



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

3.1. Requirements & Drafts of the Orchestration and Reconfiguration Plane (ORP) V1.0

Deutsches Zentrum für Luft- und Raumfahrt e. V. (DLR)



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.

All rights reserved.

Deliverable Information		
Deliverable Number	D3.1	
Deliverable Name	Requirements & Drafts of the Orchestration and Reconfiguration Plane (ORP)	
Nature	Report (R)	
Dissemination Level	Public	
Contractual Date of Delivery	31.12.2025	
Actual Submission Date	31.12.2025	
Lead Beneficiary	Deutsches Zentrum für Luft- und Raumfahrt e. V. (DLR)	
Lead Author	Henning Schlender	
E-Mail	Henning.schlender@dlr.de	
Contributing Beneficiaries	AVL	Hannes Fuchs
	DLR	Gregor Nitsche Patrick Uven Henning Schlender
	IKL	Ane Mijangos
	NEO	Othon Tomoutzoglou
	ITI	Joan Valls
	CON	Florin Anton
Reviewing Beneficiaries	AVL	Hannes Fuchs
	DLR	Gregor Nitsche
Work Package(s)	WP3	
Related Task(s)	T3.1	Continuous integration, orchestration & deployment requirement
Keywords	Software-Defined Vehicle, SDV, Horizon Europe, Automotive, CCAM, OTA, Update, Upgrade, DevOps, E/E-Architecture, Orchestration, Vehicle-to-cloud, Cloud-Edge Continuum, Reconfiguration	

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	03.12.2025	Initial version
V0.2	17.12.2025	Updated version for partners
V0.3	18.12.2025	Version for review
V0.4	05.01.2026	Reviewed Version
V0.5	07.01.2026	Review comments included, version for acceptance
V0.6	07.01.2026	Review comments included, version for clearance
V1.0	07.01.2026	Final and reviewed Version (public)

Table of contents

Table of contents.....	6
Executive Summary	7
Acronyms and Abbreviations	8
List of Figures.....	9
List of Tables.....	9
References and Background Material	9
1. Introduction	11
1.1. Project Outline	11
1.2. Work-Package Outline	12
1.2.1. Work-Package Objectives	12
1.2.2. Work-Package Description	12
1.2.3. Work-Package Status.....	14
1.3. Purpose and Scope of the Deliverable.....	14
2. The Orchestration & Reconfiguration / Deployment Plane (ORP).....	14
2.1. Development of Software Updates	15
2.2. Cloud Fleet & Update Management	16
2.2.1. Process of Developing Orchestration Solution.....	16
2.2.2. Process of Fleet Deployment.....	17
2.3. Platform Orchestration	17
2.3.1. Cloud Computation vECU Offloading	19
2.4. Continuous Monitoring.....	20
2.5. Process of Fleet Orchestration & Telemetry State	20
3. ORP Requirements	21
4. Alignment with other WPs	23
5. Summary and Outlook	23

Executive Summary

This document contains the initial requirements and a first version of the architecture of the Orchestration and Reconfiguration Plane (ORP). The requirements are developed jointly with the other partners and have been synchronized between the work-packages to ensure the close integration and common understanding for the further development steps. As such the deliverable forms an important building block for the safety and security requirements in T4.1.

Acronyms and Abbreviations

Abbrev.	Full Name
AD	Autonomous-Driving
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
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
PCoCo	Primary Coordinator Contact
PGA	Project General Assembly
PPR	Periodic Project Report
PU	Public
R	Report
SASE	SAfety and SEcurity
SD-CCAM-V	Software-Defined CCAM vehicles
SDV	Software Defined Vehicle

SEN	Sensitive
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.	12
Figure 2: Architecture of the ORP	14
Figure 3: The ORP Process	15
Figure 4: Architecture of the Software Update Development	15
Figure 5: The process of Software Update Development.....	16
Figure 6: Architecture of the Cloud Fleet & Update Management	16
Figure 7: Process of Orchestration Solution Development	17
Figure 8: Process of Fleet Deployment	17
Figure 9: Architecture of the Platform Orchestration	18
Figure 10: Process of Platform Orchestration	19
Figure 11: Cloud Computation Offloading	20
Figure 12: Fleet Orchestration	20

List of Tables

Table 1: List of ORP Requirements	21
---	----

References and Background Material

Internal Background

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

External Background

- [2] D4.1_UP2DATE-Middleware-foundationns-V1.1.pdf: https://h2020up2date.eu/wp-content/uploads/2021/10/D4.1_UP2DATE-Middleware-foundationns-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. 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:

- 01. Creation of a reconfigurable cloud connected zonal vehicle architecture blueprint based on heterogeneous COTS HW (exploiting and building on a unified HAL)**
- 02. 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)**
- 03. Realization of a dynamic update, orchestration & reconfiguration plane**
- 04. Provide harmonized and simplified development, integration and validation processes and framework for SDV applications**
- 05. Assess joint Safety and Security (SASE) of all elements from O1, O2 and O3**
- 06. Demonstration of UP2DATE4SDV solution on several CCAM relevant Use Cases**
- 07. 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 WP and their communication, internally as well as 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.

