Rail to Digital Automated up to Autonomous Train Operation

D36.5 – Onboard platform blueprint for later EU-RAIL phases established and disseminated

Due date of deliverable: 31/05/2026

Actual submission date: 13/04/2026

Leader/Responsible of this Deliverable: DB

Reviewed: Y

Document status		
Revision	Date	Description
0.1	02/02/2026	Document template
0.2	16/03/2026	First review and exchange with WP3 (Chapter 2)
0.3	24/03/2026	Content freeze and start of internal review
0.4	13/04/2026	TMT submission
0.5	07/07/2026	Final document, minor corrections after JU review

Project funded from the European Union's Horizon Europe research and innovation programme		
Dissemination Level		
PU	Public	**x**
SEN	Sensitive – limited under the conditions of the Grant Agreement	

Start date: 01/12/2022 (WP36 Kick-off 25+26/01/2023) Duration: 42 months

ACKNOWLEDGEMENTS

This project has received funding from the Europe's Rail Joint Undertaking (ERJU) under the Grant Agreement no. 101102001. The JU receives support from the European Union's Horizon Europe research and innovation programme and the Europe's Rail JU members other than the Union.

REPORT CONTRIBUTORS

Name	Company	Details of Contribution
Oliver Mayer-Buschmann	DB InfraGO	Editorial, Executive Summary, Introduction, Chapters 2, 3, 4.1, Conclusion, Review
Benjamin Labonté	DB InfraGO	Review
Thomas Martin	SBB	Chapter 4, Review
Martin Lindenmaier	SBB	Appendix C, Review
Thomas Buchmüller	SBB	Chapter 3.7, Appendix B, Review
Stefan Resch	Hitachi/GTS	Chapter 2.3, Review
Julien Spanneut	SNCF	Chapter 3, Review
Ulrich Geier	Kontron	Review
Manfred Taferner	Kontron	Review
Kevin Wriston	Kontron	Review
Henrik Larsson	TRV	Review

DISCLAIMER

EXECUTIVE SUMMARY

This deliverable concludes the activities of the *Onboard Platform Demonstrator* work package (WP36) within the Europe's Rail FP2 R2DATO programme. It consolidates the results of three implementation phases, synthesising the technical evidence and strategic insights gained from applying platform technologies, in particular Modular Platforms, in combination with FRMCS and on-board communication, diagnostics services and train-integration concepts in a relevant laboratory environment.

Digitalisation requires interoperable, upgradeable, standardised and future-proof solutions—through the integration of digital platforms and services—addressing the needs of the Railway sector and expectations and challenges of the future European transportation system.

D36.5 builds on the jointly developed results by Railway and Industry partners, supported by exchanges with related R2DATO work packages and ERJU System Pillar Task 2 domains. It intends to serve as a public reference, encouraging the adoption of the demonstrated specifications and platform-based solutions to support sector-wide harmonisation, migration and standardisation. The deliverable demonstrates that modular, multivendor on-board computing architectures provide a viable foundation for the future development, evolution and standardisation of CCS on-board systems across Europe.

SCIENTIFIC / TECHNICAL APPROACH AND SCOPE

WP36 follows a structured scientific and technical approach across the three conducted, consecutive implementation phases, each validating key elements of modular on-board platform technologies, serving multi-vendor enabling integration of mixed-criticality applications (up to SIL4) and their interaction with wayside and TCMS services. The scope and main objectives are:

D36.5-Obj-1 Provide a high-level overview of the developed demonstrator architecture, drawing conclusions from the findings and lessons learned across three consecutive implementation phases of the project. Explain the context and placement of the work within the R2DATO framework, the (WP3) architecture and sector-wide specification work.

D36.5-Obj-2 Evaluate the original remit of WP36 against evolving inputs from R2DATO standardisation activities and System Pillar domains, identifying how maturing specifications could be applied, implemented and evaluated or required adaptation during implementation.

D36.5-Obj-3 Synthesise the implementation steps, technical achievements and results across all phases, including evaluation and realisation of Technical Enablers, the progression from virtual to physical deployment, and the demonstration of realistic ETCS, FRMCS and diagnostics use cases, covering the majority of 65 specified User Stories, addressing future needs of CCS-on-board systems.

D36.5-Obj-4 Identify the most important achievements and benefits arising from the utilisation and combination of those platform technologies, resulting in a coherent understanding and platform reference implementation and blueprint, showing how those results can contribute to the Standardisation and TSI Input Plan (STIP) to support migration and evolution of the sector.

D36.5-Obj-5 Based on the demonstrated feasibility and remaining gaps, propose recommendations for the utilisation of the WP36 results towards future standardisation and demonstration, promoting technology and industrialisation and helping to improve the competitiveness of the European railway supply industry and cost-efficiency.

Rooted in the realisation, evaluation and testing, documented in earlier implementation phases

- Modular Platform capabilities (D36.2)
- FRMCS multi-domain integration and diagnostics foundations (D36.3)
- Full ETCS/FRMCS/diagnostics laboratory demonstration (D36.4)

this final deliverable provides both the technical evaluation and strategic framing as a basis for outlook and dissemination, presenting the core achievements as an initial implementation of on-board platform technologies and as a first plausible blueprint for future EU-RAIL activities.

Based on System Pillar alignment and cross-workstream integration, this deliverable contributes to sector-wide standardisation efforts, enabling important next steps as the European railway sector progresses towards a harmonised system architecture.

As the three implementation phases of WP36 progressively validated the feasibility, robustness and interoperability of the mentioned platform technology, the following section summarises the work carried out, integrating the results of the examined functional clusters and synthesising the main outcomes of the demonstrator evolution.

MAIN FINDINGS

Modular Platform Implementation

The demonstrator confirmed that a modular, COTS-based on-board platform can reliably host mixed-criticality applications, including functions up to SIL4, in a multi-vendor environment. The programming and integration models enabled deterministic execution, controlled separation between safety-related and basic-integrity functions, and services.

Throughout all phases, the approach proved compatibility with realistic on-board architectures and enabled the successful migration of a third-party ETCS Baseline 3 application into a replicated 2oo3 safety configuration. The application behaved as expected in nominal and degraded operating states and demonstrated functional equivalence to traditional bespoke on-board solutions.

FRMCS Integration and Safe Communication

FRMCS-based communication evolved from preparatory architectural work to a fully integrated end-to-end solution. Finally, a dedicated FRMCS Agent and OB_{APP}-based interface have been implemented, allowing on-board applications to access FRMCS services independently of underlying radio technologies and FRMCS-specific knowledge.

In the final phase, FRMCS communication sessions remained stable across replica failures and recovery events. Automatic session re-establishment ensured uninterrupted ETCS–RBC communication, showing that FRMCS can satisfy availability expectations when combined with platform-managed failover mechanisms.

Diagnostics and Train Integration

A unified diagnostic concept was implemented using SOVD, supporting periodic and event-driven data provision. Safety applications periodically delivered deterministic data, while Basic Integrity applications could also handle interactive request–response patterns.

The MCP-DIA prototype enabled structured diagnostics data modelling and provided a consistent interface for external tools, including a visualisation client. In parallel, the TCMS Data Service and DIA-VEC demonstrated vehicle-level data aggregation, brake-state evaluation and rule-based degradation logic. These implementations together form a coherent approach to on-board diagnostics and condition-based maintenance.

Network and System Resilience

The CCS Consist-Network validated the suitability of Ethernet-based communication for mission-critical on-board functions. A redundant ring topology with RSTP ensured continued communication during cable or switch failures, with only brief and acceptable latency increases.

At the platform level, extensive tests confirmed correct replica failover, safe-layer recovery and stable operation in degraded modes. ETCS functionality, including L1/L2 transitions, remained intact across restarts, power-loss events and FRMCS reconnection sequences.

Security and Update Considerations

Security and update mechanisms were assessed conceptually. Although full OTA update workflows were outside the demonstrator scope, the platform's technical capabilities align with relevant cybersecurity standards. The work highlights operator-side dependencies such as identity management, PKI-based credential handling and backend infrastructure as essential components for real-world deployment.

Technology Readiness and Remaining Gaps

The demonstrator achieved TRL 5–6 by delivering a functioning prototype validated in a relevant laboratory environment. Several advanced capabilities were demonstrated, including mixed-vendor integration, FRMCS-based communication, deterministic diagnostics and DevOps-based validation workflows.

Remaining gaps relate to areas requiring additional infrastructure or specification maturity, including heterogeneous hardware portability, virtualisation technologies for hardware independence of hosted Functional Systems, full OTA and configuration updates, safety protocol implementation and extended trackside Shared Services provision. These are appropriate topics for future demonstrators and operator-led pilots.

QUALITATIVE IMPACTS AND RECOMMENDATIONS FOR FUTURE WORK

Consolidated outcomes from the collaboration in all three implementation phases emphasise the following, including further development and insights within R2DATO and the System Pillar:

- Evaluated platform technologies and digital services contribute qualitatively to R2DATO KPIs (KPI Staff Productivity). Derived cost-reduction estimates and future improvements in rail supply industry competitiveness and performance are well supported. Anchoring specification work into standards and products will potentially reduce both CAPEX and OPEX, supporting the migration and the evolution challenges of the Control-Command and Signalling On-board Subsystem.
- The DevOps-based workflow proved essential for reproducibility, multi-partner integration and efficient validation. Test automation helped share laboratory capacity, align between partners, establish synergies, increase efficiency and avoid regression.
- The demonstrator provides a credible blueprint aligned with Europe's Rail ambitions for modular, interoperable and upgradable on-board architectures and offers a solid reference for future ERJU phases and standardisation activities. For industrialisation, further R&I is needed and the utilisation of the evaluated Technical Enablers in future demonstrators and pilots is reasonable and shall be explicitly recommended and promoted here.
- Since a focus shift towards the hosting of multi-vendor functional systems on Hardware-Level Platform Independence has been emphasised in the System Pillar, further work in that area may complement the results from WP36 in future phases of R2DATO. Virtualisation technology as utilised heavily in the laboratory setup to host the simulation and wayside components is recommended by the System Pillar for the functional system

deployments. Virtualisation technology in general can be used as well for on-board systems.

- Many topics need further study, specification and implementation (e.g., configuration, update and orchestration, security, etc.). The conducted work has shown that these additional transversal aspects need to be evaluated and validated in demonstrator projects and pilots, involving operators to validate operational deployment, certification-relevant roles and behaviour, and integration into real rolling-stock architectures.
- Further standardisation is needed to enable industrialisation and product development in the sector for on-board and wayside, potentially including changes in the legislative framework, role definition and clear elaborations on responsibilities and accountabilities between railway undertakings/infrastructure managers and industry partners. Related work will be continued in the System Pillar PRAMS domain and corresponding STIP-items are placed in the Standardisation and TSI Input Plan to cover further activities.

CONCLUDING REMARK

WP36 has successfully demonstrated that platform technologies can be combined into a robust, multi-vendor, scalable on-board architecture. The demonstrator achieved TRL 5–6, validated deterministic behaviour, replication strategies, and transparent failover mechanisms, and provided convincing evidence that modular on-board platforms can technically host safe (up to SIL4) and basic integrity railway applications in a multi-vendor setup.

The outcomes form the first Onboard Platform Blueprint for future EU-RAIL activities, offering reusable patterns, architectural clarity and practical experience to support future demonstrators and pilots and long-term standardisation work. WP36 thereby contributes significantly to the evolution of interoperable, upgradeable and supplier-neutral on-board CCS systems in Europe, strengthening the sector's ability to migrate toward future digital, automated and resilient railway operations.

ABBREVIATIONS AND ACRONYMS

Abbreviation	Definition
ALPI	Application-Level Platform Independence
API	Application Programming Interface
ATO	Automatic Train Operation
BI	Basic Integrity
CAPEX	Capital Expenditures
CCN	CCS Consist Network
CCS	Control-Command and Signalling
CE	Computing Element
CESAM	Systems Engineering Framework
CI	Continuous Integration
CN	Computing Node
CONEMP	Computing Environment Domain
COTS	Commercial Off-The-Shelf
CRA	Cyber Resilience Act
CS	Control System
DB	Deutsche Bahn / DB InfraGO
DEM8	Demonstrator 8 (WP36)
DEVOPS	Development (Dev) and IT operations (Ops)
DIA	Diagnostics
DMI	Driver-Machine Interface
EDT	Enabling Digital Technologies
EN	European Norm
ER	Europe's Rail
ERA	EU Agency for Railways
ERJU	Europe's Rail Joint Undertaking
ETCS	European Train Control System
ETSI	European Telecommunications Standards Institute
EU	European Union
EVC	European Vital Computer
FP2	Flagship Programme 2
FRMCS	Future Railway Mobile Communication System
FS	Full Supervision

GA	Grant Agreement
GRE	Generic Routing Encapsulation
GTS	Hitachi Rail STS
HLPI	Hardware-Level Platform Independence
HTTP	Hypertext Transfer Protocol
IAM	Identity and Access Management
IDE	Integrated Development Environment
IEC	International Electrotechnical Commission
IM	Infrastructure Manager
IO	Input/Output
IP	Internet Protocol
ISO	International Organization for Standardization
IT	Information Technology
JU	Joint Undertaking
KPI	Key Performance Indicator
MCP	Modular Computing Platform
MDCM	Maintenance, Diagnostics, Configuration & Monitoring
OB	On-board
OB_{APP}	On-Board Application interface
OCORA	Open CCS On-board Reference Architecture
OCS	On-board Communication System
OPEX	Operational Expenditures
OTA	Over The Air
PKI	Public Key Infrastructure
POSIX	Portable Operating System Interface
PRAMS	Performance, Reliability, Availability, Maintainability, Safety
RAIL	Europe's Rail
RBC	Radio Block Centre
REST	Representational State Transfer
RSTP	Rapid Spanning Tree Protocol
RU	Railway Undertaking
SERA	Single European Railway Area
SIL	Safety Integrity Level
SL3	Security Level 3

SOVD	Service Oriented Vehicle Diagnostics
SP	System Pillar
SR	Staff Responsible
STIP	Standardisation and TSI Input Plan
SV	(ETCS) System Version
SW	Software
TAS	Hitachi/GTS TAS Platform
TCMS	Train Control and Monitoring System
TCP	Transmission Control Protocol
TE	Technical Enabler
TFFR	Tolerable Functional (unsafe) Failure Rate
TOBA	Telecom On-board Architecture
TRL	Technology Readiness Level
TS	Trackside
TSI	Technical Specifications for Interoperability
UIC	International Union of Railways
VEC	Vehicle (in the context DIA-VEC: vehicle diagnostics app)
VLAN	Virtual Local Area Network
WP	Work Package
YANG	Yet Another Next Generation Data Modelling Language

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

The present document constitutes Deliverable D36.5 – "On-board platform blueprint for later EU-RAIL phases established and disseminated" within ERJU Flagship Programme FP2 (R2DATO). It consolidates the outcomes of WP36 across the specification and three implementation phases and synthesises them into a blueprint supporting the transition from laboratory demonstrations to integrated demonstrators, pilots and standardisation in future EU-RAIL phases and beyond.

WP36 has set out to demonstrate a modular, multi-vendor on-board computing platform integrating safety-critical and non-safety functions, FRMCS-based communication, digital diagnostics services, and train integration software components. Earlier deliverables—D36.2, D36.3, and D36.4—established feasibility, integrated a full laboratory setup, and executed realistic ETCS, FRMCS, diagnostics, and on-board networking demonstrations. D36.5 now evaluates these results in a broader R2DATO and System Pillar context, identifies reusable architectural patterns, and prepares recommendations for future deployment approaches and standardisation.

The overall motivation arises from the sector's transition from monolithic, vendor-specific on-board systems toward modular, scalable, interoperable, and upgradeable digital architectures. The demonstrator results provide evidence that mixed-criticality functions—including up to functions requiring SIL 4—can be reliably hosted on COTS-based modular computing platforms while maintaining deterministic behaviour, robust failover, and operational continuity.

Based on three phases of implementation and testing, the demonstrator reached TRL 5–6, with a prototype implementation validated in a relevant environment that was recreated in a laboratory.

While the deliverables D36.1 to D36.3 detailed and evolved an on-board platform architecture and the realisation of specific User Stories, D36.4 reported the full integration of the laboratory implementation.

D36.5 now merges the most important findings and central results into a strategic reference that serves as final dissemination. The following table presents the document structure to guide the reader, providing an initial overview of the chapters that follow.

Table 1: Document Structure

#	Topic	Goal
Chapter 2	Remit, Context and Placement	Structured overview of the demonstrator's context and placement to clarify contributions to standardisation and TSI evolution and how WP36 contributes to ERJU objectives by supporting future-proof, interoperable on-board computing architectures in Europe's Rail.
Chapter 3	Consolidation of the Implementation Tasks	Summarise the highlights, demonstration achievements, results and major insights gained during realisation.
Chapter 4	WP36 Onboard Platform Architecture and Reference Implementation	Present a coherent reference implementation and the proposed blueprint-architecture patterns for future Europe's Rail phases.
Chapter 5	Conclusion	Final project conclusion, identification of gaps, open issues, and recommendations for further work and exploitation of the results.

APPENDIX A - Assessment of WP26 Requirements
APPENDIX B - EVC Porting Experience and Lessons learned
APPENDIX C - Remaining User Stories, Test Cases and Results

2 REMIT, CONTEXT AND PLACEMENT

Building upon the remit as defined in the 36.5 Task and Deliverable description and the Demonstrator Specification worked out and presented in Deliverable D36.1, the Onboard Platform Demonstrator has realised a first reference implementation for platform technologies in the framework of the R2DATO Demonstrator cluster. The demonstration integrates and combines *Technical Enablers* from the R2DATO EDT cluster and provides a solid foundation, contributing to the process of enabling digitalisation by evaluating emerging technologies needed in the railway sector.

