the predictability of the system in domains such as the automotive one.

UNISS_SOL_02 is for the verification of the neural networks which will be used in the ABI_CS01. That is, given a neural network of interest and a corresponding property, the solution will provide a formal guarantee regarding the compliance of the behaviour of the neural network with the above mentioned property.
One of the aims of the challenge was to analyse the current state-of-the-art and, as consequence, what UNISS_SOL_02 needs to bridge the gap between what is currently available for and what is needed in ABI_CS01. During the challenge the YOLO architecture, which is one of the most popular learning models used for object detection, was identified as a plausible architecture for ABI_CS01.

We compared it with the architectures supported by the winner (and the runner-ups) of the 3rd International Verification of Neural Networks Competition (VNN-COMP'22), which we believe to be representative of the current state of the art. Through a first comparison between the benchmarks used during the VNN-COMP’21 (11 convolutional layers) and the YOLO architecture (109 convolutional layers in its last version), it immediately appears clear that the scalability of the current state-of-the-art verification tools is far from being enough to support our ideal architecture. Therefore, we identified different strategies to bridge the gap between our ideal architecture and the effective scalability of the state-of-the-art verification tools. In particular, we intend to investigate pruning and/or quantization as a means to produce smaller network models which should enable verification without a significant loss in performances. Furthermore, we believe that it could be possible to focus on subsets of the network architecture which are of particular interest for the task at hand and, at the same time, small enough to verify. We are also evaluating how to enhance existing verification tools and methodologies to support network architectures similar to YOLO.

The results obtained by these avenues of research will be integrated in UNISS_SOL_02.

6.4 Application in CAMEA Challenge - Power-Aware Radar Configuration- ABO

This challenge deals with a traffic monitoring system, which is a complex solution consisting of various sensors and components. The CAMEA traffic monitoring use case includes systems that are video-based or radar-based, and they can serve for applications such as travel time estimation or vehicle detection and classification. As these systems are mostly standalone and can operate at places with limited power supply, low power consumption is a key feature. Moreover, since the radar sensor is placed in a hermetically sealed box (weather-proof and water-proof), it lacks active cooling capability. Therefore, we need to keep heat dissipation of the radar chip as low as possible.

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. With expertise, engineers can manually tune a radar configuration in a standard logic way to reduce, e.g., duty cycle of transmission.

The main objective is to reduce power consumption of the device by using AI/ML to guide 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.

The STGEM solution by ABO uses AI/ML methods that can be used for power-aware radar configuration. The tool can generate much more interesting combinations of configuration parameters that minimise power consumption and heat dissipation while providing the same level of performance.

Our proposed solution fits well in the AIDOaRt framework architecture under the “AI for testing” component. The radar itself is equipped with a temperature sensor embedded in silicon and accessible via the radar SDK. A power metre is also installed on the radar, which is accessible via I2C connected to the hardware. Moreover, periodic collection of measurements has been implemented in the radar firmware. Accuracy of internal measurement readings has been compared with external measurements using an oscilloscope.

This challenge is about configuration and monitoring of a physical smart radar device. 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. The process consists of the following steps:

- Generating configurations that can minimise power consumption following the guidelines for parameter tuning and performing consistency checks for sensing quality and radar chirp timing.
- Remotely accessing the radar sensor, resetting the device, and sending a (new) configuration to the radar.
- Collecting a set of measurements per frame including temperature and power
- Analysing the results and trying to find the best candidate configurations using a set of defined metrics.

The STGEM tool automatically creates test suites using minimal domain 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 machine learning (neural networks, generative adversarial networks etc.) into DevOps processes concerning the validation of CPS in place of more traditional optimization-based methods.

STGEM can be integrated into CAMEA's API for radar configuration, tuning, and monitoring. The tool can replace the existing manual radar tuning system and can potentially result in an extension of the CAMEA use case.

6.5 Application in UC Restaurants Challenge - ReqAnalyzer - HIB

The work undertaken to solve the HIB_UCS2 use case scenario requires the use of AI. The topic is requirements management and the challenge is based around the processing of the incoming requirements/tasks for the TAMUS application in the UC restaurants. They need to be classified in two ways, first determining their topic and then assigning them to a given developer in the team with experience. This is usually done by the product owner, manually and gets to be a repetitive task that can use AI for automation.

However, our first approach (undertaken along with project partner SOFT) was based on traditional clustering and NLP which require a great deal of data to be trained, and just a corpus of less than 200 requirements was available at the time. Tests with that system during the first hackathon were not very successful. Thus, we changed gears and proposed two alternative solutions to solve this.

The best candidate, based on methods proposed in [Zero-Shot], focuses on a Zero or Few Shot embedding based classifier: The backbone of the module is based on well-known language models such as RoBERTA or BERT family. The idea is using zero-shot, commonly used in computer vision to predict unseen images, as a way to train a classifier without need of fine-tuning (zero-shot) or with only a handful of labelled task-specific examples (few-shot). The model to be implemented relies on a pre-trained ML backbone that is used to perform tokenization and create task sentence embedding and label task embedding. Then, the classifier processes the embedding results by computing the relatedness between the sequences embedding and the label embeddings using cosine similarity.

Finally, the overall similarity scores will be fed into a classification function, and the probabilities of all labels will be computed to select the maximum score as the most related label to a given task. The idea is using our own Trello dataset for this task.

