



Enabling safe & secure modular UPdates, UPgrades and DynAmic Task reallocation and Execution for Software Defined Vehicles

D5.1. Demonstrator Definitions, Metrics, and Plan V1.1

AVL LIST GMBH (AVL)

30.09.2025



Funded by
the European Union

The project, its research and all other activities leading to these results have received funding from the European Union and the granting authority CINEA under Grant Agreement No 101203060

PUBLIC

© Copyright 2025 UP2DATE4SDV Consortium Partners
101203060 — UP2DATE4SDV — HORIZON-CL5-2024-D6-01

Disclaimer:

Funded by
the European Union

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the granting authority (European Climate, Infrastructure and Environment Executive Agency - CINEA). Neither the European Union nor the granting authority can be held responsible for them.”

Deliverable Information		
Deliverable Number	D5.1	
Deliverable Name	Demonstrator Definitions, Metrics, and Plan	
Nature	Report (R)	
Dissemination Level	Public	
Contractual Date of Delivery	30.09.2025	
Actual Submission Date	31.12.2025	
Lead Beneficiary	AVL LIST GMBH (AVL)	
Lead Author	Hannes Fuchs	
E-Mail	hannes.fuchs@avl.com	
Contributing Beneficiaries	AVL	• Hannes Fuchs • Pedro Jorge
	AVL TR	• Ender Rencuzogullari
	DLR	• Henning Schlender • Patrick Uven • Gregor Nitsche • Kim Grüttner
	IKL	• Ane Mijangos Canales • Victor Julio Ramirez Duran
	BSC	• Enrico Mezzetti
	TAAT	• Alexander Kühhas
	STM FR	• Marcello Coppola
	ZSC	• Ivan Paez
	NTR	• Othon Tomoutzoglou
	KK	• Marcus Haehnel
	ITI	• Joan Josep Valls Mompó • Javier Coronel • Patricia Domingo Alvarez
	CON	• Florin Anton
Reviewing Beneficiaries	BSC	Enrico Mezzetti
	DLR	Gregor Nitsche
Work Package(s)	5	
Related Task(s)	T5.1	CCAM Showcase and Metrics refinement
	T5.2	SDV Applications and Updates requirements
Keywords	Software-Defined Vehicle, SDV, Horizon Europe, Automotive, CCAM, OTA, Update, Upgrade, DevOps, E/E-Architecture, Orchestration, Vehicle-to-cloud, Cloud-Edge Continuum, Reconfiguration, Requirements, Use-Case, Demonstrator, Showcase, Software-Stack	

Project Information	
Contract #	101 203 060
Fixed Project Start Date	01/07/2025
Project Duration	36 months
Model Grant Agreement based on	HE MGA — Multi & Mono - V1.2
Call	HORIZON-CL5-2024-D6-01
Topic	HORIZON-CL5-2024-D6-01-01
Type of Action	HORIZON-RIA
Service	CINEA/C/03
Project Website	https://up2date4sdv.eu/

Document History		
Revision	Date	Description (add and/or delete)
V0.1	01.09.2025	Initial version
V0.2	26.09.2025	Updated version: Carry over of content from Confluence collaboration portal.
V0.3	29.09.2025	Version for review
V0.4	30.09.2025	Reviewed Version
V0.5	30.09.2025	Review comments 1 included, version for acceptance
V0.6	08.10.2025	Review comments 2 included, version for clearance
V1.0	09.10.2025	Final Version (sensitive)
V1.1	10.12.2025	Final Version (public)

Table of contents

D5.1. Demonstrator Definitions, Metrics, and Plan V1.1	0
Table of contents.....	6
Executive Summary	8
Acronyms and Abbreviations	9
List of Figures.....	10
List of Tables.....	10
References and Background Material	11
1. Project Introduction	12
1.1. Project Outline.....	12
1.2. Work-Package Outline	13
1.2.1 Work-Package Objectives	13
1.2.2 Work-Package Description	13
1.2.3 Work-Package Status.....	15
1.3. Purpose and Scope of the Deliverable	15
2. Demonstrators	16
2.1. CCAM Demonstrator	16
2.2. Research Demonstrator.....	16
3. UP2DATE4SDV Showcases	18
3.1. Showcase Overview and Time Plan	18
3.1.1 Overview of Showcases	18
3.1.2 Research Demonstrator Time plan.....	19
3.1.3 Virtual CCAM Demonstrator Milestones	19
3.1.4 SDV/CCAM Demonstrator Milestones.....	20
3.2. Showcase-01: Use Case 1 demonstrated on Research Demonstrator	20
3.2.1 Involved partners	20
3.2.2 Architecture.....	20
3.2.3 Building Blocks.....	21
3.3. Showcase-02: Use Case 2, 3 and 4 demonstrated on Research Demonstrator	22
3.3.1 Involved partners	22
3.3.2 Architecture.....	22
3.3.3 Building Blocks.....	23
3.4. Showcase-03: Use Cases 5 & 6 on Virtual CCAM Demonstrator	24
3.4.1 Definition of Scenarios	25
3.4.2 Involved partners	27
3.4.3 Architecture and Building Blocks.....	28
3.5. Showcase-04: Use Cases 5 & 6 on final CCAM Demonstrator incl. ADAS HiL, physical Controller and Safe & Secure SW Updates	28
3.5.1 Involved partners	29
3.5.2 Architecture and Building Blocks.....	29
4. KPIs & Metrics	33
5. SDV Applications and Updates Requirements	45

6. Summary and Outlook	45
Annex 1: WP5 Requirements Definition	46

Executive Summary

This document covers the Milestone 1 (MS1) delivery D5.1 containing the initial results of T5.1 CCAM Showcase and Metrics refinement and T5.2 SDV Applications and Updates requirements. Therefore, D5.1 is a collection of the specifications of SDV applications of the planned project work in terms of update, upgrade and reconfiguration, as well as the Use Cases refinement and their evaluation metrics.

Please note: The basis of this document is a requirements collection in Confluence collaboratively managed by the partners. Since the individual work packages will refine and improve the requirements of the use cases during requirement collection for the single work packages, this collection is a "living" document throughout the project and will not be closed. Therefore, the deliverable D5.1 is to be considered as a current view, early in the project (M3), to the best of our knowledge created and reviewed with due diligence, but may be subject to change for following reasons: (a) A compatibility issue between the partners solutions/systems is discovered which was not expected upfront, and a work around or solution is not feasible or out of scope from effort perspective. (b) With improved knowledge and experience, a requirement is added, removed, or modified. (c) Requirement elicitation of work packages and/or building blocks results in requirements contradicting with use-case requirements and the requirements are consolidated.

Acronyms and Abbreviations

Abbrev.	Full Name
AD	Autonomous-Driving
ADAS	Advanced Driver-Assistant System
CA	Consortium Agreement
CAD	Connected and Automated Driving
CCAM	Connected Cooperative and Automated Mobility
CoCo	Coordinator Contact
COTS	Commercial Off-The-Shelf
CSA	Coordination and Support Action
D	Deliverable
Diss/Expl	Dissemination/Exploitation
DoA	Description of Action
DPC	Data Privacy Coordinator
EC	European Commission
ECU	Electronic Control Unit
EU	European Union
FSIGN	Financial Signatory
GA	Grant Agreement
HAL	Hardware Abstraction Layer
HiL	Hardware-in-the-Loop
HW	Hardware
IP	Intellectual Property
JU	Joint Undertaking
LEAR	Legal Entity Appointed Representative
LSIGN	Legal Signatory (LSIGN)
M	Month
MBE	Model-Based Engineering
MBSE	Model-Based System Engineering
NDA	Non-Disclosure Agreement
NFA	National Funding Authorities
OAL	OS/MW Abstraction Layer
ODDs	Operational Design Domains
ORP	Orchestration and Reconfiguration Plane
PaCo	Participant Contact
PC	Project Coordinator
PCM	Phase Change Memory
PCoCo	Primary Coordinator Contact
PGA	Project General Assembly
PPR	Periodic Project Report
PU	Public
R	Report

SASE	SAfety and SEcurity
SC	Showcase – A use case shown on a specific demonstration environment
SD-CCAM-V	Software-Defined CCAM vehicles
SDV	Software Defined Vehicle
SEN	Sensitive
SiL	Software-in-the-Loop
SoM	System-on-Module
SOTA	Software-Over-The-Air
SW	Software
SyGMa	System Grant Management https://ec.europa.eu/research/participants/grants-app/reporting/DLV-101203060
TL	Task Leader
TSN	Time Sensitive Network
UCL	Use Case Leader
V&V	Verification and validation
V2C	Vehicle to Cloud
VIT	Virtual Integration Testing
VRUs	Vulnerable Road Users
WP	Work Package
WPL	Work Package Leader

List of Figures

Figure 1: Overview of the UP2DATE4SDV work-packages and their Interaction.	13
Figure 2: Platform Architecture of Research Demonstrator.....	17
Figure 3: Gantt chart of showcases and related deliverables	19
Figure 4: Architecture Overview of Showcase 1	21
Figure 5: Architecture Overview of Showcase 2.....	23
Figure 6: SC-3 Virtual CCAM Demonstrator - Draft of detailed architecture	28
Figure 7: Abstract Architecture (from project proposal)	31
Figure 8: SC-4 Draft of detailed architecture.....	32

List of Tables

Table 2: Use case overview.....	18
Table 3: Overview of Showcases.....	18
Table 4: Milestones of the Research Demonstrator	19
Table 5: Milestones of the CCAM Demonstrator.....	19
Table 6: Milestones of the SDV/CCAM Demonstrator.....	20

Table 7: SC-1 building blocks	21
Table 8: SC-2 building blocks	23
Table 9: SC-4 list of project building blocks	29
Table 10: SC-4 list of non-project building blocks.....	30
Table 11: Overview of use-case specific KPIs and metrics.....	33
Table 12: Overview of proposal objectives KPIs and metrics	41

References and Background Material

Internal Background

- [1] Project Proposal Project Part B: technical alignment (not public)

External Background

- [2] D4.1_UP2DATE-Middleware-foundations-V1.1.pdf: https://h2020up2date.eu/wp-content/uploads/2021/10/D4.1_UP2DATE-Middleware-foundations-V1.1.pdf
- [3] D3.2_ArchDefinition_v02.022.pdf: https://h2020up2date.eu/wp-content/uploads/2021/10/D3.2_ArchDefinition_v02.00.pdf
- [4] Federate Building Blocks (<https://github.com/CSA-FEDERATE/Proposed-BuildingBlocks>)
- [5] Eclipse SDV – SCORE Project Architecture (<https://eclipse-score.github.io/score/main/modules/index.html>)
- [6] HAL4SDV Platform Requirements (https://www.hal4sdv.eu/files/ugd/76d064_3352d3e51b5b43028f11254027a53b3a.pdf)

1. Project Introduction

The European research project UP2DATE4SDV (Enabling safe & secure modular UPdates, UPgrades and DynAmic Task reallocation and Execution for Software-Defined Vehicles), aims to develop a comprehensive ecosystem for seamless and efficient software updates, hardware upgrades and situation-dependent reconfigurations of software-defined vehicles.

1.1. Project Outline

The project focuses on the definition and development of three abstraction layers

- the hardware abstraction layer (HAL)
- the OS/MW abstraction layer (OAL) for the operating system (OS) and middleware (MW)
- as well as on the exploration and prototyping of a secure orchestration and reconfiguration layer (ORP) between the vehicle and the cloud.

This will improve the reliability and performance of the CCAM system, extend vehicle life and reduce electronic waste. In addition, the project's new modularity concepts will accelerate the transition to safer, more sustainable automated driving and enable incremental steps towards Level 4 automation and longer vehicle lifecycles

The overall UP2DATE4SDV goal is to create a comprehensive ecosystem for updatable, upgradable, and reconfigurable software-defined connected and automated vehicles. This will be achieved by developing essential abstraction layers, advanced orchestrators, and an automated robust realization environment, all while adhering to the highest safety and cybersecurity standards. To achieve this vision, the project will target the following objectives:

- O1. Creation of a reconfigurable cloud connected zonal vehicle architecture blueprint based on heterogeneous COTS HW (exploiting and building on a unified HAL)**
- O2. Establishing a resource composable SW deployment, execution and communication reference layer, on top of existing open-source hypervisor and OS stacks (exploiting and building on a unified OAL)**
- O3. Realization of a dynamic update, orchestration & reconfiguration plane**
- O4. Provide harmonized and simplified development, integration and validation processes and framework for SDV applications**
- O5. Assess joint Safety and Security (SASE) of all elements from O1, O2 and O3**
- O6. Demonstration of UP2DATE4SDV solution on several CCAM relevant Use Cases**
- O7. Active participation around CCAM and SDV Initiative community and ecosystem**

To this goal, the demonstrators and use-cases (developed in WP5) define requirements (pink arrows) for the HW and the HAL (developed in WP1), the SW and the OAL (developed in WP2), as well as for the ORP (developed in WP3). Based on that, WP1, WP2 and WP3 provide a basic functional HW-SW-demonstrator for the use-cases – UC1- after Phase 1 (M6), UC2, UC3, and UC4 after Phase 2 (M24), and finally UC5 and UC6 after Phase 3 (M36).