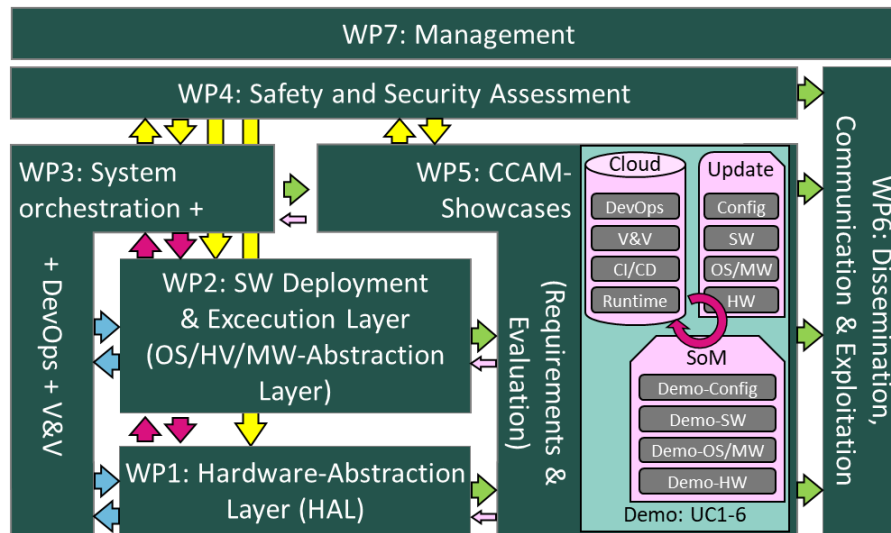


Figure 1: Overview of the UP2DATE4SDV work-packages and their Interaction.

1.2. Work-Package Outline

As shown in Figure 1, WP3 is a central work package with interfaces to all other work packages in the project. Central work item of this work package is the central, cloud-integrated orchestration & deployment plane (ORP), which has relations to all the other work items in the project, the Safety and security Assessment from WP4, the Software & Execution Layer from WP2 and the Hardware Abstraction Layer from WP1. The developments in this work package are also the basis of the CCAM-Showcases.

1.2.1. Work-Package Objectives

WP3 aims at providing a central, cloud-integrated orchestration & deployment plane (ORP), to coordinate software updates, hardware-upgrades, reconfigurations and task re-allocations at the vehicle and fleet level considering available system- and scenario-knowledge. An analysis of requirements (T3.1) is followed by the implementation of the development-, and implementation-framework (T3.2), V&V framework (T3.3), system-side update manager (T3.4) and cloud-side orchestration & deployment manager (T3.5).

1.2.2. Work-Package Description

WP3 is structured into five main tasks, each one covering a specific goal in the WP.

Task 3.1: Continuous integration, orchestration & deployment requirements. This task is to compile the requirements to define the architectures, interfaces and processes for continuously integrating, orchestrating and deploying software-updates, hardware-upgrades, reconfigurations and dynamic, mode dependent car-to-cloud/cloud-to-car task re-allocations, as well as for the integration into cloud-

based DevOps, CI/CD and V&V. Its results will be provided by D3.1 for the safety- and security requirements in T4.1.

Task 3.2: Design and implementation Framework. This task is to enhance an existing MBE-based design and implementation framework to generate reliable, certifiable code suitable for high performance automotive platforms, fitting the Operating and Software System Abstraction Layer (OAL). The design and implementation framework will ensure a seamless transition to the V&V processes (T3.3), supporting the update and deployment manager (T3.4 and T3.5). Furthermore, methods for the V&V as well as for the DSE will be integrated to facilitate the design and analysis of safety-critical SDV applications by verifying and optimizing the timing and the distribution of resources and functionalities for different operational conditions, using formal or metaheuristic algorithms such as genetic algorithms.

Task 3.3: V&V Framework. In this task, a V&V framework will be designed for next generation AI-based SDV applications, realizing new V&V methods and tools within CI/CD- and DevOps Environments. To enable high-fidelity SDV testing environments (e.g., MIL/SIL/HIL), OAL virtualization mechanisms will be provided at various abstraction levels, including the instruction level. Standard test suites will be integrated with automation infrastructure to ensure that deployed software modules meet specified non-functional properties and quality standards. Model-based V&V methods for the security- and compatibility checking of updates based on code analysis and virtual integration testing will also be integrated. Certifiability will be targeted by aligning the procedure with the WP4 SASE analysis.

Task 3.4: Vehicle to Cloud continuum orchestration and deployment implementation. In this task, the cloud-side part of the orchestration and reconfiguration plane will be implemented as middleware extension that coordinates resource composable orchestration and deployment of containers and communication channels. To support containerized workload deployment a digital-twin platform will be enhanced and integrated. The updated platform will publish available resources via device twins. From T3.5, the corresponding processes and interfaces for orchestrating and updating components will be implemented, and similarly, from T3.3, the corresponding interfaces for virtual integration testing of resource-properties according to the SASE analysis of WP4. Furthermore, data-driven orchestration and task execution will be enabled based on a reliable on-/off-loading of software via dependable deterministic communications.

Task 3.5: SASE update manager implementation. This task will develop the system-side SASE update manager for containers and communication channels to run node local container download, safety and security checks, and execution- and communication-control services from T2.2 and T2.3. To this goal, operating-system and hypervisor capabilities will be tailored for network-based updating mechanisms, micro-services will be implemented to orchestrate the deployment management, and robust authentication and validation mechanisms will be integrated based on results from T4.3. Furthermore, to enable updates, monitoring and data recovery at runtime, SASE contract based on-demand services will be implemented. To enable efficient and reliable download, distribution and management of containerized applications from cloud to edge nodes for the new STM32 central computing platform for automotive the platform-specific deployment infrastructure will be integrated and the local OTA manager of the STELLAR MCU will be developed. Finally, the implementation of the secure update mechanism will be integrated and validated by testing its robustness and efficiency, its rollback capability, and its failsafe installation.

1.2.3. Work-Package Status

Before the work package activities started in M4, there were already previous activities for WP5. The work in the work-package started in T3.1. This work on collecting the initial requirements for the continuous integration, orchestration & deployment will be completed with this deliverable D3.1. Furthermore, regular meetings have been implemented to coordinate work among the partners. This way, this deliverable forms an important building block for the safety and security requirements in T4.1. The work for T3.2, T3.3, T3.4 and T3.5 will start in M7.

