



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

D2.1 Requirements Specification V1.0

Kernkonzept (KK)

31.12.2025



Funded by
the European Union

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

PUBLIC

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

Disclaimer:

Funded by
the European Union

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

All rights reserved.

Deliverable Information		
Deliverable Number	D2.1	
Deliverable Name	Requirements Specification	
Nature	Report (R)	
Dissemination Level	Public	
Contractual Date of Delivery	31.12.2025	
Actual Submission Date	15.01.2026	
Lead Beneficiary	Kernkonzept GmbH (KK)	
Lead Author	Marcus Hähnel	
E-Mail	marcus.haehnel@kernkonzept.com	
Contributing Beneficiaries	NEO	Othon Tomoutzoglou
	ST	Marcello Coppola
	ITI	Joan Josep Valls Mompó
	IKL	Calderón Alejandro Josue
	KK	Marcus Hähnel
	TAAT	Alexander Kühhas
	ZSCALE	Ivan Paez
	DLR	Patrick Uven
	AVL	Stefan Marksteiner
Reviewing Beneficiaries	AVL	Stefan Marksteiner
	AVL	Hannes Fuchs
	CON	Florin Anton
Work Package(s)	2	
Related Task(s)	T2.1	Identification of SW deployment layer requirements and abstraction services
Keywords	Requirements; Software; Updates; Containers; Monitoring; Communication	

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	08.12.2025	Initial version
V0.2	11.12.2025	Update after review by AVL
V0.3	15.12.2025	Add DLR requirements, refactor ITI requirements
V0.4	16.12.2025	Update after KK/DLR/BSC sync for WP1/2/3 coordination
V0.5	18.12.2025	Review comments included, version for acceptance
V0.6	19.12.2025	Review comments from Florin Anton included, version for DLR clearance
V0.7	15.01.2026	Address review comments from DLR; fix layout / structure; Add Summary & Outlook; Add relationship to WP5 / WP3.
V1.0	19.01.2026	Final and reviewed Version

Table of contents

Table of contents.....	5
Executive Summary	6
Acronyms and Abbreviations	7
List of Figures.....	8
List of Tables.....	Error! Bookmark not defined.
Background Material and References	8
1 Introduction	9
1.1 Project Outline	9
1.2 Work-Package Outline.....	10
2 Deliverable introduction	12
2.1 Relationship to WP3 and WP5 Requirements.....	12
2.2 Functional Requirements	12
3 Summary & Outlook.....	19

Executive Summary

This document covers the Milestone 2 (MS2) deliverable D2.1 containing the requirements specification of the software deployment layer and the corresponding abstraction services. The deliverable is a collection of the requirements of individual parts with respect to their software components and documents the synchronized view on those software components and what other partners can expect from them.

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
PCM	Phase Change Memory
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. 10

Background Material and References

Internal Background

- [1] Project Proposal Project Part B: technical alignment (not public)
- [2] D5.1 / Demonstrator Definitions, Metrics, and Plan

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.deli

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 heterogenous 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 (pinkdeli 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.

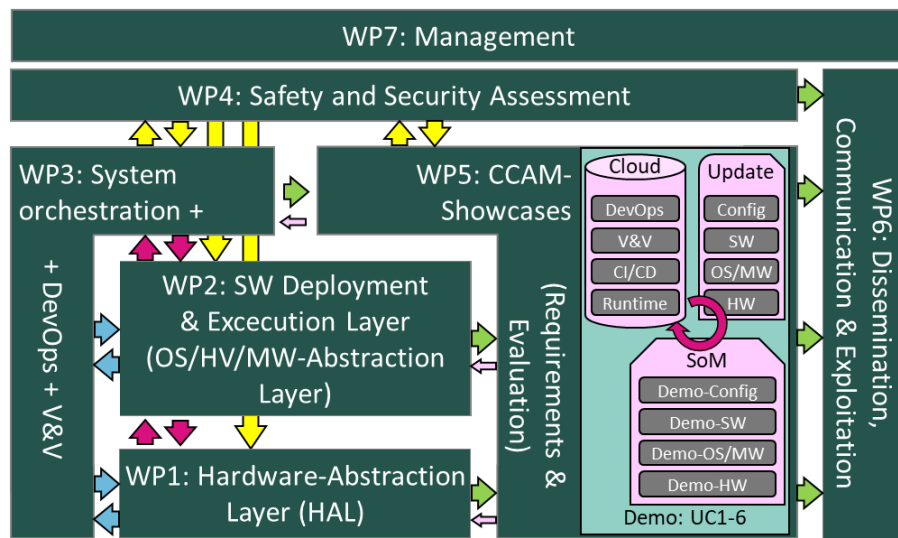


Figure 1: Overview of the UP2DATE4SDV work-packages and their Interaction.