Besides the pure communication of results (green arrows) the proper integration of the results of WP1, WP2, WP3, and WP4 within the use-cases depends on three further cooperation aspects: the safety and security assessment (yellow arrows), the update-, upgrade-, and reconfiguration-processes and the involved component-interfaces (purple arrows), and the cloud-based MBSE-representation of

actual HW-/SW-components during the DevOps cycle establishing the real-world to cloud connection according to digital twins (blue arrows). Finally, while WP6 oversees the dissemination, communication, and exploitation of the project's results, WP7 coordinates all the WPs and their communication, both internally and externally to the SDV-Initiative and to other CCAM-partnerships. Figure 1 summarizes the described interaction and communication between the work-packages (WPs) of UP2DATE4SDV.

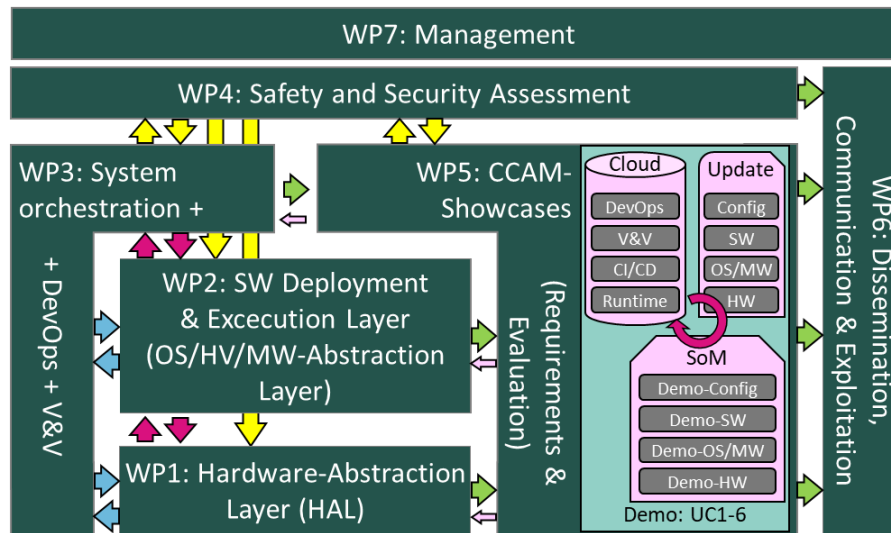


Figure 1: Overview of the UP2DATE4SDV work-packages and their Interaction.

1.2. Work-Package Outline

As shown in Figure 1, WP5 has interaction and interfaces with all other work packages and therefore is the connecting activity between the building block developments, safety and security (SASE) framework and depicts the benefits of contributions in this project for the CCAM goals. It not only has an integrating character for the effort of other activities but will provide feedback on the fitness of the building blocks developed throughout the project.

Supporting developments like DevOps integration, V&V toolchain(s), and model-based engineering will be applied

1.2.1 Work-Package Objectives

The objective of this WP is to integrate the developed technologies on different platforms and to demonstrate the value and benefits for the CCAM goals: connected, safe, secure, shared and automated mobility.

1.2.2 Work-Package Description

T5.1: CCAM Showcase and Metrics refinement (L: AVL, P: all) M1-M3

It will refine the CCAM showcases and the information about how and when the project solution will be integrated to generate them will be included in D5.1. Additionally, the existing KPIs will be supplemented with quantifiable metrics considering different aspects of the technical contributions, like the isolation quality of the modularization approach or the overall system performance, to show

the benefits for CCAM. This will be a joint effort utilizing the special areas of expertise of the different partners.

T5.2: SDV Applications and Updates requirements (L: AVL, P: all) M1-M3

In T5.2 requirements for updateability and upgradeability of Software-Defined Vehicle (SDV) applications towards safety, security, abstraction, deployment, verification & validation and DevOps will be defined. This will support the improvement and definition of KPIs for the demonstrator's implementation in accordance with the derived requirements. Areas of focus will be the observer- and monitoring-components, safety and security, dynamic reconfiguration, model-based engineering, and the integration of update-, upgrade and reconfiguration system orchestrator. For SDV applications the utilization of enhanced digital twins and containerization environments will be investigated. The requirements for the implementation will ensure compatibility with hardware support in today's high-performance System-on-a-Chip (SoCs). Furthermore, system verification and validation requirements will be defined together with testing approaches on different levels.

T5.3: Research Demonstrator Implementation and Technical KPIs evaluation (L: DLR, P: IKL, BSC, ZSC, ITI, CON) M7-M30

The UP2DATE4SDV architecture will be implemented in a public domain (open access) research demonstrator for synchronized updates and upgrades across the vehicle to cloud continuum. Different parts of the proposed technology such architecture, safety and security mechanisms, their validation, modelling and code generation frameworks as well as base technology like the low-level software layers for SoC configuration and monitoring will be contributed by the partners.

T5.4 CCAM Showcase Implementation and Benefits evaluation (L: AVL, P: CON, STM, NEO, DLR, IKL, ZSC, TTT, KK, ITI) M13-M36

The use-cases will be implemented on different platforms to depict the benefits of the technology in the CCAM context (building on top of T5.3). Achievements and performance will be reported via KPIs. The effort will contain:

- Resource observation and monitoring and/or the update-, upgrade and reconfiguration system orchestrator. (DLR, CON, BSC)
- Application of hypervisor-native container mechanisms, safety and security (SASE) mechanisms, model-based engineering frameworks and methodology including security checking of updates. (NEO, STM, IKL)
- Apply E/E architecture validation, including analysis of selected connected and automated driving functions, test design, implementation, execution and analysis of test results for different use-cases (AVL).
- Integration of existing software with new hardware, ensuring compatibility and performance, including development of related KPIs, monitoring, integration. (STM)
- Contribution to standardization. (IKL)
- Utilization of automotive DevOps pipelines for deployment on hardware/SiL/HiL/ViL/Demonstrators for V&V (AVL).
- General effort for setup, work and contribution on demonstrators and showcases (all).
- Evaluation of the use-cases and reporting on the realized benefits and learnings (all).

1.2.3 Work-Package Status

The work package activities were started at the beginning of the project and a weekly meeting with all partners was conducted. Further this deliverable was one focus area of the technical alignment face-to-face meeting conducted mid of September 2025 at Barcelona Supercomputing in Barcelona. All partners aligned on the realization of the use cases, their timeline, detailed metrics and concrete architectures for the demonstrators were drafted, refining the contents of the project proposal.

WP5 progresses as planned and currently no delays are expected.

1.3. Purpose and Scope of the Deliverable

Deliverable D5.1 marks the first deliverable of the UP2DATE4SDV project. The purpose of D5.1 was to achieve a common understanding of the show cases which will demonstrate the project use cases and the benefits of the building blocks to be developed in the UP2DATE4SDV project. By defining the relation of the partners contributions, this deliverable created the basis for collaborative work on the use cases.

The requirements derived from the show case definitions, and the SDV update requirements described in this deliverable will serve as an input for the requirements refinement of work package one to four. The aligned architectures and definition of demonstration environments enhance the understanding of required interfaces and integration efforts of project and non-project artefacts like open-source software, hardware platforms used and existing assets that partners bring into the project.

The KPIs specified in the project proposal were not supported by detailed metrics. D5.1 advances those KPIs by extending them with specific metrics which clarify how we evaluate the success of the planned contributions in this work package.

Finally, a time plan for work on the show cases was created and milestones for single showcases were defined.

2. Demonstrators

This section describes the demonstrators and showcases planned in the project. We defined showcases (SC) as a link between demonstrators and use cases (UC) defined in the project. Therefore, a showcase links a use case to a specific demonstrator and presents the capabilities of the work packages (WP) performing the scenarios described in the use-cases.

There are four showcases planned in the project (SC-1 to SC-4), which will be shown on three demonstrators, namely the "Research Demonstrator", the "Virtual SDV/CCAM Demonstrator", and the "SDV/CCAM Demonstrator (ADAS HiL + Real Controller)" demonstrator. The "Research Demonstrator" will be used to present SC-1 and SC-2, where SC-1 will present Use Case 1. The SC-3 will be showing UC-5 and UC-6 in a virtual environment and will also serve as preparation work for the final "SDV/CCAM Demonstrator (ADAS HiL + Real Controller)" showcase, where the work products of the UP2DATE4SDV project and their benefits for the CCAM and SDV use cases (UC5 and UC6) will be demonstrated. Synchronizing among each other's, these demonstrators are being coordinated by different partners.

2.1. CCAM Demonstrator

The CCAM Demonstrator will apply the building blocks and architecture for update, upgrade and reconfiguration of software and hardware of the SDV in real world use-cases using relevant and representative tools, hardware and test environments from the automotive industry.

We decided to first develop a Virtual SDV/CCAM Demonstrator utilizing a fully virtual environment for SC-3, to enable the partners to utilize this environment for integration and development, as well as allowing to focus on building the autonomous and connected vehicle scenarios described in UC-5 and UC-6. This will include necessary work on artifacts like the perception stack employed, the simulation environment, MBE approaches, DevOps integration, V&V framework and modification of interfaces of artefacts.

The work on the virtual CCAM demonstrator will then be re-used in a "SDV/CCAM Demonstrator (ADAS HiL+ Real Controller)" featuring STM MP2 and STM Stellar control units and an ADAS HIL system. The goal is to keep the scenarios and simulation environment similar, but employ representative automotive hardware. This demonstrator will also be used for the final evaluation of the CCAM benefits and the V&V and SASE framework will be applied.

2.2. Research Demonstrator

The UP2DATE4SDV architecture will be implemented in a public domain (open access) research demonstrator for synchronized updates and upgrades across the vehicle to cloud continuum. Different parts of the proposed technology will be contributed by the partners, such as architecture, safety and security mechanisms, their validation, modelling and code generation frameworks as well as base technology like the low-level software layers for SoC configuration and monitoring.

The demonstration platform (see Figure 2) consists of a powerful Simulation PC and the Turing Pi 2.5 cluster. The Turing Pi 2.5 features the possibility of switching hardware modules during run-time and supports common modules like the Raspberry Pi Compute Module 4, NVIDIA Jetson Orin Nano and NVIDIA Jetson Orin NX. The platform has four slots for hardware modules. The Simulation PC needs to run CARLA, an open source automotive simulator together with the cloud applications developed in this project. We chose Autoware as our autonomous driving stack, as it features a perception stack required as update target for UC1. The goal is to run as many components of Autoware next to the perception stack on different modules of the Turing Pi cluster. Due to the resource limitations on these

embedded devices, the remaining Autware components (i.e. with high performance requirements) will run on the Simulation PC.

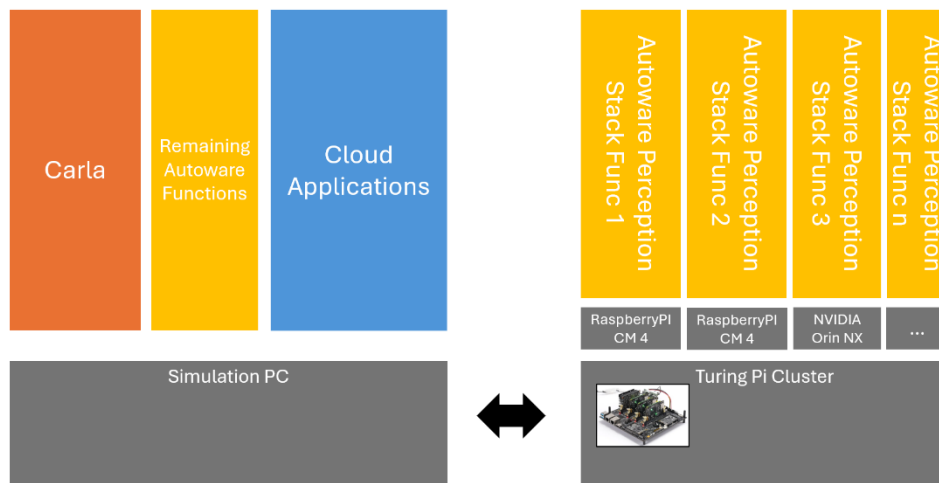


Figure 2: Platform Architecture of Research Demonstrator

3. UP2DATE4SDV Showcases

We introduced the term “showcase” to refer to a demonstration of one or more use-cases in a certain demonstration environment. The use-cases to be considered are summarized in Table 1.