1.3. Purpose and Scope of the Deliverable

This deliverable will summarize requirements and drafts of the processes and the architecture of the ORP and its tools and interfaces for the cloud-based V&V, deployment and DevOps-integration for software-updates, hardware-upgrades and dynamic reconfigurations. It will be provided as a supporting document detailing safety- and security requirements in T4.1.

2. The Orchestration & Reconfiguration / Deployment Plane (ORP)

The Orchestration & Reconfiguration / Deployment Plane (ORP) summarises all tasks on the cloud side, as well as on the local vehicle side, which are related to orchestration, reconfiguration and deployment.

The architecture (Figure 2) of the ORP consists of the following parts: The Update Development, the Cloud and the Automotive Platform. In the Cloud there is the Fleet & Update Management and the vECU Computation Offload. The Automotive Platform consists of one or more Central ECUs and several Zonal Gateways.

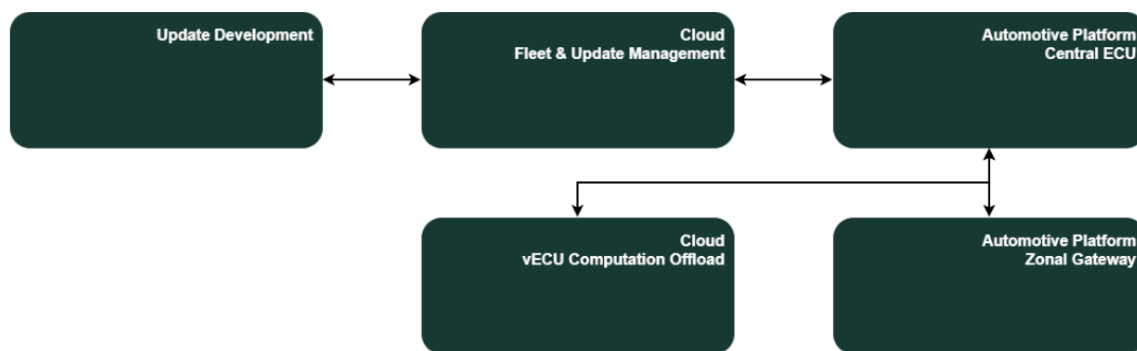


Figure 2: Architecture of the ORP

The main process of the ORP (Figure 3) consists of the following elements: development of application updates, the development of an Orchestration Solution, the Fleet Deployment, the Platform Deployment and the Monitoring. One of the most important elements in this process is the Orchestration Solution. It is a package that can combine several changes or just one update for a SDV platform. It can include the instructions for the following change operations:

1. Software update
2. Hardware upgrade
3. Software reconfiguration
4. Software offloading

A classical software update can be used to fix a bug in an application, to improve an application or to add new functionalities. The hardware upgrade can be used to replace a defect hardware module or to improve the platform capabilities. The software reconfiguration makes it possible to adapt the platform to guarantee the platforms functionality. The software offloading is a special form of reconfiguration since it will migrate applications from and to the cloud.

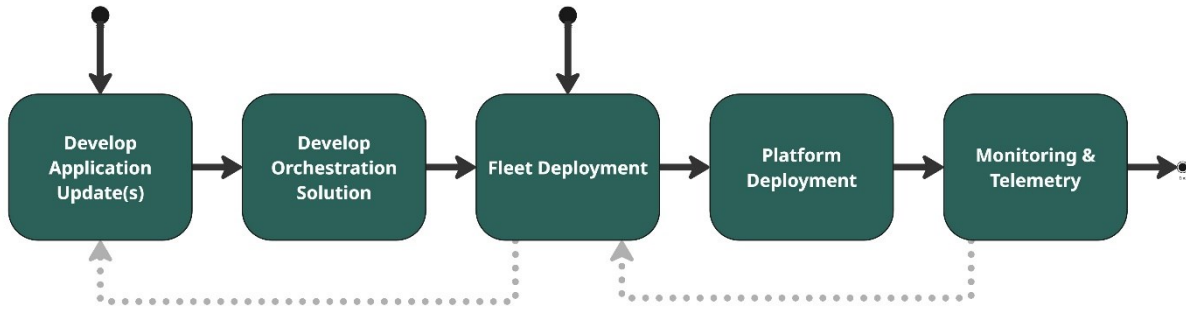


Figure 3: The ORP Process

2.1. Development of Software Updates

The developers of supplier companies work with tools in a web-based context according to a model-based development approach (Figure 4). In addition to conventional software-in-the-loop tests, non-functional properties are also specified for the individual software modules and tested using a model-based test environment (VIT). The models and non-functional properties are transmitted to the fleet operator or OEM.