1.2 Work-Package Outline

As shown in Figure 1, WP2 is a central work package with interfaces to most other work packages in the project. In particular, its work items concern the hardware abstraction layer, the container middleware and the reconfigurable and resource controllable service-oriented communication plane. The developments in this work package are the basis of the CCAM-Showcases and the interfaces exposed by the components of WP2 are the knobs used by the orchestration layer from WP3.

1.2.1 Work-Package Objectives

This work package addresses challenges in a highly dynamic and reconfigurable SDV environment, ensuring safe and secure system operation during dynamic updates. The initial task focuses on collecting and analyzing requirements, abstractions, and interfaces for a flexible, individually updatable software and communications stack for future automotive HPC systems. Based on these findings, implementation begins on the container middleware and communication plane. The container middleware will enable container support on next-generation automotive hardware and extend the L4Re Hypervisor and OS framework with native container runtime capabilities. On the communications side, a named-data networking-based middleware will be developed, complemented by resource monitoring components integrated into the OAL to oversee communication interfaces.

1.2.2 Work-Package Description

T2.1: Identification of SW deployment layer requirements and abstraction services

The goal of T2.1 is to elicit the requirements, abstractions and interfaces of execution containers to control, supervise, and individually update parts of the software stack. This includes the definition of

container formats and abstractions levels as well as of safety and security properties, the monitoring and controlling of time and resource consumption, the definition of suitable KPIs, and the definition of requirements for a service-oriented communications plane that can easily adapt to dynamic system reconfiguration. Another goal is the Analysis that will serve as the basis for a model-based security checking approach. For all these items it must be ensured that the proposed solutions are suitable for high-performance devices in general and on the SoCs used in the project in particular.

T2.2: Reconfigurable and resource controllable container middleware implementation

This task supports the implementation of reconfigurable and resource controllable container middleware. The Open Source L4Re Hypervisor and Operating System Framework shall be extended with resource and interface descriptions and a native container runtime format shall be proposed together with the corresponding infrastructure. Other focus points include resource monitoring and matching resource requirements to the SoC as well as executing containers on the STM next generation hardware for automotive to leverage STM32 MP2 and STELLAR capabilities, enhancing performance and scalability for a wide range of applications.

T2.3: Reconfigurable and resource controllable service-oriented communication plane implementation

This task focuses on the implementation of the middleware for the communication plane based on named-data networking for time and resource constrained devices, including the connection to Zenoh Flow and ROS2. Further resource monitoring and observing capabilities to monitor traffic generated and received by each node through the different communication interfaces (e.g. Ethernet, PCI, etc.) are developed. This includes the OS/Middleware Abstraction Layer (OAL)-Integration of the resource observer- and monitor-components. Alignment with the MBE work in WP3 and SASE mechanisms in WP4 shall be ensured.

1.2.3 Work-Package Status

The work package activities started in M4, after the initial kick-off of the project. All tasks of the project are actively worked on at the time of writing, with this deliverable being the final element for Task 2.1. The requirements specified here shall serve as guides for the developments in the other WP2 tasks and, simultaneously, as interface descriptions for other work packages.

1.2.4 Purpose and Scope of the Deliverable

The Operating System and Middleware Abstraction Layer (OAL) represents a foundational component in enabling next-generation Software-Defined Vehicles (SDVs). Its purpose is to abstract hardware complexity, enforce safety and security guarantees, and provide a standardized environment for deploying, managing, and updating software across heterogeneous platforms. As vehicles evolve into highly connected, service-oriented systems, the OAL ensures interoperability, portability, and dynamic adaptability between edge and cloud domains while maintaining compliance with stringent automotive standards such as ISO 26262 and cybersecurity frameworks.