Table 1: Use case overview

Use Case ID	Use Case Description
UC-1	Updating and upgrading HW/SW in the perception chain
UC-2	Maintaining reliability
UC-3	Considering hardware resource constraints
UC-4	Considering Cyber-Security
UC-5	Perception stack upgrade and mode switching
UC-6	Connected Automated Driving (CAD)

3.1. Showcase Overview and Time Plan

This section describes the showcases defined in the project. A showcase is a use case shown in a certain demonstration environment. Therefore, certain use cases can be demonstrated in different show cases. This is the case for the CCAM use cases UC5 and UC6, which will be shown on the "Virtual SDV/CCAM Demonstrator" and on the "SDV/CCAM Demonstrator (ADAS HiL + Real Controller)".

3.1.1 Overview of Showcases

Table 2: Overview of Showcases

Show Case (SC) ID	Demonstrator	Use Case or Individual Demonstration	Targeted Completion Date
SC-1	First version demonstrator for UC1	UC-1: Updating and upgrading HW/SW in the perception chain	M6 31 Dec 2025
SC-2	Research Demonstrator	UC-2: Maintaining reliability	M36 30 Jun 2028 M16 D5.2 (initial) M30 D5.4 (mid) M36 D5.5 (final)
		UC-3: Considering hardware resource constraints	M36 30 Jun 2028 M16 D5.2 (initial) M30 D5.4 (mid) M36 D5.5 (final)
		UC-4: Considering Cyber-Security	M36 30 Jun 2028 M16 D5.2 (initial) M30 D5.4 (mid) M36 D5.5 (final)
SC-3	Virtual SDV/CCAM Demonstrator	UC-5: Perception stack upgrade and mode switching UC-6: Connected Automated Driving (CAD)	M30
SC-4	SDV/CCAM Demonstrator (ADAS HiL + Real Controller)	UC-5: Perception stack upgrade and mode switching UC-6: Connected Automated Driving (CAD)	M36 M24 D5.3a (initial) M36 D5.3b (final)

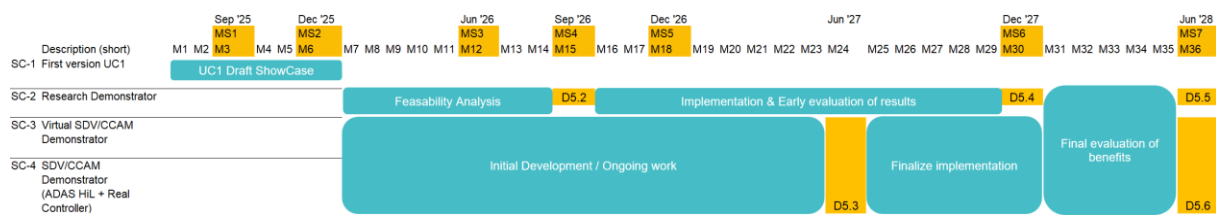


Figure 3: Gantt chart of showcases and related deliverables

Additionally, to the project milestones (MS) the showcases defined internal milestones, named MSSCX for (M)ile(S)tone (S)how(C)ase X, where the X stands for the showcase number.

3.1.2 Research Demonstrator Time plan

In Table 3 the milestones of the Research Demonstrator are described.

Table 3: Milestones of the Research Demonstrator

Date:	Milestone:	Description:
31.12.25	MSSC1-1	SHOWCASE-01 done
30.06.26	Project MS3	Expected input of early results from technical WPs HAL, HW, OAL, Orchestrator, realization framework, SASE update cycle, architecture definition, SASE mechanisms -> Early evaluation starts
30.09.26	MSSC2-1 (Project MS4)	Early evaluation finished
30.06.27	MSSC2-2	UC2, UC3, and UC4 are integrated
31.12.27	Project MS6	Expected input of final results from technical WPs
30.06.28	MSSC3-3 (Project MS7)	EOP -> Showcase done

3.1.3 Virtual CCAM Demonstrator Milestones

In Table 4 the milestones of the CCAM Demonstrator are described.

Table 4: Milestones of the CCAM Demonstrator

Date:	Milestone:	Description:
30.09.26	MSSC3-1 (Project MS4)	Basic virtual test environment is available
30.06.27	MSSC3-2	All use cases are working on the virtual demonstrator
31.12.27	MSSC3-3 (Project MS6)	Expected input of final results from technical WPs
30.06.28	MSSC3-4 (Project MS7)	Final evaluation of benefits available (reported together with physical SDV/CCAM Demonstrator)

3.1.4 SDV/CCAM Demonstrator Milestones

In Table 5 the milestones of the Research Demonstrator are described.

Table 5: Milestones of the SDV/CCAM Demonstrator

Date:	Milestone:	Description:
30.06.27	MSSC4-1	First full setup is available (ADAS HiL, physical controller, gateway controller); carry over of simulation environment from virtual demonstrator
31.12.27	MSSC4-2 (Project MS6)	Expected input of final results from technical WPs
30.06.28	MSSC4-3 (Project MS7)	Final evaluation of benefits available (reported together with Virtual CCAM Demonstrator)

3.2. Showcase-01: Use Case 1 demonstrated on Research Demonstrator

This is a "state-of-the art" demonstrator, based on existing technology from project partners, which will result in learning and understanding of the functionality required from the building-blocks created in later WPs. The focus lays on demonstrating the in-place update of SW. This will be shown on the initial research demonstrator which should be presented in M6 (after phase 1). Here, L4Re virtual machines will be the unit-under-update. The KPIs UC1.1 - UC1.4, will be shown on the Autoware perception chain.

For Use-Case 1 the following points are covered:

- In-place updating of a software module
- Replacement of a hardware module with a higher performance module
- A hardware module is added to the system
- The update process is managed by a cloud orchestrator

3.2.1 Involved partners

DLR, AVL, KK, ZSC, TTT

3.2.2 Architecture

In Figure 4 you can see the planned architecture for Showcase 1 based on the Research Demonstrator (see Figure 2). The simulation PC will host the Carla simulation and our cloud applications like the Update Registry and the Cloud Orchestrator. Besides that, it will host the remaining Autoware functions that the Turing Pi cluster is not able to run. On the cluster we are planning to use a mixture of several compute modules. On every module the L4Re Hypervisor will host a Linux VM and an Update Middleware. In the Linux VM our payload, the Autoware functions, will run with the ROS 2 middleware and Zenoh. The Update Middleware hosts the Platform Orchestrator, which is responsible for local deployment of new software functions.

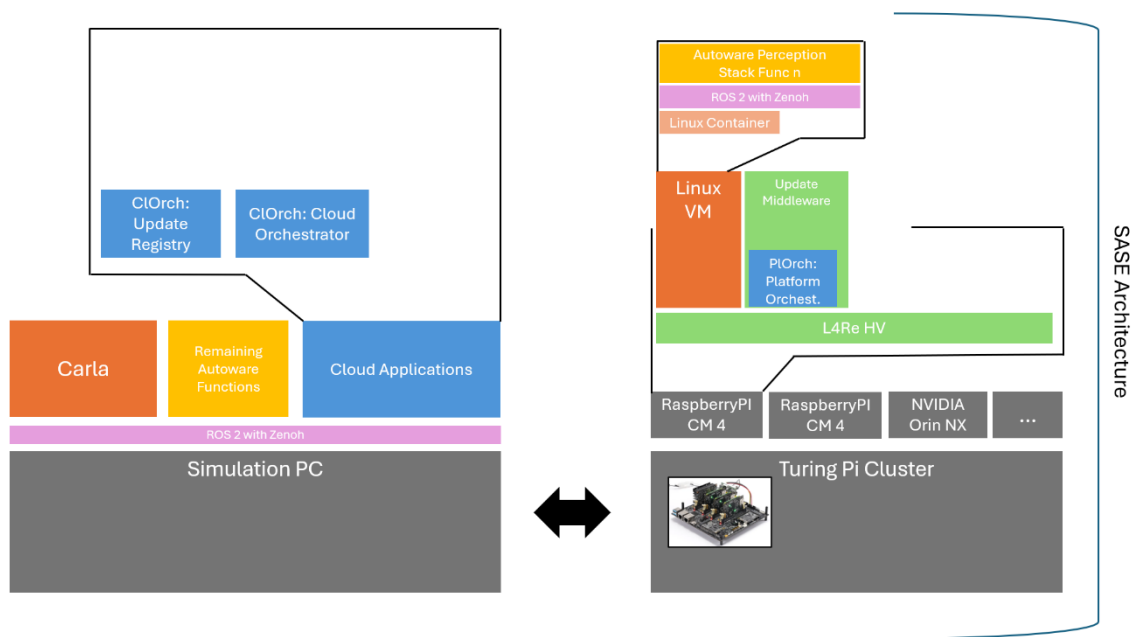


Figure 4: Architecture Overview of Showcase 1

3.2.3 Building Blocks

Table 6: SC-1 building blocks

Building Block	Related WP	Description
BB-DLR-1: PIOrch	WP3	Apply update package to target unit and adjust monitoring accordingly
BB-DLR-2: CIOrch	WP3	- Handles Update Package including resource monitors, system configuration - Specify specific non-functional properties of target system, consistency and compatibility checking, generation of target configuration and monitoring
BB-DLR-3: TuringPi	WP5	- Runs the automotive software stack - Provides HW platforms: Raspberry Pi CM4, NVIDIA Jetson Orin Nano/NX
BB-IKL-1 SASE Architecture	WP4	- Architecture - Definition of architectural components (orchestrator, update manager, online/offline monitor...)
BB-KK-1: HV	WP2	Secure & Safe isolation of workloads; individually restartable / updateable workloads (VMs) (L4Re)
Autware	WP5	Open Source Driving Stack
Carla	WP5	Open Source Driving Simulator
ROS 2	WP2	Communication Middleware
Zenoh	WP2	Communication protocol

3.3. Showcase-02: Use Case 2, 3 and 4 demonstrated on Research Demonstrator

This Showcase covers the Use Cases 2, 3 and 4. It builds on the results of Showcase 1 and will be therefore implemented on the Research Demonstrator Platform.

For Use Case 2 the following points are covered:

- The orchestrator platform is aware of the criticality of running functions
- The in-vehicle orchestrator is able to stop or relocate software modules according to the health status of the compute nodes
- The in-vehicle orchestrator detects network issues and relocates software modules and restores original configuration when the network issues are resolved

For Use Case 3 the following points are covered:

- The function deployment considers resource limitations of the compute nodes and only allows deployment if enough resources are available
 - We consider resource properties like the type of hardware, processing power, size of available memory and network properties
- The observation and observance of resource properties can be shown on more than one real hardware module
- The observation and observance of resource properties can be shown on virtual hardware module

For Use Case 4 the following points are covered:

- The security properties of the Use Case and the research demonstrator are analysed
- Model-based security checking is able to detect malicious code in update packages and software defects and deviations from the intended behavior

3.3.1 Involved partners

- DLR, BSC, AVL AVL-TR, CON, IKL, KK, ZSC, TAAT, ITI

3.3.2 Architecture

In Figure 5 you can see the planned architecture for Showcase 2 based on the Research Demonstrator (see Figure 2) and Showcase 1 (see Figure 4). The simulation PC will host the Carla simulation and our cloud applications, which are meant to support the software development process, the Update Registry and the Cloud Orchestrator. Besides that, it will host the remaining Autoware functions that the Turing Pi cluster is not able to run. On the cluster we are planning to use a mixture of several compute modules. On every module the L4Re Hypervisor will host a Linux VM and an Update Middleware. In the Linux VM our payload, the Autoware functions, will run with the ROS 2 middleware and Zenoh. It will also include an Intrusion Detection System and additional security mechanisms. The Update Middleware hosts the Platform Orchestrator, which is responsible for local deployment of new software functions. At different levels resource monitors will observe the overall system state.