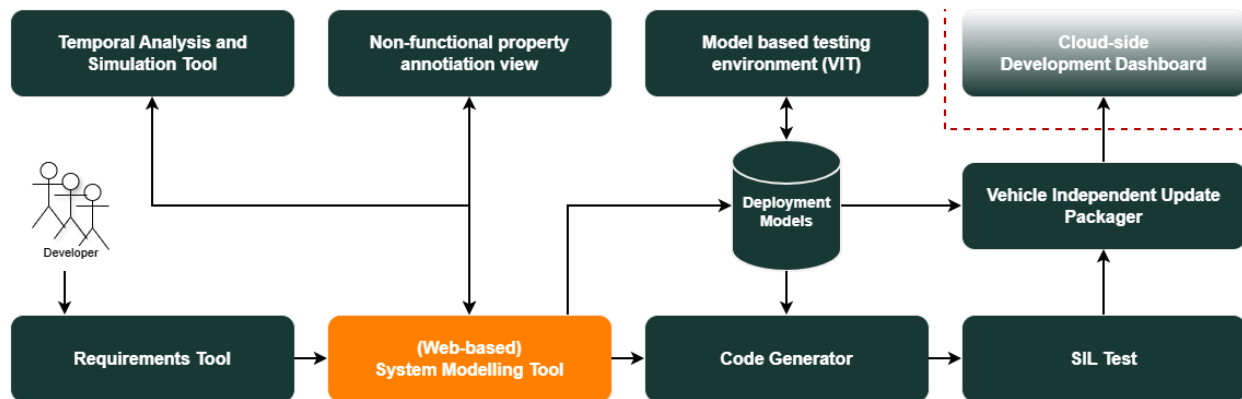


Figure 4: Architecture of the Software Update Development

In UP2DATE4SDV, these non-functional properties are used later in the process, for example for virtual testing and the generation of monitors (Figure 5). These properties can be tracked from specification, over development, testing, deployment and finally run-time to ensure the satisfaction of requirements.

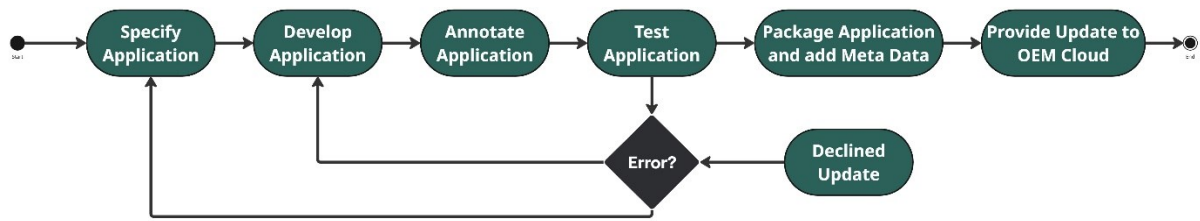


Figure 5: The process of Software Update Development

2.2. Cloud Fleet & Update Management

The interface to the suppliers' developers is the Development Dashboard (Figure 6). This is used to store the update packages sent in a database. The central element is the Cloud Orchestrator Core, which provides tools for model-based testing of software updates, as well as a vECU environment. Here, integration tests defined by the fleet developer are carried out to test the software updates in the context of the SDV. When developing an orchestration solution, the Orchestrator Core accesses both the fleet data and current telemetry data based on monitors from the fleet in order to develop possible solutions. The Rollout Manager then takes care of the fleet deployment.

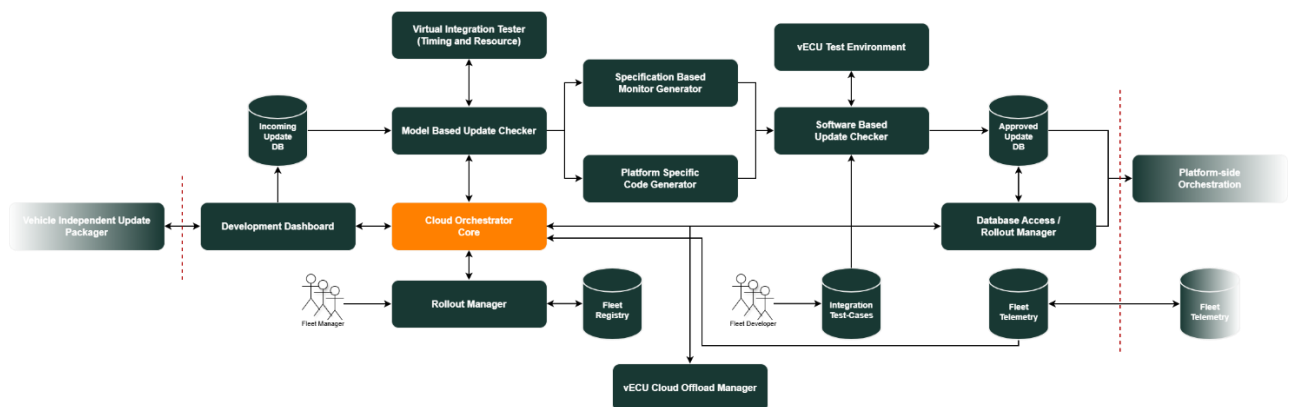


Figure 6: Architecture of the Cloud Fleet & Update Management

2.2.1. Process of Developing Orchestration Solution

This process (Figure 7) takes place at the OEM and is triggered when a new update for an application is provided by the supplier or when the fleet manager wants to make a change to an SDV fleet.

First, the new update is tested in the usual way to rule out any errors on the part of the supplier. Then an orchestration solution is developed. This contains all the changes required for this process, such as software updates or reconfigurations. If it is determined that a hardware upgrade is necessary for the platform, this is specially noted, as this orchestration solution may only be executed once this hardware upgrade has been carried out. A model-based virtual integration test with the overall system is used to determine whether the new solution has all the necessary resources available and whether it can be executed safely within the scope of its specification. As part of the simulation-based testing, further corner cases are tested at a functional level. If errors occur during these tests, the update is declined.

If the process is completed without errors, monitors are generated based on the specification of the resources, which later monitor compliance with the specification on the platform. The orchestration solution is then handed over to Fleet Deployment.

