



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

D3.3 Safe and Secure Orchestration and Reconfiguration Plane [Initial] V1.0

Neotera S.r.l. (NEO)

30.06.2026



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	D3.3	
Deliverable Name	Safe and Secure Orchestration and Reconfiguration Plane [Initial]	
Nature	Report (R)	
Dissemination Level	Public	
Contractual Date of Delivery	30.06.2026	
Actual Submission Date	30.06.2026	
Lead Beneficiary	Neotera srl (NEO)	
Lead Author	Othon Tomoutzoglou	
E-Mail	o.tomoutzoglou@neotera.eu	
Contributing Beneficiaries	AVL	Hannes Fuchs
	CON	Anton Florin
	DLR	Gregor Nitsche Patrick Uven Henning Schlender
	IKL	Alejandro J. Calderon Jorge Arboleda Joel Martinez
	KK	Marcus Hähnel
	NEO	Othon Tomoutzoglou
	STM-IT	Giancarlo Asnaghi
	STM-FR	Marcello Coppola
	TAAT	Alexander Kühhas
	ZSC	Ivan Paez
Reviewing Beneficiaries	DLR	Patrick Uven (WPL)
Work Package(s)	WP3	
Related Task(s)	T3.4	Vehicle to Cloud continuum orchestration and deployment implementation
	T3.5	SASE update manager implementation
Keywords	Software-Defined Vehicle, SDV, Horizon Europe, Automotive, CCAM, OTA, Update, Upgrade, DevOps, E/E-Architecture, Orchestration, Vehicle-to-cloud, Cloud-Edge Continuum, Reconfiguration, ORP, SASE, Digital Twin, Containerisation	

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	25.05.2026	Initial version
V0.2	02.06.2026	Updated version after partners feedback
V0.3	24.06.2026	Merged with partners input
V0.4	26.06.2026	Reviewed Version
V0.5	30.06.2026	Review comments included, version for acceptance
V0.6	01.07.2026	Review comments included, version for clearance
V1.0	01.07.2026	Final and reviewed Version

Table of contents

Table of contents.....	5
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. Technical Deliverable-Map.....	12
1.3. Work-Package Outline.....	13
1.3.1. Work-Package Objectives.....	13
1.3.2. Work-Package Description	13
1.3.3. Work-Package Status.....	13
1.4. Purpose and Scope of the Deliverable	14
1.4.1. Relation to D3.2.....	14
2. Work Package and Architecture Overview	15
2.1. WP3 Overall Architecture.....	15
2.2. Boundaries between Dev (T3.2), V&V (T3.3) and Update (T3.4/T3.5).....	17
3. Baseline Architecture	18
3.1. Relation to D3.1 (T3.1) - Requirements.....	18
3.2. T3.4 Contributions.....	19
3.2.1. ContinuumX Cloud Substrate (NEO).....	19
3.2.2. ClOrch Cloud Orchestrator (DLR).....	21
3.2.3. ClOrch - ContinuumX seam	21
3.2.4. Digital-Twin Platform Integration (NEO, STM-FR).....	22
3.2.5. Cloud Computation Offload / vECU Runtime (CON)	22
3.2.6. T3.4 Enabling Partners.....	23
3.3. T3.5 Contributions.....	23
3.3.1. Platform Orchestrator (PIOrch and PIOrCA, DLR).....	23
3.3.2. ContinuumX Edge Components (NEO)	24
3.3.3. Local Deployment and Execution Surface	24
3.3.4. TSN Bridge and PCM Mapper	25
3.3.5. SASE Update-Manager Services	25
3.3.6. Zonal Gateway / Stellar G6 baseline	25

3.4.	Cross-WP Inputs Consumed by D3.3	25
4.	Interface Baseline (M12)	26
4.1.	Interface Register - Summary	26
5.	Process Baseline (M12)	28
5.1.	Process Overview	28
5.2.	Step 1: Inputs from T3.3	28
5.3.	Step 2: Develop Orchestration Solution (T3.4)	29
5.4.	Step 3: Provide continuum context, digital-twin and resource publication (T3.4)	29
5.5.	Step 4: Fleet rollout decision (T3.4)	29
5.6.	Step 5: Platform receives / polls update intent (T3.5)	29
5.7.	Step 6: SASE update-manager checks (T3.5)	29
5.8.	Step 7: Execute local change operations (T3.5)	29
5.9.	Step 8: Zonal firmware path (T3.5)	30
5.10.	Step 9: Monitor and recover (T3.5)	30
6.	Summary and Discussion/Conclusion	30
7.	Outlook	31
	Appendix A - Component List	32

Executive Summary

This document is D3.3 “Safe and Secure Orchestration and Reconfiguration Plane” [Initial], delivered at project month M12 as an initial baseline. It establishes the architecture, interface families, and process structure of the UP2DATE4SDV Orchestration and Reconfiguration Plane (ORP) to a degree sufficient for partner alignment and to inform the parallel WP4 SASE assessment. Concrete protocol schemas, API specifications, runtime contracts, and full implementation evidence are left for M30 maturation.

The ORP baseline integrates two task areas within WP3:

T3.4 (Vehicle-to-Cloud continuum orchestration, NEO-led): the cloud-side ORP layer, implementing the ContinuumX cloud-edge delivery substrate, digital-twin publication, and the ClOrch policy/decision orchestrator (DLR).

T3.5 (SASE update manager, DLR-led): the system-side layer running on the Central ECU and Zonal Gateway, including PlOrch, local execution adapters, SASE checks, and the TSN Bridge / PCM Mapper path to the Stellar G6 firmware memory.

Sixteen (16) interface families (I-1 to I-16) are identified and named at M12. The M12 register captures scope and data-family descriptions. Message schemas, API contracts, and test evidence are M30 targets.

The component registry cross-reference (Appendix A) aligns every baseline component with its CID from the UP2DATE4SDV component registry.

Acronyms and Abbreviations

Abbrev.	Full Name
AD	Autonomous-Driving
AUP	Approved Update Package
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: Overview of the UP2DATE4SDV technical deliverable map.....	12
Figure 3: M12 baseline ORP architecture: M12 baseline ORP architecture.....	17
Figure 4: M12 interface baseline: M12 interface baseline.....	26
Figure 5: M12 process baseline with T3.3/T3.4/T3.5 roles	28

List of Tables

Table 1: The three physical tiers and their components	16
Table 2: List of requirements defined in D3.1 which are relevant for T3.4 & T3.5.	18
Table 3: ContinuumX Cloud substrate components.....	20
Table 4: PIOrch and PIOrchA components.....	23
Table 5: ContinuumX Edge components.....	24
Table 6: Interfaces register	26
Table 7: Component list.....	32

References and Background Material

Internal Background

[1] D5.1 – Demonstrator Definitions, Metrics, and Plan

[2] D3.1 – Requirements & Drafts of the Orchestration and Reconfiguration Plane (ORP)

[3] D3.2 – DevOps and V&V Framework [Initial]

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:

- 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 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.

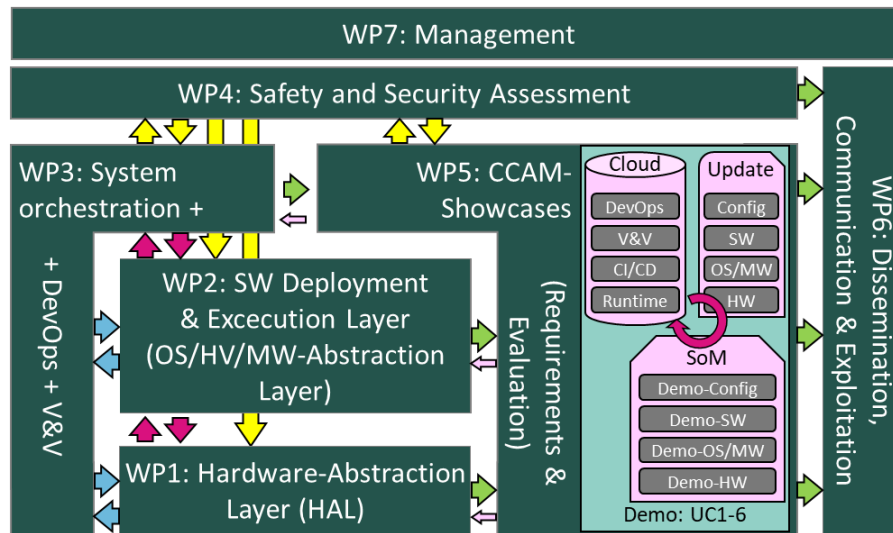


Figure 1: Overview of the UP2DATE4SDV work-packages and their Interaction.

1.2. Technical Deliverable-Map

UP2DATE4SDV has defined 38 deliverables of which 10 have an organizational nature and 28 a technical nature. To provide orientation to the reader, Figure 2 shows a technical deliverable map that gives a structural overview of all technical deliverables and their content-wise relation.