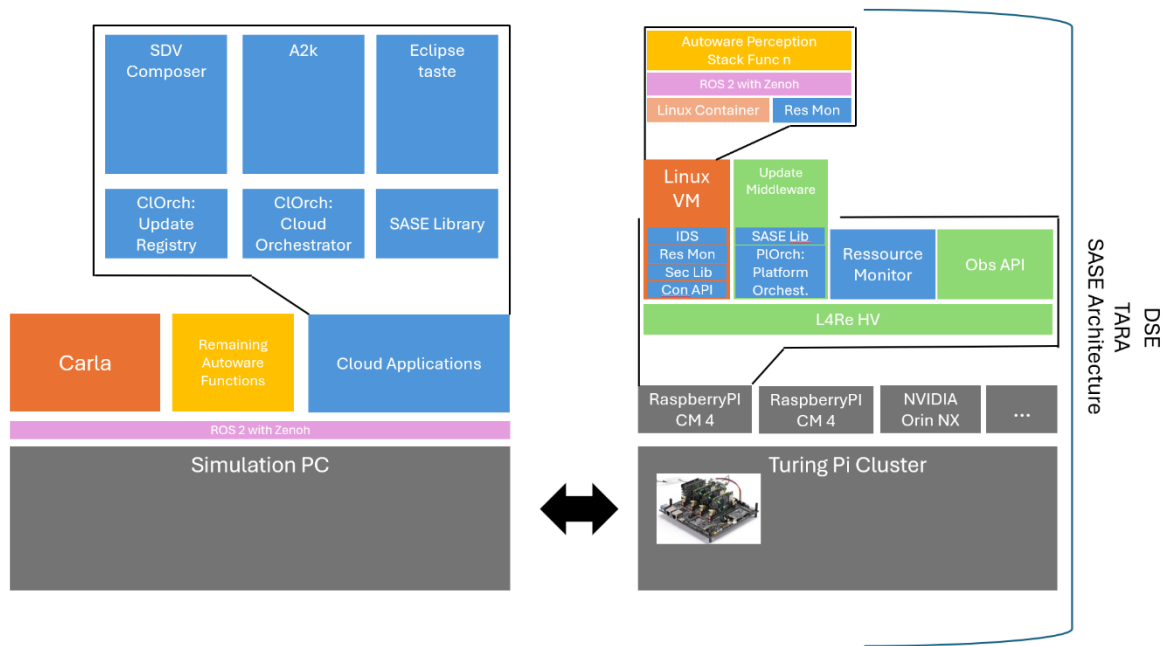


Figure 5: Architecture Overview of Showcase 2

3.3.3 Building Blocks

Table 7: SC-2 building blocks

Building Block	Related WPs	Description
BB-AVL-1: TARA	WP4	TARA of Architecture/SW Update Process
BB-BSC-1: Obs API	WP1	Low-level monitoring primitives
BB-BSC-2: Con API	WP1	- Expose (SASE) platform configuration options - Primitives to enforce HW configuration to support SASE-aware or SASE-enabling configuration
BB-DLR-1: PIOrch	WP3	Apply update package to target unit and adjust monitoring accordingly
BB-DLR-2: ClOrch	WP3	- Handles Update Package including resource monitors, system configuration - Identifies specific non-functional properties of target system, consistency and compatibility checking, generation of target configuration and monitoring
BB-DLR-3: TuringPi	WP5	- Runs the automotive software stack - Provides HW platforms: Raspberry Pi CM4, NVIDIA Jetson Orin Nano/NX
BB-IKL-1 SASE Architecture	WP4	- Architecture - Definition of architectural components (orchestrator, update manager, online/offline monitor...)
BB-IKL-2 SASE Mechanisms Library	WP4	Library of SASE mechanisms

BB-IKL-3 TASTE Based Toolchain	WP3	Software design and implementation tool
BB-ITI-1: a2k	WP3	System design and early evaluation tool
BB-ITI-2: DSE	WP3	Multi-objective optimisation of deployments in the edge-cloud continuum using DSE techniques with genetic algorithms
BB-ITI-3: IDS	WP4	Detection of intrusions in the network using AI
BB-KK-1: HV	WP2	Secure & Safe isolation of workloads; individually restartable / updateable workloads (VMs)
Autoware	WP5	Open Source Driving Stack
Carla	WP5	Open Source Driving Simulator
ROS 2	WP2	Communication Middleware
Zenoh	WP2	Communication protocol

3.4. Showcase-03: Use Cases 5 & 6 on Virtual CCAM Demonstrator

This showcase will demonstrate Use Case 5 and Use Case 6 in a virtual environment without physical control units.

For UC5 we want to demonstrate following points:

- Switch of sensor: After boot of the vehicle (ROS2 booted) a new sensor is configured (e.g. LIDAR added), and LIDAR data is transmitted from CARLA via the data bus.
 - A vehicle internal service recognizes the change in configuration and informs the Cloud orchestrator about the new HW Sensor available.
 - The cloud orchestration defines which SW should be deployed to the vehicle.
 - A LiDAR perception algorithm is deployed to the vehicle and started as soon as deployment is complete.
 - The vehicle control software can start using the additional information source (e.g., sensor fusion now also integrates the LIDAR sensor data or detections derived from the LIDAR data)
- Switch of location of the vehicle (either in-city vs offroad / highway) or geolocation (e.g., different road marking in US vs England (left side driving)).
 - Switch of location requires different vehicle control software features/parameterization (e.g., to avoid driving on the wrong lane).
 - Switch to optimized algorithms / parameters for algorithms depending on in-city or highway/off-road.
- Switch of weather condition (fog/rain; day/night)
 - The change of weather could either be detected by the vehicle, and the vehicle orchestrator could request a different set of algorithms or communicated by a script in CARLA which changes the weather to the cloud orchestrator.

For UC6 we want to demonstrate following points:

- The vehicle controller utilized cloud provided information, which is used in two demonstrations:

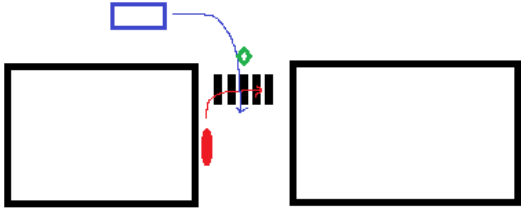
- Visibility extension (in cloud/fog) where the vehicle is informed about a obstacle (fallen tree/construction site) outside of the visual line of sight.
- Dense city driving situation approaching crossings with complicated setup (pedestrians, cyclists, cars). Information provided by a smart signal light provides positions of pedestrians, e-scooter drivers and cyclists around to vehicles approach the crossing.
- The V2V interaction is demonstrated by chaining high level autonomy vehicles (level 4) with low level autonomy vehicles (level 2) negotiated through a cloud service (anonymized vehicle position and destination location to give level 2 vehicles the possibility to chain with level 4 vehicles).
 - Requires the cloud service and vehicle controller functionality.

The goal is to design the virtual showcase SHOWCASE-03 in a way that artifacts (scenarios, SW, models, etc.) can be carried over to SHOWCASE-04 on the real HW and ADAS HiL without changing the content of the demonstration but only the environment.

3.4.1 Definition of Scenarios

Scenario	Notes
Add LiDAR (KPI UC 5.1)	• Add lidar sensor to a vehicle which previously only had camera sensors and show performance change in rain/fog/bad lighting conditions. Emulate a static upgrade of the vehicle in the workshop to allow for additional functions/ODD support. It requires additional hardware (LiDAR) and an according upgrade of SW (perception stack and algorithms). Presentation: • Run scenario before modification (without LiDAR) • Upgrade sensor (add the LiDAR) • After vehicle starts, the in-vehicle orchestration recognizes the new sensor and informs the cloud orchestrator about the current state of the vehicle. • Cloud orchestrator decides based on the data which SW is required for the given vehicle and triggers a software update. • Perception stack/algorithm is updated and activated in the vehicle. • Re-run scenario after modification (LiDAR sensor data is utilized by the autonomous driving functions) Scenario: Cyclist without lighting on the country road at night triggers the AEB (automatic emergency braking). Benefit: Report distance of detection, distance of cyclist after full stop.
Mode Switch due to ODD change (location & weather) (KPI UC 5.2 & 5.3)	Mode switch of the perception stack requires: Part 1 accounting for KPI UC 5.2 • Start driving in a country with metric system and right-side driving or unload the vehicle with the wrong configuration (e.g., transported via ship to UK) • Demonstrate a vehicle started for the first time in a changed location, for example UK and need to recognize that it now requires due to the location change an ODD change, due to requiring left-hand driving.

	○ Detection of ODD (Operational Design Domain) change. Impact of driving location: ▪ Text below road signs changes based on location (e.g., "only at school days (7:00-9:00)") - language (e.g., Chinese, Arabic, German umlaut, ...) ▪ Right side driving vs. left side driving ▪ Speed written in MPH vs KPH ○ Alternative options: ▪ Turkey requires stricter deceleration than Germany due to different driving behavior. ▪ E.g. for L2+ driving system, driver monitoring system is required in EU but not in US. So, this feature can be disabled/enabled via OTA. • Cloud orchestrator decides based on the data which SW is required for the given vehicle and triggers a software update. ○ Taking according to modifications (switch of parameters, activation of certain functions, language pack download, ...) • Verification → Vehicle drives on the right side of the road and considers street signs as MPH instead of KPH Part 2 accounting for KPI UC 5.3 • During simulation, weather conditions get worse (heavy rain and bad visibility) • Parameter to weight radar/lidar more than camera input is adjusted to account for the degradation of camera performance • Mode switching due to decreased performance in the vehicle (e.g., HW fault, sensor issue, etc.)
V2V communication with bad visibility (KPI UC 6.1, 6.2, 6.3 Scenario 1)	• Highway scenario where one vehicle detects obstacles at the highway (ghost-driver, truck tire parts, hole/damage in the road surface) • An in-vehicle application informs the cloud service about the obstacle(s) • The information is available to following cloud connected vehicles, which receive the information about the obstacle even if it is outside the line of sight and adopts the driving style (avoid lane, decrease speed, warning signals) • A non-connected vehicle is also driving the same scenario and will be used for comparison Verification/show of benefit → Compare scenario outcome of cloud connected vehicle vs. offline vehicle without cloud connection
V2I (Vehicle-to-Infrastructure) Smart Signal light (KPI UC 6.1, 6.2, 6.3 Scenario 2)	• On a crossing there is a smart signal light which transfers information about pedestrians, cyclists and scooter drivers (generally traffic participants) to the cloud. ○ Note: Ground truth of CARLA can be used. • Vehicle is approaching crossing and wants to do a turn. • Scooter with 25 km/h is approaching the crossing on the sidewalk - not in line of sight of the vehicle until very late. • Vehicle receives location information, trajectory and speed of traffic participants from the cloud and fusion with own perception data is done. • Vehicle should stop when scooter crosses the street

	→ Verification/show of benefit → Compare scenario outcome with cloud connected vehicle vs. offline vehicle without cloud connection.  (blue: ego vehicle; red: scooter ~25km/h on sidewalk, green: smart signal light)
V2V Chaining of vehicles (KPI UC 6.1, 6.2, 6.4)	- Assuming a highway situation - L4 automated vehicle drives and communicates planned route to the cloud. - L2 automated vehicle uses an App that allows chaining to a L4 vehicle (driver has to start chaining if he is lined up behind the L4 vehicle) - As soon as chaining is started the vehicle uses L2 capabilities to stay in the same lane as the L4 vehicle and also does lane changes accordingly - Combination of V2V communication and cloud communication - Related projects/papers: - Paper about platooning of vehicles (https://disi.unitn.it/~msegata/bibfiles/segata2017platooning.pdf) - Sumo extension for platooning (https://plex.car2x.org/)

3.4.2 Involved partners

All partners are involved in the CCAM demonstrators.