2.1 STRATEGIC INCORPORATION INTO THE ERJU FRAMEWORK

WP36 directly contributes to the ERJU FP2 R2DATO mission, enabling future digital, automated, and interoperable on-board systems. Throughout the demonstrator evolution, the team validated integration and migration, combining FRMCS-based and on-board communication and diagnostics concepts around a unified approach towards a multi-vendor on-board computing platform, hosting an ETCS application. The work shows that mixed-criticality applications can technically coexist on a modular, COTS-based on-board platform, supporting robust failover mechanisms, harmonised diagnostics, FRMCS-based communication and a modular on-board communication network.

These findings are fully aligned with the overall ERJU strategy to replace monolithic solutions with modular, scalable, expandable, and lifecycle-efficient architectures and products.

The following Figure 1 visualises how WP36 is embedded into the ER JU Framework. This simplified and non-exhaustive view lists the most relevant stakeholders and counterparts of WP36 as:

- EU-RAIL, responsible for Sector Alignment, Governance of Technical Development & Stakeholder Coordination, approving work programmes and funding envelopes, ensuring System Pillar outputs become official EU system specifications and feed into TSIs
- R2DATO Technical Enablers of the ETD cluster
    - TE16 Modular Platforms
    - TE5.1 FRMCS On-board Integration
    - TE17 On-board Communication Network
- R2DATO WP3 “Technical Engineering and Requirements Consistency”
- Relevant System Pillar Task 2 Domains that have been input sources for WP36 and may as well be interested in the presented results and findings from the conducted work:
    - CONEMP domain for Computing Environment Specifications and Transversal topics (like Configuration Management, SW-Update, Diagnostics and interfaces to Shared Services),
    - Train-CS domain for CCS-On-board Architecture, Migration, Modularity,
    - PRAMS domain for Performance, Reliability, Availability, Maintainability, Safety and Evolution in Certification and Authorisation of the CCS System,
    - Cyber Security domain for Secure Communication, Shared Services with Security-by-Design, supporting a harmonised system architecture to replace national approaches,

      Regulatory Compliance, aligning with EU regulations, including the Cyber Resilience Act (CRA) and IEC 62443 standards, etc.

Please note that the depiction of arrows is exemplary, and that other stakeholders have been omitted for the sake of simplicity. In addition, exchange between Technical Enablers and System Pillar domains rather happens directly and not via WP36. All the mentioned WPs, TEs and domains

directly or indirectly contribute to SERA and future TSI updates via established processes, e.g., via change requests or in the framework of the Standardisation and TSI Input Plan (STIP).

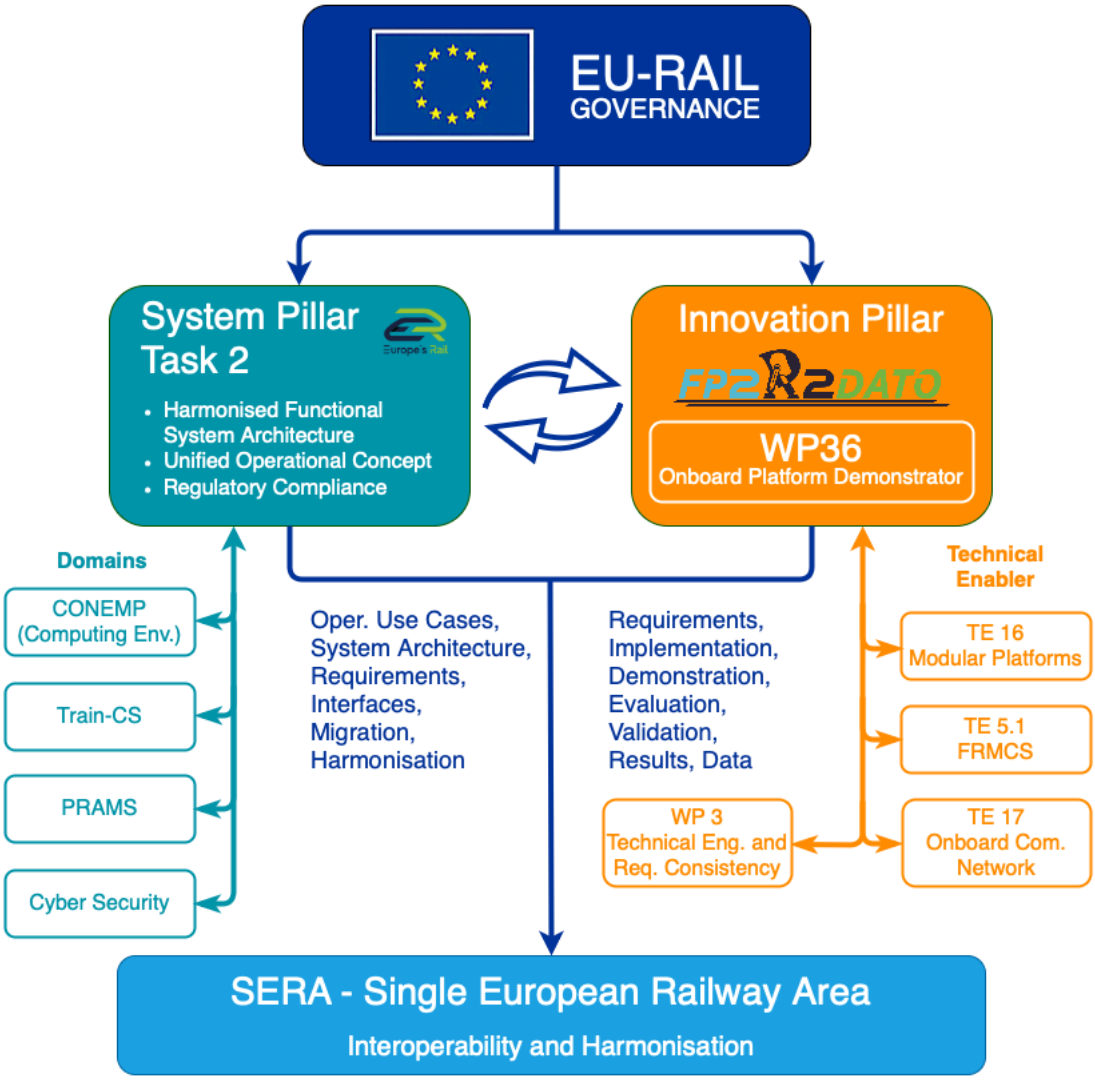


Figure 1: WP36 embedded into the ER JU Framework

2.2 WP36 ALIGNMENT WITH R2DATO WP3

This chapter intends to place and identify WP36 within architectural content provided by WP3's Deliverable: *D3.4 - Architecture baseline* report [1] for R2DATO technologies. The reader may study D3.4 for a better understanding. In this chapter some specific content from this report is cited and presented as guiding findings "in quotes". Two diagrams have been used and complemented (only with red frames) to indicate the WP36 placement, since they cover objectives, capabilities and functions that were applied in the On-board Platform Demonstrator.

D3.4 follows the CESAM framework [2], providing multiple architectural views for "Technical Engineering and Requirements Consistency". In the following the *Enterprise Context* view and the *Constituent breakdown* view of D3.4 are presented. This contributes to a structured and coherent understanding of how WP36 fits into the wider R2DATO architecture as a systemic description of the WP36 design context.

Note: For the sake of readability and since D3.4 is currently under R2DATO review and not publicly available at the time of writing this chapter. Therefore, the cited ("quoted") text and diagrams cannot

be fully referenced scientifically. References are provided to the original document where feasible but integrated content may still become subject to minor changes in the final D3.4 report. The used content is supposed to be almost final, and its usage has been aligned and reviewed by the responsible WP3 contributors.

To introduce the purpose of D3.4 and establish the context to WP36, several relevant design objectives from the D3.4 report are reiterated here, as they also guide and relate to the remit of WP36:

- "The architecture work within the R2DATO project, particularly under WP3, plays a pivotal role in aligning and consolidating the diverse developments across the project's work packages (WPs). Its primary objective is to act as a bridge between the technical work of the WPs and the overarching System Pillar, ensuring coherence, integration, and shared understanding."[1]
- "While the System Pillar is developing its reference architecture and R2DATO WPs are applying specific design choices, there is an opportunity to define an "applicable architecture" to integrate existing system knowledge and identify system coherence risks/issues as early as possible in the design process."[2]
- "Building an architecture for R2DATO systems design is the opportunity to build a systematic traceability between the needs (e.g. contribution of R2DATO towards EU Rail KPIs) and the products (and concrete prototypes) developed in WPs."[2]
- "The system architectures should help to analyse the business (e.g. cost and benefit) and legal (e.g. TSI) implications of a design choice (e.g. accountability or competition in the railway sector)."[2]

In the *WP3 R2DATO Enterprise Context view* demonstrators are considered as an intermediate "TACTICAL" link between strategic and operational business impact, to:

- "Verify complete traceability for operation with respect to the stakeholders involved"[3],
- "Improve EU rail supply industry competitiveness"[3],
- "Improve performance and capacity"[3].

by integrating digital technologies with the intention to further optimise the rail production process.

These statements and objectives align well with the purpose of platform technologies, both in the *Enterprise Context* and within the technical *Constituent Breakdown*. Although WP36, as a laboratory demonstrator with a TRL of 5-6, contributes mainly qualitatively to R2DATO KPIs (KPI Staff Productivity), cost-reduction estimates, rail supply industry competitiveness improvements, and performance are supported by the applied emerging technologies. In the mid and long term, anchoring and standardising these technologies in products may reduce both CAPEX and, even more importantly, OPEX, while supporting the migration and evolution of the CCS-On-board Subsystem.

For details on those KPIs and estimates, please refer to R2DATO Deliverable: *D2.2 - KPI Monitoring Report* [3], Chapter 7.2 "Baselines By The Demos 2-8", Figure 16: "Baseline of [DEM8]", Figure 24: "Qualitative improvement of [DEM8]", Figure 31: "Estimation of quantitative impact for [DEM8]" and Figure 32: "Ownership of the KPIs by [DEM2] – [DEM8]".

Note: DEM8 refers to WP36 in Deliverable D2.2.

[1] Cited from R2DATO D3.4 Chapter 2.2 "Purpose"
[2] Cited from R2DATO D3.4 Chapter 1 "Context of FP2 - R2DATO WP3"
[3] Cited (slightly modified) from R2DATO D3.4 "Figure 15: Enterprise Context view" (please refer to Figure 2)

The following Figure 2 visualises the placement of WP36 as a demonstrator in the "TACTICAL" Business Impact level (by adding a red frame area to the original WP3 Enterprise Context view).

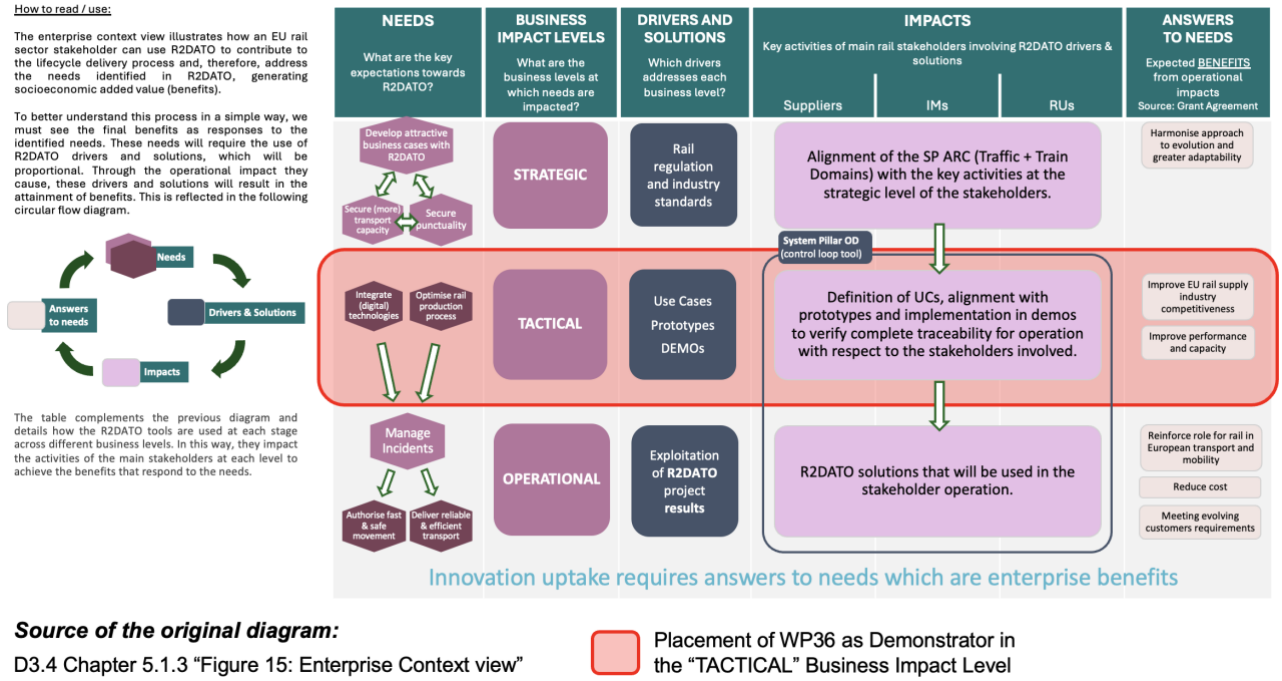


Figure 2: Placement of WP36 in the WP3 Enterprise Context view

As visualised in Figure 2, demonstration work in Flagship Area projects such as R2DATO plays an essential role towards sector-wide acceptance of new technologies, ensuring that research results are validated in realistic operational contexts and can serve as credible evidence for European-level standardisation and TSI evolution. Technical Enablers and their demonstrations are typically linked to TSI updates via STIP-items within the framework of the Standardisation and TSI Input Plan. Further context and details are presented in Chapter 2.3.

D36.5 consolidates the outcomes of the WP36 implementation phases into a coherent and strategically aligned on-board platform blueprint supporting future EU-RAIL activities, the ERJU System Pillar, the evolution of the TSI CCS, and the long-term industrialisation of modular on-board solutions. The demonstrated results show that a vendor-neutral, modular on-board architecture—integrating FRMCS communication services, diagnostics, and train integration capabilities—is not only technically feasible but also operationally robust and aligned with Europe's long-term objectives of a fully digitalised, interoperable, and future-proof rail system.

The blueprint and evidence produced throughout WP36 offer concrete architectural patterns, validated integration models, and cross-ERJU alignment that together form a solid foundation for the next steps of deployment and standardisation.

Building on these outcomes, Deliverable D36.5 provides clarity on how platform technologies demonstrated in the laboratory can be extended, scaled, and industrialised. Key aspects for future pilots and beyond include:

- Scalability of modular architectures to heterogeneous on-board hardware and multivendor software ecosystems, enabling flexible and cost-efficient deployment strategies across different vehicle classes and operators.
- Integration of FRMCS on a Modular Platform, demonstrating how future safety-related communication services of hosted applications can be supported, abstracted, and securely connected to wayside operational environments.

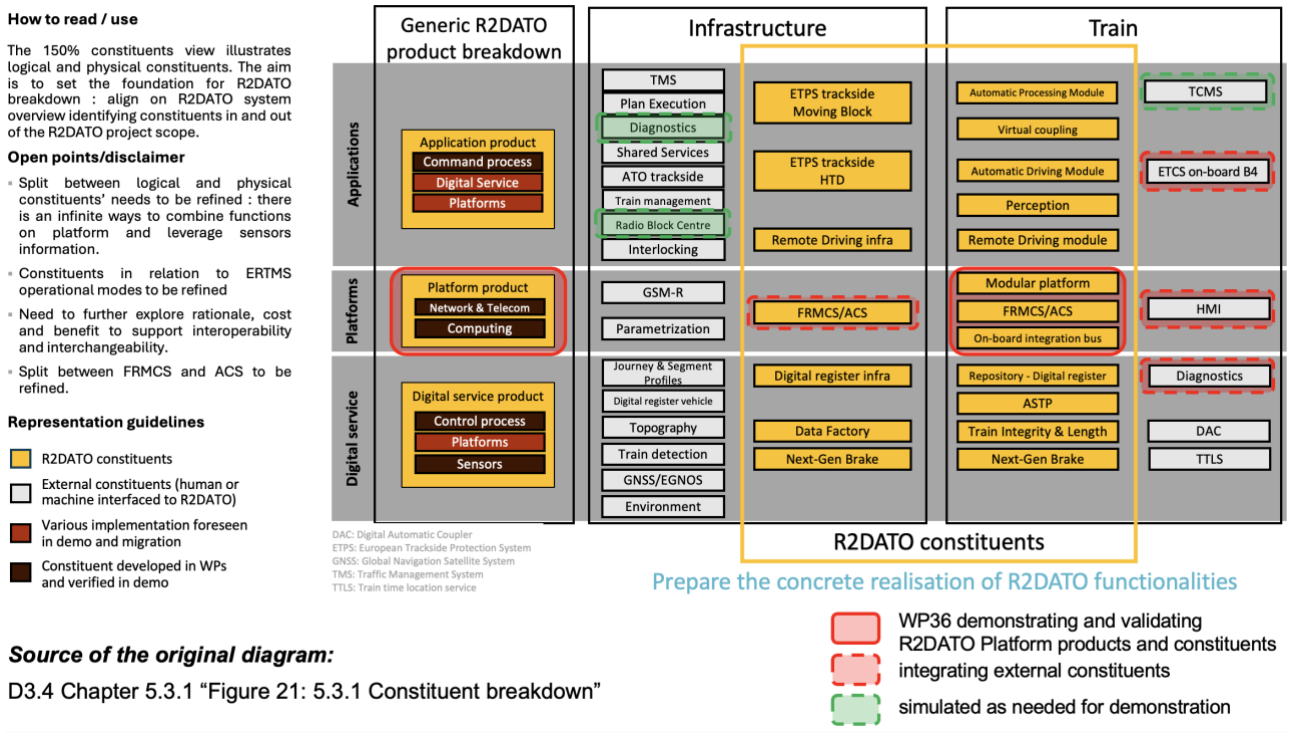
- Identification of operator-side infrastructure needs, such as Identity and Access Management (IAM), Public Key Infrastructure (PKI), secure update governance, and OTA-capable backend solutions that are essential for full operational deployment.
- Preparation for field-grade implementation, outlining the steps required to evolve from laboratory demonstrators to precommercial pilots, certification-relevant validation, and industrial rollout.
- Bridging diagnostics and TCMS data services with predictive maintenance ecosystems, enabling condition-based maintenance and fleet-level analytics rooted in harmonised data models and open interfaces.