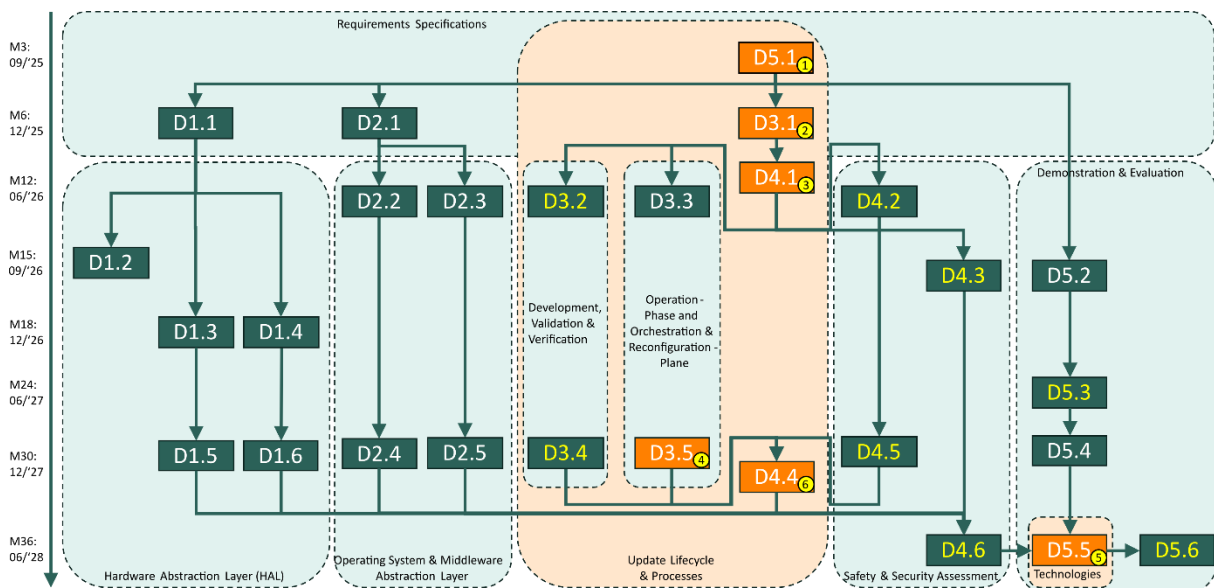


Figure 2: Overview of the UP2DATE4SDV technical deliverable map, showing the deliverables of the technical work-packages WP1-WP5, with restricted deliverables highlighted by the yellow text, a main-track highlighted by the orange nodes, and a recommended reading-order in the yellow circles. The arrows between deliverables are indicating the content development.

Deliverables are shown as rectangles, which are connected by arrows if the content of one deliverable is based on the contents of previous ones. The text in the rectangles shows the deliverable number and it is highlighted in yellow, in case of restricted deliverables, whereas public deliverables are labelled in white. The light-green and light-orange areas form technical content-categories, close but not equal to the work-packages. They show the hardware abstraction layer (HAL) in WP1, the Operating-System and Middleware Abstraction-Layer (OAL) in WP2, the Safety and Security Assessment in WP4, and the Demonstrators and the Evaluation in WP5, but they combine the Orchestration and Reconfiguration-Plane of WP3 with the safety- and security-analysis and corresponding measurements from WP4, forming the area of the Update Lifecycle and Processes, where Development, Validation and Verification, and Orchestration and Deployment of Updates are covered. Furthermore, the four initial deliverables of the work-packages requirements-definitions are also combined in a separate area at the very top. To get an overall impression of the full project, the main story-track of deliverables is highlighted in orange, and a suggested reading-order is indicated by numbers in yellow circles.

1.3. Work-Package Outline

1.3.1. Work-Package Objectives

WP3 provides a central, cloud-integrated Orchestration and Reconfiguration Plane (ORP) that coordinates software updates, hardware upgrades, reconfiguration and task re-allocations at vehicle and fleet level, taking available system- and scenario-knowledge into account. The work package begins with an analysis of requirements (T3.1) and proceeds with the implementation of the design and implementation framework (T3.2), the verification and validation framework (T3.3), the cloud-side orchestration and deployment manager (T3.4) and the system-side update manager (T3.5).

1.3.2. Work-Package Description

The work in WP3 is organized into five tasks spanning months 4 to 30. T3.1 (Continuous integration, orchestration and deployment requirements) compiles the requirements that define the architectures, interfaces and processes for continuously integrating, orchestrating and deploying software updates, hardware upgrades, reconfigurations and dynamic, mode-dependent vehicle-to-cloud and cloud-to-vehicle task re-allocations, including their integration into cloud-based DevOps, CI/CD and V&V; its output is deliverable D3.1. T3.2 (Design and implementation framework) enhances a model-based engineering framework that generates reliable, certifiable code suited to high-performance automotive platforms and aligned with the Operating and Software System Abstraction Layer (OAL), ensuring a smooth transition into the V&V processes. T3.3 (V&V framework) provides the verification-and-validation and DevOps environment, including virtual integration and software-in-the-loop testing, security and compatibility checking, and CI/CD pipelines, culminating in the Approved Update Package (AUP). T3.4 (cloud-side orchestration and deployment) realizes the cloud-side ORP that develops orchestration solutions and manages fleet-level roll-out and computation offloading. T3.5 (system-side update manager) realizes the in-vehicle orchestration that receives, validates and applies updates on the automotive platform. Deliverables D3.2 and D3.3 report the results of these tasks.

1.3.3. Work-Package Status

First partners aligned on the requirements for the ORP (Orchestration and Reconfiguration Plane) and delivered D3.1, which captures the collected requirements and provided an overview about the envisioned process and relationships.

In the previous months more details and content were derived from the alignment on WP5, the research- and CCAM- demonstrators, as well as from WP4 from the work on the SASE framework.

Partners collected components, interfaces and aligned on an overall structure of the whole WP3 content, capturing all parts from model-based engineering, over code generation, the V&V framework, as well as digital twins and finally the parts of the ORP, namely the cloud orchestration and the platform part.

This work concludes into a first delivery of D3.2 “DevOps and V&V Framework [Initial]”, and D3.3 “Safe and Secure Orchestration and Reconfiguration Plane [Initial]” which outlines the planned work, utilized solutions, required to deliver the work package 3 content in the project, as well as a collection of open points.

1.4. Purpose and Scope of the Deliverable

D3.3 [Initial] 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. The M12 baseline herein establishes:

- Baseline ORP architecture: three tiers, component placement, task boundaries.
- M12 interface register: interface families, data-family descriptions. Endpoint contracts target M30.
- M12 process baseline. Sequence diagrams and timing target M30.
- Component cross-reference: every component referred to in the body of this deliverable is consolidated in Appendix A with its owner, location, task assignment, and short role description.

The M30 (Final) revision of D3.3 will provide implementation-complete prototypical software, runtime contracts, and validation evidence.

1.4.1. Relation to D3.2

This deliverable (D3.3) is the counterpart of D3.2 within WP3 and is intended to be read together with it. D3.2 (DevOps and V&V Framework) covers the development of update artefacts (T3.2) and their verification and validation (T3.3); it concludes with the Update Package candidate and, once this passes the V&V gate, the AUP. D3.3 covers the Orchestration and Reconfiguration Plane (ORP), comprising the cloud-side orchestration (T3.4) and the system-side orchestration (T3.5), and focuses on the deployment, roll-out, reconfiguration and offloading of those update packages.

The AUP, together with its V&V evidence and CI/CD trigger, is the main hand-over point from D3.2 to D3.3, exchanged through interfaces I-1, I-2 and I-3. D3.2 concludes with the approved package; D3.3 picks up with its deployment and lifecycle management.

The relation is bidirectional. Where an orchestration operation in T3.4/T3.5 modifies a platform-specific package (for example a software reconfiguration or a computation-offloading decision), the modified package is routed back to the T3.3 V&V framework for re-validation before roll-out, as described in D3.2 and in section 2.2 of this deliverable. The vECU testing capability provided by the vSDV Composer is the shared asset across this boundary, used both for cloud-side V&V (T3.3, REQ-ORP-0016) and as the basis of the runtime vECU offloading addressed here (T3.4, REQ-ORP-0024).

2. Work Package and Architecture Overview

This section orients the reader to Work Package 3 and to the architecture that underpins the present deliverable. It is shared in structure with deliverable D3.2: where D3.2 documents the development (T3.2) and verification-and-validation (T3.3) activities, D3.3 documents the orchestration and reconfiguration plane (T3.4 and T3.5). The two deliverables are intended to be read together, and this section establishes the common ground between them. The status of Work Package 3 and the split of its content across the two deliverables are described in Section 1.3.3.

The section is organized as follows:

- 2.1 presents the overall WP3 architecture, introducing the three-tier deployment topology, cloud, Central ECU, and Zonal Gateway, and the placement of components across it; this architecture is the reference against which the detailed contributions in Section 3 are described.
- 2.2 sets out the boundaries between the three concerns that together form the software lifecycle for software-defined vehicles, development (T3.2), verification and validation (T3.3), and update orchestration and reconfiguration (T3.4 and T3.5), and clarifies which concerns are documented in D3.2 and which in this deliverable.