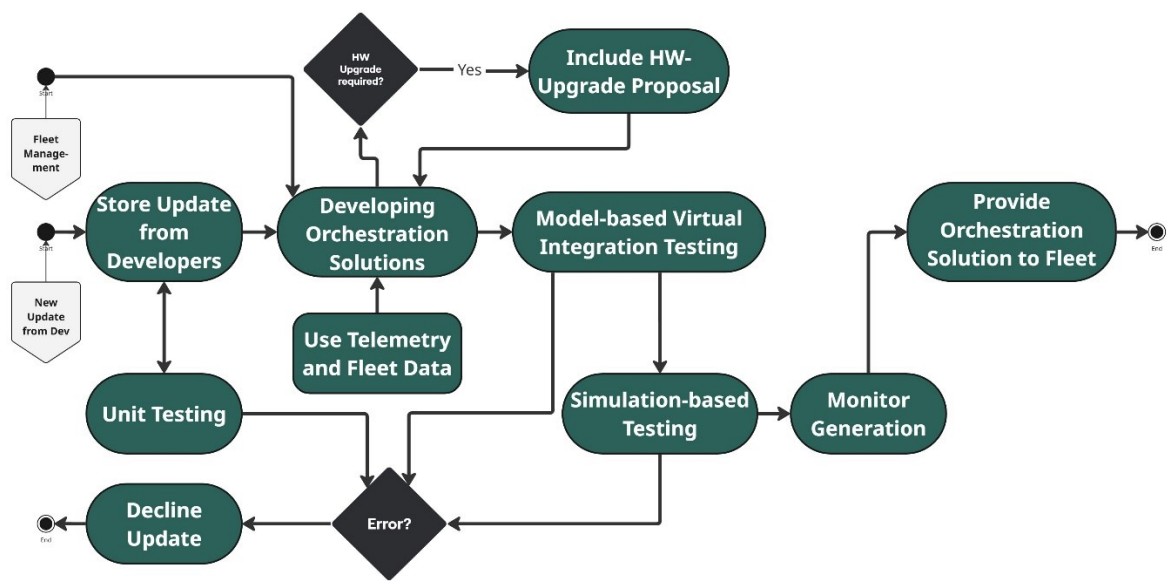


Figure 7: Process of Orchestration Solution Development

2.2.2. Process of Fleet Deployment

Various strategies can be used for fleet deployment (Figure 8), for example to avoid overloading the networks. These strategies must be defined by the fleet manager and can also vary from model series to model series. If an orchestration solution requires a hardware upgrade, this must be notified in advance to the responsible vehicle operators. It must be ruled out that vehicles are running the orchestration solution before this upgrade has been carried out.

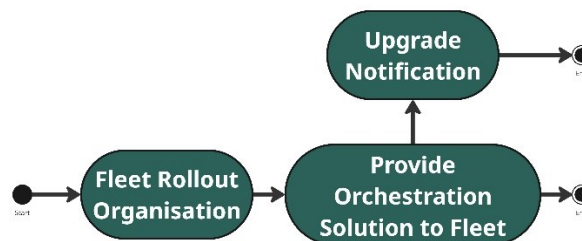


Figure 8: Process of Fleet Deployment

2.3. Platform Orchestration

The platform orchestration executes the orchestration solution after the fleet deployment in the vehicle. To do this, it requires access to monitoring and scheduling data for the individual hardware components. The platform orchestrator works together with local OTA agents. These execute manipulation commands, such as downloads, in the individual target modules.

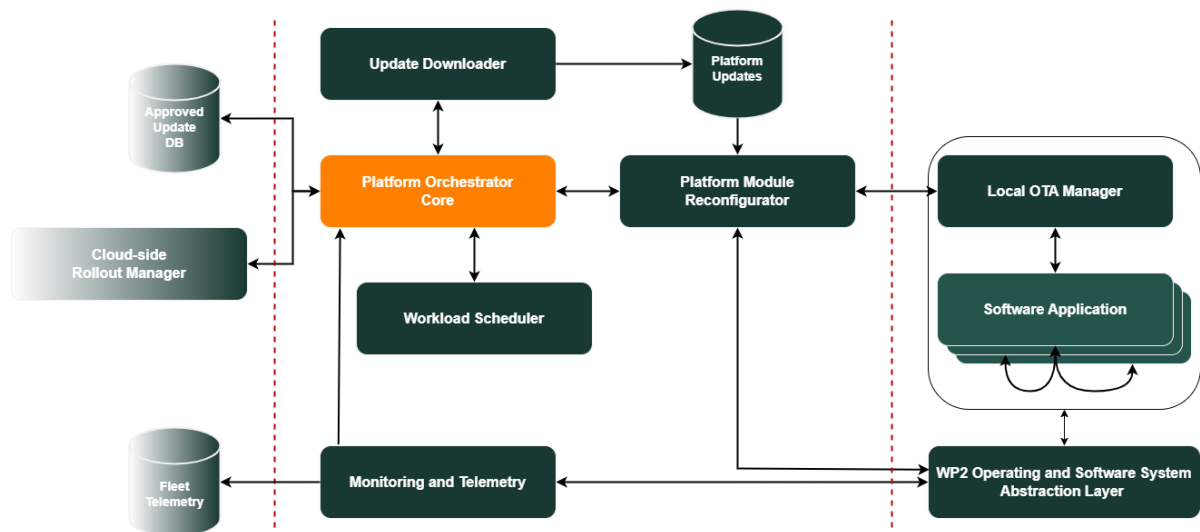


Figure 9: Architecture of the Platform Orchestration

The Platform Orchestrator continuously polls the cloud for new orchestration solutions (Figure 10). If one is offered, the metadata is used to check whether this orchestration solution was really developed for the current platform and whether the transfer went correctly. The system then checks whether a hardware upgrade is necessary for this solution. If this is the case, it is checked whether the upgrade has already been carried out. It is also checked whether the vehicle is in a safe state to carry out changes. Before such a solution is processed, the current system configuration is saved for rollbacks. The individual change operations from the orchestration solution are then processed. In the case of a software update, the status of the target application is first saved by a local agent in the target partition and then a download command is sent for the respective new application. This agent performs a series of tests, updates the resource monitors, updates the application state and activates the new application or deactivates the old one if required. A final system test is then carried out. In the case of a hardware upgrade, the new hardware is first detected and then configured and tested. The affected application is then migrated by saving its status and downloading it to the target location just like a software update. The same applies to a reconfiguration. In this case, the states of the application actions are saved and downloads are processed the local agents at the corresponding locations.