Taken together, these elements position WP36 as a strategic enabler for the next phase of the R2DATO programme. They ensure that the validated concepts, architectural patterns, and integration experiences captured in this deliverable can serve as an actionable reference for integrated demonstrators in future EU-RAIL phases and the broader European standardisation landscape.

By closing the conceptual gap between R&I activities and upcoming industrial deployments, Chapter 2 concludes with a clear view on how the presented results can be transferred, expanded, and utilised to support sector-wide migration towards modular, interoperable on-board platforms.

Consolidation of the Implementation Tasks, provides a summary of the WP36 evaluation, results and central findings, to depict how the Onboard Platform Demonstrator is supporting validation, verification and migration of platform technologies in R2DATO. For more details, the former four WP36 deliverables (e.g., D36.4, which presents the full demonstrator setup, validation activities and results) are referenced for readers seeking further detail.

As shown in the following Figure 3 from D3.4, WP36 evaluates and integrates important R2DATO constituents on the “Platform products” level, supplemented by digital services, applications and other external constituents.



Source of the original diagram:

D3.4 Chapter 5.3.1 “Figure 21: 5.3.1 Constituent breakdown”

Figure 3: Placement of WP36 as “Platform Product” in the WP3 Constituent breakdown

As shown in the red framed boxes, WP36 directly supports R2DATO capabilities and migration aspects described in WP3 (please refer to D3.4 Chapters 5.2.1 “Technical Capabilities”, 5.2.2 “Functional Breakdown” and 5.2.3 “Functional Modes (with focus on Migration)”). By evaluating platform approaches, WP36 enables sustainable solutions in the form of underlying digital services and contributes to paving the way for interoperable and, where appropriate, interchangeable products. This is achieved by implementing the following WP3 capabilities and functions:

- “Providing safe, secured, and seamless communication between train and infrastructure”,
- “Providing safe, secured, and seamless on-board communication”,
- “Support the implementation of on-board applications”,
- “Providing safe and secured digital modular platform”.

The WP36 main topics directly align with this constituent breakdown and are complemented with digital services and applications:

- Computing Platform, FRMCS,
- On-board Communication Network,
- ETCS-OB application,
- TCMS (Data Service),
- Diagnostics Services.

Through these elements, WP36 supports both the strategic needs of R2DATO, as described in WP3, and the broader needs of the sector. Specifically, WP36 improves modularity, maintainability, evolvability, and sustainability by applying platform-based approaches that can reduce costs and enable more affordable on-board CCS solutions in terms of both CAPEX and OPEX.

2.3 Input from Standardisation

The WP36 demonstrator acts as a link between technology readiness and the functional and system-level specifications developed in the ERJU System Pillar, R2DATO work packages, and external standardisation bodies (such as ETSI and UIC). By providing concrete implementation evidence, the demonstrator contributes to ongoing specification work in several areas, including:

- Functional allocation principles across modular platform, communication paradigms, and hosted applications,
- Architectural patterns for replicated platform functions, digital services as diagnostics, and on-board to trackside communication,
- Clarification of safety responsibility boundaries, and where safety protocols (e.g., EuroRadio), can be provided by platform transport functions,
- Update, configuration, and cybersecurity concepts that may support future CCS governance processes.

These results support the R2DATO Technical Enablers and the System Pillar domains responsible for TSI evolution and harmonised on-board architecture definitions. The primary input sources from the System Pillar and the selected Technical Enablers have already been shown in Figure 1 and the corresponding R2DATO constituents and system capabilities/functions were introduced in the previous chapter.

Since specification work progressed in parallel to design and implementation, alignment proved to be challenging. Although regular exchanges with R2DATO work packages and System Pillar representatives took place, varying levels of maturity, late availability of deliverables, and evolving baselines required WP36 to balance new inputs with project internal stability. This resulted in several design decisions that combined available specification content with existing solutions and

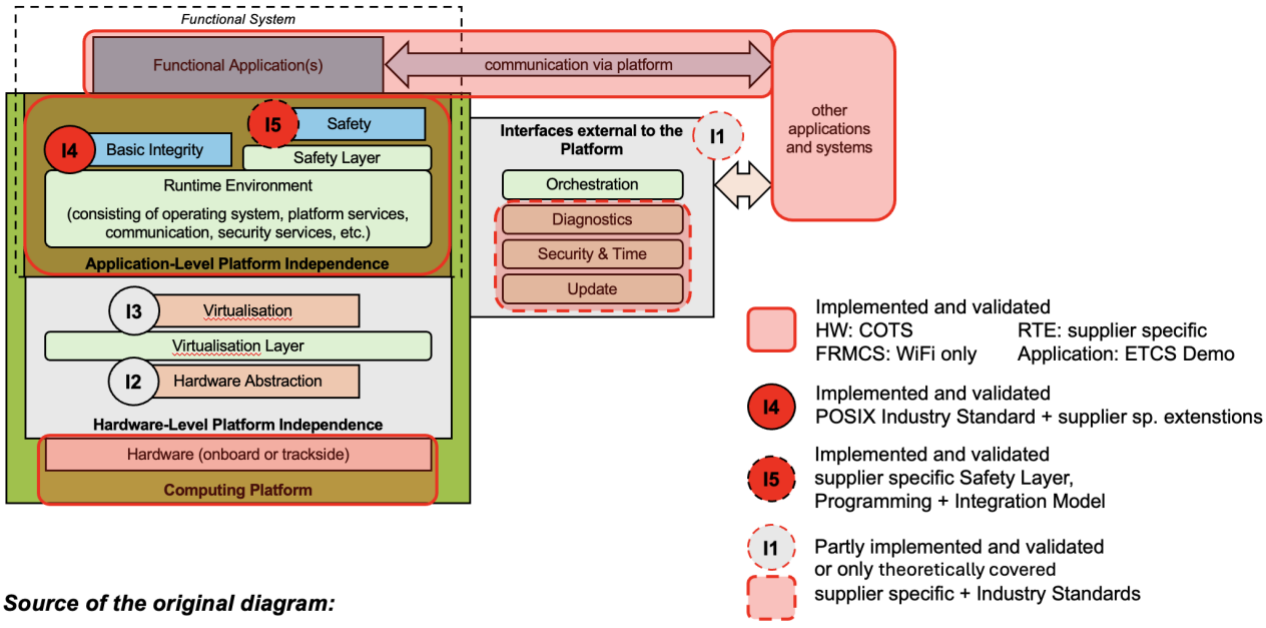
preliminary standards. Such conditions are typical for R&I driven demonstrators and underline the importance of structured planning, controlled change management, and, where necessary, the temporary use of vendor-specific solutions or established industry practices and standards.

The evolution of the FRMCS specifications illustrates this dynamic. Based on the development of FRMCS specifications, the technical realisation of the OB_{APP} interface, which links on-board applications (like ETCS/ATO) to the FRMCS Gateway (TOBA), moved towards a REST-based and HTTP/2- enabled communication in the V2 specification set in early 2025. The project decided to adopt the updated version rather than remain on an outdated baseline. Despite introducing additional effort and some integration risk, this proved feasible and ensured alignment with the forward-looking FRMCS architecture.

A more significant impact emerged from the Modular Platform specification work in WP26, which had been expected to deliver a mature definition of Application-Level Platform Independence (ALPI, interfaces I4 and I5; see Figure 4) early in the project. However, the System Pillar Computing Environment domain shifted focus towards Hardware-Level Platform Independence (I2/I3). Consequently, WP36 had to integrate applications and platform services based on the vendor-specific platform implementation, using industry-standard APIs (POSIX with supplier-specific extensions) in combination with existing programming and integration models of the supplier.

Hitachi/GTS delivered a certified and industry-used solution, forming the basis for the WP36 Modular Platform implementation. Alignment between the realised implementation and the WP26 requirements is provided in the APPENDIX A - Assessment of WP26 Requirements. It must be recognised that these requirements feed into ongoing standardisation efforts and may continue to evolve.

The following Figure 4 illustrates which elements of the Modular Platform architecture defined in D26.3 (Chapter 5) could be implemented, and which aspects remained only partly addressed or out of scope due to immature specifications, resource limitations, or project priorities. Virtualisation technologies, for example, were not included—not because they were not suitable for on-board systems—but because they were not required to fulfil the WP36 remit within the available timeframe and specification context.



Source of the original diagram:

D26.3 Chapter 5 MODULAR PLATFORMS ARCHITECTURE

"Figure 14: Modular Platform architecture showing key components and interfaces"

Figure 4: Placement of WP 36 into WP26 Modular Platform Architecture

The purely technical realisation and implementation do not, however provide sufficient clarity on responsibilities, which addresses a central demand for the System Pillar's emerging CCS architectures. Future certification concepts will be needed to strengthen the feasibility of a multivendor ecosystem, where applications can be supplied, integrated, tested, validated and assessed more independently than today.

Such evidence—for example, regarding modular hosting of up to SIL4 applications—can be derived from work that happened in WP26 and is presented in the deliverable D26.4 "Summary of findings and recommendations from study on modular certification and homologation".

The assessment strategy of such a modular system starts with the assessment of the modular platform as a generic product according to EN 50129. Applications running on top can then be assessed individually as generic applications. The specific application instance, as well as the integration of all applications, then needs to be assessed as specific application. To ensure that the responsibilities of the involved actors are clear—particularly the integration aspect of applications of different vendors—please refer to the chapter of WP26 listed above.

The blueprint, presented later in detail in Chapter 4 WP36 Onboard Platform Architecture and Reference Implementation aligns with relevant input sources, beyond the Technical Enablers and System Pillar domains illustrated in Figure 1, as:

- EuroSpec efforts on software update [4] harmonisation,
- OCORA [5] principles for open on-board architectures,
- EN 50159/50716 system-level non-interference considerations,
- IEC 62443 / TS 50701 cybersecurity zoning concepts, etc.

The presented content focuses mainly on Modular Platform blueprints, enabling FRMCS failover. and the provision of standardised diagnostics interfaces, supporting future updates to the TSI CCS, especially regarding:

- On-board architecture modularity,
- Communication-layer responsibilities,
- Integration of FRMCS transport into the Modular Platform Concept and CCS functional chain.

Still, the demonstrator clarifies technical feasibility and provides evidence on applicable boundaries between platform, safety layer, and hosted applications as introduced in the WP26's Modular Platform Concept, with a focus on Application-Level Platform Independence (ALPI). Please refer to D26.3, Chapters 5 and 6 and the corresponding requirements that get presented and assessed in APPENDIX A - Assessment of WP26 Requirements.

This combined evidence is intended to accelerate consensus around European interoperable on-board computing standards and may be considered in future pilots and industrialisation activities.

2.4 INPUT TO STANDARDISATION AND TSI EVOLUTION

Demonstrator work packages act as a bridge between research outcomes and the Standardisation and TSI Input Plan (STIP). In essence, they translate research into deployable, trusted evidence that enables new technologies to evolve into future European standards and, where appropriate, future TSI obligations.

The STIP is the negotiated pipeline through which ERJU research results are prepared for inclusion into:

- European standards (CENELEC, EN, ETSI, ISO),
- ERA-led TSI revisions,
- Operational rules,
- System architecture baselines.

Demonstration WPs support this process by generating validated evidence for their allocated STIP items. Each STIP item requires:

- Performance evidence,
- Safety argumentation,
- Operational feasibility data.

The following Figure 5—taken from the Europe's Rail online presence[4]—provides a high-level overview of the STIP process. Further supporting information and the STIP itself have been published online as approved documents in "Version V2.0".[5]

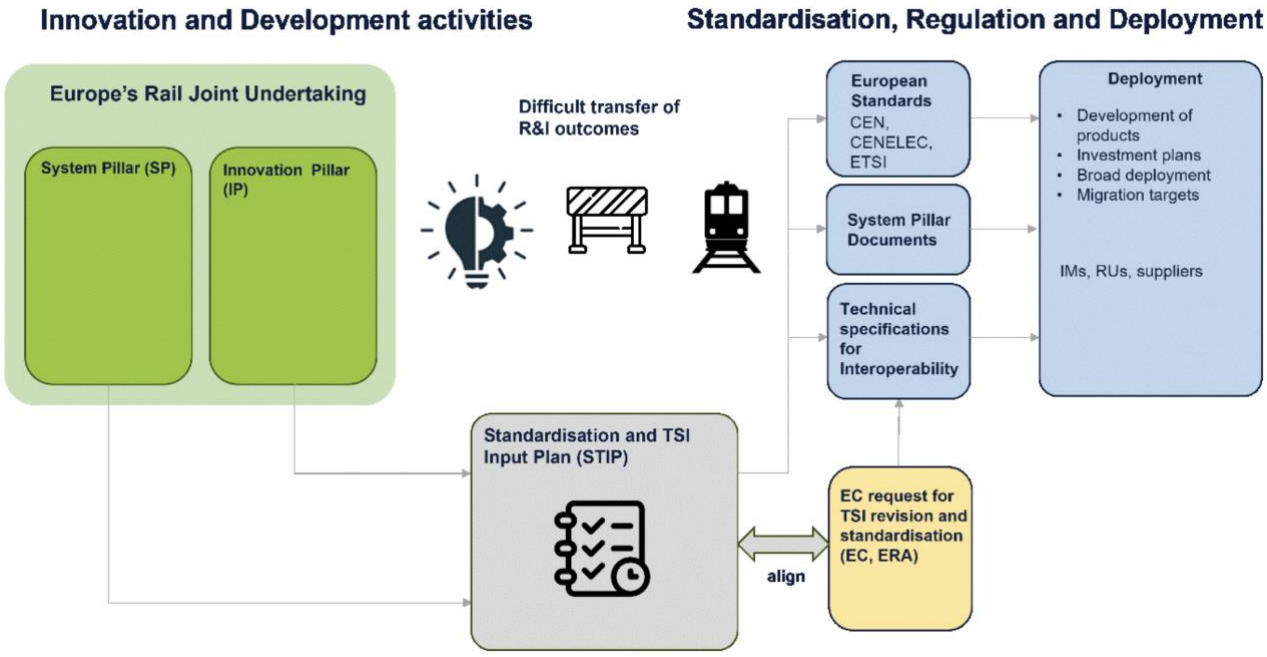


Figure 5: Streamlining Railway Harmonisation: Introducing the EU-Rail STIP

[4] ER JU home page, [Online] available March 2026: https://rail-research.europa.eu/latest-news/streamlining-railway-harmonisation-introducing-the-eu-rail-stip/

[5] ER JU home page, [Online] available March 2026: https://rail-research.europa.eu/standardisation-and-tsi-input-plan/

WP36 and other demonstrators form an essential first step in de-risking innovations before they reach ERA and the drafting of TSIs, as ERA's System Authority will only accept inputs that:

- are technically mature,
- have safety argumentation,
- demonstrate interoperability.

Demonstrator-WPs produce the required Technical Maturity Level (TRL) and generate interoperable results that show consistency with the System Pillar architecture.

STIP items must align with:

- System capabilities/functions,
- Interfaces definitions, and,
- Migration paths.

Demonstrations validate this alignment, providing actionable input for several standardisation fronts, such as the future evolution of the TSI CCS and the broader European harmonisation process.

The following STIP-items can be mapped to TEs and corresponding system capabilities/functions:

Table 2: Relevant STIP-Items

STIP ID	STIP Topic	Technical Enabler	ERJU System or Product Capability / Function
STIP_4	Standardisation of Computing Environment	TE16 Modular Platform	Providing safe and secured digital modular platform Support the implementation of on-board applications
STIP_68	Ethernet CCS Consist-Network (full stack)	TE 17 Onboard Communication Network	Providing safe, secured, and seamless on-board communication
Multiple (STIP_48+)	FRMCS, e.g. V1' / V2 Specification Documents	TE 5.1 FRMCS	Providing safe, secured, and seamless communication between train and infrastructure

Work in WP36 is related to additional STIP Topics:

- STIP_7 Specification of Standardized Maintenance Interface,
- STIP_10 Service Function Diagnosis,
- STIP_2.0_175 Cybersecurity,
- STIP_79 Modular Safety Case Structure.