Taken together, the two subsections provide the context required to read the baseline architecture, the interface baseline, and the process baseline that follow. The orchestration and reconfiguration plane is the focus of this deliverable; the development and V&V activities that produce and qualify the update artefacts consumed by the plane are described in D3.2, and are referred to here only insofar as they define the boundary between the two deliverables.

2.1. WP3 Overall Architecture

The ORP reference architecture is designed in alignment with the evolution of **next-generation E/E architectures towards software-defined vehicles (SDVs)**, as illustrated in Figure 3 and summarized in Table 1. The underlying SDV trend is driven by the decoupling of hardware and software life cycles, the increasing centralization of strategically relevant software functions, and the need to support frequent software updates, secure reconfiguration, and scalable orchestration across heterogeneous vehicle platforms. In this context, OEMs are following different architectural paths with common elements, including **central consolidation, zonal consolidation, safe and secure actuation, and Ethernet-centric architectures**. The ORP addresses these trends by providing a distributed orchestration and deployment layer that can operate across central, zonal, and cloud contexts while supporting secure update, upgrade, and reconfiguration workflows.

The architecture in Figure 2 is structured as a **three-tier distributed system** comprising **Cloud, Central ECU or central computing, and Zonal Gateway** domains. This decomposition reflects the transition from classical domain-based vehicle control towards a more flexible SDV architecture, where central computing hosts strategic software with frequent updates, while zonal controllers manage local functions and reduce wiring complexity. The tier-to-component mapping in Table 1 identifies the ORP building blocks deployed at each tier, and the detailed component descriptions in the following subsections elaborate these elements based on the consolidated definitions in Appendix A.

At the **Cloud tier**, ORP functions are executed on cloud infrastructure, either in a **public Internet cloud** or in a **local Internet-cloud equivalent** used for development, integration, and validation. In SDV terms, this tier supports the consolidation of update intelligence, fleet-level orchestration, and cross-

vehicle deployment planning. The cloud hosts the highest-level orchestration and composition services, including **CIOrch (CID 40)**, the **ContinuumX cloud tier (CIDs 90–93)**, the **vSDV Composer (CID 83)**, the **Cloud SASE Library (CID 30)**, the **Eclipse Zenoh cloud tier (CID 81)**, and the **data-driven orchestration service (CID 82)**, as listed in Table 1. These components realise the cloud-side control plane for secure coordination, deployment planning, and data-informed reconfiguration decisions.

The **Central ECU tier** is deployed on **STM32 MP2, Raspberry Pi 4, and NVIDIA Orin** platforms based on the **Arm Cortex-A class**. This tier represents the vehicle-side central computing layer that is characteristic of modern SDV architectures and is responsible for bridging cloud-level orchestration with zonal execution. The Central ECU hosts the main orchestration and deployment mediation functions, including **PIOrch (CID 50)** and the corresponding **PIOrCA adapters for Podman (CID 51)** and **L4Re (CID 52)**, the **ContinuumX edge tier (CIDs 94–96)**, the **TSN Bridge (CID 71)**, **L4Re OS and VMM (CIDs 76 and 78)**, **native L4Re network update support (CID 77)**, the **Eclipse Zenoh vehicle tier (CID 85)**, the **micro-services deployment manager (CID 88)**, the **SASE libraries (CID 84)**, the **seL4 hypervisor (CID 70)**, and the **TSN Driver**, all of which are referenced in Table 2. This tier provides the execution substrate for orchestrating updates and reconfigurations while preserving separation between strategic software control and safety-relevant real-time functions.

The **Zonal Gateway tier** is implemented on **STM32 Stellar G6** devices based on the **Arm Cortex-R class** and corresponds to the **zonal consolidation** and **safe & secure actuation** principles. It hosts the localized execution and mediation components required at the edge of the vehicle network, namely the **PCM Mapper (CID 72, STM-IT)**, **SASE mechanisms (CID 84, IKL)**, the **TSN Driver**, and **L4Re (CID 76, KK)**, as summarized in Table 2. In this tier, safety-relevant actuation remains separated from strategic software, and all updates or reconfiguration actions are mediated via the Central ECU rather than via direct cloud access. This hierarchy supports the incremental integration of zonal controllers and physical proximity-based functions while maintaining robust control over safety-critical pathways.

The resulting ORP architecture is therefore consistent with the broader SDV direction towards **Ethernet-centric architectures**, higher configuration flexibility, and a clearer separation between computation, communication, and actuation. By structuring ORP across cloud, central, and zonal tiers, the architecture enables controlled propagation of software updates, hardware upgrades, and dynamic reconfigurations while supporting scalable deployment across heterogeneous E/E platforms. The corresponding tier-specific component allocation is summarized in Table 1, which serves as the architectural reference for the ORP implementation described in the later sections.

Table 1: The three physical tiers and their components

Tier	Hardware Platform	Primary ORP Components
Cloud	Cloud infrastructure (Real Internet Cloud or Local “Internet Cloud”)	CIOrch (CID 40); ContinuumX cloud tier (CIDs 90-93); vSDV Composer (CID 83); Cloud SASE Library (CID 30); Eclipse Zenoh cloud tier (CID 81); data-driven orchestration (CID 82)
Central ECU	STM32 MP2, PI 4, NVIDIA Orins (Arm Cortex-A series)	PIOrch (CID 50) and PIOrCA adapters for Podman (CID 51) and L4Re (CID 52), all DLR; ContinuumX edge tier (CIDs 94-96); TSN Bridge (CID 71); L4Re OS and VMM (CIDs 76, 78); native L4Re network update support (CID 77); Eclipse Zenoh vehicle tier (CID 85); micro-services deployment manager (CID 88); SASE libraries (CID 84); seL4 hypervisor (CID 70).
Zonal Gateway	STM32 Stellar G6 (Arm Cortex-R series)	PCM Mapper (CID 72, STM-IT); SASE Mechanisms (CID 84, IKL); L4Re (CID 76, KK). No direct cloud access; all updates mediated via Central ECU.

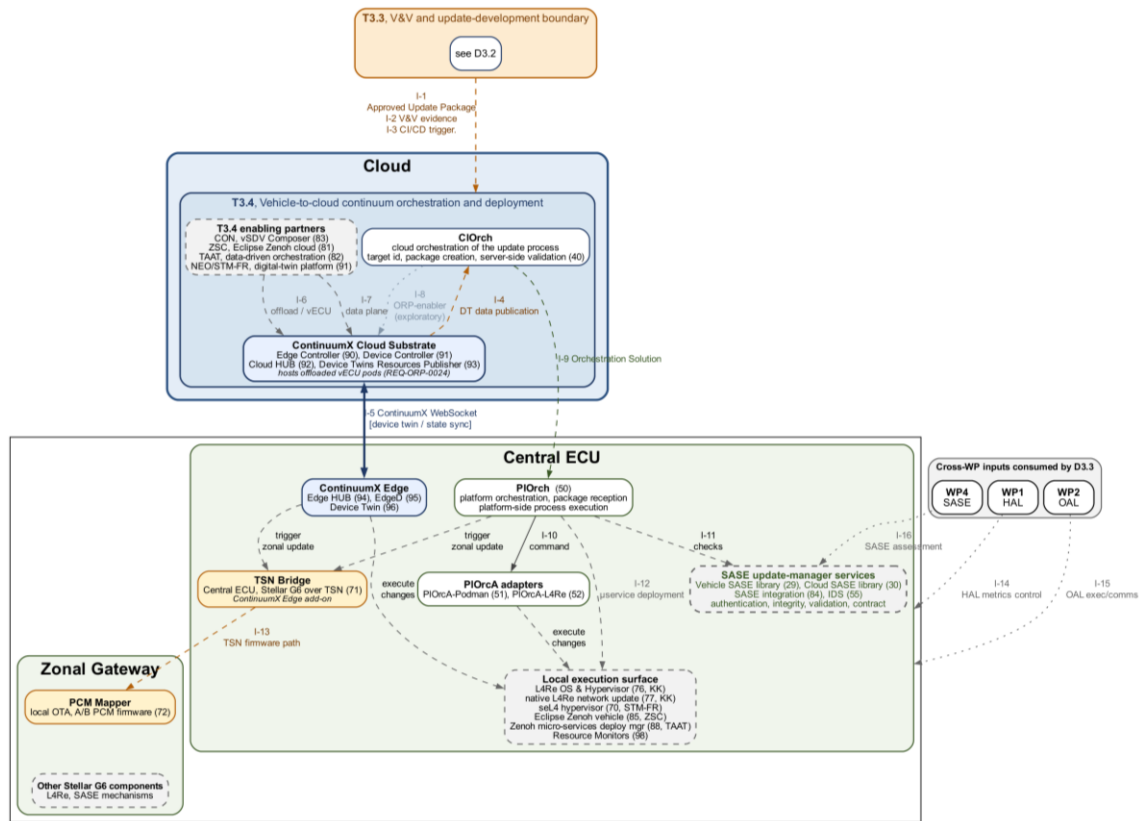


Figure 3: M12 baseline ORP architecture: M12 baseline ORP architecture

2.2. Boundaries between Dev (T3.2), V&V (T3.3) and Update (T3.4/T3.5)

WP3 spans three concerns that together form the software lifecycle for software-defined vehicles: development (T3.2), verification and validation (T3.3), and update orchestration and reconfiguration (T3.4 and T3.5). Development and V&V produce and qualify the update artefacts and are documented in deliverable D3.2; the present deliverable (D3.3) covers the update orchestration and reconfiguration plane (T3.4 cloud-side and T3.5 system-side). Consistent with D3.2, an Update Package candidate is produced and validated in T3.2/T3.3; once it passes the V&V gate it becomes the AUP, which is the hand-over point to this deliverable, together with its V&V evidence and CI/CD trigger, exchanged through interfaces I-1, I-2 and I-3 (see Section 4). The remainder of this subsection summarizes that boundary; the development and V&V activities themselves are described in D3.2.