3.4.3 Architecture and Building Blocks

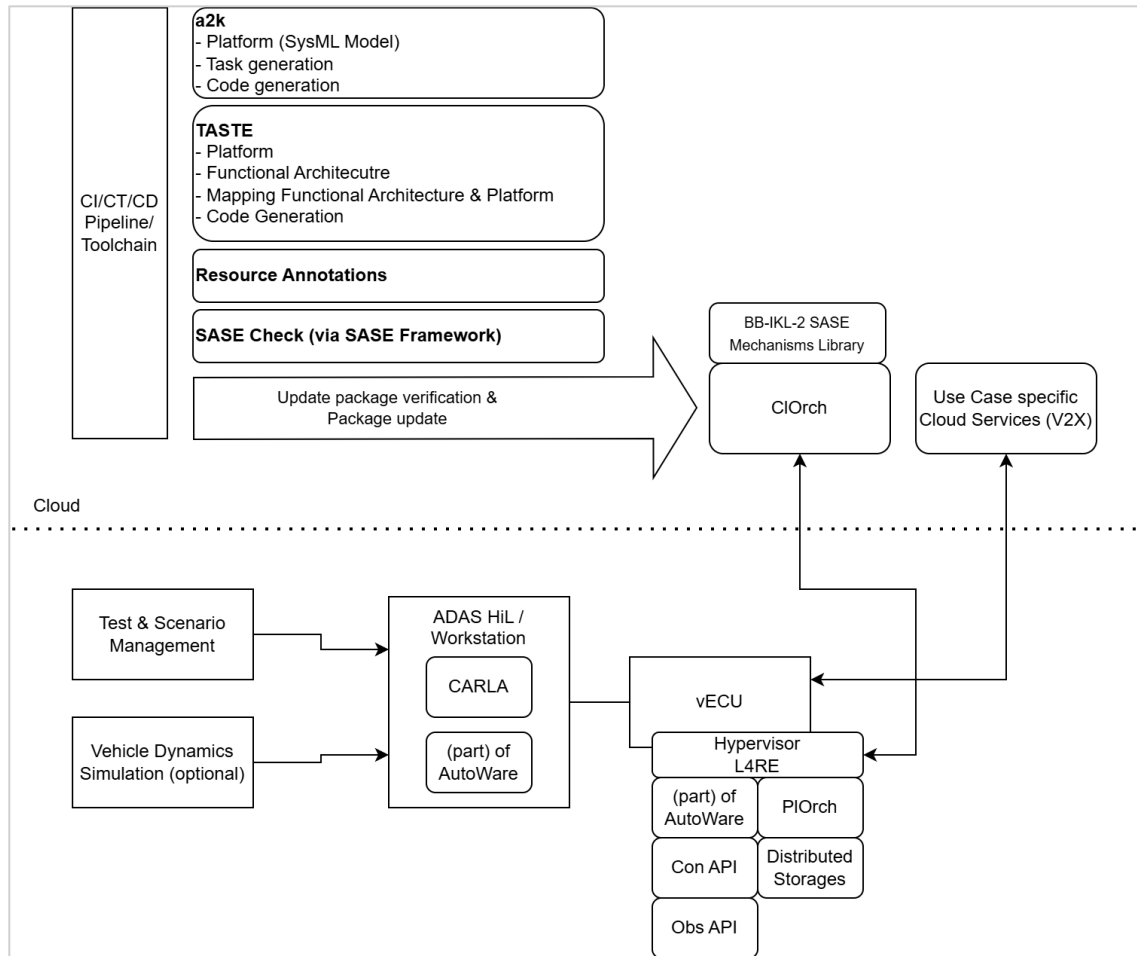


Figure 6: SC-3 Virtual CCAM Demonstrator - Draft of detailed architecture

3.5. Showcase-04: Use Cases 5 & 6 on final CCAM Demonstrator incl. ADAS HiL, physical Controller and Safe & Secure SW Updates

This showcase is the final CCAM showcase demonstrating UC-5 "Perception stack upgrade and mode switching" and UC-6 "Connected Automated Driving" (CAD). The use-case architecture is driven by the goal to utilize industry relevant toolchains and hardware, as well as demonstrate the developed building blocks to show their benefits on autonomous driving scenarios and connected autonomous driving scenarios. The STM32MP2 will manage the cloud connection and is able to run perception algorithms, while the Stellar micro-controller will serve additional control unit.

This showcase demonstrates how the UP2DATE4SDV architecture enables safe and secure software updates, hardware upgrades, and dynamic task reallocation on a mixed-criticality platform built from STM32 devices. The showcase is built around an STM32MP2 discovery or evaluation board connected via a Time-Sensitive Networking (TSN) interface to one or more Stellar micro-controllers. On the STM32MP2, a Microkernel runs as a hypervisor at Non-Secure Exception Level 2 (NS-EL2), hosting:

1. Best-effort OS partition: a virtual machine (VM) in which Linux (NS-EL1) and the container runtime (NS-EL0) execute, and

2. Critical partitions: a Real-Time OS (RTOS) inside a VM or Microkernel-native components (or unikernels) with real-time guarantees

Constituting a mixed-criticality system (MCS). OP-TEE runs in the Secure Exception Level 1 (S-EL1) and provides trusted services and secure firmware loading. ST is also working to enable a preliminary container-runtime support for the Stellar family of MCUs. In this architecture the microkernel acts purely as a hypervisor: it hosts the Linux guest in a virtual machine while critical workloads on the Stellar MCUs are handled via secure firmware updates rather than dynamically launched protection domains. Containers and orchestrated services always run inside the Linux guest (normal world), not as microkernel partitions.

By integrating the hardware abstraction layer (HAL), reconfigurable container middleware and service-oriented communication (OAL), the orchestration & reconfiguration plane (ORP), and the safety & security (SASE) update mechanisms, the platform supports in-place software updates, hardware swaps and additions, and reliability- and resource-aware deployment.

ContinuumX platform enables cloud orchestration and digital twin integration. The Phase Change Memory (PCM) Mapper enables coordinated monitoring and control of zonal gateways, and sensors. By leveraging digital-twin driven orchestration, Neotera enables seamless over-the-air updates, workload migration, and resource-aware task reallocation between edge and cloud, thus highlighting the scalability and robustness of STM32-based SDV platforms.

3.5.1 Involved partners

- DLR, BSC, AVL AVL-TR, CON, IKL, KK, NEO, STM, ZSC, TAAT, ITI

3.5.2 Architecture and Building Blocks

Project Building Blocks

Table 8: SC-4 list of project building blocks

Building Block	Related WP	Comment
Safety & Security (SASE) Update Manager	WP4	Defines the safe update process, reference architecture and security mechanisms. On STM32MP2 the secure world leverages OP-TEE together with the Microkernel hypervisor to run trusted services and verify signed firmware before it is deployed to the Stellar MCU. These mechanisms ensure that updates are checked before deployment, that the system can detect malicious code or deviations from intended behaviour, and that failures can be rolled back safely.
Reconfigurable Container Middleware & Service-Oriented Communication (OAL)	WP2	Implements a container runtime on STM32MP2 inside the Linux guest and a service-oriented communication plane based on Zenoh. The OAL manages container start, stop and resource allocation within the Linux environment and monitors resource requirements via the HAL. Containers are not deployed as separate Microkernel protection domain.

Orchestration & Reconfiguration Plane (ORP)	WP3	ContinuumX provides cloud-edge orchestration and dynamic task reallocation for containerized workloads. It integrates digital- twin models and uses monitoring data from HAL to make resource-aware scheduling decisions. In the mixed- criticality system, ContinuumX orchestrates containers in the Linux guest and triggers secure firmware loads on Stellar MCUs via the PCM Mapper. It does not launch new Microkernel protection domains. The framework supports over-the-air updates, container migration and rollback.
Hardware Abstraction Layer (HAL)	WP1	Provides deterministic real-time behaviour and monitoring for the STM32MP2 and Stellar MCU devices. On the STM32MP2, the Microkernel (seL4) runs as a hypervisor at NS-EL2 on top of Arm Trusted Firmware, enabling the realization of a mixed-criticality system (MCS) on top. This arrangement isolates secure and non-secure workloads alongside time-critical and best-effort workloads. The HAL exposes interfaces to interact with the underlying PCM technology to control the Stellar MCU. On the Stellar MCU, the Microkernel (L4re) runs as a hypervisor for the use case applications (Tire speed, weather detection), the PCM update manager and communication drivers
Demonstration & Evaluation	WP5	Implements the research demonstrator and evaluates the technical KPIs defined in the project's WP5 specification. Ensures that the showcase meets the defined metrics and provides feedback into WP1-WP4 deliverables.

Non-project Building-Blocks / Artifacts

Table 9: SC-4 list of non-project building blocks

Dependency	Status	Comment	Provided by/Owner
STM32MP257 Discovery Kit / Evaluation Board	Available	Provides edge Linux environment for running containerised perception and control applications.	ST
Stellar MCU Evaluation Board	Available	Provides target for PCM firmware update demonstration.	ST
ContinuumX Cloud-Edge Basic Platform	Available	Operational in Neotera's lab and tested over AMD64 architecture.	NEO
Linux Kernel, OP-TEE remoteproc	Available	Linux Kernel for STM32MP2. OP-TEE and remoteproc framework to securely load firmware on the Stellar MCU. Used to manage security-critical tasks and provide isolation.	ST
Microkernel	Porting & deployment planned	seL4 provides strong isolation and real-time scheduling for the STM32MP257.	ST

Cloud Infrastructure (Kubernetes)	Available	Hosts the cloud portion of the ContinuumX orchestrator and digital-twin services. Allows partners to access the demonstrator remotely.	NEO
ADAS HIL	Available	Provides high performance compute and real-time functionality.	AVL
L4RE	Available	Open-source Hypervisor	KK
a2k		System Modelling	ITI
TASTE			-
CARLA	Available	Open-source environment simulation.	-
Autoware	Available	Open-source perception stack.	-

Based on the initial architecture draft from the project proposal, shown in Figure 7, the derived architecture proposal of the final CCAM showcase based on the previously identified and described building blocks is shown in Figure 8. Same as with the requirements, the presented architecture is a current snapshot and can still be subject to changes during elicitation of work package requirements, or due to incompatibilities found in the implementation phase of the show case. The partners agreed on the shown architecture, but there was not enough time yet to check the full compatibility of all interfaces or project and non-project artifacts.

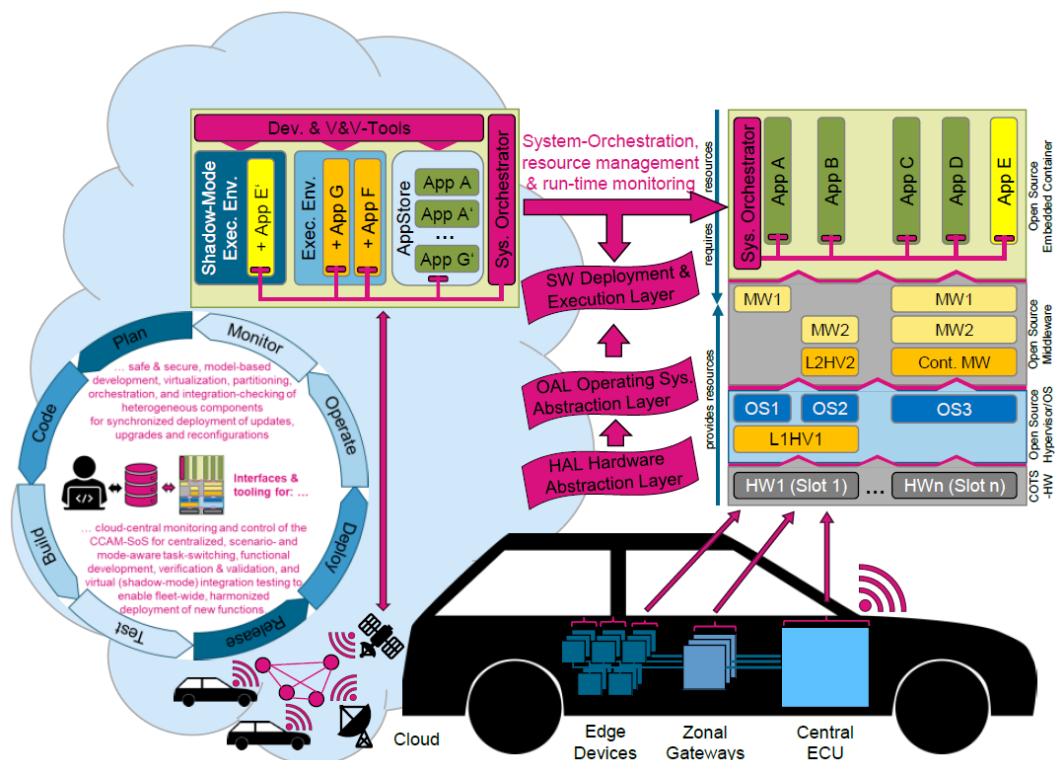


Figure 7: Abstract Architecture (from project proposal)