During offloading, the same process is performed by an agent in the cloud. If an error occurs during any of these processes, Fleet Management is notified and rolled back to the previous orchestration solution.

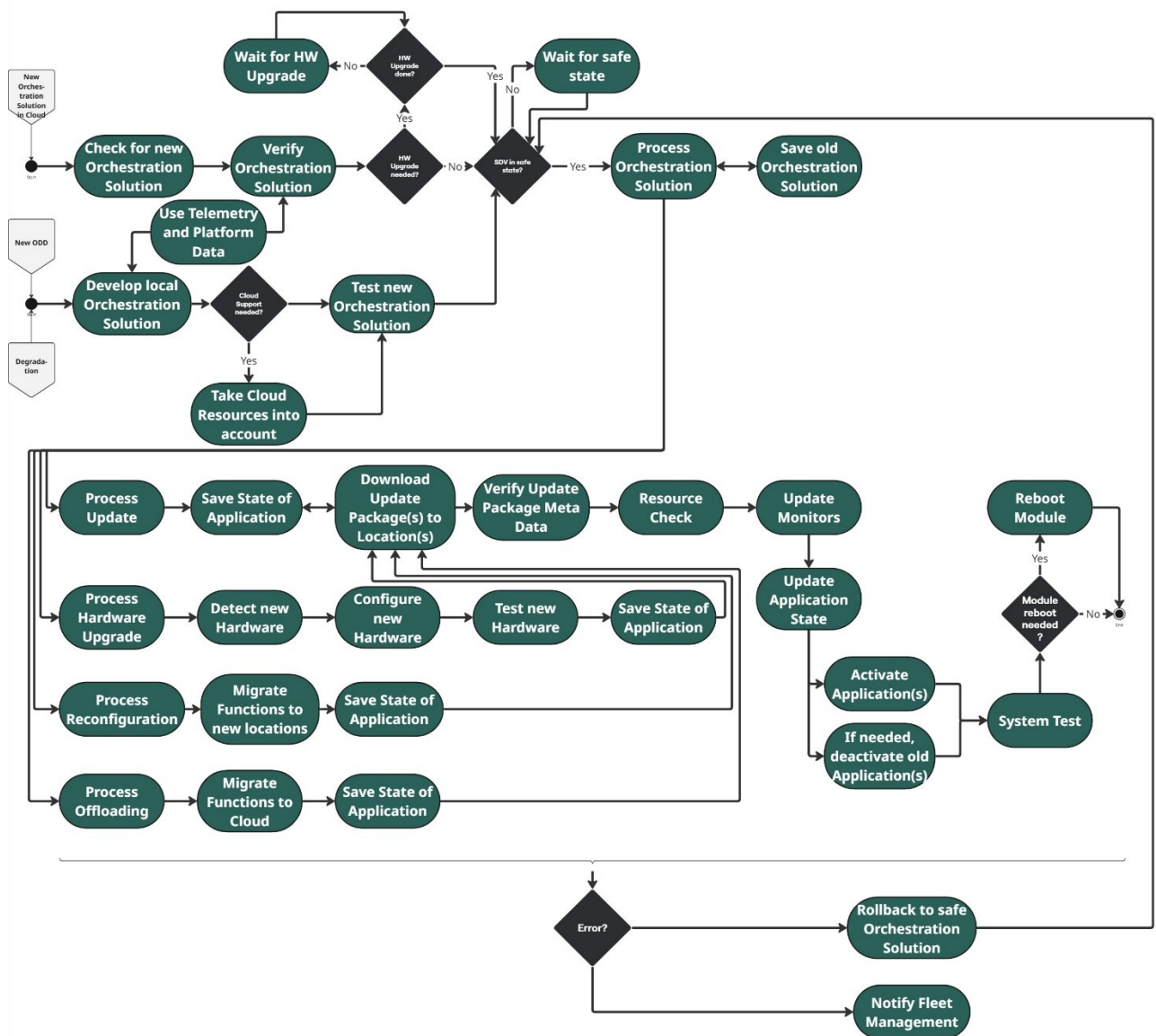


Figure 10: Process of Platform Orchestration

2.3.1. Cloud Computation vECU Offloading

Offloading is special in the sense that the target platform does not exist on the vehicle, but in the cloud (Figure 11). This can happen if there are not enough resources for one or more functions for an orchestration, for example due to a breakdown, or specifically as part of a fleet strategy. It is also possible to maintain a digital twin of a vehicle or just a part of it in the cloud.