T3.3 is the V&V and update-development boundary. Its full description, component list, and implementation details are covered in deliverable D3.2 (DevOps and V&V Framework [Initial]). For the purposes of this deliverable, T3.3 is represented by the three interface families through which the AUP enters D3.3: I-1 (update package), I-2 (V&V evidence), and I-3 (CI/CD trigger). The relation between T3.3 and the update tasks is not only forward. The V&V flow can also be entered from the update side: T3.4 may trigger re-validation of a platform-specific update package after it has been modified, so that the modified package passes again through the T3.3 virtual-testing gate before it is deployed.

Within this deliverable, this backward relation applies where an orchestration operation changes a platform-specific package: a software reconfiguration or a computation-offloading decision (T3.4) that alters the deployed software produces a modified package that must be re-validated through the T3.3 V&V framework (D3.2) before roll-out. In such cases the AUP consumed over interfaces I-1, I-2 and I-3 is the re-validated result of that loop, not only a first-pass package.

The vECU testing capability provided by the vSDV Composer (CON) is the shared asset across this boundary: the same virtual-ECU environment used for cloud-side V&V in T3.3 (REQ-ORP-0016) underpins the runtime vECU offloading addressed in T3.4 (REQ-ORP-0024). The T3.4 contributions that build on this capability are described in Section 3.2.5.

3. Baseline Architecture

3.1. Relation to D3.1 (T3.1) - Requirements

In D3.1, requirements for the Orchestration and Reconfiguration Plane (ORP) were defined. Those requirements address different parts of WP3. The requirements relevant for T3.4 and T3.5, the cloud-side and system-side orchestration, deployment, reconfiguration and offloading, are listed in the table below. The requirements relevant for T3.2 and T3.3 are listed in D3.2.

Table 2: List of requirements defined in D3.1 which are relevant for T3.4 & T3.5.

ID	Requirement
REQ-ORP-0001	The ORP must be able to update software components in SDVs (called platforms).
REQ-ORP-0003	The platform must be able to download an update package and save it locally.
REQ-ORP-0004	The platform must have a managed database for downloaded updates.
REQ-ORP-0005	The cloud must have a managed database for tested updates.
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-0009	The ORP must have a central component on the cloud side to orchestrate the cloud-side part of the update process.
REQ-ORP-0010	The platform must have a central component on the platform side to orchestrate the platform-side part of the update process.
REQ-ORP-0011	The cloud-based orchestrator and the platform-based orchestrator must be able to communicate to inform of new updates, request available updates and provide update packages.
REQ-ORP-0014	The cloud side must be able to manage the access to the Approved Updates Database.
REQ-ORP-0015	The cloud side must be able to manage the update rollout over a fleet of SDVs.
REQ-ORP-0017	The platform must be able to rollback to a stable version after a failed update process.
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-0028	The platform must have interfaces to the Software Application Deployment, Separation and Execution Layer.
REQ-ORP-0030	The platform must be able to detect that a new configuration is available.
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-0032	The cloud side must maintain a database with information regarding the updates already done and which are not done yet.
REQ-ORP-0033	The platform must alert the end user if, due to reconfiguration, previous functionality is lost or altered.

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 decisions.
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.
REQ-ORP-0040	The cloud-based and platform-based orchestrators shall implement a secure handshake protocol and mutual authentication for update notifications and device-twin synchronisation.
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-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-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-0047	The vehicle's edge platform is running with active connections to the cloud and multiple workloads deployed; resource demand or workload type changes are handled.
REQ-ORP-0048	The cloud orchestration must get informed about vehicle hardware configuration changes (e.g., LIDAR sensor) and trigger the deployment of necessary software.

3.2. T3.4 Contributions

T3.4 implements the cloud-side ORP as a middleware extension coordinating resource-composable orchestration and deployment of containers and communication channels, and integrates a digital-twin platform that publishes available resources via device twins. At the M12 baseline, the cloud orchestration substrate of the ORP is composed of two tightly coupled but conceptually distinct parts: ContinuumX, provided by NEO as the edge-cloud device-management and container-orchestration backbone, and ClOrch, provided by DLR as the cloud orchestration core. Together, these components implement the cloud-side execution environment for orchestration, deployment, and update management within UP2DATE4SDV. The functional interaction between both elements is governed by the proposed ClOrch - ContinuumX seam and the associated interface contracts.

3.2.1. ContinuumX Cloud Substrate (NEO)

ContinuumX is Neotera's native containerized application orchestration and device-management platform for hosts at the Edge, based on the CNCF project KubeEdge.

Within the UP2DATE4SDV project, ContinuumX constitutes the foundational technical contribution provided by NEO to the orchestration and reconfiguration plane. It will be extended and augmented throughout the project to fulfil the specific functional and non-functional requirements arising from the SDV paradigm. ContinuumX treats each SDV as a **Kubernetes-managed edge node**. The onboard Central ECU is registered within the ContinuumX control plane, enabling the deployment, monitoring, and full lifecycle management of vehicle software functions as containerised workloads executing

directly on the vehicle. This design enforces a clear separation between latency-sensitive and cloud-dependent concerns: safety- and latency-critical functions — including sensor fusion, perception pipelines, and vehicle motion control — execute entirely on the SDV without dependency on cloud connectivity, while non-latency-critical services such as route optimisation, telemetry aggregation, and OTA update orchestration are synchronised with the cloud tier when connectivity is available.

A defining characteristic of the SDV architecture is the ability to **dynamically deploy, update, and reconfigure vehicle software functions** across the full vehicle lifecycle — at the factory, at the dealership, and in the field. ContinuumX supports this capability through CIOrch (CID 40). The **DeviceTwin** mechanism maintains a continuously synchronised digital representation of each onboard ECU, sensor, and actuator, enabling remote reconfiguration and desired-state enforcement from the cloud control plane. This directly supports the SDV concept of a **persistent digital thread** between the vehicle and its backend services, allowing functional updates to be validated, deployed, and monitored at fleet scale. The device-twin runtime will be jointly modified and augmented by NEO and STM-FR as part of the collaborative development activities foreseen within the project.

At the M12 baseline, the cloud-tier components of ContinuumX are described in Table 3 and comprise four principal elements:

- **ContinuumX Edge Controller (CID 90)** — responsible for synchronising pod, ConfigMap, and Secret changes from the cloud control plane to the edge nodes, while propagating edge-side status and events back to the cloud, ensuring continuous reconciliation between the desired and observed state of each vehicle node.
- **ContinuumX Device Controller (CID 91)** — manages device metadata and device status as Kubernetes Custom Resources, ensuring consistent synchronisation of device-related updates between the cloud and the edge tier.
- **ContinuumX Cloud HUB (CID 92)** — constitutes the cloud-side endpoint of the WebSocket communication channel to the edge, acting as the routing point between the cloud-side controllers and the Edge HUB. It thereby provides one endpoint of **Interface I-5** as defined in the UP2DATE4SDV system architecture.
- **ContinuumX Device Twins Resources Publisher (CID 93)** — a new component to be developed by NEO within Task T3.4. It is responsible for publishing device-twin resources and associated metadata to authorised consumers across the system, providing the resource-context publication endpoint of **Interface I-4**.

Table 3: ContinuumX Cloud substrate components

CID	Component	Role in T3.4
90	ContinuumX - Edge Controller	Synchronizes pod, ConfigMap and secret changes from the cloud to edge nodes and propagates edge-side status and events back to the cloud control plane.
91	ContinuumX - Device Controller	Cloud-side device-management component representing device metadata and status as Kubernetes Custom Resources, and synchronising device updates between cloud and edge.
92	ContinuumX - Cloud HUB	Cloud-side endpoint of the WebSocket channel to the edge; routes messages between the cloud controllers and the Edge HUB. One endpoint of Interface I-5.
93	ContinuumX - Device Twins Resources Publisher	New component to be developed by NEO in T3.4. Publishes device-twin resources and metadata to interested consumers, providing the resource-context publication endpoint of Interface I-4.

3.2.2. ClOrch Cloud Orchestrator (DLR)

ClOrch (CID 40) is DLR's Cloud Orchestrator Core. Its scope at this baseline covers cloud orchestration of the update process, including target identification, update-package creation, server-side validation, and communication toward the platform orchestrator.