This is currently being implemented for tests, but we're proposing an alternative in case this first approach does not deliver the best results either. This is based on Unsupervised Learning, same as in our first hackathon attempt, but some changes are made: The module takes in all the existing tasks and applies pre-processing, followed by tfidf (term frequency-inverse document frequency) with Principal Component Analysis (PCA), and clustering. When a new task is to be assigned, the module predicts a cluster for the new task under assignment in the existing clusters, and based on the five most similar neighbouring task groups' allocation. To prepare the input requirements for the tfidf, we used the standard pre-processing pipeline: First, we remove all the symbols and tokenize the task into words and sentences. Then we tag each token with its Part-of-Speech (POS) tags to guide the lemmatization. We lemmatize each of the tokens to its root so as to avoid different interpretations of the same word in different forms. Finally, we remove any potential stop-words in the tasks and consider the lower-cased and lemmatized text as the final output of the pre-processing. TF-IDF serves to represent the tasks as vectors - PCA is used to make a dimensionality reduction and limit the number of features to be considered. Finally, clustering is used to group the tasks. As many clusters as the number of groups for tasks are defined.

With these two approaches we expect that better results will be achieved on our first approach.

6.6 Application in VCE Challenge - Modeling Recommendations- UAQ/JKU

VCE is a global manufacturing company in the construction machine industry. In this regard, VCE mainly works in the context of Product Line Engineering (PLE). Much of the development effort regards managing and configuring these product lines of various machines for various missions and contexts. Indeed most construction machines contain a large set of customizable or variable options in the system specification, split across different subsystems. Traditionally, managing the system architectures is primarily based on informal artefacts and documents; this is often an inefficient method for development given the modern system complexity. As a consequence, there is a need for enhancement of the current methods and practices in order to tackle the growth in system complexity. Model-based practices are one key technology to address the growing complexity of systems with adding more advanced capabilities to engineering workflows. However, previous experience has shown that adopting model-based practices is more complex in an industrial context. Specifically, there must be more maturity regarding tooling capabilities, advanced analysis capabilities, and activities relating to verification and validation (V&V). In more detail, the target of model-based practices is at the earlier stages of development relating to system architecture.

Built on top of a more comprehensive architecture that is still under development, this challenge aims to provide modelling recommendations by combining process mining and recommendation capabilities provided by two separate tools, i.e. Modeling Process Mining Tool (MPMT) and MORGAN. MPMT acts as a modelling action listener and can be installed on Sirius-based editors, e.g. AutomationML, which provides by default a modelling event notification mechanism. MPMT is capable of generating modelling traces as IEEE XES models. It is worth noting that we defined a tailored XES metamodel that allows a detailed characterization of each captured event. Once such traces are produced, MORGAN is employed to retrieve relevant modelling operations. Since AI-based approaches rely on a huge amount of data, we generate the training set using a model generator, i.e., VIATRA. The rationale behind this choice is that this tool provides means to generate "realistic" models rather than random ones, which is required for generating meaningful recommendations. At the time of writing, we did not parameterize the VIATRA generator, and it still produces a model containing random values, thus affecting the quality of the final recommendations. However, we demonstrated that the whole pipeline can come in handy for designing CPSs from scratch.

We are working on setting up an internal survey by involving VCE engineers. In such a way, we can collect real feedback on the proposed solution. Furthermore, we employ XES traces that are manually generated starting from a real use case, i.e. the Dumper system.

7 Conclusion

In this document (deliverable D4.2), we have presented the current status of the activities carried out in WP4 (M13-M24), and the next steps required to successfully advance to the final phases of the project.

In the solution descriptions and the use case challenges, we have explained in detail how they relate to CPS and AI/ML. We now have a mapping of the related solutions to the interface level of that AI-Augmented Tool Set. As well a mapping of the Use Case Requirements and the Use Case Data Requirements to respective interfaces was done. The provided challenges in Section 6, *AIDOaRt AI-Augmented Toolkit Solutions in Use Cases*, are showing a good match of solutions and requirements. Nevertheless, further use case challenges will help address potential gaps.

Over all, there is already a good coverage of the interfaces from both sides, Requirements and Solutions. Nevertheless, the mapping of the Requirements shows that the interfaces *IF-PATTERN-DISCOVERY* and *Bug Pattern Discovery* are still not addressed by any solution; while *ML-based Dataset Balancing*, *IF-COLLABORATION-SERVICE*, *ML-based Prediction for Human–Machine Interaction (HMI)*, *AI for Equivalence Class Prediction*, *AI for Requirements Ambiguity Check* and *AI for Test Model Generation* are not consumed in any requirement.

In the following months we will also carry out activities to further ensure that the results of WP4 are in line with the results of WP2 and WP3. These activities will be conducted in conjunction with WP5.

The results above form the basis to deliver the final version of the AI-Augmented toolkit at M32 of the project.

8 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 toolkit - initial version, 2022. (Available online at: https://www.aidoart.eu/download/18.3022ca0a184b9251f0a105c0/1670243232895/D4.1%20AIDOaRt%20AI-Augmented%20toolkit%20-%20%20Initial%20Version.pdf)

[AIDOART-D4.2] AIDOaRt consortium: AIDOaRt AI-Augmented toolkit - 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)

[RiFa21] G. Rimassa and F. M. Facca. Digital Autonomy inthe 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