STIP Topics that may be supported by TE 16 Modular Platform in the future (non-exhaustive):

- STIP_67 On-board Repository (exemplary as pot. mixed-criticality application),
- STIP_70 Multi-Display On-board.

2.5 ENABLER FOR FUTURE PILOTS AND INDUSTRIALISATION

Deliverable D36.5 consolidates WP36 results into a blueprint aligned with these STIP expectations and future TSI-relevant developments. As shown in earlier sections, the demonstrated platform features—modularity, deterministic behaviour, replication concepts, unified diagnostics, and FRMCS-integrated communication—provide valuable evidence supporting the evolution of European interoperable and interchangeable on-board systems.

By translating implementation results into structured evidence, WP36 enables the acceleration of finding consensus on European level. These results form a key contribution to the STIP pipeline and ensure that modular onboard technologies can be considered for future European level standardisation, migration planning, and harmonised architecture development, and the long-term industrialisation of modular on-board solutions. The demonstrated results show that a vendor-neutral, modular on-board architecture—integrating FRMCS communication services, diagnostics, and train-integration capabilities—is not only technically feasible but also operationally robust and aligned with Europe's long-term objective- of a fully digitalised, interoperable, and future-proof rail system.

The blueprint and evidence produced throughout WP36 offer concrete architectural patterns, validated integration models, and cross-ERJU alignment that together form a solid foundation for the next steps of deployment and standardisation.

Building on these outcomes, Deliverable D36.5 provides clarity on how platform technologies demonstrated in the laboratory can be extended, scaled, and industrialised. Key aspects for future pilots and beyond include:

- Scalability of modular architectures to heterogeneous on-board hardware and multivendor software ecosystems, enabling flexible and cost-efficient deployment strategies across different vehicle classes and operators.
- Integration of FRMCS on a Modular Platform, demonstrating how future safety-related communication services of hosted applications can be supported, abstracted, and securely connected to wayside operational environments.
- Identification of operator side infrastructure needs, such as Identity and Access Management (IAM), Public Key Infrastructure (PKI), secure update governance, and OTA capable backend solutions that are essential for full operational deployment.
- Bridging diagnostics and TCMS data services with predictive maintenance ecosystems, enabling condition-based maintenance and fleet-level analytics rooted in harmonised data models and open interfaces.
- Preparation for field grade implementation, outlining the steps required to evolve from laboratory demonstrators to precommercial pilots, certification relevant validation, and industrial rollout.

Taken together, these elements position WP36 as a strategic enabler for the next phase of the R2DATO programme. They ensure that the validated concepts, architectural patterns, and integration experiences captured in this deliverable can serve as an actionable reference for demonstrators in future EU-RAIL phases, and the broader European standardisation landscape.

By closing the conceptual gap between R&I activities and upcoming industrial deployments, Chapter 2 concludes with a clear view on how the presented results can be transferred, expanded, and utilised to support sector-wide migration towards modular, interoperable on-board platforms.

3 CONSOLIDATION OF THE IMPLEMENTATION TASKS

The three-phase evolution of the WP36 demonstrator within ERJU FP2 R2DATO provides a coherent, evidence-based confirmation that a modular on-board computing platform can integrate mixed-criticality applications from multiple suppliers, including functions potentially up to SIL 4. Across all phases, the demonstrator validated key architectural elements—such as deterministic behaviour, robust replication, FRMCS-based and on-board communication, diagnostics, and train-integration concepts—reaching a TRL 5-6, within a realistic laboratory setup, while ensuring reliability, resilience and interoperability.

Together, these validations position the demonstrator as one of the first integrated proofs that FRMCS + Modular Platform + Modern Onboard Communication can operate reliably in a unified laboratory deployment.

Phase 3 (D36.4) concluded this progression by integrating an existing ETCS application into a full operational and expanded final physical laboratory setup, thereby validating the platform technologies under realistic failure and network conditions. The subcontractor Clearsy was commissioned to document its porting experience and provide a summary report, which is included in APPENDIX B - EVC Porting Experience and Lessons learned.

In fact, software from six different partners and subcontractors has finally been integrated on the Modular Platform, complemented by multiple pre-existing software components in the laboratory environment. This has been technically realised either by following programming and integration models of the used TAS Platform on the embedded rolling stock ready COTS Computing Elements or by the utilisation of virtualisation environments and containers in the surrounding laboratory environment. It has been proven that a multi-vendor approach is feasible as long as the partners are integrating their software components following predefined and specified implementation guidelines and rule sets, with clearly defined responsibilities and roles.

SBB took over the integration lead, acting as the provider of the main laboratory environment, that hosted the CCS on-board constituents. These components were remotely connected to the Kontron laboratory, which operated the FRMCS core infrastructure. Hitachi supplied the TAS Platform and embedded hardware boards, along with the necessary guidance, documentation and technical support to enable its use and to facilitate the integration of both, pre-existing and newly developed software provided by Kontron, DB, SBB, Clearsy and AVL-Ditest.

3.1 DEMONSTRATED FEASIBILITY OF A MODULAR, MULTI-VENDOR PLATFORM

Across the three implementation phases, the project built consistently upon the Hitachi/GTS TAS Platform, demonstrating that a modular, COTS-based platform can host mixed-criticality applications—from basic-integrity functions to software requiring up to SIL 4—without compromising determinism or reliability. The demonstrator further confirmed the feasibility of integrating applications of different suppliers by porting and operating a third-party ETCS Baseline 3 (SV 2.1) application in a 2oo3 configuration, showing that otherwise safety-critical functions can be decoupled from the hardware also for third-party provided applications without compromising determinism or reliability.

This result closed the loop established in earlier phases, where communication, replication, failover, and safety API models were proven reusable for potentially safety-critical and non-safety-critical applications of different suppliers. The ETCS integration in D36.4 successfully validated the platform's suitability for integrating pre-existing railway functions in a complex operational environment.

3.2 ROBUST FRMCS INTEGRATION WITH TRANSPARENT FAILOVER

The integration of FRMCS advanced progressively over the three phases, culminating in a fully replicated end-to-end, on-board-to-trackside communication chain.

- D36.2 introduced the early architectural building blocks.
- D36.3 implemented the FRMCS on-board Gateway, OB_{APP} interface, and first end-to-end- messaging.
- D36.4 validated fully replicated FRMCS communication chains, enabling ETCS–RBC SUBSET-26 exchanges with transparent failover between replicas. When a computing element (CE) or FRMCS bridge replica fails, another replica seamlessly reestablishes the RBC connection, ensuring uninterrupted operation.

The demonstrator thus provides strong evidence that FRMCS based communication can meet ETCS L2 availability expectations under modular platform conditions. ETCS–RBC communication over FRMCS behaved reliably even during replica restarts, platform failover events, and link interruptions, demonstrating the viability of FRMCS for future safety-relevant applications when combined with platform-managed redundancy.

3.3 Platform-Level Resilience and Recovery Confirmed

Across D36.3 and D36.4, the platform's fault detection, isolation, and reintegration capabilities were systematically tested in different scenarios:

- Automatic restart of failing replicas, demonstrated with software fault injection,
- Recovery from complete power loss of one computing element (CE),
- Continued ETCS operation in degraded (2oo2) configuration.

The tests in D36.4 go further by showing that even during ETCS L1/L2 transitions and FRMCS session resets, the application continues operating safely. These results show that platform-managed failover mechanisms can ensure continuous behaviour of hosted applications and train control, even without bespoken hardware redundancy beyond the platform's replication.

3.4 Diagnostics and TCMS Data Integration

The diagnostics functionality matured significantly across the phases, resulting in the MCP-DIA C-Sample implemented in D36.4. The demonstrator established a unified SOVD-based diagnostics architecture capable of supporting both periodic and event-driven data provision.

Validated capabilities include:

- REST-based and configurable network access to platform internal and external clients,
- A consistent modelling approach for cross-application diagnostics,
- Seamless access to diagnostics data via standardized SOVD interfaces,
- Deterministic diagnostics from replicated safety applications,
- Syslog integration, providing harmonised logging information of the safety layer and hosted applications.

In parallel, the Train Integration functional cluster implementation – initiated in D36.2 and shaped in D36.3 reached a full demonstratable state in D36.4:

- Integrating a working TCMS Data Service (TCMS_DS) in the lab based on WP31 concepts,
- Integrating the DIA-VEC application on the target platform, evaluating and aggregating vehicle brake states (via TCMS_DS) and executing rule-based degradation logic,
- Automated end-to-end testing via Azure DevOps pipelines, ensuring reproducible evaluation.

Together, these components demonstrate how vehicle-level data aggregation, diagnostics, and condition-based maintenance can be realised on a Modular Platform.

3.5 Communication and Network Reliability Validated

Networking behaviour was validated extensively in D36.4, confirming that mission-critical connectivity can be ensured in modular environments, using standard networking technologies, in line with Subset-147 [6] specifications but without bespoke railway-specific custom protocols.

The demonstrator validated several demanding networking scenarios:

- Uninterrupted communication during CCN cable failure thanks to RSTP ring redundancy (temporary delay only).
- Location-transparent communication across replicated bridges.
- Simultaneous FRMCS sessions for multiple on-board applications.

Additional User Stories have been added to APPENDIX C - Remaining User Stories, Test Cases and Results to conclude the selected networking scenarios.

3.6 Security and Update Concepts Defined (Theoretical Level)

Security and software update user stories—introduced in D36.2 and conceptually refined in D36.3—are theoretically evaluated in D36.4**:**

- Alignment with TS 50701, IEC 62443, EN 50159.
- Need for integration with the operator's IAM, PKI/certificates, and secure update governance.
- Supported updates and rollback functions by the TAS Platform. Since OTA requires operator infrastructure, this was not realised in the course of the demonstrator.

Although operator environments were beyond project scope, the study analysed which update and security functions can be provided by the platform, and which must be ensured by operators or future standards.

3.7 Summary of the User Stories

At the start of the project, a structured set of User Stories was defined to capture the central questions related to all covered technical domains of the demonstrator. These User Stories served as an organising mechanism to ensure coverage of the essential aspects of Modular Platform, FRMCS integration, communication behaviour, diagnostics, IT/OT Security, SW Update, train-integration functionality and application migration. Each User Story was associated with one of several thematic clusters and assigned to a specific implementation phase, supporting early results where feasible while maintaining flexibility as the demonstrator design evolved.

During the course of the project, the User Stories were reviewed and refined repeatedly. Some were consolidated where overlaps became apparent, others were identified as out of scope for practical implementation, for example due to excessive complexity or limited relevance to the WP36 remit. In such cases, analytical assessments were conducted to preserve the underlying insights. This iterative process resulted in a sharpened set of evaluations aligned with the central objective of demonstrating Modular Platform capabilities and their interaction with key on-board services.

The following table provides a consolidated overview of all User Stories and their processing status. For each entry, the User Story ID, thematic cluster, short description, status (tested, analysed, omitted), and the deliverable in which it is documented (D36.1 to D36.5) are presented to ensure transparency and traceability across the WP36 development flow.

Table 3: Status and Documentation of all User Stories

Group / Cluster	User Story according to D36.1	Title (for the description, please refer to [2] D36.1 User Stories document)	Status	Docu-mented in
Deploy& Orchestration				
	[2.1.1]	Run a Parallel Basic Integrity Application on Platform	tested	D36.2
	[2.1.2]	Run a Basic Integrity Application on Safety Layer / RTE	tested	D36.2
	[2.1.3]	Run a Safe Application (up to SIL 4) on Safety Layer / RTE	tested	D36.2
	[2.1.4]	Run Multiple Applications	tested	D36.2
	[2.1.5]	Execute Declarative Configuration	tested	D36.2
	[2.1.6]	Restart Failing Replicas	tested	D36.4
	[2.1.7]	React to Hardware Failure	tested	D36.4
	[2.1.8]	Failover to Cold Backup Hardware	omitted	(D36.4)
	[2.1.9]	Replace Failing Hardware	tested	D36.2
Modularity				
	[2.2.1]	Change to Different Hardware	omitted	(D36.4)
	[2.2.2]	Change to Different Platform or Safety Layer / RTE	omitted	(D36.4)
	[2.2.3]	Use Diverse Hardware	omitted	(D36.4)
Communication				
	[2.3.1]	Communicate Safe App on RTE <-> Safe App on RTE	tested	D36.2
	[2.3.2]	Communicate Safe App on RTE <-> External Safe App	omitted	(D36.4)
	[2.3.3]	Communicate Safe App on RTE <-> Basic Integrity App on RTE	tested	D36.2
	[2.3.4]	Communicate Safe App on RTE <-> Parallel Basic Integrity App	tested	D36.2
	[2.3.5]	Communicate Safe App on RTE <-> External Basic Integrity App	tested	D36.4
	[2.3.6]	Communicate Basic Integrity App on RTE <-> Basic Integrity App on RTE	tested	D36.2
	[2.3.7]	Communicate Basic Integrity App on RTE <-> Parallel Basic Integrity App	tested	D36.2
	[2.3.8]	Communicate Basic Integrity App on RTE <-> External Basic Integrity App	tested	D36.2
	[2.3.9]	Communicate Parallel Basic Integrity App <-> External Basic Integrity App	tested	D36.4
	[2.3.10]	Communicate Independent of RTE Instance	omitted	(D36.4)
	[2.3.11]	Communicate Independent of Replication	tested	D36.4
FRMCS				
	[2.4.1]	Communicate over FRMCS Transparent to the Application	tested	D36.3
	[2.4.2]	Reuse FRMCS Platform Functions	tested	D36.4
	[2.4.3]	Integrate an Application with FRMCS over Loose Coupling	tested	D36.3
	[2.4.4]	Failover to Redundant FRMCS Platform Functions	tested	D36.3
	[2.4.5]	Host FRMCS Onboard Services on the Modular Computing Platform	tested	D36.3
Diagnostics (MDCM)				
	[2.5.1]	Collect Diagnostics Data from External Basic Integrity App	tested	D36.4
	[2.5.2]	Collect Diagnostics Data from Basic Integrity App on RTE	tested	D36.4
	[2.5.3]	Collect Diagnostics Data from Safe App on RTE	tested	D36.4
	[2.5.4]	Collect Diagnostics Data from Safety Layer / RTE	tested	D36.4
	[2.5.5]	Provide Diagnostics Events to External Basic Integrity App	tested	D36.4
	[2.5.6]	Provide Diagnostics Events to Basic Integrity App on RTE	tested	D36.4
	[2.5.7]	Provide Diagnostics Events to Safe App on RTE	omitted	(D36.4)
	[2.5.8]	Provide Diagnostics Events to Safety Layer / RTE	tested	D36.4
	[2.5.9]	Exchange Diagnostics Data with Trackside Diagnostics Service	omitted	(D36.4)
	[2.5.10]	Collect and Aggregate Diagnostics Data from Vehicle	tested	D36.4
Software Update				
	[2.6.1]	Update a Basic Integrity Application on Platform	analysed	D36.4
	[2.6.2]	Update a Basic Integrity Application on Safety Layer / RTE	analysed	D36.4

Group / Cluster	User Story according to D36.1	Title (for the description, please refer to [2] D36.1 User Stories document)	Status	Docu-mented in
	[2.6.3]	Update a Safe Application on Safety Layer / RTE	analysed	D36.4
	[2.6.4]	Update the Safety Layer / RTE	analysed	D36.4
	[2.6.5]	Update the Platform	analysed	D36.4
	[2.6.6]	Rollback a Software / Configuration Update	analysed	D36.4
	[2.6.7]	Perform a Software / Configuration Update over the Air (FRMCS)	analysed	D36.4
Onboard Network				
	[2.7.1]	Allow Exchange of Data over the Network	tested	D36.5
	[2.7.2]	Provision of Uninterrupted Connectivity	tested	D36.4
	[2.7.3]	Allow Management Access to Onboard Components	omitted	(D36.4)
	[2.7.4]	Allow Ease of Access to the Network	omitted	(D36.4)
	[2.7.5]	Supply List of Onboard Network Components	omitted	(D36.4)
	[2.7.6]	Provision of Date Time Reference	omitted	(D36.4)
	[2.7.7]	Prioritise Specific Network Traffic	tested	D36.5
	[2.7.8]	Monitor the Network Health Status / Guaranteed Characteristics	tested	D36.5
IT/OT Security				
	[2.8.1]	Realize Authentication and Authorization Mechanisms	analysed	D36.4
	[2.8.2]	Change Identity and Access Management Settings	analysed	D36.4
	[2.8.3]	Encrypt all Data on the Network	analysed	D36.4
	[2.8.4]	Realize Secure Communication over FRMCS	analysed	D36.4
	[2.8.5]	Separation of Network Zones	analysed	D36.4
	[2.8.6]	Detect Unusual Network Traffic	analysed	D36.4
	[2.8.7]	Detect Unauthorised Network Device	analysed	D36.4
Train Abstraction				
	[2.9.1]	Provide Safe Access Train Hardware via a Functional Vehicle Adapter	analysed	D36.4
	[2.9.2]	Provide Access to Train Hardware via a TCMS Data Service	tested	D36.4
Functional Apps				
	[2.10.1]	Execute a Simplified ATO Control Loop	omitted	(D36.4)
	[2.10.2]	Execute a Simplified ETCS Control Loop	tested	D36.4
	[2.10.3]	Cache Map Data from Digital Register	omitted	(D36.4)

3.8 CENTRAL FINDINGS

The demonstrator reached a high level of technical maturity, corresponding to TRL 5-6, through a structured and incremental approach based on three consecutive implementation phases. The results provide strong evidence supporting the validity of the Modular Onboard Computing Platform concept, in alignment with the objectives of Europe's Rail and the R2DATO programme.