ClOrch is tasked with managing the deployment of software modules on all connected vehicle platforms. Each platform's hardware/software layout can be described using an appropriate configuration format. These configurations shall then be parsed by ClOrch to generate a series of steps required in order to reach the state described in the configuration. This series of steps is what we call an orchestration solution and needs to be delivered to the vehicle by the rollout manager. Along with the orchestration solution, a vehicle platform shall also be delivered a simplified version of the platform configuration, stripped of deployment metadata which can be used to validate the platform's state against.

A platform configuration should usually be obtained from the model-based update checker from T3.3. The process of validating a model and submitting the resulting desired platform configuration to ClOrch should be able to be triggered in several ways, e.g. by user interaction when provisioning a new platform but ClOrch should also have the ability to autonomously trigger this process when a module update becomes available or when sanity checks fail.

3.2.3. ClOrch - ContinuumX seam

It is useful to distinguish three layers when relating ClOrch and ContinuumX. What is orchestrated is the Orchestration Solution defined in D3.1: a package of change instructions covering one or more of the four change operations: software update, hardware upgrade, software reconfiguration and software offloading.

The decision and sequencing layer develops and drives that Orchestration Solution: ClOrch on the cloud side and PlOrch on the system side decide the targets, sequence the roll-out and gate on V&V. This is the orchestration authority of the ORP, and it is in line with the project requirements.

The execution layer carries the resulting change operations out. ContinuumX is one of the execution substrates of this layer, alongside the platform adapters (PlOrchA), the TSN Bridge and the PCM Mapper. ContinuumX natively contains container and workload-lifecycle orchestration; within UP2DATE4SDV this native capability is an execution mechanism subordinate to ClOrch's decisions, not an independent ORP decision layer.

ClOrch operates as the self-sufficient orchestration authority of the ORP. Its pipeline, platform configuration from the T3.3 model-based update checker, Orchestration Solution development, fleet rollout via the Rollout Manager, delivery to PlOrch over Interface I-9, does not depend on ContinuumX.

Within the ORP role model, ContinuumX contributes two enabler functions: (a) the digital-twin data path, device state, attributes and resource context collected by the ContinuumX edge components (CIDs 94-96), synchronised to the cloud (CIDs 90-92), and published to interested consumers by the Device Twins Resources Publisher (CID 93); and (b) the TSN Bridge (CID 71), positioned as a ContinuumX Edge add-on enabling the Central ECU to Stellar G6 firmware path (Interface I-13).

ContinuumX, natively contains container orchestration and workload-lifecycle capability that functionally overlaps with cloud orchestration. In UP2DATE4SDV the policy and decision authority of the ORP rests with ClOrch, in line with the project requirements; ContinuumX's native orchestration capability is not used as the ORP decision layer.

ContinuumX depends on ClOrch for any ORP policy or decision, it takes no autonomous rollout decisions within the ORP. Conversely, ClOrch is the primary candidate consumer of the ContinuumX

digital-twin data via Interface I-4. Other components may subscribe to the same publications as consumers.

A potential ORP-enabler interface (I-8) through which CIOrch's policy/decision layer could drive the ContinuumX workload-lifecycle capability, could make ContinuumX available as an additional ORP execution substrate. This option is recorded here as an exploration the partners are interested in pursuing. Deployment directives include orchestration solution plus a description of the target platform state. The platform description can be annotated with device state and resource context for reporting back to CIOrch.

Realizing I-8 would require a translation module, provisionally the "ContinuumX ORP Adapter", analogous in pattern to the PIOrcA adapters (CIDs 51, 52), that maps CIOrch orchestration decisions onto ContinuumX Kubernetes-native scheduling primitives (node placement, resource constraints, rollout / rollback strategy, and transactional application of an orchestration solution). This module is recorded only as a prerequisite should I-8 be pursued.

CIOrch expects communication via the zenoh protocol. Currently it is unclear how feasible the adaption to a different protocol is.

3.2.4. Digital-Twin Platform Integration (NEO, STM-FR)

In parallel to the Cloud Orchestrator and ContinuumX integration, NEO and STM-FR co-develop the digital-twin platform integration within T3.4. The platform exposes available resources through device twin publisher (CID 93) to interested consumers. ContinuumX provides the cloud-edge integration backbone (CIDs 90-92) and supports the liquid computing capability, enabling seamless management of workloads and data across the edge-cloud continuum.

This digital-twin integration is designed to complement the PCM Mapper at the Zonal Gateway level and to support real-time software update operations with minimal or no downtime where technically feasible. It also provides the contextual basis for orchestration decisions, since cloud-side planning and deployment can exploit resource availability, platform state, and runtime context derived from the twin representation

3.2.5. Cloud Computation Offload / vECU Runtime (CON)

Cloud computation offloading addresses the case described in D3.1 Section 2.3.1, where the target platform for one or more functions does not exist on the vehicle but is instantiated in the cloud, for example when on-board resources are insufficient following a fault, or as a deliberate fleet strategy. A virtual ECU (vECU) representing the vehicle, or a part of it, is created and operated in the cloud.

The vECU is used in two complementary roles. (a) As a virtual test target for cloud-side V&V (REQ-ORP-0016); this is the simulation / virtual-testing role, carried out in the T3.3 DevOps and V&V framework using CON's vSDV Composer and the CI/CD pipeline. (b) As a cloud-hosted execution instance that hosts vehicle functions on vECUs (REQ-ORP-0024). The present section concerns role (b), the runtime offload path within T3.4; role (a) is documented in D3.2.

vSDV Composer (CON, CID 83) is the authoring, integration and simulation environment for the virtual ECU, built on the CAEdge virtual workbench, and is the basis for the vECU V&V role (a). For the runtime offload role (b), CON provides the deployable vECU artefacts: consistent with the vSDV Composer "Bring Your Own Payload" (BYOP) model described in D3.2, these are containerized partition images where the workload can be containerized, and VM-based instances (e.g. AMI-based partitions extending the vECU concept to EC2 VM instances) where full operating-system fidelity is required. These artefacts are handed to the cloud substrate for offloaded execution.

ContinuumX (NEO) provides the cloud substrate that deploys and executes the containerized vECU artefacts supplied by CON, addressing REQ-ORP-0024 together with the publication of available resources via device twins (CID 93). It hosts and operates the offloaded vECU pods but holds no orchestration decision authority. The decision on whether and what to offload would rest with the cloud orchestration authority (CIOrch, DLR), in line with the Orchestration Solution it develops.

Interface I-6 (Cloud Offload / vECU) is the interface family for this scenario, covering the offloaded-workload context and the associated run-control and health signalling between CIOrch, the vSDV Composer / vECU source, and the ContinuumX hosting substrate.

A cloud-side vECU may consume the published device-twin and resource data (CID 93) to parameterize its execution and to maintain a cloud-side digital twin of the vehicle or a part of it. The corresponding data exchange is carried by Interface I-4.

3.2.6.T3.4 Enabling Partners

Besides CON, DLR, NEO and STM-FR the following partners contribute to T3.4:

- TAAT: data-driven orchestration and task execution; the relevant component is CID 82.
- ZSC: reliable on/off-loading via dependable deterministic communications; Eclipse Zenoh (CID 81) is the relevant ZSC component. Additional details about the communication plane can be found in D2.3.

3.3. T3.5 Contributions

T3.5 develops 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. It is led by DLR; contributing partners include CON, IKL, KK, NEO, STM-IT, TAAT, ZSC. STM-FR also contributes to the T3.5 execution platform through the sel4 Hypervisor port on the STM32 MP2 Central ECU (CID 70, WP1 / T1.4).

3.3.1.Platform Orchestrator (PIOrch and PIOrcA, DLR)

The system-side platform orchestration components owned by DLR comprise:

Table 4: PIOrch and PIOrchA components

CID	Components	Role in T3.5
50	PIOrch - Platform Orchestrator	Platform Orchestrator Core, responsible for platform orchestration of the update process: package reception, processing, and platform-side process execution.
51	PIOrcA for Linux Containers / Podman	Adapter component enabling PIOrch to apply changes to Linux containers via Podman.
52	PIOrcA for L4Re VMs/Containers	Adapter component enabling PIOrch to apply changes to applications, containers and virtual machines hosted on L4Re.

The platform orchestrator is tasked with executing an orchestration solution provided by CIOrch in order to reach a desired system state in a safe manner while also providing monitoring feedback to CIOrch during the update as well as during operation. The orchestrator itself does not make any changes to the system state. Instead it uses a network of platform orchestration agents (PIOrcAs)

which it instructs to apply desired changes. Communication between PIOrch/PIOrCAs is implemented using the Zenoh protocol.

PIOrch polls some state information about hosted modules from agents but must also rely on monitoring data provided by monitoring infrastructure (see D2.3) in order to identify safe states for the execution of update operations.

Overall sanity checks of a proposed updates must be performed by the orchestrator and missing software modules must be provisioned from the rollout manager. For the execution of update operations, the update agents provide an interface to the orchestrator. The current implementation mandates a unified interface for all PIOrCAs for addition/removal of software packages and the starting/stopping of services. This can however require more processing logic in PIOrCAs than what is desired and may be subject to change.