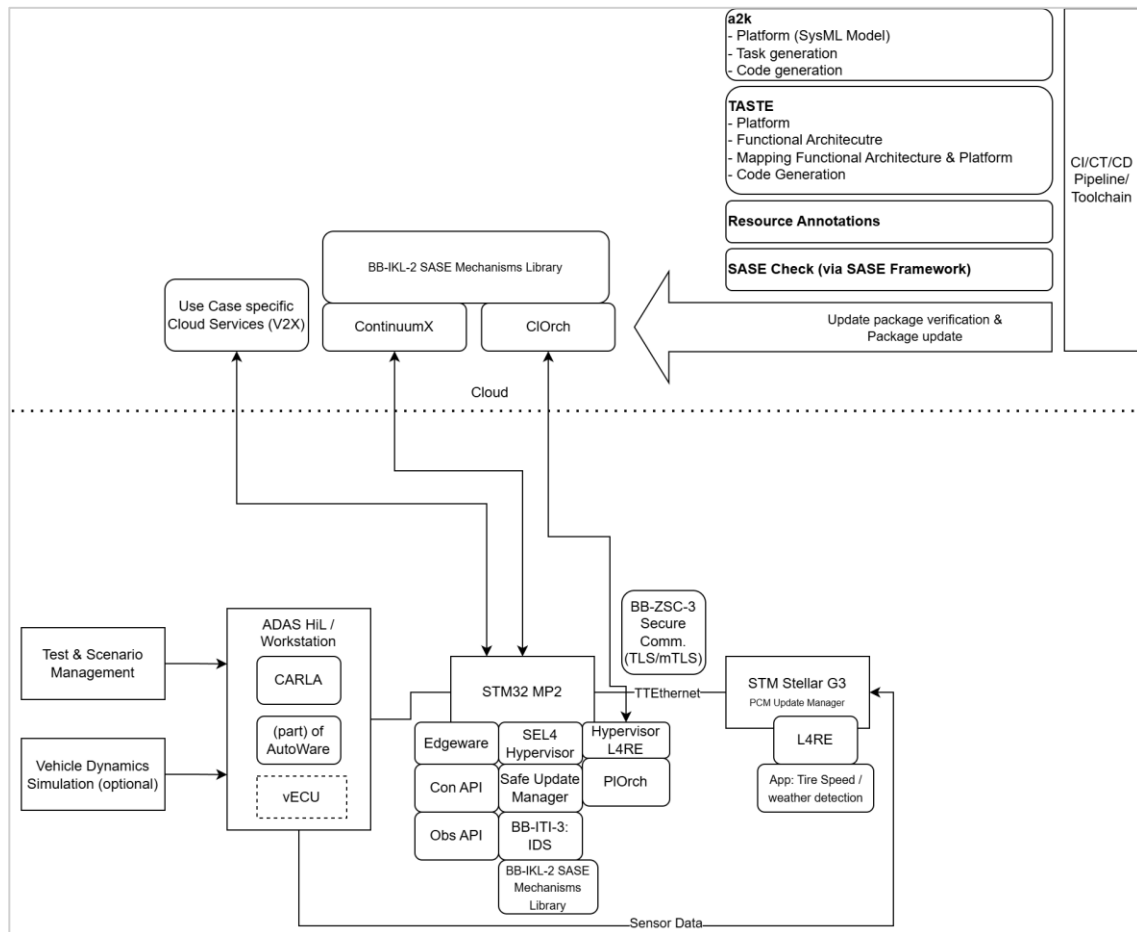


Figure 8: SC-4 Draft of detailed architecture

The showcase comprises a central ECU built on an STM32MP2 discovery/evaluation board connected via TSN to one or more Zonal Gateways built on Stellar MCUs, and a cloud control plane. On the STM32MP2 we have a mixed criticality system (MCS), where a Microkernel runs as a hypervisor at NS-EL2 and hosts a virtual machine running Linux and the container runtime, (NS-EL1/NS-EL0). and a Real-Time OS (RTOS) (NS-EL1) inside a VM or Microkernel-native components (NS-EL0) with real-time guarantees, in the rich execution environment. OP-TEE operates in the secure world (S-EL1) and manages calls to the secure firmware loader via the remoteproc framework. The cloud hosts the ContinuumX orchestrator, which coordinates software updates, container deployment and dynamic task reallocation, and uses digital-twin models for edge devices and sensors on both the edge and cloud side. Monitoring data (CPU load, memory usage, network metrics) collected by the HAL is sent via the service-oriented communication plane to the orchestrator. The SASE update manager runs, at least partially, within the secure world and performs authentication and security checks before any update is applied. Sensor events (e.g., switching from LiDAR to camera) are detected and trigger reconfiguration of perception software. The architecture supports offline operation: if network connectivity drops, the orchestrator continues to execute services locally on the STM32MP2 and resynchronizes once connectivity is restored.

4. KPIs & Metrics

This section describes the KPIs and metrics defined for the project. Each KPI is covered by at least one show case. For this deliverable we show the use case specific KPIs and their metrics in Table 10, and the related proposal objective KPIs and their metrics in Table 11.

Table 10: Overview of use-case specific KPIs and metrics

KPI ID	KPI Description	Metric(s)	Covered by
KPI UC1.1	In-place update of SW is demonstrated	This KPI will be shown on the Research Demonstrator Metrics: - A software module could be a module on the function (e.g. ROS 2 nodes), OCI container, hypervisor partition level. - The process of updating is defined as the deactivation of the module (e.g. by migration the functionality to another software module or by making sure it's not in use), replacing it with a different module, and reactivating the new module. - The behavior of the system is the same afterwards.	SC-1
KPI UC1.2	Replacement of HW module is demonstrated by replacing an older module with a newer higher performance module.	This KPI will be shown on the Research Demonstrator If feasible, we will also show this KPI using vECUs. Otherwise, this KPI will be shown on the research demonstrator on Turing Pi Metrics: - A hardware module is defined as a slot-in card for the TuringPi. The used interface is defined by NVIDIA NANO/NX. - The upgrade is defined as a deactivation of a module, including software migration, physical removal of the current module, physical replacement with a better module and activation of the better module and reallocation of the previous software. Better means a more powerful module. - The behavior of the system is the same afterwards.	SC-1
KPI UC1.3	An additional HW unit is added to the vehicle and subsequently utilized by the system.	This KPI will be shown on the Research Demonstrator. Metrics:	SC-1

	The SW update/upgrade is triggered and orchestrated by the cloud control-plane	- A hardware module is defined as a slot-in card for the TuringPI. The used interface is defined by NVIDIA NANO/NX. - The process of adding is defined as physically adding a hardware module to an empty slot in the TuringPI. A software module is migrated to the new hardware module. - The process is orchestrated by a cloud control-plane. - The behaviour of the system is the same afterwards.	
KPI UC1.4	A cloud control plane orchestrates the update process.	This KPI will be shown on the Research Demonstrator. Metric - A software module could be a module on the function (e.g. ROS 2 nodes), OCI container, hypervisor partition level. - The process of updating is defined as the deactivation of the module (e.g. by migration to the functionality to another software module or by making sure it's not in use), replacing it with a different module and reactivating the new module. - The process is orchestrated by a cloud control-plane. - The behaviour of the system is the same afterwards.	SC-1
KPI UC2.1	Criticality of executed services and functions is available to the target orchestrator.	This KPI will be presented either on the Research Demonstrator or on the CCAM Demonstrator Metric - A software service or function is a software module (e.g. ROS 2 nodes) - The criticality is defined as the Automotive-Safety-Integrity-Level (ASIL) assigned to the function or component according to ISO 26262 - The target orchestrator is able to get this information for every software module from the Cloud Registry as part of meta data	SC-2
KPI UC2.2	In-vehicle target orchestrator stops services or re-locates them according to the health status of	This KPI will be shown on the Research Demonstrator. Metric	SC-2

	the compute nodes and reliably ensures continuous execution of safety critical functions. <i>Comment: From O2, the federation of containers and dependable execution environments must keep track which services are intended to run and further on which compute nodes (locally or in the cloud) execution takes place. Further O1 needs to provide an understanding of the cloud connection performance regarding bandwidth and latency.</i>	• The function is defined by its input and output. • The processing time is defined as the time the function takes to produce the output. • During nominal use, the typical processing time is upheld • During changes (e.g. re-location etc.), the time of functional downtime is minimal (maximal downtime timeframe TBD after function selection) • The information required to steer orchestration decisions is available.	
KPI UC2.3	In-vehicle target orchestrator reliably detects network disconnections, low bandwidth or worse than required network latency, and therefore starts required services/containers on local compute nodes to ensure continuous functionality.	This KPI will be shown on the Research Demonstrator. Metrics: • The function is defined by its input and output. • The processing time is defined as the time the function takes to produce the output. • During nominal use, the typical processing time is upheld • During changes (e.g. re-location etc.), the time of functional downtime is minimal (maximal downtime timeframe TBD after function selection) • The information required to steer orchestration decisions is available.	SC-2
KPI UC2.4	If network connection quality returns to acceptable performance, the desired distribution of services between local and cloud compute nodes should be re-established. Connected services should be continued.	This KPI will be shown on the Research Demonstrator. Metrics: • The function is defined by its input and output. • The processing time is defined as the time the function takes to produce the output. • During nominal use, the typical processing time is upheld • During changes (e.g. re-location etc.), the time of functional downtime is minimal (maximal downtime timeframe TBD after function selection) • The information required to steer orchestration decisions is available.	SC-2

KPI UC3.1	Function deployment considers resource limitations of the connected vehicle and consequently only allows deployment and execution of new services if it is safely possible.	This KPI will be shown on the Research Demonstrator. Metrics: • A software module could be a module on the function (e.g. ROS 2 nodes), OCI container, hypervisor partition level. • A hardware module is defined as a slot-in card for the TuringPI. The used interface is defined by NVIDIA NANO/NX. • The process of deployment is defined as the placing of a software module on a hardware module. • The process is orchestrated by a cloud control-plane. • If the resources, namely CPU time, of the hardware module is not sufficient for the software module, the deployment is not executed.	SC-2
KPI UC3.2	The demonstration is shown on more than one real hardware module (e.g., NXP 32G, QUALCOMM, Nvidia).	This KPI will be shown on the Research Demonstrator. Metrics: • A software module could be a module on the function (e.g. ROS 2 nodes), OCI container, hypervisor partition level. • A hardware module is defined as a slot-in card for the TuringPI. The used interface is defined by NVIDIA NANO/NX. • The process of deployment is defined as the placing of a software module on a hardware module. • The process is orchestrated by a cloud control-plane. • If the resources, namely CPU time, of the hardware module is not sufficient for the software module, the deployment is not executed. • This is shown on more than one hardware module.	SC-2
KPI UC3.3	The resource constraints considered include hardware (sensors, computer modules), AI accelerators, general properties of the computer modules like processing power, size of available	This KPI will be shown on the Research Demonstrator. Metrics: • A software module could be a module on the function (e.g. ROS 2 nodes), OCI container, hypervisor partition level.	SC-2

	memory and network properties like bandwidth and latency.	- A hardware module is defined as a slot-in card for the TuringPI. The used interface is defined by NVIDIA NANO/NX. - The process of deployment is defined as the placing of a software module on a hardware module. - The process is orchestrated by a cloud control-plane. - If the resources of the hardware module is not sufficient for the software module, the deployment is not executed. - The considered resources are the type of hardware and processing power, size of available memory and network properties like bandwidth and latency. - This is shown on more than one hardware module.	
KPI UC3.4	Validation of consideration of resource constraints can be shown in a virtual environment, for example on Continentals CAEdge (presuming that a virtual ECU is available that models the same performance and resource capabilities than the real hardware).	This KPI will be shown on the Research Demonstrator. Metrics: - We can show that specifications of virtual and real hardware are identical. - The observed values of the virtual device are identical to the observed values on the real hardware. - Constraints based on the values observed are identically upheld.	SC-2
KPI UC4.1	Security properties of the use-case / demonstrator, and the update system are analysed.	Research Demonstrator & CCAM Demonstrator Metrics: - The Safety Integrity Level and Security Level of the demonstrator is defined - The corresponding Security properties are specified based on the SL - These security properties are integrated and checked in the demonstrator. - If an update is requested, it will only be executed if the security properties check phase is correct. Otherwise, a rollback to the previous version will be done.	SC-2 SC-5