A major outcome of the work is the definition and realisation of the Onboard Computing Platform Blueprint. This architecture introduces a clear abstraction from the underlying transport layer and is deployed redundantly across multiple Computing Elements. It combines SIL components, with BI elements including the Communication Bridge and the FRMCS Agent. The overall design intentionally minimises the potentially safety-critical scope, which would also limit a potential certification effort.

Across all implementation phases, WP36 has shown that modular computing platforms can reliably host mixed-criticality applications, including existing ETCS functions, while supporting replicated execution, deterministic behaviour and stable integration of communication, diagnostics and vehicle-data services. The work demonstrated that platform technologies, when combined coherently, offer tangible benefits for migration, evolvability and cost-effectiveness of future on-board CCS solutions, particularly by reducing dependency on supplier-specific implementations. The alignment activities with R2DATO WP3 and System Pillar domains strengthened the understanding

of how such technologies fit into the wider system architecture and identified practical considerations for future harmonisation.

The Onboard Communication Network demonstrated the suitability of Ethernet-based communication combined with VLAN segmentation, in line with SUBSET-147 [6] at Layers 1 and 2. Deterministic behaviour and correct routing were validated, including under redundant ring topology conditions, confirming the feasibility of modern network concepts for safety-related on-board environments.

The diagnostics architecture proved to be both vendor-neutral and scalable, aligned with the OCORA MDCM. An adoption path towards SOVD was established, enabling REST/HTTP-based diagnostics services for railway applications. The implementation supports application logging, platform health monitoring and the exposure of diagnostic data through standardised APIs to external systems.

Train integration capabilities were successfully demonstrated through the implementation of the TCMS Data Service based on YANG models and EuroSpec [7] specifications, as well as the DIA-VEC brake diagnostics system using rule-based evaluation. End-to-end data flows were validated from simulated on-board subsystems through the Modular Platform to external visualisation environments.

Regarding IT/OT security, the TAS Platform as a product has been assessed to conform to IEC 62443-4-1 and IEC 62443-4-2, supporting security levels up to SL3.

3.9 Main Findings (Deliverable D36.2 – WP36 Demonstrator, Phase 1)

The first phase confirmed the feasibility of the Modular Platform concept. The TAS platform successfully demonstrated the capability to host mixed-criticality applications, combining BI software with SIL1 to SIL4 applications within a unified architecture aligned with the interfaces I4/I5 principles.

The programming model of TAS Platform was applied with the use of different TaskSet types. Model 1 TaskSets support replicated safety-critical computations, while Model 3 TaskSets are used for basic integrity functions such as singular input/output and low-level communication. Safe and deterministic communication between these domains is achieved with the use of voted and non-voted queues.

Core safety mechanisms, including deterministic scheduling, replication strategies and 2oo3 voting, were successfully used. These results confirm that a third-party application can easily use the platforms mechanisms while meeting its performance requirements in a mixed-criticality environment.

Twelve User Stories defined in D36.1 were fully implemented and validated using automated testing within a virtual environment. At the same time, the system architecture was further refined, with increased detail provided across key functional domains including the Modular Platform, FRMCS on-board, diagnostics, train integration and IT/OT security.

An automated testing framework based on Azure DevOps was established, enabling reproducible test execution, systematic logging and artefact management. The virtual test environment, based on container technologies such as Docker, proved to be particularly effective for early-stage mixed-criticality validation.

Overall, Phase 1 established a solid foundation for subsequent developments, including the integration of FRMCS, the implementation of TCMS-based diagnostics and the transition towards physical demonstrator environments.

3.10 Main Findings (Deliverable D36.3 – WP36 Demonstrator, Phase 2)

The second phase marked the transition from a purely virtual demonstrator to a fully integrated physical laboratory setup, enabling realistic end-to-end communication between on-board and trackside systems. This step required significant architectural updates, particularly to support FRMCS integration and the introduction of MCP-DIA diagnostics within the Modular Platform.

The communication architecture, combining safety-related and basic integrity components, demonstrated robust failover capabilities within a 2oo3 configuration. In particular, communication sessions remained active even when the hosting Computing Element was powered down, with automatic and transparent re-establishment of sessions at application level.

In parallel, the MCP-DIA diagnostics prototype was implemented as a SOVD-based diagnostics server running on the Modular Platform. It supports request-response interactions, logging services and bulk data handling. An end-to-end diagnostics chain was successfully demonstrated, from on-board applications through the diagnostics server to external clients, including visualisation via a mock MDCM interface.

The phase also established a complete CI/CD pipeline using Azure DevOps, covering build, deployment, automated testing and log collection and cleanup. Automated deployment to physical laboratory hardware was achieved, and test executions were fully reproducible, enabling efficient regression testing and offline analysis.

3.11 Main Findings (Deliverable D36.4 – WP36 Demonstrator, Phase 3)

The third phase focused on full system integration and advanced demonstration scenarios. A key achievement was the successful deployment of a third-party ETCS application on the Modular Platform as a replicated 2oo3 and potentially safety-critical application. The system was fully integrated with FRMCS communication, diagnostics and a comprehensive simulation environment.

The ETCS application demonstrated functional equivalence with traditional architectures through realistic closed-loop simulations, including interactions with balises, RBC, DMI and train dynamics. Communication with the RBC was achieved via the FRMCS stack using the OB_{APP} interface, enabling the exchange of SUBSET-026 messages within a loosely coupled architecture.

The platform demonstrated strong resilience capabilities. Transparent failover mechanisms ensured that communication sessions were maintained without interruption in the event of failures. The system was able to detect, isolate and reintegrate failing replicas, while continuing operation in a degraded 2oo2 mode. Failover and recovery mechanisms were validated across the communication stack, FRMCS Agent, control handler and platform voting logic, including re-synchronisation after reboot.

The diagnostics chain was fully operational end-to-end, covering data provisioning, periodic reporting, server-side processing, REST-based access and syslog integration. In addition, vehicle-level diagnostics were successfully integrated through the combination of TCMS Data Services and DIA-VEC, enabling the evaluation of brake states, fault events and degradation logic based on rule-based processing. Data dissemination through publish/subscribe mechanisms to external visualisation tools was also validated.

Network robustness was demonstrated through redundancy tests, showing no loss of communication sessions during network link failures. The redundant ring topology ensured continuous on-board-to-trackside communication, with only minor and acceptable latency variations.

Finally, a complete ETCS operational scenario was successfully executed in the laboratory environment. This included driver identification, train data input, RBC connection via FRMCS, mode transitions from Standby to SR and FS, balise-triggered transitions and L1/L2 handovers, including during replica failover situations. All scenarios were validated through automated testing, confirming the overall robustness and consistency of the platform.

3.12 Limitations, Remaining Gaps and Path Forward

While WP36 achieved a high level of technical maturity and delivered a comprehensive demonstrator aligned with the R2DATO remit, several limitations shaped the scope and outcomes of the work. These limitations are common for R&I-driven activities and should be viewed as natural boundary conditions rather than shortcomings. In many areas compromises and prioritisation were necessary.

Constraints Related to Platform Diversity and Hardware Availability

WP36 was implemented with a single supplier platform and one set of COTS computing hardware. Planned evaluations of heterogeneous hardware, additional platform suppliers, and portable deployment across different environments could therefore not be performed. These constraints arose from limited asset availability and the absence of a second platform provider. Nevertheless, the demonstrator outcomes remain valid and identify clear opportunities for future expansion.

Dependency on Evolving Specifications

WP36 was conducted in parallel to several evolving specifications within WP26, FRMCS and the System Pillar. Many specifications were not yet stable or available at the required maturity level. As a result, WP36 relied on existing baselines, established industry-standards and practises. For the Modular Platform vendor-specific API extensions, programming and integration models were used. This situation provided important insights into change management and highlighted areas requiring further study.

Scope Restrictions in Safety, Vehicle Integration, and Trackside Interactions

Due to laboratory constraints, several areas could not be fully demonstrated. This includes complete safety-protocol implementations, extended vehicle-adapter functionality, trackside diagnostics exchange, and the use of productive FRMCS radio technology. In the demonstrator, FRMCS communication was realised via Wi-Fi; however, since the concept is designed to be independent of the underlying radio technology and performance measurements were out of scope, this had no impact on the demonstrator results. These topics would require additional hardware, operator infrastructure, or real rolling-stock environments. Such aspects are better suited for future pilots than for a laboratory demonstrator.

Transversal Topics Requiring Additional Work

Key transversal aspects—such as over-the-air updates, configuration and orchestration workflows, IAM/PKI management, and operational cybersecurity—were analysed conceptually but not implemented end-to-end. Their practical validation depends on operator-side backend systems and will require dedicated follow-up activities.

System Pillar Alignment and Future Standardisation Needs

Because the System Pillar Computing Environment domain shifted its focus towards Hardware-Level Platform Independence (HLPI), several ALPI-related concepts from WP26 (interfaces I4/I5) could only be evaluated towards requirement compliance, not validating an established standard. WP36 nonetheless provides essential evidence supporting the Modular Platform Concept and identifies concrete next steps for pilots aligned with the final System Pillar CE specifications.

Limited Evaluation of Multi-Vendor Integration in Operational Conditions

Although WP36 successfully demonstrated multivendor modular integration, certification related aspects and real-world operational governance could not be fully evaluated. Operator-led pilots will be required to explore certification strategies, lifecycle responsibilities, and deployment in operational rolling stock.

Concluding Remark

None of the identified limitations undermine the validity of the demonstrated concepts. Instead, they clearly define the first development steps and provide a roadmap for maturing the Modular Platform concept toward industrialisation, large-scale validation and future European research in field deployment and long-term European harmonisation.

4 WP36 ONBOARD PLATFORM ARCHITECTURE AND REFERENCE IMPLEMENTATION

This demonstrator exercise has validated the integration of key EDT cluster enablers—notably the Modular Platform (TE16), FRMCS integration (TE5.1) and on-board communication components (TE17)—according to the remit and implementation consolidation set out in Chapters 2 and 3. The work concentrated on the Modular Platform as the principal technical focus: a supplier-specific platform instance was used to exercise the WP26 MPC architecture and requirements, demonstrating mixed-SIL deployments, redundancy behaviours and the communication patterns required for both potential safety-critical and basic-integrity functions.

Virtualization and containerization approaches were evaluated but were not required for the on-board reference implementation within WP36; these technologies remain relevant for future use where scalability, maintainability or multi-tenant hosting are.

Continuation and maturation are recommend in future implementation waves, shifting emphasis as indicated by the System Pillar CONEMP domain to virtualization and containerization approaches leveraging the interfaces I2 and I3 (for an overview of the System Pillar CONEMP proposed interfaces, please refer to Figure 8 in Chapter 4.2.2).

4.1 BLUEPRINT FOUNDATIONS FOR FUTURE EU-RAIL PHASES

Deliverable D36.5 collects and disseminates the demonstrated results into a blueprint by synthesising insights from the implementation phases and aligning with the coherent R2DATO architecture and organisational framework. This blueprint aligns with:

- Reusable patterns for communication, replication, diagnostics, and on-board data integration
- Interfaces and APIs suitable for standardisation
- Test and validation workflows, including DevOps-driven deployment and automated integration pipelines

These foundations can support and prepare EURAIL to execute largescale pilots, strengthen cross-vendor interoperability, and develop harmonised requirements for future on-board systems.

4.2 BLUEPRINT ARCHITECTURE

4.2.1 Physical Architecture

The Onboard Demonstrator consists mainly of the TAS Platform and all its integrated software components including the ETCS on-board application, the FRMCS TOBA system as the link to the FRMCS infrastructure, the FRMCS Core hardware and software, the routing and connectivity devices for communication as well as the simulation and test environments deployed on a dedicated laboratory server hardware.

The lab is geographically distributed across two sites: on-board hardware and software, plus the trackside simulators, are located at SBB in Switzerland, while the FRMCS core hardware and software are hosted at Kontron in Germany.

Figure 6 presents a simplified architecture overview; for a detailed view, see Chapter 2.3 of deliverable D36.3. At its core is the TAS Platform, which comprises three COTS Computing

Elements (CE) and their power supplies, hosting a supplier-specific implementation of the Modular Platform Concept (see deliverable D26.3).

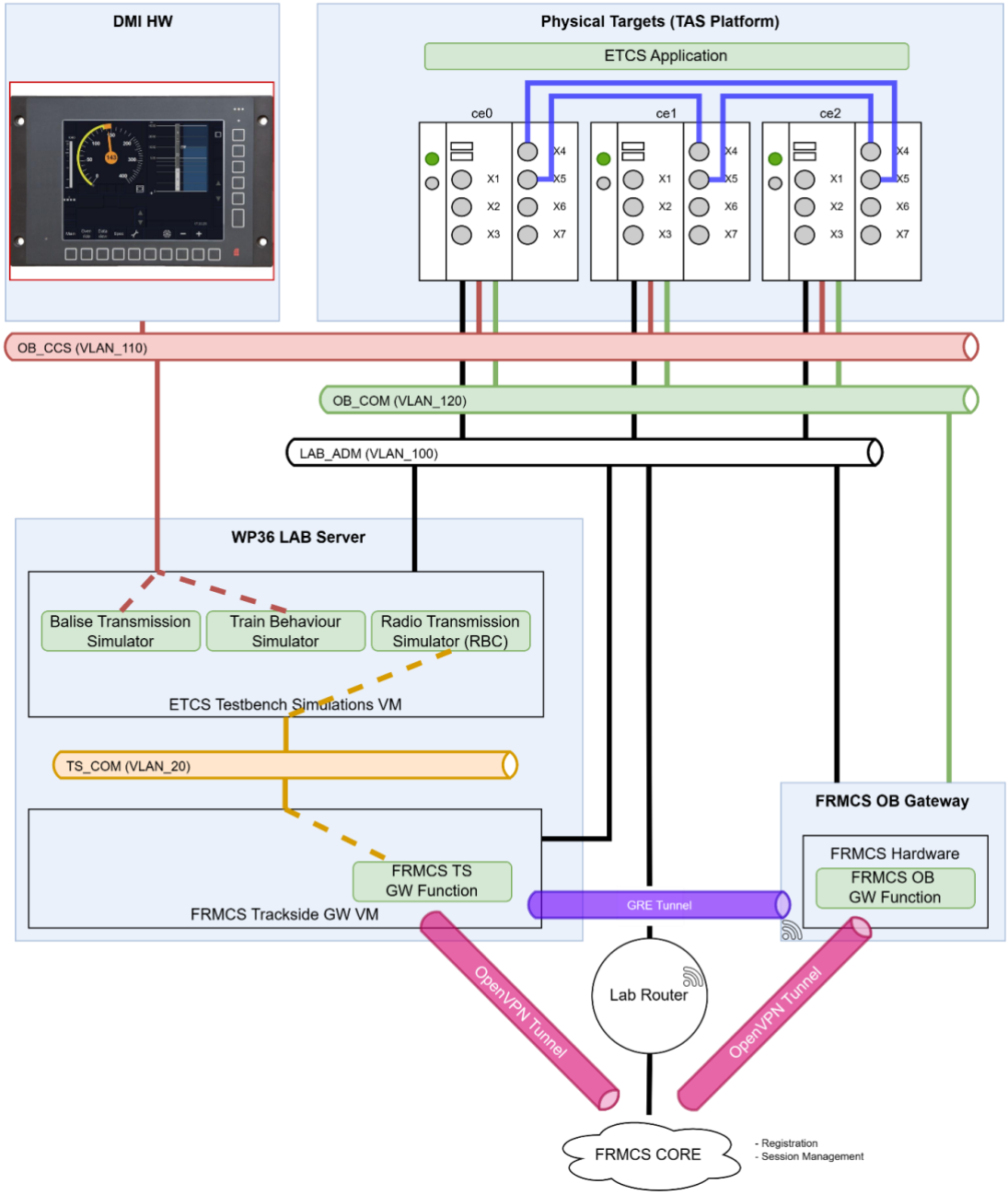


Figure 6: Physical Architecture Overview (simplified)

Internal communication within the platform uses point-to-point Ethernet links between all three CEs. For external communication with trackside and other on-board systems, the CEs connect to the CCS Consist Network, spinning over four HW switches arranged in a ring. The switches are configured to use the Rapid Spanning Tree Protocol (RSTP) so that routes are automatically recalculated in the event of a switch failure or a physical link loss. For increased availability, each Computing Element is attached to a different CCN switch.

Communication with trackside systems is provided via FRMCS. A TOBA box hosts the FRMCS on-board gateway plus the modems/Wi-Fi required to connect to the FRMCS core via the Lab Router (access point) and the internet. For simplicity, the lab setup uses Wi-Fi instead of 5G to bridge the air gap to the access point. The FRMCS on-board gateway uses an OpenVPN tunnel to the FRMCS

core at Kontron (Germany). Similarly, the simulated FRMCS trackside gateway running on a virtual machine on the lab server connects to the same FRMCS core via a second OpenVPN tunnel.

On-board applications and trackside services exchange traffic over a GRE tunnel that the FRMCS core establishes between the on-board and trackside gateways when a session is initiated.

VLANs with different priorities separate communication between the ETCS application running on the TAS platform, the physical DMI, and the trackside services hosted on a virtual machine on the lab server.