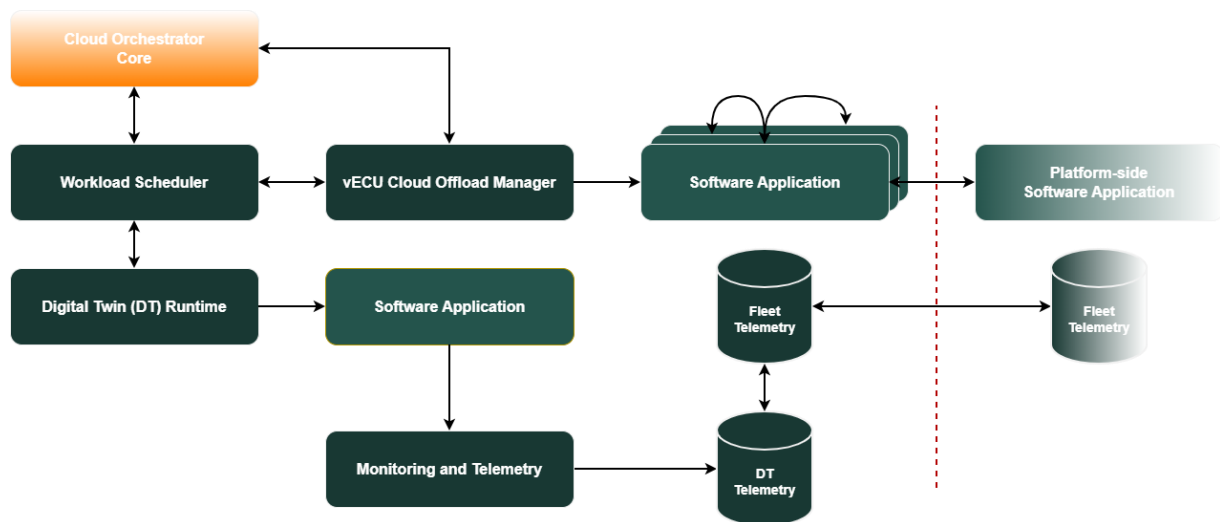


Figure 11: Cloud Computation Offloading

2.4. Continuous Monitoring

In Figure 9 monitoring and telemetry collection on the platform side was introduced. Non-functional monitoring will continuously collect and analyze the status of the deployed modules and inform the platform orchestration core about specification violations even after the update process. This allows the orchestration core to make decisions even after changes are finished to initiate reaction and provide feedback to the cloud orchestration.

2.5. Process of Fleet Orchestration & Telemetry State

The platform orchestration state and telemetry data are continuously collected and analyzed (Figure 12). If errors occur or deviations from the specified behavior of the system or its subcomponents occur, the fleet management is notified. They must then find out how the problems occurred and whether the suppliers need to be involved.

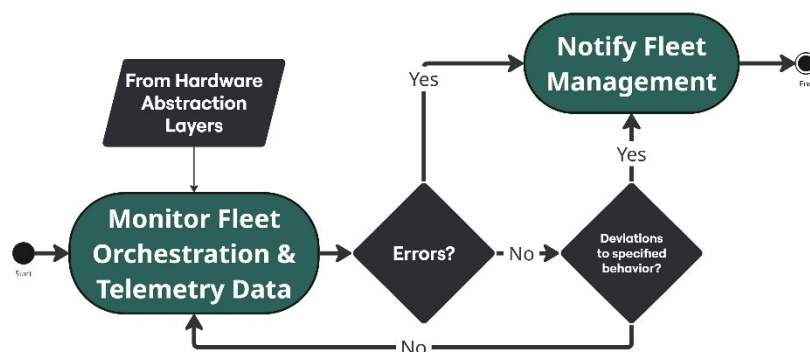


Figure 12: Fleet Orchestration

3. ORP Requirements

The following table lists all the requirements for the ORP, which were collected during this work phase (see Table 1).

Table 1: List of ORP Requirements

ID	Requirement
REQ-ORP-0002	The ORP must support a fleet of SDVs.
REQ-ORP-0001	The ORP must be able to update software components in SDVs (called platforms).
REQ-ORP-0006	The platform must be able to reconfigure the applications depending on the available resources.
REQ-ORP-0007	The platform must be able to reconfigure the applications depending on the network quality.
REQ-ORP-0016	The cloud side must be able to test platform specific code on virtual ECUs (vECUs).
REQ-ORP-0017	The platform must be able to rollback to a stable version after a failed update process.
REQ-ORP-0003	The platform must be able to download an update package and save it locally.
REQ-ORP-0025	The developer side must be able to generate vehicle independent code for testing.
REQ-ORP-0026	The developer side must be able to bundle vehicle independent update packages.
REQ-ORP-0027	The developer side must enable model based testing.
REQ-ORP-0029	The Vehicle Independent Update Package Builder (REQ-ORP-0026) must get the module specification from the web-based development tooling (REQ-ORP-0012).
REQ-ORP-0033	The platform must alert the end user if due to reconfiguration previous functionality is lost or altered
REQ-ORP-0032	The cloud side must maintain a database with information regarding the updates already done and which are not done yet
REQ-ORP-0031	The platform must be able to detect that a new configuration or update/upgrade besides being available is also compatible and compliant with the existing/available HW resources
REQ-ORP-0035	The developer side must check that the updated model satisfies all timing constraints.
REQ-ORP-0036	The developer side must use a common requirement format (e.g. ReqIF)
REQ-ORP-0037	The developer side tools must be able to exchange models in a shared format (e.g., AADL)
REQ-ORP-0030	The platform must be able to detect that a new configuration is available.
REQ-ORP-0040	The cloud-based and platform-based orchestrators shall implement a secure handshake protocol and mutual authentication for update notifications, device-twin synchronization and telemetry exchange. The interface shall support encrypted transport (e.g., TLS) and include mechanisms to verify the integrity and origin of update packages.
REQ-ORP-0039	The platform-side ORP shall implement a local PCM Mapper / OTA manager capable of performing atomic A/B firmware updates on STELLAR MCUs using Phase-Change Memory (PCM), which enables to store A/B firmware on the same memory zone without