By defining these requirements, the OAL establishes a robust foundation for safe, secure, and efficient orchestration of software components in distributed automotive architectures. This ensures continuity of services, resilience against failures, and future-proof scalability—key enablers for realizing the vision of autonomous, connected, and software-driven mobility.

2 Deliverable introduction

The Operating System and Middleware Abstraction Layer (OAL) represents a foundational component in enabling next-generation Software-Defined Vehicles (SDVs). Its purpose is to abstract hardware complexity, enforce safety and security guarantees, and provide a standardized environment for deploying, managing, and updating software across heterogeneous platforms. As vehicles evolve into highly connected, service-oriented systems, the OAL ensures interoperability, portability, and dynamic adaptability between edge and cloud domains while maintaining compliance with stringent automotive standards such as ISO 26262 and cybersecurity frameworks.

This document outlines the key requirements and contributions of the OAL within the context of an EU-funded research initiative. These requirements address critical aspects such as secure and atomic OTA updates, dynamic resource reallocation across edge-cloud environments, real-time execution of mixed workloads with isolation, and safe temporal and spatial partitioning. Furthermore, the OAL introduces mechanisms for containerized application management, performance monitoring, and service-oriented communication, leveraging protocols like Eclipse Zenoh and DDS over TSN for deterministic data exchange.

By defining these requirements, the OAL establishes a robust foundation for safe, secure, and efficient orchestration of software components in distributed automotive architectures. This ensures continuity of services, resilience against failures, and future-proof scalability—key enablers for realizing the vision of autonomous, connected, and software-driven mobility.

2.1 Relationship to WP3 and WP5 Requirements

These Requirements and the design of the OAL also interface with WP3 and WP5. The base mechanism the orchestration and reconfiguration plane (ORP) from WP3 uses to run software, to monitor the platform and to configure the system are provided by WP2 as the foundational hardware abstraction layer.

The hypervisor is a building block for WP5 by providing secure & safe isolation of workloads as well as individually updateable and restartable workloads. The communication middleware (ROS2 / Zenoh) is used by the orchestrator to provide communication.

2.2 Functional Requirements

RTOS Running Mixed Workloads with Isolation and Real-Time Guarantees	
Requirement ID	WP2_ITI_001
Contributing Partner	ITI
Use Cases	UC-6
KPIs	O2.1
Given	RTOS is running mixed workloads (safety and non-safety critical tasks)
When	a real-time task schedules or requests memory/resources
Then	the RTOS must ensure tasks execute interference free, with spatial and temporal freedom from interference ensuring certifiability up to ISO26262 ASIL-D and conformance to cyber security standards.

Safe temporal and spatial isolation

Requirement ID	WP2_KK_01
Contributing Partner	KK
Use Cases	UC-2
KPIs	O5; O2.1
Given	there are multiple applications VMs or “software partitions” with differing ASIL level
When	the software partitions are executed on the same chip
Then	the hypervisor shall maintain temporal and spatial freedom from interference between the partitions

Containerized applications and system partitions in the Hypervisor / Operating System

Requirement ID	WP2_KK_02
Contributing Partner	KK
Use Cases	UC-1
KPIs	O2.1
Given	multiple software functions are implemented
When	these functions should be run interference free within one SoC
Then	the operating system shall provide a mechanism to package these applications with their dependencies and execute these packages (“containers”) on the target

Hypervisor-native network driver infrastructure

Requirement ID	WP2_KK_03
Contributing Partner	KK
Use Cases	UC-6
KPIs	O3.1
Given	a cloud connected vehicle
When	a software partition of the system shall be updated or a network device shall be shared between multiple partitions
Then	the operating system shall provide the means to access the network hardware from multiple such partitions

Hypervisor and OS adaptability to changing hardware

Requirement ID	WP2_KK_04
Contributing Partner	KK
Use Cases	UC-1; UC-5
KPIs	O3.1

Given	An SDV platform with multiple sensors and other hardware components attached to the central compute platform.
When	When any device or other hardware component is added to or removed from the device
Then	that hypervisor and operating system shall be reconfigurable, such that the new devices can be used by their applications and VMs as determined by the system integrator without impacting the operation of unrelated applications / VMs.