KPI UC4.2	Model based security checking is implemented which is able to detect malicious code in update packages.	This KPI will be shown on the Research Demonstrator. Metrics: • The Safety Integrity Level and Security Level of the demonstrator is defined • The corresponding Security properties are specified based on the SL • Before every update deployment, using a model of the system, the software update is checked to be compliant with the security properties. • It must be possible to detect malicious code (code not compliant with the security properties) • The update will not be deployed in that case. Only authorized and uncorrupted updates will be deployed	SC-2
KPI UC4.3	Model based security checking is implemented which is able to detect software defects and deviations from the intended system behavior according to the requirements and system security properties analysis.	This KPI will be shown on the Research Demonstrator. Metrics: • Before every update deployment, using a model of the system, the software update is checked through static analysis and/or model checking tools. • It must be possible to detect software defects and deviations in the code in terms of safety and security (e.g.: using MISRA-C, SAST...) • If deviations or defects are detected, then the update will not be deployed.	SC-2
KPI UC5.1	The switch of sensors can be detected and can be considered by the cloud orchestrator to provide the according software to the vehicle.	This KPI will be shown on the CCAM Demonstrator. Metrics: • The vehicle software (in-vehicle orchestrator) provides an interface to get information about hardware changes during start-up. • A vehicle software component (either vehicle controller, perception stack or gateway software), recognizes a change of hardware and uses before mentioned interface to inform the in-vehicle orchestrator. • The in-vehicle orchestrator informs the cloud orchestrator about the change so that subsequent actions can be taken.	SC-3 SC-4

KPI UC5.2	Depending on the position (environment) of the vehicle, the employed perception software algorithms can be switched/changed on-the-fly.	This KPI will be shown on the CCAM Demonstrator. Metrics: • The cloud control plane/cloud orchestrator is aware of the vehicle position and current ODD (weather condition, exceptional events like unexpected obstacles like accidents). • The vehicle transmits its position to the relevant cloud services. Ideally, cloud services can subscribe to certain types of vehicle information. • The cloud service / orchestrator understands the current software configuration of the vehicle incl. software used, versions of software deployed, and tasks executed. • The cloud service / orchestrator can trigger a change in software configuration according to the current requirements (e.g., left-hand vs. right-hand driving location, change of ODD, ...). • The boundary conditions regarding hardware resources (memory, CPU, available sensors) are considered. • The in-vehicle orchestrator executes the software configuration change and activates the required tasks.	SC-3 SC-4
KPI UC5.3	Depending on the weather condition and lighting, the employed perception software algorithms can be switched/changed on the fly.	This KPI will be shown on the CCAM Demonstrator. Metrics: • The cloud control plane/cloud orchestrator is aware of the vehicle position and current ODD (weather condition, exceptional events like unexpected obstacles like accidents). • The vehicle transmits its position to the relevant cloud services. Ideally, cloud services can subscribe to certain types of vehicle information. • The cloud service / orchestrator understands the current software configuration of the vehicle incl. software used, versions of software deployed, and tasks executed. • The cloud service / orchestrator can trigger a change in software configuration according to the current requirements (e.g., left-hand vs. right-hand driving location, change of ODD, ...).	SC-3 SC-4

		- The boundary conditions regarding hardware resources (memory, CPU, available sensors) are considered. - The in-vehicle orchestrator executes the software configuration change and activates the required tasks.	
KPI UC6.1	A vehicle controller is implemented which utilizes cloud provided information about the scene to enhance the vehicles decision making and safety.	This KPI will be shown on the CCAM Demonstrator. Metrics: - The vehicle utilizes an application that retrieves environment information from a cloud service (e.g., traffic jams, unexpected obstacles, construction sites, traffic participant position information for smart cities, ...). - A cloud service exists which provides the required information for the use-case scenarios.	SC-3 SC-4
KPI UC6.2	Supplemented information is fused with perception system output.	This KPI will be shown on the CCAM Demonstrator. Metrics: The perception stack of the vehicle can fuse information about the environment received or fetched from cloud services (e.g., traffic jams, unexpected obstacles, construction sides, traffic participant position information for smart cities, ...) with the own perception stacks detections.	SC-3 SC-4
KPI UC6.3	At least two different scenarios demonstrate the benefit of technology: - Scenario 1: Highway driving situation with road and/or traffic issues outside the vehicle's perception systems range in low visibility scenarios (fog, night), is demonstrated. - Scenario 2: Dense city driving situation approaching crossings with complicated setup (pedestrians, cyclists, cars). Information provided by a smart signal light provides positions of pedestrians and	This KPI will be shown on the CCAM Demonstrator. Metrics: - Scenario 1 and Scenario 2 are executed, and the benefit is documented by comparing the impact of the software update/upgrades and V2X connectivity on the safety of the driver. - The reported metric should be if a crash/accident occurred, and if not the distance of the emergency brake or evasive maneuver start before the obstacle to understand the increase of safety margins.	SC-3 SC-4

	cyclists around to vehicles approach the crossing.		
KPI UC6.4	Demonstrate chaining of high capability autonomous vehicles with lower-level autonomous vehicles (e.g. level 4+ with level 2) on the highway through cloud to vehicle connection and vehicle to vehicle connection. - Level 2 assisted driving vehicles can chain with vehicle of higher autonomous driving functionalities (e.g., follow the vehicle with higher capabilities). - Vehicle external cloud-connected service provides anonymized vehicles and their position and navigation system goals, therefore informs vehicles of lower levels about possibilities to “chain” or “follow” those vehicles for a common part of their path.	This KPI will be shown on the CCAM Demonstrator. Metrics: - At least one vehicle is chained with a vehicle that supports a higher level of autonomous driving. - A vehicle application exists which provides the vehicle location, level of autonomy supported, as well as the planned route to the cloud service. - A vehicle application can request potential chaining targets for the current destination. It will receive a list of best candidates for matching depending on current position and navigation target. - The vehicle driver or autonomous driving function needs to reach a rendezvous point at a certain time to successfully chain the vehicles. - The chaining functionality is executed accordingly until the driver manually deviates from following the vehicle or disables the feature.	SC-3 SC-4

The following table shows the KPIs and metrics of the project objectives which will be addressed in the show cases. Please note that the deliverable D5.1 focuses on the CCAM use cases and the SDV update requirements described in WP5 and therefore naturally not all objective KPIs will be already addressed here. Remaining objectives unrelated to the use cases will be covered by later deliverables.

Table 11: Overview of proposal objectives KPIs and metrics

KPI ID	KPI Description	Metric(s)	Covered by
KPI O1.1	Definition and provision of a time and resource composable SoM hardware platform	- The HW platform supports at least 2 different SoM based SoCs (e.g. NVIDIA and STM Stellar) - The physical communication between the SoMs is organized through Gbit class Time Sensitive Network (TSN)	SC-1

KPI O1.2	Configurations for monitoring, and SASE operation and updates concerning local node operation and internode communication for at least two platforms - Support common monitoring and configuration services for at least two different HW platforms (e.g. NVIDIA Jetson, STM Stellar)	- Support common monitoring and configuration services for at least two different HW platforms (e.g. NVIDIA Jetson, STM Stellar) - Configurations provided allow measuring at least execution cycles and memory bandwidth usage for each platform - At least one node configuration per platform limits timing interference below given bounds, with those bounds being at least 10% execution interference	SC-2
KPI O2.1	Interference-free execution containers and container federations to express dependable execution environments	- Proof-of-concept implementation of container execution on the STM32 MP2 can execute at least two different containers independently. - Proof-of-concept implementation of a microkernel container runtime demonstrated in a relevant environment; feedback obtained from at least two independent testers. - Conduct tests to ensure applications inside containers can't exceed allocated computational/memory resources	SC-4
KPI O2.2	Definition and provision of a service-oriented reconfigurable communication plane across nodes and into the cloud	- Demonstration of a communication protocol independent service-oriented intra-node, node-to-cloud and cloud-to-node communication technology (e.g. Eclipse Zenoh) - Show that the time required to migrate a session from one router to another is in the order of hundreds of ms.	SC-2
KPI O3.1	Process definitions for software-updates (in place) and hardware-upgrades (in place), task-reallocation (vehicle-to-cloud/cloud-to-vehicle) and system reconfiguration (scenario-dependent switching between configurations), as well as fleet orchestration (synchronization of updates/upgrades) aligned with O4	- At least one use-case (scenario) for each process defined - Reference model in a process description language for each process available	SC-2
KPI O3.2	Implementation of the orchestration manager plane	Proof-of-concept implementation of the cloud-based orchestration and deployment manager for at least three of the four processes available	SC-2

		• Proof-of-concept implementation of the in-vehicle orchestration and deployment manager for at least three of the four processes available (based on HAL of O1 and OAL of O2) • Proof-of-concept orchestrators for 2 different targets (e.g. NVIDIA Jetson, ARM-R based, ARM-A based) available	
KPI O3.3	Integrate Virtual Integration Testing (VIT) of O4 to verify update-compatibility	• VIT for update compatibility based on timing, HW/SW configuration (incl. security-needs), resource capabilities (computation, memory and bandwidth) available • VIT for update compatibility is automatically executed in the orchestration manager plane when the system is updated, upgraded or reconfigured	SC-2
KPI O4.2	Provide harmonized and simplified development, integration and validation processes and framework for SDV applications	• Generate code automatically for 2+ SDV applications matching the UP2DATE4SDV HW+SW architecture	SC-4
KPI O4.4	Provide harmonized and simplified development, integration and validation processes and framework for SDV applications	• Virtualization of at least 8 functionalities of UP2DATE4SDV Middleware on 2 HWs	SC-4
KPI O4.6	Provide harmonized and simplified development, integration and validation processes and framework for SDV applications	• At least 2 applications successfully virtualized	SC-4
KPI O4.8	Provide harmonized and simplified development, integration and validation processes and framework for SDV applications	• Increase the automatization of the V&V procedure by 30%.	SC-4
KPI O4.9	Provide harmonized and simplified development, integration and validation processes and framework for SDV applications	• At least a test suite of 50 test cases focusing on SASE evaluation of UP2DATE4SDV update/upgrade/reconfiguration framework.	SC-4
KPI O4.10	Provide harmonized and simplified development, integration and validation processes and framework for SDV applications	• 95% of coverage of upgrade/update/reconfiguration functionalities by the test-suite generated.	SC-4
KPI O5.2	Assess joint Safety and Security (SASE) of all elements from O1, O2 and O3	• Positive assessment of the aimed ASIL (e.g., ASIL C) and SL (e.g., SL-2) in one of the UC applying UP2DATE4SDV	SC-2 SC-4

		update/upgrade/reconfiguration procedure, components and reference architecture.	
KPI O5.3	Assess joint Safety and Security (SASE) of all elements from O1, O2 and O3	Reaching the aimed %95 diagnostic coverage regardless of the HW used (cope at least two COTS).	SC-2 SC-4
KPI O5.4	Assess joint Safety and Security (SASE) of all elements from O1, O2 and O3	At least 5 safety and security mechanisms for AI-driven SDV applications. At least 10 safety and security preventive evaluators and countermeasures spread in the UP2DATE4SDV zonal architecture.	SC-2 SC-4
KPI O6.1	Demonstration of UP2DATE4SDV solution on several CCAM relevant Use Cases	Ratio of use cases system requirements successfully implemented according to T5.1 metrics. Global performance indicators defined in the project proposal Part B section 1.2.1.	SC-3 SC-4

5. SDV Applications and Updates Requirements

For T5.2 we identified and collected requirements for the updateability and upgradability of the Software-Defined Vehicle (SDV) towards safety, security, abstraction, deployment, verification & validation, and DevOps.

These requirements for the implementation will ensure compatibility with hardware support in today's high-performance System-on-a-Chip (SoCs). Furthermore, system verification and validation requirements are defined together with testing approaches on different levels.

Please find the collection of requirements in the Appendix A.

6. Summary and Outlook

This deliverable D5.1 provides a comprehensive foundation for the UP2DATE4SDV project's demonstrator strategy, showcasing the initial integration of modular update, upgrade, and reconfiguration capabilities for Software-Defined Vehicles (SDVs). Through the definition of use cases, demonstrator architectures, and associated KPIs, the document outlines how the project will validate its core technological innovations across virtual and physical environments. The demonstrators are designed to reflect real-world challenges in CCAM scenarios, enabling iterative refinement of requirements and metrics. The collaborative approach across work packages ensures alignment between hardware abstraction, middleware orchestration, and safety/security mechanisms. Deliverable D5.1 captures the current state of planning and integration at project month three (M3) and sets the stage for future deliverables that will build upon these definitions. In conclusion, D5.1 marks a critical milestone in translating the UP2DATE4SDV vision into actionable and measurable implementations, laying the groundwork for safe, secure, and dynamic SDV ecosystems.

During the initial phase of WP5 from M1 to M3, weekly meetings with all partners were conducted. Starting from October 2025, we will replace the WP5 alignment meetings with a monthly meeting series but start weekly meetings for the aligning on the described showcases, starting with SC-1.