	doubling the memory. The mechanism shall support seamless roll-back to the previous version in case of update failure.
REQ-ORP-0038	The cloud-side ORP orchestrator shall integrate digital-twin models with resource-monitoring metrics (CPU, memory, network, energy) from SDVs to perform resource-aware scheduling, dynamic task off-loading and workload migration across the vehicle-to-cloud continuum.
REQ-ORP-0043	The ORP shall have one or more functions to support the identification of dependencies for software update packages with the target
REQ-ORP-0044	The ORP should have one or more functions to initiate actions if it is notified of the failure of a software update operation.
REQ-ORP-0045	The platform based ORP shall have one or more functions to identify the ECUs to which a software update package applies.
REQ-ORP-0041	The ORP shall have one or more functions for receiving, storing and processing of vehicle configuration information.
REQ-ORP-0042	The cloud based ORP shall have one or more functions to associate software update packages with targets.
REQ-ORP-0046	The cloud based ORP must validate that the necessary in-vehicle resources in the target are available before any deployment (V&V phase)
REQ-ORP-0009	The ORP must have a central component on the cloud side to orchestrate the cloud sided part of the update process.
REQ-ORP-0004	The platform must have a managed database for downloaded updates.
REQ-ORP-0011	The cloud based orchestrator (REQ-ORP-0009) and platform based orchestrator (REQ-ORP-0010 - UP2DATE4SDV - Confluence) must be able to communicate with each other to inform of new updates, request a list of available updates and provide update packages.
REQ-ORP-0012	Multiple developers must be able to model, specify and develop a platform collaboratively.
REQ-ORP-0013	It must be possible to extend models by non-functional properties
REQ-ORP-0014	The cloud side must be able to manage the access to the Approved Updates Database.
REQ-ORP-0010	The platform must have a central component on the platform side to orchestrate the platform sided part of the update process.
REQ-ORP-0028	The platform must have interfaces to the Software Application Deployment, Separation and Execution Layer.
REQ-ORP-0005	The cloud must have a managed database for tested updates.
REQ-ORP-0015	The cloud side must be able to manage the update rollout over a fleet of SDVs.
REQ-ORP-0018	The cloud side must have a database to store updates, which must be tested.
REQ-ORP-0019	The cloud side must have a component to manage the testing of new updates.
REQ-ORP-0020	The cloud side must provide a Development Dashboard.
REQ-ORP-0021	The cloud side must be able to generate monitors for extra-functional properties from the software specification of the updates.
REQ-ORP-0022	The cloud side must be able to fulfill Virtual Integration Tests for the updates.
REQ-ORP-0023	The cloud side must be able to store vehicle telemetry data.
REQ-ORP-0024	The cloud side must be able to host vehicle functions on virtual ECUs.

REQ-ORP-0034	It must be possible to define testing pipelines with specific test steps for software updates OR provide an interface which allows to change the state of an update (e.g., from "ready to test" to "test passed" etc.).
REQ-ORP-0047	the vehicle's edge platform is running with active connections to the cloud and multiple workloads deployed resource demand or workload type changes (for example, a compute-heavy function needs to migrate from cloud to edge) the system reallocates resources dynamically, ensuring no disruption of active, safety-critical functions and preserving service continuity
REQ-ORP-0049	The system must be designed following a Model Based System Engineering methodology
REQ-ORP-0049-0001	A model of the base system must be constructed.
REQ-ORP-0049-0002	All timing constraints of the system must be satisfied in the model of the target system.
REQ-ORP-0049-0003	The model must be updated whenever there are hardware/software modifications or updates.
REQ-ORP-0049-0004	The model should be (re-)configured using optimization techniques to systematically explore optimal allocation of resources.
REQ-ORP-0048	The cloud orchestration must 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.

4. Alignment with other WPs

The first two phases of this project (MS1, MS2) were concerned with formulating the requirements for the various components of the update middleware. The requirements are developed jointly with the other partners in this project to ensure close integration in further development. Coordination with the other work packages was ensured through intensive dialogue. The interfaces to the Hardware Abstraction Layer (HAL) and the Operating and Software System Abstraction Layer (OAL) are particularly important, as the ORP would not function properly without them.

5. Summary and Outlook

The deliverable D3.1 contains the requirements and a first version of the architecture of the Orchestration and Reconfiguration Plane (ORP). The requirements are developed jointly with the other partners and have been synchronized between the work-packages to ensure the close integration and common understanding for the further development steps. As such the deliverable forms an important building block for the safety and security requirements in T4.1.

The next deliverables for WP3 are scheduled in six months. D3.2 (DevOps and V&V Framework [Initial]), due in M12, will implement and describe initial interface- and process-definitions together with a documentation of the cloud-based design & implementation framework and the cloud-based V&V-framework, and its connection to the ORP. For the design and implementation framework, it will describe how an existing MBE-based design and implementation framework is enhanced to generate reliable, certifiable code suitable for the deployment on high-performance automotive platforms, fitting the OAL, and how methods for V&V as well as for DSE can be integrated, to facilitate the design and analysis of safety-critical SDV applications by verifying and optimizing the timing and the distribution of resources and functionalities. In parallel, D3.3 (Safe and secure orchestration and reconfiguration plane [Initial]), also due on M12, will initially specify prototypical software-

implementations, interface- and process-definitions and the orchestration & reconfiguration plane, consisting mainly of cloud- and system-side update management components and interfaces to cloud-side V&V-, CI/CD-, and DevOps-components. It will outline the development and integration of middleware extensions for resource composable orchestration, deployment, and communication channels, as well as the implementation of a digital-twin platform for containerized workload deployment. By providing detailed documentation and interface definitions, this deliverable aims to facilitate the seamless integration of SASE capabilities with existing V&V-, CI/CD-, and DevOps tools