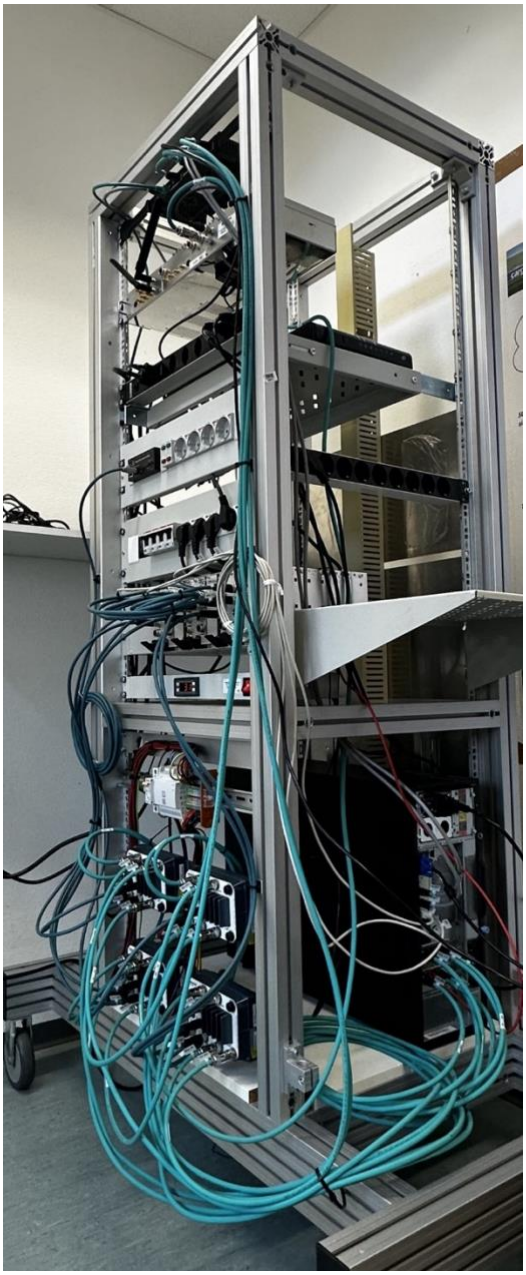


Figure 7: Physical Laboratory Setup

Figure 7 shows the physical rack at SBB in Switzerland. The four CCN switches are visible at the lower left, and the lab server is at the lower right. The TAS Computing Elements occupy the centre-left of the rack. At the very top is the lab router (access point), with the FRMCS TOBA box directly beneath it.

For further details about the physical architecture please refer to chapter 2.3 of deliverable D36.3.

4.2.2 Functional Cluster Modular Platform

The Onboard Platform Demonstrator is built on the TAS Platform. This modular implementation provides supplier-specific abstractions between applications, the runtime, and the safety layer. Its interfaces are based on the POSIX/Linux API and extend functionality in a manner developed and overseen to meet SIL4 requirements. In the Modular Platform Concept of WP26 this API refers to Application-Level Platform Independence. In the layered architecture defined by the System Pillar CE domain, the related Interfaces I4 and I5 reside above the Runtime Layer as depicted in the following Figure 8.

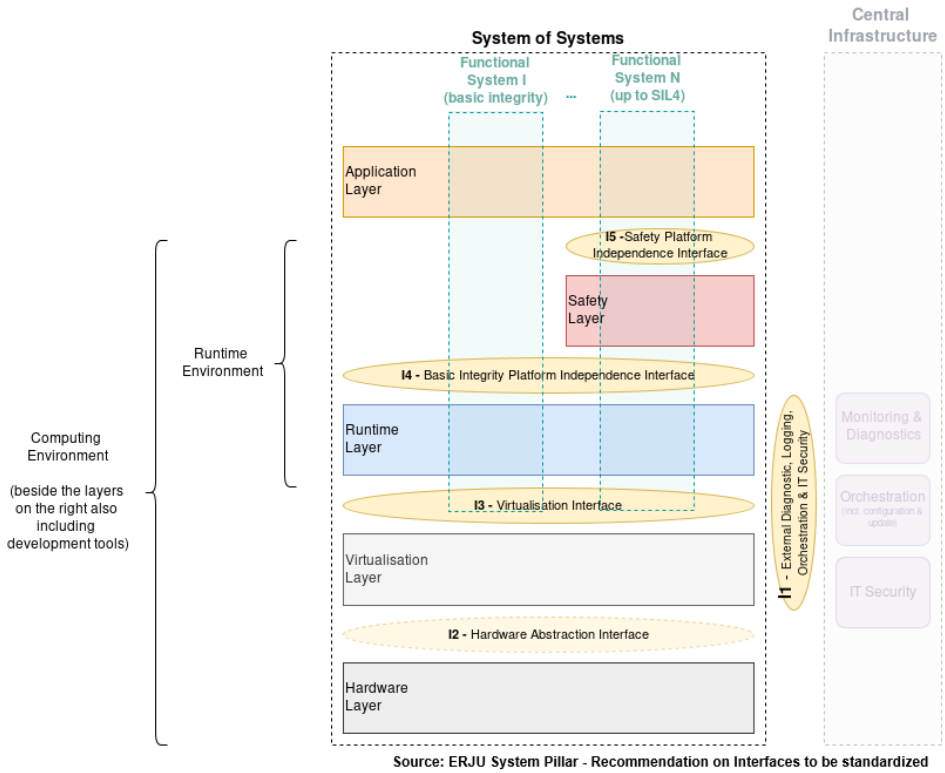


Figure 8: System Pillar proposed Layered Architecture and Interfaces

1. TAS Platform Architecture

The TAS Platform is built on modern, industry-standard software technologies and a modular, scalable architecture that supports real-time, multitasking applications.

It runs on commercial off-the-shelf (COTS) hardware suitable for the railway environment. The platform's core comprises key software subsystems: the operating system, communication services, and fault tolerance mechanisms.

The communication subsystem implements standard services and safety enhanced semantics, plus safety relevant protocols that align with CENELEC requirements. Fault tolerance is provided across hardware boards or virtual machines through composite fail safety mechanisms. Individual boards or virtual machines are referred to as Computing Elements (CE). Multiple CEs (for example in a 2oo3 configuration) form a Computing Node (CN), which is the unit that implements and delivers a safe function.

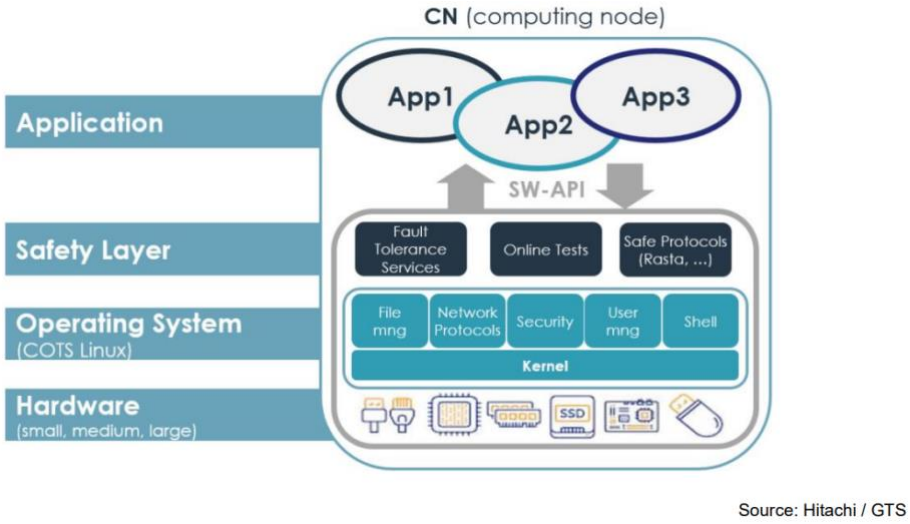


Figure 9: TAS Platform Layered Architecture

The TAS Platform as well follows a layered architecture (see Figure 9). The top layer contains the system's functional components—the applications. The middleware section (black boxes) provides an implementation of the API and a bridge between the programming models included in the safety layer, and the operating system on top of the hardware layer.

The TAS Platform's POSIX/Linux API with extended functionality implements its own flavour of the I4 and I5 interfaces as proposed by the System Pillar's Modular Platform architecture (see Figure 8). Together with its programming model the TAS Platform offers, regarding I4 and I5, a supplier-specific implementation related to the Modular Platform Concept (as developed in WP26).

For more details, see D36.2, chapter 2.1.1.

2. TAS Platform Programming Model

The TAS Platform programming model structures each application as a collection of TaskSets that implement its functionality. TaskSets come in two types:

- Model 1 TaskSets: for safety critical computation. They are replicated, use a restricted POSIX subset, and produce outputs that can be voted.
- Model 3 TaskSets: for non-safety functionality. They are not replicated for safety, may perform I/O and communicate with other POSIX threads. Redundant I/O can be achieved by running multiple Model 3 TaskSets on different computing elements.

Model 1 TaskSets communicate only via TAS Platform message queues.

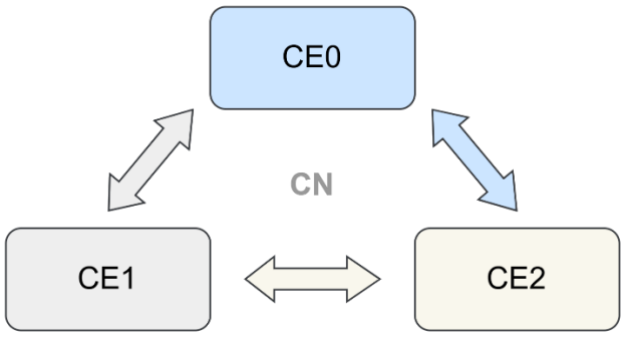


Figure 10: 2oo3 Redundancy Architecture

In the 2oo3 redundancy setup shown in Figure 10, each Computing Element (CE) runs an instance of every Model 1 TaskSet; Model 3 TaskSets run on a single CE.

Application execution has three phases—start-up, operation, and shutdown—with different API usage rules:

- Start-up: Model 1 TaskSets must allocate all required resources and prepare for operation. They signal readiness by calling a defined completion function, which blocks until all replicated instances have started on the required CEs and all checks pass.
- Operation: Model 1 TaskSets execute safety critical functionality using a very restricted subset of the API. No dynamic resource allocation is allowed. I/O and external communication must be done through specific communication libraries (for example OCS) or delegated to Model 3 TaskSets.
- Shutdown: The TAS Platform prevents normal application execution; only diagnostic tasks are permitted.

4.2.3 Functional Cluster FRMCS Onboard

A safe and secure end-to-end communication between applications hosted on the Modular Platform and their peers on the trackside is a key functionality for an on-board system. Therefore, a common architecture for safe and secure external communication is an important aspect of the Onboard Platform Demonstrator.

The approach is to centralize FRMCS-specific protocol details and communication implementation within a separate, reusable gateway component and an FRMCS agent. This allows for greater flexibility and modularity within the system architecture.

Figure 11 depicts the architecture, focusing on one computing element (CE). In the demonstrator's 2oo3 redundancy setup, all three CEs use the same architecture but, at any given time, only one CE has an active Bridge/FRMCS Agent. The other two CEs are ready to take over, in case the active CE fails. The architecture uses a combination of Model 1 and Model 3 Tasksets communicating via message queues (voted and POSIX queues).

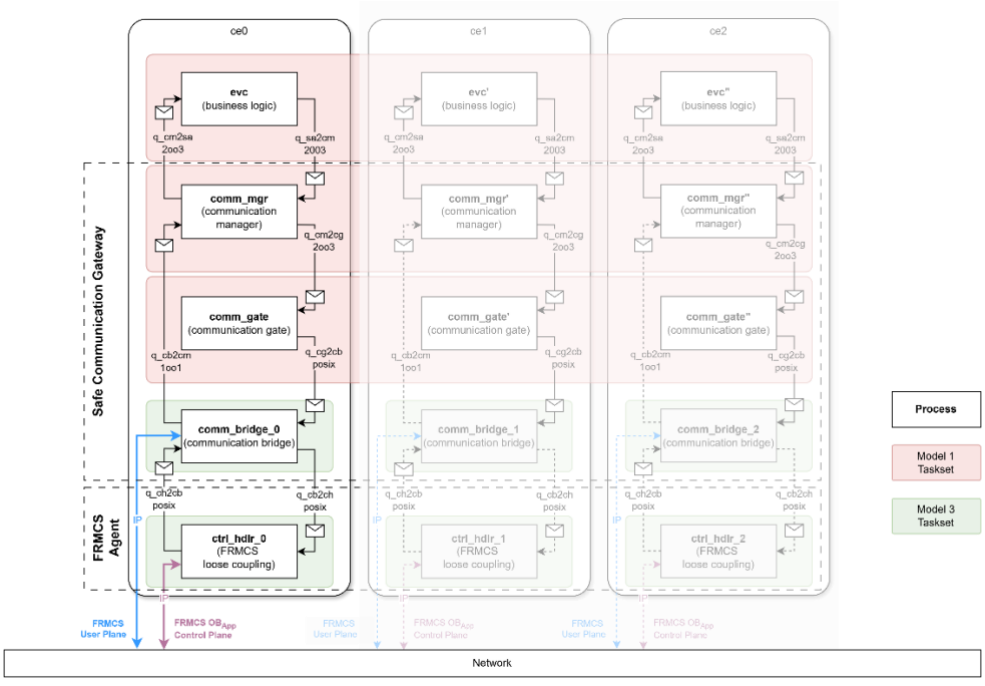


Figure 11: Architecture for safe & secure Communication via FRMCS

Business Logic (EVC): It represents the safe application logic that requires safe and secure communication with trackside systems.

Communication Manager: It manages and implements safety protocols for communication sessions required by the safe application logic, handling session control requests (establish/terminate), lifecycle monitoring (heartbeats, sequence numbers, timestamps) per CENELEC standards, and automatically triggering recovery and re-establishment when failures are detected. It also selects the computing element responsible for physical Ethernet communication and, if that CE fails, automatically re-establishes all active sessions via other CEs, without requiring an application safety response if safety timeouts are not breached.

Communication Gate: It is a platform-specific helper task that aids recovery of replicated Model 1 TaskSets communicating with non-replicated Model 3 TaskSets via message queues.

Communication Bridge: It bridges the platform's internal message-queue communication and the CCN Ethernet network, managing user-plane connections with peer services by handling OSI layers 4–6 and maintaining TCP/IP connections to offboard services. It uses the Control Handler to set up IP tunnels and addresses.

FRMCS Agent (Control Handler): It mediates control-plane exchanges with the on-board FRMCS TOBA Box via the OB_{App} interface (see Figure 12), receiving requests from the Communication Bridge, translating them into OB_{App} control sequences, and establishing/managing IP tunnels for user-data sessions with offboard services; it returns assigned IP addresses to the Communication Bridge and sets up the FRMCS User Plane connection to the trackside gateway.

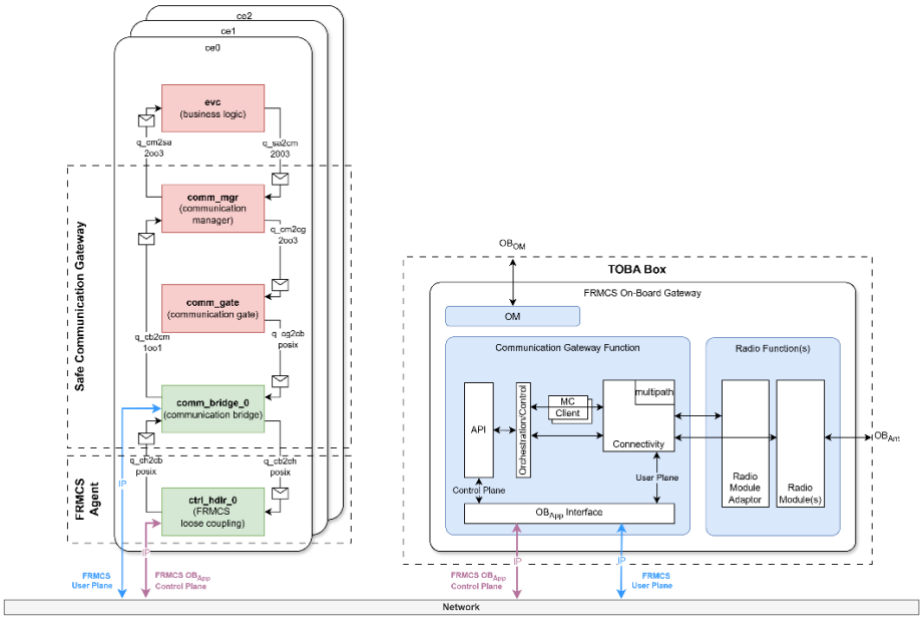


Figure 12: FRMCS On-Board Gateway

In this demonstrator, each application includes its own instance of the above-shown Safe Communication Gateway (see Figure 12, dotted area) and FRMCS Agent.

This demonstrator offers a blueprint architecture for safe and secure FRMCS based communication with trackside systems supporting automatic, transparent (to the safe application logic) failover in case of a single CE failure, that can be reapplied for multiple applications.

FRMCS On-Board Gateway redundancy has been discussed theoretically in chapter 3.1.3 of deliverable D36.3. Both discussed options can be accommodated into the above described safe & secure communication architecture.

For further details please refer to D36.2 chapter 2.2, D36.3 chapter 3.1.

4.2.4 Functional Cluster Diagnostics

The primary objective of the diagnostics implementation is to demonstrate how health and performance data can be obtained on the Modular Platform and provided to and external diagnostics client through a common interface.

The realization follows the recently published automotive diagnostic standard SOVD. The standard defines a HTTP-REST-based approach to access diagnostic data, and to call procedures. For a particular SOVD instance, the REST API and associated services are configured using an OpenAPI-based scheme.