The prototype *Podman* PIOrCA is implemented as a Linux application, requiring access to the Podman socket for gathering container information, importing/removing container images and starting/stopping/removing containers. As the specification is largely shared, this implementation also has a decent chance of working with Docker sockets without further modifications. A Podman PIOrCA aims to be a relatively direct bridge from zenoh to Podman socket. This does come with the caveat of Podman itself handling much of the processing of a newly added container image, giving up a degree of control and making this mechanism unsuitable for updates which have hard real-time constraints.

3.3.2. ContinuumX Edge Components (NEO)

ContinuumX also includes edge-side components running on the Central ECU. At the M12 baseline these comprise:

Table 5: ContinuumX Edge components

CID	Component	Role at M12 (full description in Appendix A)
94	ContinuumX - Edge HUB	Edge-side endpoint of the WebSocket channel to Cloud HUB. Routes messages between cloud and edge modules, maintains the cloud connection, and reports edge-host and device status changes. One endpoint of Interface I-5.
95	ContinuumX - EdgeD	Manages pod lifecycles and runs containerised workloads on the edge node: pod-change handling, health checks, secrets and ConfigMaps, volumes, container-runtime interaction, garbage collection, and status reporting to the cloud.
96	ContinuumX - Device Twin	Manages local device state, attributes and twin data; links devices to the edge node, stores device information locally, synchronises status and twin updates with the cloud, and offers query interfaces to applications.

At this baseline, the workload/pod-lifecycle traffic inside ContinuumX is not an ORP concern; it would become one only under the potential ORP-enabler interface, whereby ContinuumX would leverage the CIOrch policy/decision layer. The TSN Bridge (CID 71) is positioned as a ContinuumX Edge add-on.

3.3.3. Local Deployment and Execution Surface

The Central ECU local deployment and execution surface comprises the following components (full descriptions in Appendix A):

- L4Re OS and Hypervisor (CID 76, KK): provides safe and secure isolation of virtual machines and native applications, with a POSIX-compatible programming environment.

- L4Re Virtual Machine Monitor (CID 78, KK): provides execution profiling capabilities for bottleneck identification.
- Native L4Re network update support (CID 77, KK): supports dynamic loading of components and configurations from the network, including restarting a virtual machine into an updated image (A/B booting).
- seL4 Hypervisor (CID 70, STM-FR): port of the seL4 microkernel to the STM32 MP2 platform, providing a base virtualisation layer present on the Central ECU. seL4 is a WP1 / T1.4 contribution.
- Eclipse Zenoh: vehicle tier (CID 85, ZSC): communication substrate for the vehicle-side update-management and on-demand services. Additional details about the communication plane can be found in D2.3.
- Zenoh-based micro-services deployment manager (CID 88, TAAT): runtime execution and deployment management for software components on the embedded target, covering pre-deployment, deployment, and post-deployment monitoring with rollback on anomaly.
- Resource Monitors (CID 98, DLR): parse observer data and deliver verdicts on specification adherence.

3.3.4. TSN Bridge and PCM Mapper

Two components implement the firmware-update path from the Central ECU to the Zonal Gateway (e.g., Stellar G6):

CID	Component	Owner	Role at M12 (full description in Appendix A)
71	TSN Bridge (Central ECU side)	NEO	Central ECU component facilitating the transfer of update packages to the Stellar Zonal Gateway over TSN, enabling STM OTA functionality for the PCM memory. Controlled by the vehicle's edge platform.
72	PCM Mapper (Stellar G6 side)	STM-IT	Zonal Gateway component mediating communication with the MCU PCM memory and providing the local OTA functionality on the Stellar G6.

3.3.5. SASE Update-Manager Services

The IKL-owned components in the T3.5 / WP4 SASE scope are:

- Vehicle SASE library (CID 29, IKL): vehicle-side safety- and security-mechanism library covering diagnostics, monitoring, encryption, and authentication.
- Cloud SASE library (CID 30, IKL): cloud-side counterpart of CID 29.
- SASE libraries (CID 84, IKL): integration of the SASE library set into the T3.5 update manager.

3.3.6. Zonal Gateway / Stellar G6 baseline

At the M12 baseline, the Zonal Gateway tier hosts the PCM Mapper (CID 72, ST). The Stellar G6 has no direct cloud connectivity and firmware updates are mediated by the Central ECU.

3.4. Cross-WP Inputs Consumed by D3.3

Per Figure 3, the orchestration and reconfiguration plane consumes inputs from WP1 (HAL), WP2 (OAL) and WP4 (SASE). The relevant components are:

- From WP1: seL4 Hypervisor (CID 70, STM-FR); Observability API (CID 73, BSC); Controllability API (CID 74, BSC); VM Resource Observer (CID 8, DLR); Container Resource Observer (CID 9, DLR); IEEE 802.1Qbv scheduling support (CID 99, TAAT).
- From WP2: L4Re OS and Hypervisor (CID 76, KK); Resource Monitors (CID 98, DLR); Eclipse Zenoh (CID 80, ZSC); ROS 2 time-triggered executor (CID 100, TAAT).
- From WP4: Vehicle SASE library (CID 29, IKL); Cloud SASE library (CID 30, IKL). Intrusion Detection System (CID 55, ITI) is T3.3 scope; see D3.2. Its potential integration with the D3.3 SASE monitoring path is a topic for M30 alignment between ITI and the T3.5 partners.

4. Interface Baseline (M12)

Figure 4 shows the M12 interface baseline.

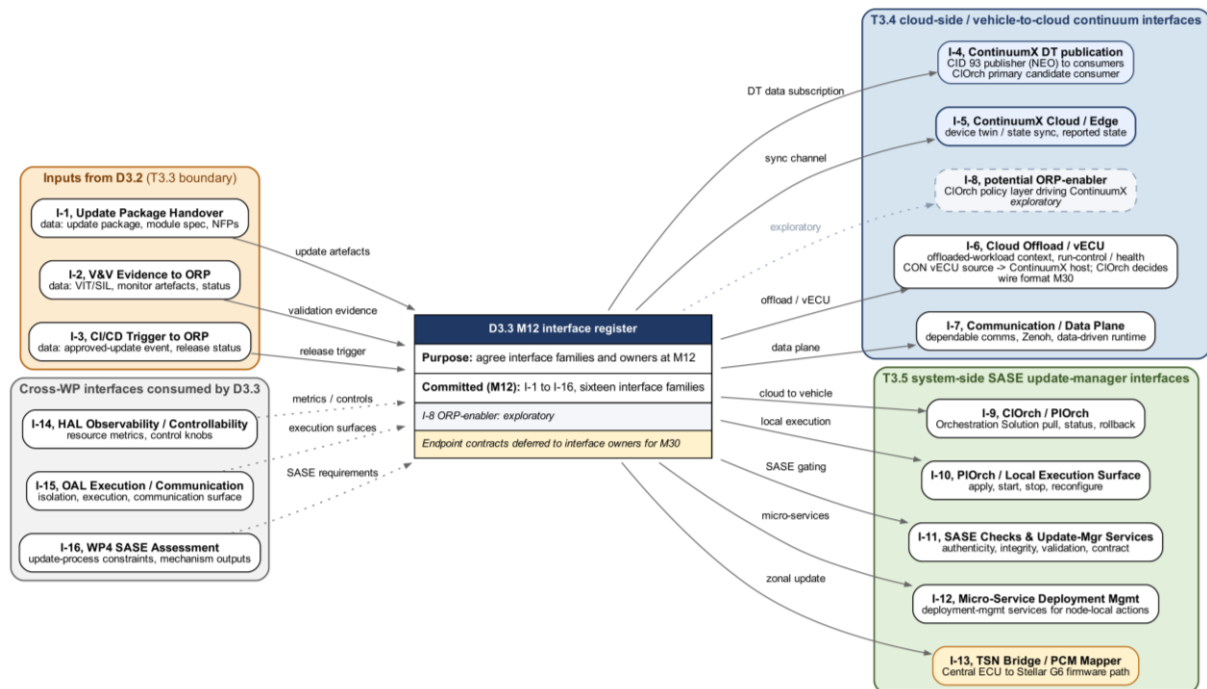


Figure 4: M12 interface baseline: M12 interface baseline

4.1. Interface Register - Summary

The M12 interface register names sixteen (16) interface families (I-1 to I-16) drawn from Figure 4. For each interface the register records the data-family and/or scope. Endpoint contracts, message schemas, APIs, protocols, and evidence, are M30 deliverables to be authored by the owner partners.

Table 6: Interfaces register

ID	Name	Scope	Data family / scope
I-1	Update Package Handover	T3.3 (ORP)	Approved Update Package (the validated Update Package candidate from D3.2), with module specification and non-functional properties (NFPs).