The next deliverables regarding WP5 are:

- D5.2 - Basic Technology Implementation and Evaluation [Initial] - considering the Research Demonstrator (M15), and
- D5.3 - Demonstrator Technology Implementation and Evaluation [Initial] - considering the CCAM Demonstrator (M24), and
- D5.4 - Basic Technology Implementation and Evaluation [Intermediate] - considering the Research Demonstrator (M30), and
- D5.5 - Basic Technology Implementation and Evaluation [Final] - considering the Research Demonstrator (M36), and
- D5.6 - Demonstrator Technology Implementation and Evaluation [Final] - considering the CCAM Demonstrator (M36)



Funded by
the European Union

D5.1. Demonstrator Definitions, Metrics, and Plan

Annex 1: WP5 Requirements Definition

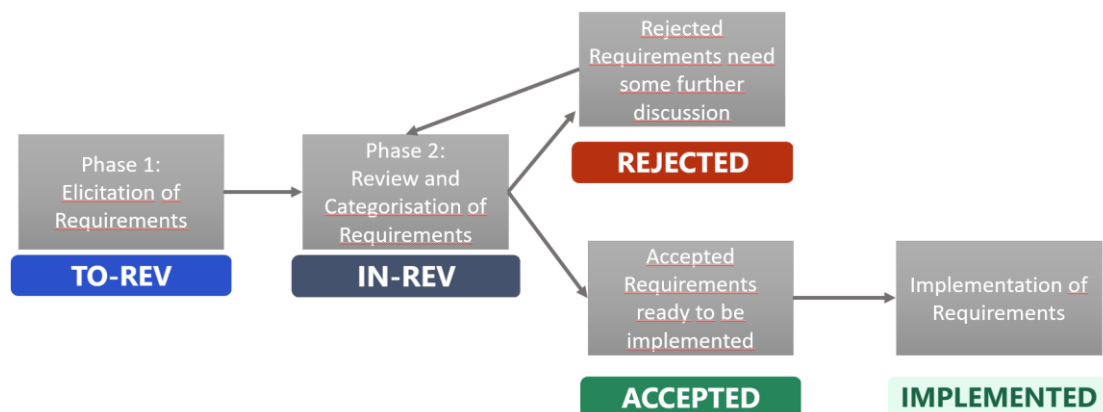
UP2DATE4SDV

Exported on 2025-09-26 15:31:21

Table of Contents

Table of Contents	47
Process of defining the Requirements	48
Explanation about the Requirements Form	48
Requirements Management	49
Short-List of Requirements	50

Process of defining the Requirements



The requirements definition process was organised as shown in the diagram above. Initially, the requirements were collected (Phase 1: Elicitation) directly in Confluence and then reviewed and categorised by the WP5 team (phase 2) and either accepted or rejected.

- Accepted requirements are ready for implementation.
- Rejected requirements must undergo a further review process or remain rejected.

The comment function of the Confluence system was used to discuss the requirements and clarify open points. This allows direct responses to questions and thus enables traceable documentation of the discussions on each individual requirement. The comments were then successively integrated back into the requirements. At the end of this process, the requirements document was automatically generated from the Confluence. As the project progressed, the requirements could be added to and commented on in a traceable manner. As the requirements are processed directly in Confluence, there can be no inconsistencies due to any parallel filing structures. It is possible to create an up-to-date requirements document at any time.

Explanation about the Requirements Form

ID	Unique ID, like REQ-0001 or REQ-001-001 for a derived requirement.
Related Requirements	Link to related requirements. Has to be done manually.
Derived Requirements	Works automatically if those are nested beyond this requirement.
Priority	MUST This requirement has high priority. SHOULD This requirement has medium priority. COULD This requirement has low priority.
Type	F-REQ It is a functional requirement. NF-REQ It is a non-functional requirement.

Requirement	Put the requirement here.
Explanation	Details about the requirement. Refer to the corresponding showcase and add the tag UP2DATE4SDV-SC-1 ...
Building Block	This helps to cluster the requirements and map them to the architecture(s).
Work Package	The Work Packages this requirement is relevant for, like WP-1 ... Please also add the correct tag to this page: UP2DATE4SDV-WP-1 ...
Use-Case	The Use-Cases this requirement is relevant for, like UC-1 ..., or RD Please also add the correct tag to this page: UP2DATE4SDV-UC-1 ...
Contact Person	Name a person who can give details about this requirement. Please use the Confluence handle, when available.
Status of the Requirement	TO-REV The Requirement needs to get a Review by the WP5 Team. This should be the initial State! IN-REV The Requirement is in Review by the WP5 Team. ACCEPTED The Requirement has been accepted by the WP5 Team. REJECTED The Requirement has been rejected by the WP5 Team. IMPLEMENTED The Requirement has been implemented.
Milestone	To which Milestone should this requirement be implemented?
Last Modified	Give the date of the last modification the table below contains further information about this.

Requirements Management

Date	What was modified?	Who did it?
01.07.2025	Short description of the changes that were made in this requirement.	Who modified or saved the modifications here?
...		

Short-List of Requirements

ID	Priority	Requirement
REQ-0001	MUST	The CARLA simulation must be able to change the sensor configuration of the vehicle by adding a LIDAR sensor.
REQ-0001-0001	MUST	The vehicle SW stack must be able to recognize a new hardware added (e.g. LIDAR sensor).
REQ-0001-0002	MUST	The HAL needs to recognize and support a new hardware sensor added (e.g. LIDAR sensor).
REQ-0001-0003	MUST	The in-vehicle orchestration needs to execute the software to support a new hardware sensor added (e.g. LIDAR sensor).
REQ-0001-0004	MUST	The cloud orchestration can get informed about the vehicle hardware configuration changes (e.g., LIDAR sensor) and triggers the deployment of necessary SW to the vehicle according to the sensor type/specification.
REQ-0003	MUST	Security validation of the update procedure must be done on the most representative demonstrator in UC5 and/or UC6.
REQ-0006	MUST	The system must be designed following a Model Based System Engineering methodology
REQ-0006-0001	MUST	A model of the base system must be constructed.
REQ-0006-0002	MUST	All timing constraints of the system must be satisfied in the model of the target system.
REQ-0006-0003	MUST	The model must be updated whenever there are hardware/software modifications or updates.
REQ-0006-0004	SHOULD	The model should be (re-)configured using optimization techniques to systematically explore optimal allocation of resources.
REQ-0002-0003	MUST	The research demonstration platform must provide the possibility of adding additional hardware modules to the system.
REQ-0002-0002	MUST	The research demonstration platform must provide the possibility of upgrading hardware modules.
REQ-0002	MUST	A research demonstration platform must be developed.
REQ-0007-0007-003-001	MUST	The intrusion detection system must have access to network traffic.
REQ-0007-0007-003-002	MUST	Communication datasets must be created to train AI models for network-level intrusion detection modules.
REQ-0007-0004	MUST	Secure boot for startUp
REQ-0007-0005	MUST	Rollback must be performed in case of non safety/ non security or interruption / unsatisfactory update
REQ-0007-0006	MUST	A registration of every update event must be carried

ID	Priority	Requirement
REQ-0007-0007	MUST	The update must be transmitted via secure communication based on IEC 62443.
REQ-0007-0007-001	MUST	Techniques and measures must be implemented to support safe and secure communications.
REQ-0007-0007-002	MUST	The platform must have at least one cryptography engine
REQ-0002-0010	MUST	The demonstration of the Use-Case 3 must be shown on more than one real hardware module.
REQ-0007-0007-003	SHOULD	The system should be able to detect intrusions in the communication channels.
REQ-0002-0016	MUST	A time and resource composable SoM hardware platform must be provided.
REQ-0002-0017	MUST	At least two hardware platforms must be supported by monitoring and SASE operation and updates concerning local node operation and internode communication.
REQ-0002-0019	MUST	The process definitions for software updates, hardware pgrades, task reallocation, system reconfiguration and fleet orchestration must be aligned with KPI O4.X.
REQ-0002-0021	MUST	Virtual Integration Testing (VIT) of KPI O4.X must be integrated to verify update-compatibility.
REQ-0002-0022	MUST	A harmonized and simplified development, integration and validation processes and framework for SDV applications must be provided.
REQ-0002-0023	MUST	The hardware modules of the demonstrator must support hardware based isolation mechanisms.
REQ-0002-0024	MUST	The hardware platform must support virtualization extensions
REQ-0002-0001	MUST	The research demonstration platform must provide the possibility of updating software module.
REQ-0002-0011	MUST	The function deployment must consider resource constraints like the type of hardware and processing power, size of available memory and network properties like bandwidth and latency.
REQ-0008	MUST	Security validation of the update procedure must be done on the most representative demonstrator in UC5 and/or UC6.
REQ-0005	MUST	E/E functional architecture needs to be defined
REQ-0007	MUST	The update must be aligned with SASE standards (e.g. IEC 62443, IEC 61508, ISO 26262).
REQ-0007-0001	MUST	Safety mechanisms and diagnostics must be executed periodically to guarantee the correctness of the safety functions
REQ-0007-0002	MUST	Security properties must be respected for all updates
REQ-0007-0003	MUST	Ensure only authorized and uncorrupted updates while preserving authenticity, confidentiality and integrity

ID	Priority	Requirement
REQ-0002-0020	MUST	The orchestration manager must be implemented.
REQ-0004-0002	MUST	The update procedure shall resume safely after any power cycle or sleep/wake event, without causing ECU corruption or incomplete installation
REQ-0004	MUST	E/E validation of the update/upgrade procedure must be done on the most representative demonstrator in UC5 and/or UC6.
REQ-0004-0001	MUST	The update/upgrade procedure shall not disrupt ongoing in-vehicle communication
REQ-0002-0018	MUST	A service-oriented reconfigurable communication plane across nodes and into the cloud must be defined and provided.
REQ-0002-0012	MUST	The Validation of the consideration of resource constraints must be shown in a virtual environment.
REQ-0004-0003	MUST	The update/upgrade procedure shall remain stable under varying supply voltage and current conditions
REQ-0002-0025	MUST	The in-vehicle network must support deterministic, time-guaranteed communication for safety-critical data.
REQ-0002-0026	MUST	The execution platform must provide deterministic scheduling and resource isolation for applications.
REQ-0002-0027	MUST	The system must provide an autonomous driving perception stack.
REQ-0002-0028	MUST	The scheduling platform must use criticality information to prioritize safety-critical functions.
REQ-0002-0004-0001	MUST	The cloud orchestration system must provide CI-pipelines to automate the steps from development result to vehicle deployment.
REQ-0002-0004	MUST	The research demonstration platform must provide the possibility of orchestrating the update process from the cloud.
REQ-0002-0007	MUST	The in-vehicle target orchestrator ensures continuous functionality reliably during network issues by local migration.
REQ-0002-0006	MUST	The in-vehicle target orchestrator reliably ensures continuous execution of safety critical functions.
REQ-0002-0008	MUST	The in-vehicle target orchestrator ensures continuous functionality reliably after network issues by backwards migration.
REQ-0002-0029	SHOULD	The components of the research demonstrator should be public domain.
REQ-0002-0013	MUST	The security properties of the research demonstrator and the update system analysed.
REQ-0009	MUST	Phase-Change Memory (PCM) based Update Storage
REQ-0009-0001	MUST	Write Granularity
REQ-0010-0001	MUST	Automated regression testing and simulation environments must be run in the models for pre-deployment validation.

ID	Priority	Requirement
REQ-0009-0003	MUST	Programming Speed
REQ-0010	MUST	Include pre-deployment verification steps to confirm correct integration and behavior in terms of security
REQ-0010-0003	MUST	static analysis and/or model checking tools must be used in the defined security model to analyse the code for all updates.
REQ-0010-0002	MUST	SASE mechanisms must be checked in pre deployment in the models
REQ-0002-0014	MUST	Model based security checking must be implemented to detect malicious code in update packages.
REQ-0002-0015	MUST	Model based security checking must be implemented to detect software defects and deviations from the intended system behavior.
REQ-0002-0009	MUST	The function deployment must consider resource limitations of the connected vehicle.
REQ-0009-0004	MUST	Memory Extensibility
REQ-0002-0005	MUST	The criticality of the executed services and functions is available to the target orchestrator.
REQ-0009-0002	MUST	Retention
REQ-0011-0001	MUST	Digital-Twin Integration
REQ-0011-0002	MUST	Dynamic Task Reallocation
REQ-0011-0003	MUST	Secure PCM-Based Update Orchestration
REQ-0012	MUST	Reconfigurable Container Middleware and Service-Oriented Communication
REQ-0013	MUST	Mixed-Criticality Hardware Abstraction Layer and Microkernel Architecture
REQ-0011-0004	MUST	Rollback and Fault Recovery
REQ-0011	MUST	Cloud-edge orchestrator for dynamic task reallocation and secure updates