Diagnostics clients can read health and performance data using SOVD API's HTTP-GET method in combination with corresponding URIs describing the targeted entity (e.g. an application hosted on the platform) and type of diagnostic information, e.g. logfiles (bulk-data), faults, or parameters (data).

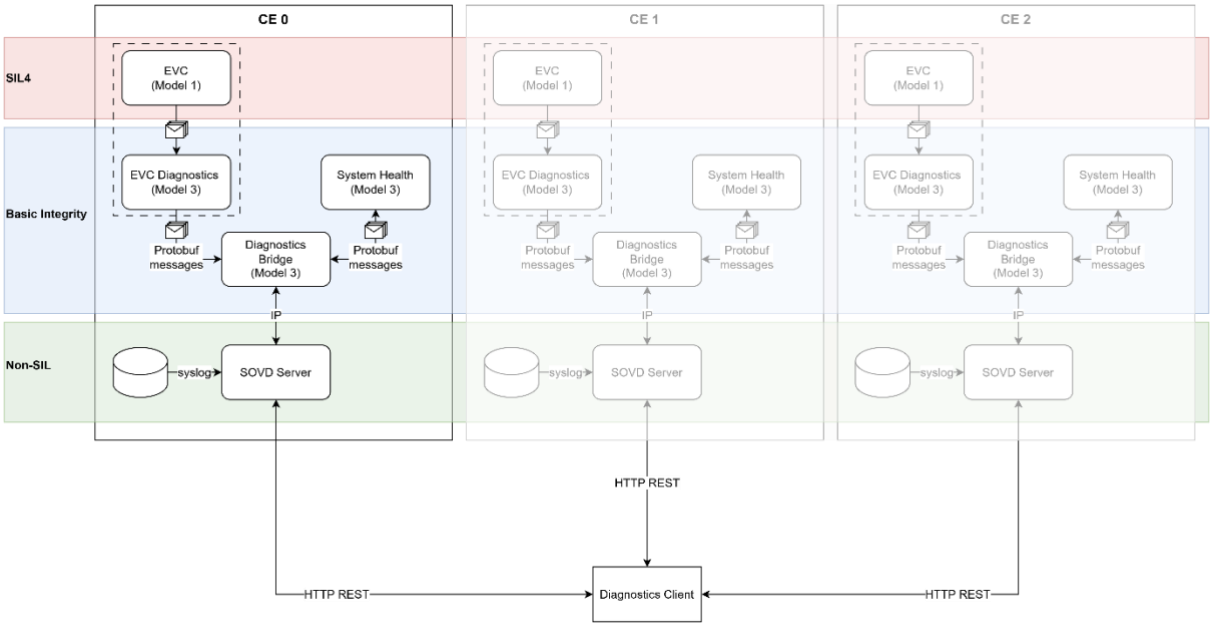


Figure 13: SOVD based Diagnostics

The SOVD server uses a request–response pattern to retrieve diagnostic data in response to HTTP REST requests from diagnostics clients. For Model 3 TaskSets, this pattern extends all the way to the original source of the requested information (for example, "System Health" in Figure 5). However, because SIL4 applications must be deterministic, their corresponding Model 1 TaskSets cannot handle requests forwarded by the SOVD server. At startup, each Model 1 TaskSet begins periodically sending its diagnostic data to the SOVD server. The SOVD server stores the latest values in its cache and answers any HTTP REST request directly from that cache.

The Diagnostics Bridge is a helper TaskSet that connects the message-queue transport to TCP/IP.

Each Computing Element runs its own SOVD server which provides diagnostic data for the tasks executing on that specific Computing Element. Consequently, for Model 1 TaskSets running in a redundant configuration across multiple Computing Elements, the diagnostics client is responsible for consolidating the diagnostic data received from the TaskSet replica.

The proposed architecture makes it straightforward to integrate additional applications that expose their own diagnostic data. A SOVD Definition Language, together with a dedicated authoring tool, enables dynamic extension of the SOVD server so it can serve the new diagnostic data provided by those applications. Even a diagnostics client doesn't need any changes to be able to retrieve the new diagnostic data.

For further details please refer to chapter 2.3 of deliverable D36.2, chapters 2.3 and 3.2 of deliverable D36.3 as well as chapter 2.1 of deliverable D36.4.

5 CONCLUSION

Deliverable D36.5 marks the final report on the demonstration of platform technologies in R2DATO Wave-1. It brings together the specification, development and demonstration work of WP36 and summarises the consolidated results of the activities conducted across the project phases. It reflects the progression from initial specification tasks toward a validated demonstrator implementation, highlighting the insights gained from applying modular platform technologies, FRMCS integration, diagnostics concepts and train-integration approaches in a realistic laboratory environment. As outlined in the Introduction and the objectives placed in the Executive Summary, the motivation for this deliverable was to synthesise the demonstrator outcomes into a coherent blueprint and to assess how the insights gained contribute to future Europe's Rail activities. The remit, as described in Chapter 2, situates WP36 within the wider R2DATO architecture and the System Pillar framework and clarifies how these results support future standardisation, system evolution and preparation for subsequent pilots. In this respect, the work carried out fully addresses the remit while providing rich practical input for future architectural, technical and organisational developments.

Starting from the early specification tasks of D36.1, WP36 developed, implemented and refined an on-board platform architecture that applies and exercises the Modular Platform Concept developed in WP26. The demonstrator confirms that mixed-criticality applications of various suppliers — including functions up to SIL 4—can be hosted with deterministic behaviour, predictable execution cycles and platform-managed replication mechanisms on COTS computing elements. The successful integration of a third-party ETCS Baseline 3 application into a replicated setup demonstrates that migration of existing CCS functionality to modular environments is technically feasible. This is highly relevant for long-term lifecycle optimisation, increased flexibility for operators and a healthier multivendor ecosystem.

WP36 also actively supported alignment with R2DATO WP3, ensuring that the demonstrator results map to the Enterprise Context, Constituent Breakdown and Capability views. By positioning the work within these architectural frameworks, the deliverable strengthens the traceability between technological demonstration and the strategic objectives of the programme. In parallel, WP36 contributed practical insights to several System Pillar domains, including CONEMP, Train-CS, PRAMS and Cybersecurity, by providing real-world evaluation results for platform behaviour, communication principles and diagnostics concepts. While WP36 does not define sector architecture, its reference implementations offer valuable evidence that can support ongoing System Pillar work and future harmonisation discussions and demonstration in R2DATO later phases.

The demonstrator also illustrates the benefits of combining multiple platform related technologies. The integration of FRMCS communication, modular platform services, diagnostics and train integration concepts shows how coherent platform abstractions can support future migration paths, evolvability and reduction of supplier-specific implementations. These insights reinforce the importance of standardised interfaces and shared services in enabling a resilient and competitive European supply landscape.

A key contribution of D36.5 lies in its relevance to future standardisation, especially through the Standardisation and TSI Input Plan (STIP). The results directly support multiple STIP items—including modular platforms (STIP_4), on-board communication network evolution (STIP_68), FRMCS integration (STIP_48+), configuration and diagnostics (STIP_7, STIP_10) and cybersecurity (STIP_2.0_175). By demonstrating feasibility and documenting integration experience, WP36 helps de-risk future sector-wide decisions, strengthening the evidence base needed for harmonised European specifications and TSI evolution.

While a few exciting areas for future development remain—such as heterogeneous hardware portability, configuration, orchestration and secure OTA update, capabilities and enhanced integration of trackside diagnostics, etc.—they offer significant potential to build on the strong momentum achieved so far. These opportunities outline a clear and encouraging pathway for future demonstrators, pilots and industrialisation activities. With active operator and industry engagement,

and expanded infrastructure, the programme is well positioned to broaden its impact and deliver even greater value in the next phase.

The demonstrator experience also delivered important lessons learned, underlining the benefits of clear programming models for integrating safe and non-safe functions, deterministic recovery strategies, harmonised diagnostic modelling and the use of robust DevOps automatisation for multi-partner integration. The ETCS porting exercise particularly highlighted the value of structured development environments and stable tooling for predictable system behaviour and efficient migration.

The work of WP36 has been widely disseminated through:

- The public release of all prior D36 deliverables and their test results,
- Publication of the demonstrator on the Digital Rail Germany website,
- Presentations at System Pillar plenary meetings,
- Contributions to R2DATO Sounding Board sessions, the project Kick-off, Midterm and Final Events, appearances at InnoTrans and other public communication formats.

These activities ensure transparency, stakeholder alignment and reuse of results across Europe's Rail and the wider sector.

In conclusion, WP36 has demonstrated that the combined use of modular platform technologies, modern communication services, diagnostics concepts and train integration principles can significantly support the migration, evolvability and cost-effectiveness of future CCS on-board solutions. The evidence presented in this deliverable forms a robust foundation for future Europe's Rail phases, supports ongoing System Pillar work, and provides a clear contribution towards harmonised, upgradeable and supplier-neutral on-board architectures. By delivering practical insights and validated implementation patterns, WP36 helps move the sector closer to a future in which interoperable and sustainable solutions can be deployed efficiently and confidently across Europe.

REFERENCES

[1] “Europe's Rail R2DATO, D3.4 – Architecture baseline,” 2026. [Online]. Available: https://rail-research.europa.eu/pages/fp2-r2dato/deliverables/.

[2] “CESAM Systems Architecture Framework,” 2026. [Online]. Available: https://www.cesames.net/en/cesam-framework/.

[3] “Europe's Rail R2DATO, D2.2 - KPI Monitoring Report,” 2026. [Online]. Available: https://rail-research.europa.eu/pages/fp2-r2dato/deliverables/.

[4] “EuroSpec, Software Updates v1.0, 2023,” 2023. [Online]. Available: https://eurospec.eu/download/software-updates-v1-0/.

[5] “OCORA, Release R6,” 2025. [Online]. Available: https://github.com/OCORA-Public/Publications/tree/master/12_OCORA%20Release%20R6/.

[6] “UNISIG, CCS Consist Network Communication Layers,” 2023. [Online]. Available: https://www.era.europa.eu/system/files/2023-09/index090_-_SUBSET-147_v100.pdf.

[7] “EuroSpec, TCMS Data Service V1.0,” 2019. [Online]. Available: https://eurospec.eu/tcms-data-service/.

[8] “Europe's Rail R2DATO, D36.3 – Demonstrator implementation phase 2 concluded, and test results consolidated,” 2025. [Online]. Available: https://rail-research.europa.eu/pages/fp2-r2dato/deliverables/.

[9] “Europe's Rail R2DATO, D36.2 – Demonstrator implementation phase 1 concluded, and test results consolidated,” 2024. [Online]. Available: https://rail-research.europa.eu/pages/fp2-r2dato/deliverables/.

[10] “Europe's Rail R2DATO, D36.1 – Demonstrator Specification, User Stories & Test Cases,” 2023. [Online]. Available: https://rail-research.europa.eu/pages/fp2-r2dato/deliverables/.

[11] “ASAM SOVD - Service-Oriented Vehicle Diagnostics, API Specification 2022,” 2022. [Online]. Available: https://www.asam.net/standards/detail/sovd/.

APPENDIX A - ASSESSMENT OF WP26 REQUIREMENTS

Table 4: MPC Requirements from D26.3

Note: Fulfilled means here "in the context of the demonstrator implementation".

ID	Requirement	WP36 Assessment
R001	The MPC shall execute Functional Applications with SIL ranging from BI up to SIL4. *Rationale: The MPC is a universal platform for all needs. However, actual implementations can limit the supported SIL where necessary, e.g. for trackside BI systems.*	**Fulfilled**
R002	Where multiple Functional Applications are present, the MPC shall execute Functional Applications independent of their individual SIL. *Rationale: The SIL can be mixed within Functional Systems and across multiple Functional Systems and its Functional Applications. The MPC implementations have to support arbitrary mixed SIL within their limitations of maximum SIL.*	**Fulfilled**
R003	The MPC shall conform to the interface specification of the System Pillar Computing environment domain. *Rationale: Interfaces I1 to I5 are specified by the SP CE domain. The OCORA source requirements only referred to I4 and I5.*	**SP Specs are not available in needed maturity yet**
R004	Where application-level platform independence is used, the MPC shall transparently encapsulate the safety and fault tolerance mechanism. *Rationale: All safety-related functions not inherent in the application logic shall be implemented as part of the platform. As platform vendors may use their specific approaches to handling safety and fault tolerance, it must be fully encapsulated in the platform. Applications must not include any platform specific code related to safety or fault tolerance.*	**Fulfilled**
R005	The MPC shall enforce a clear separation between the platform hardware and the runtime environment. *Rationale: A clear separation between hard- and software simplifies platform life-cycle management. Typically, the hardware has a much shorter lifetime than the software running on top of it.*	**Fulfilled**

ID	Requirement	WP36 Assessment
R006	The MPC vendor shall be responsible to ensure (by provision of tooling, documentation, generic product certification, etc.) that a solution using the platform is certifiable according to CENELEC without the explicit involvement of the Computing Platform vendor. *Rationale: A full decoupling of the life-cycles of the Computing Platform and the Functional Applications requires that a deployment of Platform and Applications can be homologated without the explicit involvement of the Computing Platform vendor.*	**Fulfilled**
R007	The Application vendor shall be responsible to ensure, by providing all required artefacts, that a Functional System can be integrated on a MPC and homologated without the explicit involvement of the Application vendor. *Rationale: A full decoupling of the life-cycles of the MPC and the Functional Systems requires that a deployment of Platform and Applications resp. Systems can be homologated without the explicit involvement of the application(s) resp. System(s) vendor(s).*	**Fulfilled**
R008	Where application-level platform independence is used, the MPC shall define a unified set of safety related application conditions (SRACs) (at least to the extent that they relate directly to the Functional System) which all safety critical Functional Systems must comply with in order to be certifiable according to CENELEC safety standards. *Rationale: In order to be able to port Functional Systems from one Platform implementation to another, it is key that all Platform implementations delegate the exact same set of safety related application conditions to the Functional Systems. Otherwise, Functional Systems would have to be modified to comply with different conditions on different platform implementations.*	**Not part of the evaluation**
R009	The MPC shall meet the relevant security system requirements as defined by the System Pillar Cyber Security domain. *Rationale: Using a standards-based approach, ensures that adequate controls, processes and procedures are in place to ensure the protection of the platform confidentiality, integrity and availability.* *Remark: The standards situation is being cleared up and the relevant guidelines are expected from the System Pillar. For the HLPI approach, additional investigations can become necessary. IEC 62443:2013 SL3 can be assumed to be the baseline.*	**Fulfilled towards existing standards**

ID	Requirement	WP36 Assessment
R010	The MPC shall ensure the authentication and authorisation of Functional Systems. *Rationale: The Platform has to ensure that only authenticated Functional Systems and their components can use the platform. Further Functional Systems and their components need to be able to trust that the entities they are receiving messages from or transmitting messages to are the entities they claim to be.*	**Possible but not realised**
R011	The MPC shall ensure independence (e.g., in CPU and memory usage) between Functional System components to fulfil the CENELEC norm EN 50129:2018 or later. *Rationale: This is required for CENELEC compliance EN 50129:2018 or later.*	**Fulfilled**
R012	The MPC shall offer a management interface to set a Functional System to be active or inactive where inactive means that it is not executed and can be moved/replaced/updated. *Rationale: The operations accessible from the outside of a platform are on the Functional System level.*	**Not part of the evaluation**
R013	The MPC shall provide the Functional System the ability to report that it needed to deactivate itself. *Rationale: A Functional System might see the need to deactivate itself due to safety goals not being met. The MPC needs to be aware of this change.*	**Fulfilled**
R014	The MPC shall provide a management interface on which information is provided when a Functional System changes state. *Rationale: If a Functional System deactivates itself this likely requires some external action.*	**Fulfilled**
R015	The MPC shall be able to dynamically map relevant Functional System components during operation to other Physical Computing Elements in response to hardware failures. *Rationale: This is intended to mitigate hardware failures.*	**Not part of the evaluation**
R016	The MPC shall provide a management interface that allows activation and deactivation of Physical Computing Elements. *Rationale: The operators shall be able to add and remove physical computing element, e.g., hardware.*	**Fulfilled**

ID	Requirement	WP36 Assessment
R017	For a Functional System it shall be transparent on which Physical Computing Elements its components are deployed to. *Rationale: The functionality within the Functional Systems does not need to know where it is deployed. SRACs have to be satisfied, of course.*	**Fulfilled**
R018	The MCP shall be able to run multiple Functional Systems concurrently (at the same time). *Rationale: Hardware with a multicore processor architecture is commonly available today. It allows running Functional System components side-by-side sharing resources.*	**Fulfilled**
R019	Where application-level platform independence is used, the MPC shall be able to assign deterministic execution behaviour to Functional System components in regular scheduling intervals, based on configuration. *Rationale: Determinism is paramount in a safety critical environment. Therefore, each relevant Functional System component must have a defined execution period (scheduling: time interval).*	**Fulfilled**
R020	Where application-level platform independence is used, the MPC shall be able to assign deterministic execution behaviour to relevant Functional System components triggered by an one-shot-timer event. *Rationale: Besides a regular scheduling interval, additional execution of a Functional System component might be needed, therefore one-shot-timer triggered execution shall be provided by the platform.*	**Fulfilled**
R021	Where application-level platform independence is used, the MPC shall be able to schedule Functional System components for execution triggered by an event such as the receiving of a message by a Functional System component. *Rationale: Enable reaction to explicit events prior to the next regular scheduled start.*	**Fulfilled**
R022	Where application-level platform independence is used, the MPC shall be able to execute Functional System components for a guaranteed execution budget, which shall be defined in the configuration. *Rationale: Determinism is paramount in a safety critical environment. Therefore, relevant Functional System components must have a guaranteed execution e.g., to be scheduled for configured number of time ticks.*	**Fulfilled**