Performance monitoring and evaluation

Requirement ID	WP2_KK_05
Contributing Partner	KK
Use Cases	UC-3
KPIs	O1.2
Given	a virtual machine or application running on the hypervisor
When	running the application
Then	it shall be possible to collect performance data for execution time (profiling) and/or hardware utilization (performance counters)

Modular updateable architecture

Requirement ID	WP2_KK_DLR_01
Contributing Partner	KK; DLR
Use Cases	UC1-; UC-5; UC-6
KPIs	O1.1; O2.1; O3.1; O6
Given	A complex SDV CAD stack
When	individual software components shall be updated or added to the stack
Then	The OS shall allow to do so without interfering with the operation of components unrelated to the updated by providing individual exchangeable software modules.

Operating System and Hypervisor interfaces for re-configuration

Requirement ID	WP2_KK_DLR_02
Contributing Partner	KK; DLR
Use Cases	UC1; UC-2; UC-3; UC-5
KPIs	O1.1; O2.1; O3.1; O6
Given	The presence of an ORP and a Hypervisor
When	The ORP determines that a change in the configuration of the system is required

Then	The operating system and the hypervisor shall provide interfaces to configure the system such that the ORP can adapt the software stack to the changing system requirements.
-------------	--

Enables a reconfigurable, service-oriented communication plane for dynamic data flows within the vehicle, from vehicle-to-vehicle, vehicle-to-infrastructure, and vehicle-to-the-cloud (V2X).

Requirement ID	WP2_TAAT_ZSCALE_02
Contributing Partner	TAAT; ZSCALE
Use Cases	UC-1
KPIs	O2.2
Given	a vehicle function is distributed across in-vehicle ECUs and cloud services
When	a dynamic task reallocation or a software update is orchestrated between the vehicle and the cloud
Then	the Eclipse Zenoh protocol shall manage the data flow, ensuring seamless and location-transparent communication across domains

Provides a safety-certifiable communication layer for real-time data exchange for critical functions within the zonal vehicle architecture.

Requirement ID	WP2_TAAT_01
Contributing Partner	TAAT
Use Cases	UC-2
KPIs	O1.1
Given	safety-critical applications are distributed across a zonal E/E architecture in an SDV
When	these applications exchange data that requires bounded latency and jitter (e.g. fused sensor data)
Then	the communication shall be managed by a DDS over Time-Sensitive Networking (TSN) solution to guarantee deterministic delivery suitable for ISO 26262

Platform & Interface Contract (Runtime + HAL + Container Declarations). ST: OCI runtime + HAL metrics/controls. NEO: Container declarations (CPU/memory, etc.) and network telemetry to place/migrate workloads on constrained nodes and re-balance cloud-edge. Shared: schemas and Service Level Objectives (SLOs). Optionally, use a defined Zenoh profile (topics/QoS/monitoring) to enhance interoperability.

Requirement ID	WP2_ST_NEO_001
Contributing Partner	ST; NEO
Use Cases	UC-1; UC-2; UC-3

KPIs	O1.2; O2.1; O2.2; UC2.3; UC2.4; UC3.1
Given	STM32MP2 hosts containerized services, and the orchestrator must place/migrate workloads,
When	containers/services are deployed, started, stopped, or migrated,
Then	(a) the platform shall expose stable schemas for telemetry/controls, and (b) the orchestrator shall consume these schemas and container declarations to take placement/migration decisions that satisfy declared requirements.

Safe Update Pipeline & PCM Orchestration (Signatures + SASE + Digital Twin + A/B). ST: secure boot, signed artifacts, crypto primitives, PCM A/B. NEO: signature verification, SASE/model checks, digital twin, transaction control.

Requirement ID	WP2_ST_NEO_002
Contributing Partner	ST; NEO
Use Cases	UC-1; UC-2; UC-4
KPIs	O3.3; UC1.1; UC1.4; UC4.2; UC4.3
Given	a proposed firmware/software update or task reallocation,
When	the update plan is submitted for execution,
Then	the pipeline shall verify signatures, execute SASE/model checks in a digital-twin step, and only proceed if constraints are met; if Stellar firmware is involved, then the update shall perform an atomic PCM A/B swap with the ability to rollback.

Resource & Network-Aware Scheduling with Degraded-Mode Fallback. ST: continuous telemetry and enforcement hooks. NEO: enforce limits, place/migrate, re-balance cloud–edge with fallback.