I-2	V&V Evidence to ORP	T3.3 (ORP)	VIT results, monitor artefacts, security/compatibility status.
I-3	CI/CD Trigger to ORP	T3.3 (ORP)	Approved update event / release status.
I-4	ContinuumX -> Digital-twin data consumers	T3.4	Digital-twin data publication / subscription. ContinuumX Device Twins Resources Publisher (CID 93) -> consumers.
I-5	ContinuumX Cloud - Edge	T3.4	Device-twin and state synchronization, reported state. (ContinuumX-internal workload/pod traffic is not an ORP concern at this baseline.)
I-6	Cloud offload / vECU	T3.4	Offloaded-workload context and run-control / health signalling between CIOrch (DLR), the vSDV Composer / vECU source (CON), and the ContinuumX hosting substrate (NEO). Supports the runtime vECU offload path (REQ-ORP-0024).
I-7	Communication / data plane	T3.4	Dependable communication, Zenoh, data-driven runtime.
I-8	CIOrch -> ContinuumX (potential)	T3.4	Potential ORP-enabler: CIOrch policy/decision layer could drive ContinuumX workload lifecycle as an additional ORP execution substrate. Recorded without commitment at M12.
I-9	CIOrch - PIOrch	T3.5	Orchestration Solution delivery: PIOrch continuously polls the cloud for new Orchestration Solutions (poll model); status / failure / rollback notification from PIOrch.
I-10	PIOrch - Local Execution Surface	T3.5	Apply / start / stop / import /remove / reconfigure to PIOrchA-Podman / PIOrchA-L4Re.
I-11	SASE checks and update-manager services	T3.5	Authenticity / integrity / validation, contract services...
I-12	Micro-service deployment management	T3.5	Deployment-management services for node-local actions
I-13	TSN Bridge - PCM Mapper	T3.5	Central ECU to Stellar G6 firmware path.
I-14	HAL Observability / Controllability	Cross-WP	Resource metrics, control knobs...
I-15	OAL execution / communication services	Cross-WP	Isolation, execution, communication surfaces..
I-16	WP4 SASE assessment interface	Cross-WP	Update-process constraints, mechanism outputs.

5. Process Baseline (M12)

Figure 5 shows the M12 process baseline with T3.3, T3.4, and T3.5 roles. The baseline flow is authoritative for M12; implementation sequence, message schemas, and evidence traces mature for M30.

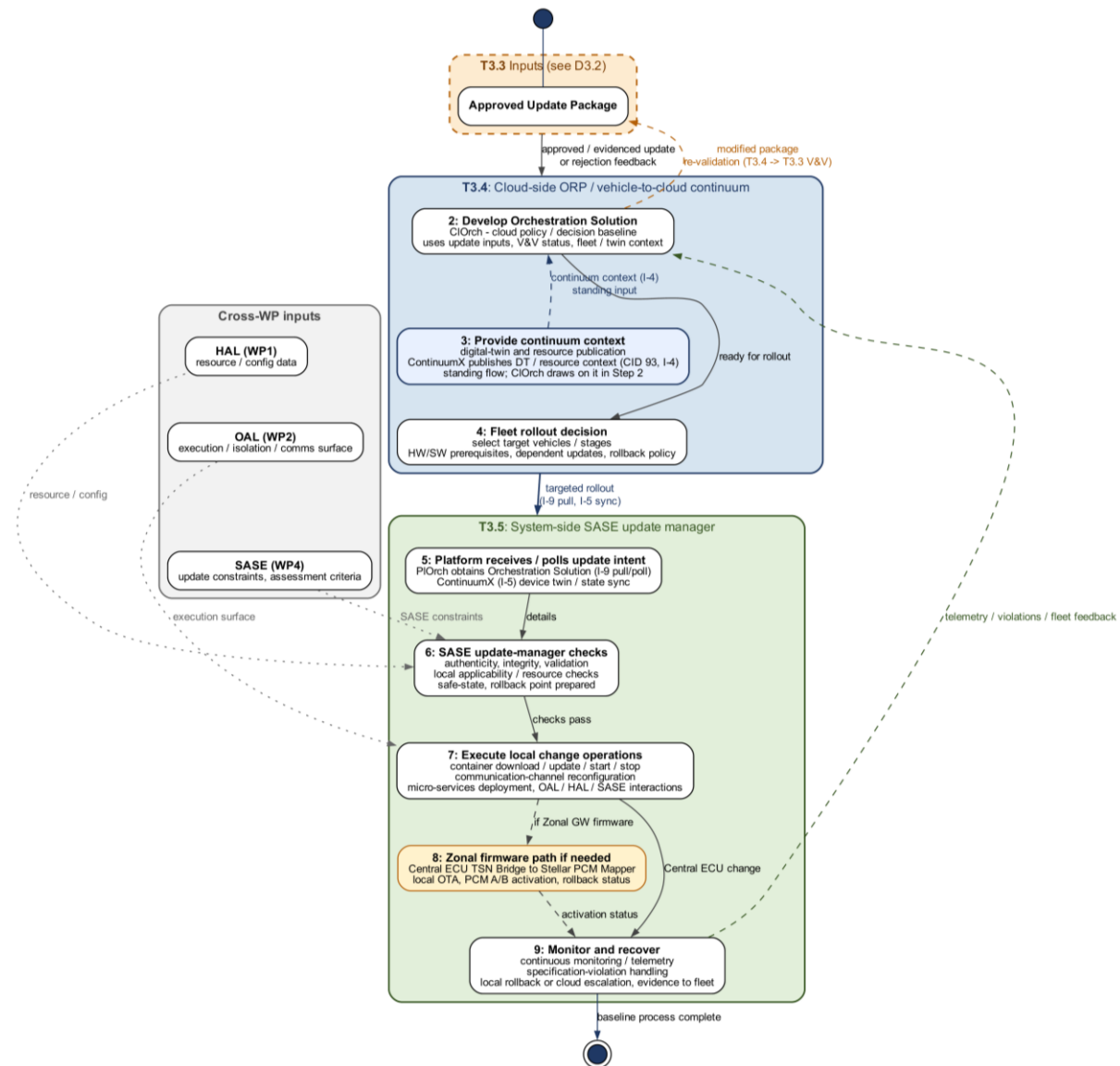


Figure 5: M12 process baseline with T3.3/T3.4/T3.5 roles

5.1. Process Overview

Per Figure 5, the ORP process at M12 consists of nine numbered steps. Step 1 sits in the T3.3 V&V boundary; Steps 2-4 in the T3.4 cloud-side scope; Steps 5-9 in the T3.5 system-side scope.

5.2. Step 1: Inputs from T3.3

Step 1 lies in the T3.3 verification-and-validation boundary and is summarized here only to establish the entry point of the orchestration process. The developer side prepares the update artefacts, the update package, its module specification, non-functional properties, and development evidence, which are then qualified through the V&V and DevOps activities, yielding an approval or rejection

decision. The output of this step is the AUP, handed over to the cloud-side ORP through the interfaces defined in the Interface Baseline.

5.3. Step 2: Develop Orchestration Solution (T3.4)

Triggered by the AUP from the preceding step, ClOrch (the cloud orchestrator) develops the Orchestration Solution that encodes the cloud-side policy and deployment decisions. In doing so, it draws on the update inputs and V&V status together with the continuously published fleet, resource and device-twin context that ContinuumX provides over Interface I-4 (described in the next step).

5.4. Step 3: Provide continuum context, digital-twin and resource publication (T3.4)

The Device Twins Resources Publisher (CID 93) continuously publishes the device-twin and resource context over Interface I-4. This publication is a standing data flow rather than a per-update action: it maintains an up-to-date view of available resources and device state that ClOrch draws on when developing an Orchestration Solution in the previous step. ContinuumX thereby acts as the digital-twin and resource enabler of the cloud-side ORP, while the orchestration decisions themselves remain with ClOrch.

5.5. Step 4: Fleet rollout decision (T3.4)

ClOrch determines how the Orchestration Solution is rolled out across the fleet. It selects the target vehicles, verifies their hardware and software prerequisites, coordinates any dependent updates, and defines the rollback policy to be applied should deployment fail. The outcome is a fleet-level rollout plan that is handed to the system side for execution.

5.6. Step 5: Platform receives / polls update intent (T3.5)

On the system side, PlOrch (the platform orchestrator) obtains the Orchestration Solution from ClOrch over Interface I-9 using a polling model. In parallel, the ContinuumX Edge HUB (CID 94) maintains a WebSocket synchronization channel to the Cloud HUB (CID 92) over Interface I-5, which in the ORP context carries device-twin and state synchronization.

5.7. Step 6: SASE update-manager checks (T3.5)

Before any change is applied, the SASE update-manager services validate the received Orchestration Solution. They verify the authenticity, integrity and validity of the update, check its local applicability against available resources, evaluate the safe-state and SASE constraints, and prepare a rollback point. Only updates that pass these checks proceed to execution.

5.8. Step 7: Execute local change operations (T3.5)

PlOrch, through its platform adapters, carries out the local change operations on the Central ECU. These comprise downloading, updating, starting and stopping the affected containers, reconfiguring communication channels, and managing the deployment of micro-services, in interaction with the OAL, HAL and SASE layers. Operations that target the Zonal Gateway are handled through the dedicated firmware path described in the next step.

5.9. Step 8: Zonal firmware path (T3.5)

Where an update targets the Zonal Gateway, it is applied over the dedicated firmware path: the Central ECU reaches the Stellar Phase-Change-Memory Mapper through the TSN Bridge, which performs the local over-the-air update and the A/B firmware activation. The activation and rollback status is returned to the Central ECU for inclusion in the monitoring evidence.