Requirement ID	WP2_ST_NEO_003
Contributing Partner	ST; NEO
Use Cases	UC-2; UC-3
KPIs	O3.2; UC2.2; UC2.3; UC2.4; UC3.1; UC3.3;
Given	containers declare resource needs and network conditions may degrade,
When	the orchestrator detects resource pressure or poor connectivity,
Then	it shall place/migrate workloads to satisfy declared requirements and switch to local fallback on STM32MP2;

End-to-End Rollback & Audit. ST: A/B rollback capability and version evidence.

Requirement ID	WP2_ST_NEO_ITI_001
Contributing Partner	ST; NEO; ITI

Use Cases	UC-1; UC-2
KPIs	UC1.1; UC2.2; O3.1
Given	An OTA update for a software module has been received and post-deployment validation detects faulty behavior or policy violations, an issue is confirmed, the orchestrator SHALL atomically roll-back to the last known good configuration via the platform's A/B mechanism and persist an auditable record of what/when/why/who.
When	
Then	

Ensures interoperability and portability of software components across different hardware and software platforms.

Requirement ID	WP2_IKL_001
Contributing Partner	IKL
Use Cases	UC-1; UC-5
KPIs	O2.1
Given	different modular and updatable software components
When	these software components are deployed across a distributed system
Then	a standardized container format shall be used to package software components and their dependencies, ensuring portability and interoperability

Enables efficient resource allocation and management for containerized applications, considering the constraints of the underlying hardware.

Requirement ID	WP2_IKL_002
Contributing Partner	IKL
Use Cases	UC-3
KPIs	O1.4; O2.3
Given	that containerized applications have specific resource requirements
When	containers are deployed on resource-constrained hardware
Then	the container abstraction shall include resource requirements (CPU, memory, etc.) to enable the orchestrator to make informed scheduling and deployment decisions

Ensures that containerized applications meet the required safety and security levels.

Requirement ID	WP2_IKL_003
Contributing Partner	IKL
Use Cases	UC-2; UC-4
KPIs	O5.2; O5.3
Given	that applications have different safety and security requirements

When	containers are used to deploy these applications
Then	the container abstraction shall include SASE (Safety and Security) properties to ensure that the system can enforce the required safety and security levels for each container

Module based resource observation and monitoring

Requirement ID	WP2_DLR_001
Contributing Partner	DLR
Use Cases	UC-3
KPIs	UC3.1; UC3.3
Given	Any kind of software hierarchy is established/running on top of a hardware platform, each part of each software layer (module) must
When	always
Then	provide information about resource utilization to (included) observers for monitoring. The layer next to the hardware platform must also provide information about the hardware resource utilization.

Detection of resource under-/over consumption

Requirement ID	WP2_DLR_002
Contributing Partner	DLR
Use Cases	UC-3
KPIs	UC3.1; UC3.3
Given	Resource monitoring is established on the platform and observers provide the necessary information to monitors
When	a module is over or under consuming it's specified resources
Then	the monitor shall detect this problem and provide a trigger to react accordingly.

Provide resource utilization information for update middleware & process

Requirement ID	WP2_DLR_003
Contributing Partner	DLR
Use Cases	UC-1; UC3
KPIs	UC1.3; UC1.4; UC3.3; UC3.4
Given	An update, upgrade or reconfiguration is required
When	the ORP wants to make changes to the platform (i.e. changes to the software layers or orchestrate a hardware change)
Then	the OAL must provide resource utilization information to the ORP for decision making and validation of correct orchestration.

3 Summary & Outlook

This document contains the requirements collected by the project partners in collaboration with the other tasks that interface with the OAL. It represents the first step in building a comprehensive updatable SDV architecture and implementing demonstrators to showcase the feasibility and features of the approach.

Based on the requirements gathered and summarized in this document the partners in WP2 have already started work on both the reconfigurable and resource controllable container middleware platform (T2.2) as well as on the reconfigurable and resource controllable service-oriented communication plane implementation.

The next upcoming deliverables for WP2 are due in month 12, as part of milestone MS3 and will comprise the initial designs and prototypes for the container runtime and the communication plane (D2.2 initial and D2.3 initial).