5.10. Step 9: Monitor and recover (T3.5)

After deployment, the system continuously monitors the updated software and collects telemetry. Specification violations are handled locally where possible, through a rollback to the prepared safe point, or escalated to the cloud when they cannot be resolved on the platform. The resulting evidence is reported back to the fleet and cloud, closing the orchestration loop.

6. Summary and Discussion/Conclusion

This deliverable provided a first view on the architecture, interfaces and processes of the Orchestration and Reconfiguration Plane (ORP) defined in tasks T3.4 and T3.5. The architecture, interface register and process baseline presented here are initial baselines derived from partner alignments and give a preview of the implementation work ahead.

The presented architecture organizes the ORP across three deployment tiers, cloud, Central ECU and Zonal Gateway, and assigns the cloud-side orchestration to ClOrch (T3.4) and the system-side update management to PlOrch (T3.5). Orchestration in the ORP acts on the Orchestration Solution, a package of update, upgrade, reconfiguration and offloading instructions, with ClOrch and PlOrch as the decision authority. ContinuumX provides the cloud-to-edge execution substrate, the device-twin and resource publication, and, for the offloading case, the hosting of vECU workloads; its native workload-lifecycle capability is subordinate to ClOrch's decisions. This separation of a single orchestration authority from the delivery, lifecycle and data-plane functions that execute its decisions is the central design choice of the baseline.

The interface baseline identifies sixteen interface families spanning the cloud-side (T3.4), system-side (T3.5) and the cross-WP inputs consumed by the ORP. At this stage the register records the scope and data families of each interface; the concrete message schemas, API contracts and protocol bindings are to be matured for the final revision. The process baseline describes the end-to-end flow from the AUP through orchestration-solution development, fleet roll-out, platform-side execution and monitoring, including the backward relation through which modified packages are re-validated by the T3.3 V&V framework.

The boundary with the development and verification activities of D3.2 was clarified, with the AUP, its V&V evidence and the CI/CD trigger as the main hand-over point. The shared virtual-ECU capability of the vSDV Composer is the asset linking the V&V role in T3.3 with the runtime offloading addressed here.

Safety and security were not covered in this deliverable; the corresponding mechanisms will be addressed in cooperation with WP4 through the SASE framework. The contents of this document will be refined and updated in line with the developments and learnings in the project, until a comprehensive description of the final ORP is provided in the M30 revision of this deliverable.

7. Outlook

In the next phase of the project the baseline presented here will be taken into implementation, and the partners' contributions will be integrated into a coherent Orchestration and Reconfiguration Plane. The interface families will be developed from scope-level descriptions into concrete message schemas, API contracts and protocol bindings, and the process baseline will be exercised against the project use cases. Open points recorded in this deliverable, among them the assignment of the vECU Cloud Offload Manager role, the wire format of the offload interface, and the exploratory ORP-enabler interface, will be resolved together with the responsible partners and reflected in the final revision at M30.

Appendix A - Component List

This appendix consolidates the components referred to in the body of this deliverable into a single reference table. The component identifiers (“CID”) used here are working labels used uniformly across consortium documents during the WP3 baseline-consolidation phase; they are presented for the convenience of the reader of this deliverable and do not constitute an external or contractual identification scheme.

The component-role summaries in the right-hand column are short descriptions reflecting the role of each component at the M12 baseline; they paraphrase the working descriptions agreed among the contributing partners during baseline preparation. Each owner remains responsible for the authoritative description of its components in subsequent revisions of this deliverable and in its own contributions for M30.

Table 7: Component list

CID	Component Name	Owner	Location	Task	Role at the M12 baseline
Cloud Tier					
40	CIOrch	DLR	Cloud	T3.4	Cloud orchestration of the update process: target identification, update-package creation, server-side validation, and communication toward the platform orchestrator.
90	ContinuumX - Edge Controller	NEO	Cloud	T3.4	Synchronises pod, ConfigMap and secret changes between the Kubernetes API server and edge nodes; relays edge-side status and events back to the cloud control plane.
91	ContinuumX - Device Controller	NEO, STM-FR	Cloud	T3.4	Cloud-side device-management component representing device metadata and status as Kubernetes Custom Resources, and synchronising device updates between cloud and edge.
92	ContinuumX - Cloud HUB	NEO	Cloud	T3.4	Cloud-side endpoint of the WebSocket channel toward the edge; routes messages between cloud controllers and Edge HUB.
93	ContinuumX - Device Twins Resources Publisher	NEO, STM-FR	Cloud	T3.4	New component, to be developed by NEO in T3.4. Publishes device-twin resources and metadata to interested consumers and subscribers.
83	vSDV Composer	CON	Cloud	T3.4	Virtual SDV composer environment supporting validation of the T3.4 middleware extension. Description and detailed scope to be confirmed by CON.
81	Eclipse Zenoh (V2X / cloud tier)	ZSC	Cloud	T3.4	Communication substrate supporting V2X and cloud-tier dependable communication.
82	Data-driven orchestration and task execution	TAAT	Cloud	T3.4	Runtime control mechanism (scheduling- or data-driven) for software-component execution; periodic and event-driven task categorisation.
30	Cloud SASE Library	IKL	Cloud	T4.3	Cloud-side safety- and security-mechanism library covering

					diagnostics, monitoring, encryption, and authentication.
Central ECU Tier					
50	PIOrch	DLR	Vehicle	T3.5	Platform orchestration of the update process: package reception, processing, and platform-side process execution.
51	PIOrchA for Linux Containers / Podman	DLR	Vehicle	T3.5	Adapter component enabling the platform orchestrator to apply changes to Linux containers via Podman.
52	PIOrchA for L4Re VMs / Containers	DLR	Vehicle	T3.5	Adapter component enabling the platform orchestrator to apply changes to applications, containers, and virtual machines on L4Re.
94	ContinuumX - Edge HUB	NEO	Vehicle	T3.5	Edge-side endpoint of the WebSocket channel toward Cloud HUB; routes messages between cloud and edge modules; maintains the cloud connection; reports edge-host and device status changes.
95	ContinuumX - EdgeD	NEO	Vehicle	T3.5	Manages pod lifecycles and runs containerised workloads on the edge node: health checks, secrets, ConfigMaps, volumes, container-runtime interaction, garbage collection, and status reporting.
96	ContinuumX - Device Twin	NEO, STM-FR	Vehicle	T3.5	Manages local device state, attributes, and twin data; links devices to the edge node, synchronises with the cloud, and exposes query interfaces to applications.
71	TSN Bridge (Central ECU side)	NEO	Vehicle	T3.5	Central ECU component facilitating the transfer of update packages to the Stellar Zonal Gateway over TSN, enabling STM OTA functionality for PCM memory. Ownership split with STM-IT to be resolved.
70	seL4 Hypervisor	STM-FR	Vehicle	T1.4	Port and integration of the seL4 microkernel hypervisor for the STM32 MP2 Central ECU; provides a base virtualisation layer on which the wider Central ECU stack is built. WP1 / T1.4 contribution; consumed by the T3.5 execution surface.
76	L4Re OS and Hypervisor	KK	Vehicle	T2.2	Safe and secure isolation of virtual machines and native applications; POSIX-compatible programming environment.
78	L4Re Virtual Machine Monitor	KK	Vehicle	T1.4	VM monitor providing execution profiling for bottleneck identification.
77	Native L4Re Network Update Support	KK	Vehicle	T3.5	Dynamic loading of components and configurations from the network; A/B booting of VM images updated over the network.
85	Eclipse Zenoh (vehicle tier)	ZSC	Vehicle	T3.5	Vehicle-tier communication substrate for update management, on-demand

					services, and runtime data-recovery communication.
88	Zenoh-based micro-services deployment manager	TAAT	Vehicle	T3.5	Runtime execution and deployment management for software components on the embedded target: pre-deployment, deployment, and post-deployment monitoring with rollback on anomaly.
98	Resource Monitors	DLR	Vehicle	T2.2	Parse observer data and deliver verdicts on specification adherence.
29	Vehicle SASE Library	IKL	Vehicle	T4.3	Vehicle-side safety- and security-mechanism library: diagnostics, monitoring, encryption, authentication.
84	SASE Libraries (T3.5 integration)	IKL	Vehicle	T3.5	Integration of the IKL SASE library set into the T3.5 update manager.
8	VM Resource Observer	DLR	Vehicle	T1.4	Collection of resource usage and provisioning at virtual-machine-manager level.
9	Container Resource Observer	DLR	Vehicle	T1.4	Collection of resource usage and provisioning at Linux-container-manager level.
73	Observability API	BSC	Vehicle	T1.4	Low-level API for monitoring performance, resource usage, and memory bandwidth.
74	Controllability API	BSC	Vehicle	T1.3	Low-level API to inspect and set configurations enabling SASE-relevant processes.
79	Zenoh-pico (embedded)	ZSC	Vehicle	T1.4	Lightweight Zenoh implementation suitable for constrained devices.
Zonal Gateway Tier					
72	PCM Mapper (Stellar G6 side)	STM-IT	Vehicle	T3.5	Zonal Gateway component mediating communication with the MCU PCM memory and providing the local OTA functionality on the Stellar G6.