ID	Requirement	WP36 Assessment
	Remarks: *Configuration for all different scheduling paradigms need to be provided, e.g., execution time guarantees might differ between interval and event triggered scheduling and as well between on-board and trackside applications needs.*	
R023	The MPC shall be able to provide strict deadlines and maximum tolerable jitter for deterministic scheduling of Functional System components. *Rationale:* *Real-time computing is key for designing and/or developing predictable, safe CCS functional applications.*	**Fulfilled**
R024	Where application-level platform independence is used, the MPC shall detect and handle errors according to EN 50129:2018 (with both the definition of "error" and the handling of these according to EN 50129:2018). *Rationale:* *This is required to fulfil the norm EN 50129:2018.*	**Fulfilled**
R025	Where application-level platform independence is used, the MPC shall monitor whether a Functional System component is able to conclude processing within a defined time period. *Rationale:* *It is important that Functional System components can conclude on processing, e.g., incoming messages within a certain CPU resource. The platform has to monitor this to be able to potentially react.*	**Fulfilled**
R026	Where application-level platform independence is used, the MPC shall inform the Functional System component if one or multiple of its components have (once or multiple times) not been able to conclude processing in the allocated processing time and take action if configured so. *Rationale:* *Only the Platform knows about the exceeding and informs the Functional Actor that it might have taken appropriate actions.*	**Fulfilled**
R027	Where application-level platform independence is used, when there are multiple replicas of the same Functional Actor, the MPC shall ensure that individual replicas process the same messages. *Rationale:* *This is required, as otherwise it could not be guaranteed that the Replicas yield the exact same output.*	**Fulfilled**
R028	Where application-level platform independence is used, MPC shall provide standardised mechanisms for communication between Functional Systems components. *Rationale:* *Functional Systems and their respective Functional Actors shall be able to exchange data with each other using a standardised message paradigm.*	**Fulfilled**

ID	Requirement	WP36 Assessment
R029	~~Where application-level platform independence is used, the MPC shall provide the ability to Functional Actors to access local inputs and outputs.~~ *Rationale:* *In case there are local I/Os directly connected to the hardware of the MPC, these must be made accessible to the relevant Functional System components.* *Remark:* *Some on-board Computing Platforms need to supply up to SIL4 outputs (e.g., Emergency Brake) and it is the responsibility of the MPC implementation to realise such functional safe I/Os.*	**Crossed out in original table**
R030	The MPC shall provide the ability to Functional Systems to communicate via communication networks. *Rationale:* *Functional Systems deployed on different Physical Computing Elements need to communicate with each other.* *Remark:* *It is assumed that the Computing Platform needs to provide network access via commonly used network protocols as e.g., TCP/IP.*	**Fulfilled**
R031	The MPC shall allow time synchronisation with an external time server via standard protocols. *Rationale:* *Time synchronisation aims to coordinate otherwise independent clocks. Even when initially set accurately, real clocks will differ after some amount of time due to clock drift, caused by clocks counting time at slightly different rates. The usage of standardised protocols ensure compatibility, interoperability, simplify product development and speed up time-to-market.*	**Fulfilled**
R032	The MPC shall include a monitoring and diagnostics interface accessible locally and via remote connection. *Rationale:* *In order to analyse the system behaviour and performance during development, test and operation, a diagnostics interface is needed between Computing Platform deployments and the MDCM (Monitoring, Diagnostics, Configuration, and Maintenance).*	**Fulfilled**
R033	Where application-level platform independence is used, the MPC shall support monitoring of the execution of Functional System components, for instance by capturing KPIs related memory usage, processor load, etc. *Rationale:* *To support fault analysis as well as to monitor proper operation of deployed system.*	**Fulfilled**

ID	Requirement	WP36 Assessment
R034	The MPC shall be able to provide logging and tracing information to an external entity. *Rationale:* *Logging and tracing are critical when analysing system behaviour and faults. Having a unified logging and tracing concept dramatically simplifies the analysis.*	**Fulfilled**
R035	The MPC shall provide safe and secure mechanism to update the run-time environment locally and remotely. *Rationale:* *The ability of updating the platform software is essential. To minimize maintenance cost, the normal update deployment mechanism shall be remotely (e.g., over-the-air) with no need for physical presence of any maintenance personnel on site (e.g., on the train). In case remote updates fail for any reason, it must be possible to perform local updates with physical access to the MPC. Updates shall be uploaded via industry standard interfaces.* *Remark:* *Remote updates must not affect the proper operation of MPCs and might need explicit planned scheduling to be applied.*	**Fulfilled**
R036	The MCP shall provide safe and secure mechanisms to update the MPC configuration locally and remotely. *Rationale:* *The ability of updating the platform configuration is essential. To minimize maintenance cost, the normal update deployment mechanism shall be remotely (e.g., over-the-air) with no need for physical presence of any maintenance personnel on site (e.g., on the train). In case remote updates fail for any reason, it must be possible to perform local updates with physical access to the MPC. Updates shall be uploaded via industry standard interfaces.* *Remark:* *Remote configuration updates must not affect the proper operation of MPCs and might need explicit planned scheduling to be applied.*	**Fulfilled**
R037	The MPC shall provide safe and secure mechanisms to update Functional Systems as a whole or their selected individual components locally and remotely. *Rationale:* *The ability of updating Functional Systems as a whole or some of their components is essential. To minimize maintenance cost, the normal update deployment mechanism shall be remotely (e.g., over-the-air) with no need for physical presence of any maintenance personnel on site (e.g., on the train). In case remote updates fail for any reason, it must be possible to perform local updates with physical access to the MPC. Updates shall be uploaded via industry standard interfaces.*	**Fulfilled**

ID	Requirement	WP36 Assessment
	Remark: *Software updates of the Functional System are performed by the platform, but software updates of entities controlled by the Functional System are in the responsibility of the Functional System itself (e.g., update of field elements).*	
R038	The MPC shall leverage existing specifications for interfaces where feasible. *Rationale:* *Proven interface specifications offer the necessary maturity level for the MPC. Also, existing implementations for platform components can potentially be reused.*	**Not part of the evaluation** **Specifications were not available**
R039	The MPC shall minimize the number of function calls that are part the interfaces. *Rationale:* *A reduced set of function calls eases certification, acceptance and potentially portability.*	**Fulfilled**
R040	The MPC shall define interfaces in a way that is evolvable over time and always enables backward-compatibility. *Rationale:* *It is expected that the interfaces can evolve over time. Future evolved versions are expected to be decently backward compatible to make sure that existing applications can be integrated in MPCs implementing future versions.*	**Not part of the evaluation**
R041	The MPC shall minimize the number of differences in the interface specifications for on-board and trackside environments. *Rationale:* *A common platform simplifies the specification, and thus portability and reusability of common elements between the different environments.*	**Fulfilled**
R042	The MPC shall provide capabilities for Functional Systems to obtain presence and state information of other Functional Systems. *Rationale:* *If there are dependencies between Functional Systems, they need to know each other's state (e.g., active, inactive, degraded) and react if necessary.*	**Fulfilled**
R043	Where application-level platform independence is used, the MPC shall provide a mechanism to Functional System components for obtaining the current replica-synchronized time. *Rationale:* *Functional System components need consistent, replica-synchronised time information which is exactly the same for all replicas and can be used to create output that is voted on.*	**Fulfilled**

ID	Requirement	WP36 Assessment
	Remark: *This is important if the time stamp has an impact on any (voted) output of a Functional System component.*	
R044	The MPC shall provide a mechanism to Functional System components for obtaining the current un-synchronised time. *Rationale:* *Functional System components need un-synchronised time information e.g., replica for specific logging.* *Remark:* *"Unsynchronised time" corresponds to the time at the point when a Functional System component requests this (and for which different components of the same Functional System may obtain a different result).*	**Fulfilled**
R045	Where application-level platform independence is used, the MPC shall complement messages with timestamps. *Rationale:* *Timestamps are important, so that receiving Functional System component can check whether and how strongly received messages are outdated and possibly take appropriate action (i.e., either discard such messages or take other action).*	**Fulfilled**
R046	Where application-level platform independence is used, the MPC shall inform the appropriate Functional System component if it has not been able to conclude processing within a defined time period (once or multiple times, as configured) and (if configured) shut down the Functional System component. *Rationale:* *If a Functional System component is not able to conclude processing within a defined time period (once or multiple times), it either has to scale down its load (e.g., by shifting tasks to other Functional Systems, where possible), or the platform has to shut it down.*	**Fulfilled**
R047	Where application-level platform independence is used, the MPC shall provide a standardized communication mechanism, to exchange messages between Functional System components. *Rationale:* *A standardized communication mechanism is needed in order to abstract safe and secure communication and to enable a transparent encapsulated safety and fault tolerance mechanism, realized by the platform.*	**Fulfilled**
R048	Where application-level platform independence is used, the MPC shall provide a communications method with location transparency to the Function System components. *Rationale:* *Addressing and routing should be transparent to the applications, and thus communication messages routed to the appropriate recipients without needing to know their physical execution environment and location.*	**Fulfilled**

ID	Requirement	WP36 Assessment
R049	Where application-level platform independence is used, the MPC shall provide a communications method with replication transparency to the Function System components. *Rationale:* *The appropriate Functional Systems components have to be agnostic towards the fact that they might be replicated. All complexities with communications coming from replicated execution has to be handled by the platform.*	**Fulfilled**
R050	Where application-level platform independence is used, the MPC shall allow communications only by its own communication mechanisms. *Rationale:* *Side channel communication not using the platform methods has to be avoided. Only this way, standardized and safe communication can be established, and transparent replication implemented.*	**Fulfilled**
R051	Where application-level platform independence is used, the MPC shall supervise configured maximum message delivery times. *Rationale:* *Functional System components depend on reliable, reasonably deterministic communication with other Functional System components. It is hence required that the platform supervises defined maximum message deliverable times in, e.g., the scheduling of relevant Functional System components.*	**Fulfilled**
R052	Where application-level platform independence is used, the MPC shall ensure the correct ordering of all messages sent and received by Functional System components. *Rationale:* *Functional System components can rely on the correct message distribution order without the need to implement order checking logic.*	**Fulfilled**
R053	Where application-level platform independence is used, the MPC shall conform to EN 50159. *Rationale:* *In terms of safe communication, EN 50159 provides guidelines which shall be followed to ensure that we have a common base for communication requirements.* *Remark:* *Issues are identified by the platform (e.g., through the usage of message sequence numbers or some other platform-specific mechanism).*	**Fulfilled**

ID	Requirement	WP36 Assessment
R054	Where application-level platform independence is used, the MPC shall supplement messages with their time of creation. *Rationale:* *Time stamping is needed, so that receivers are able to determine how old messages are, and whether they should still be processed or discarded, etc.* *Remark:* *This might require the notion of synchronized platform clocks (also among distributed platforms, see also R043) at least to the extent/granularity (e.g., on the order of tens of ms) that is required to detect outdated messages. When exactly the time stamping happens needs to be discussed.*	**Fulfilled**
R055	The MPC shall provide a bi-directional interface to exchange diagnostics information with Functional System components. *Rationale:* *Functional System components can use the interface, e.g., to receive diagnostics information or to report diagnostic information to the platform.* *Remark:* *Diagnostic information could for instance comprise the health status of the platform or a Functional System component.*	**Fulfilled**
R056	The MPC shall provide an interface towards Functional System components for logging and tracing. *Rationale:* *Applications have the need to provide log and trace data to the platform.*	**Fulfilled**
R057	The MPC shall allow configuration of different logging and tracing levels and categories per platform and on a Function System component level. *Rationale:* *Depending on the required information, it is important to be able to enable logging and tracing only for certain Functional System components and not for the entire system.*	**Fulfilled**
R058	The Physical Computing Element shall be based on COTS components. *Rationale:* *Leveraging COTS hardware provides several benefits, including cost-effectiveness, readily available components, and ease of integration.*	**Fulfilled**

ID	Requirement	WP36 Assessment
R059	The SRACs of the Functional System shall not limit the use of shared hardware resources. *Rationale: To maximize the efficiency and flexibility of hardware usage it is essential to be able to aggregate Functional Systems of various suppliers on the same Physical Computing Element(s).*	**Not part of the evaluation**
R060	The safety environment shall identify incorrect deployment of safety critical Functional System compartment. *Rationale: It is imperative that safety-critical functions employing composite safety, such as replication and voting, are executed on distinct hardware devices.*	**Fulfilled**
R061	The safety environment shall not restrict mixed criticality on a physical computing element. *Rationale: To enable the wide range of applications with different critical levels to coexists and to attain resource efficiency, flexibility, improved system utilization, optimized performance, and scalability.*	**Fulfilled**
R062	The safety environment shall not restrict the creation/initialization of new compartments during runtime. *Rationale: To support dynamic configuration the system shall support the aggregation/deployment of Functional Systems during runtime.*	**Not part of the evaluation**
R063	The safety environment shall support start/stop of Functional System compartments on another physical computing element during runtime. *Rationale: To replace defective HW it is essential that the safety environment shall support Functional System compartment management.*	**Not part of the evaluation**
R064	The virtual computing element shall support the remote restart of an individual FS compartment at runtime. *Rationale: This will allow to automate the recovery of FS during runtime.*	**Not part of the evaluation**
R065	The safety environment shall support the synchronization of restarted/updated compartments. *Rationale: To synchronize the safe applications replica with other running replicas after update/recovery.*	**Fulfilled**
R066	Communication between functional systems (I0) shall be realized with 2 redundant channels for availability. *Rationale: In order to support safety-critical communication, fulfilling availability requirements the FS system shall have redundant and independent communication channel.*	**Fulfilled**

ID	Requirement	WP36 Assessment
R067	The OI shall provide APIs and tools to orchestrate the FS Compartments. *Rationale: To facilitate the remote deployment of FS it is essential to implement mechanisms that automate, manage, and coordinate Functional Systems in compliance with their certification requirements.*	**Not applicable, further standardisation needed**
R068	The virtualisation environment shall guarantee the assigned CPU resources (cores, memory, memory bandwidth, network bandwidth, network latency) for a FS continuously (7 days /24 hours / 60 minutes / 60 seconds) without any influence or dependency to existence and behaviour of other FS aggregated on same computing element. *Rationale: The VE must provide the committed resources to all FS compartments and ensures no interference between the virtual computing elements on same physical computing element.*	**Not applicable, further standardisation needed**
R069	The configuration of the virtualization environment shall comply with the FS deployment rules. *Rationale: To ensures the function and availability of the intended FS, it is crucial to adhere to its deployment rules.*	**Not applicable, further standardisation needed**
R070	The virtualization environment shall provide standard OI functionalities. *Rationale: Having a common set of OI functionalities for operating all Functional System(s) offers benefits such as resource optimization, scalability, high availability, automation, and cost efficiency.*	**Not applicable, further standardisation needed**
R071	The virtualization environment shall be based on COTS solutions. *Rationale: As there are already a wide range of COTS virtualization solutions available, developing a new one for railways is prohibitive due to complexity and cost.*	**Not applicable, further standardisation needed**
R072	Different virtualization approaches shall be allowed. *Rationale: There are several types of virtualizations approaches available in different domains, each with its own benefit. Therefore, the virtualization solution could allow different implementation approaches as long as a standard Orchestration Interface (OI) is provided.*	**Not applicable, further standardisation needed**

ID	Requirement	WP36 Assessment
R073	~~The orchestration interface (OI) implementation shall not bind to a specific approach/programming language.~~ *Remark:* *In the MPC, this interface is expected to be connected to the Platform Management, thus only available internally. As such, no guidance on approaches and programming languages is necessary. The requirement has been removed.*	**Crossed out in original table**
R074	The Virtualization environment shall provide remote orchestration. *Rationale:* *To enable the efficient management, scalability, cost reduction, it is essential to have a centralised FS management.* *Remark:* *In the MPC, this interface is expected to be connected to the Platform Management.*	**Not applicable, further standardisation needed**
R075	The Virtual Environment shall allow uninstallation of individual compartment deployed on a virtual computing element without interrupting neighbouring compartments on the same physical hardware. *Rationale:* *To ensure the safety and availability of other Functional system compartments running on the same physical computing element.*	**Not applicable, further standardisation needed**
R076	The Virtualization Environment shall support remote creation of virtual computing elements without interfering with already running virtual computing elements on the shared physical hardware. *Rationale:* *To enable remote dynamic configuration of virtual computing elements.*	**Not applicable, further standardisation needed**
R077	The Virtualization Environment shall support remote deletion of virtual computing element without impacting other running virtual computing element on the shared physical hardware. *Rationale:* *To enable remote dynamic configuration of virtual computing elements.*	**Not applicable, further standardisation needed**
R078	The Virtualization Environment shall provide full hardware abstraction. Changes in the underlying COTS HW may not have any impact to the FS running on the virtualisation environment. *Rationale:* *Virtualization Environment must be compatible to all the hardware architecture such as x86, ARM, PowerPC, and others to support seamless integration of FS.* *Remark:* *There is no requirement towards full hardware emulation across different CPU architectures.*	**Not applicable, further standardisation needed**

ID	Requirement	WP36 Assessment
R079	The Virtualization Environment shall support to do updates of the virtualisation environment computing-element-wise "one after the other" without affecting the virtualisation environment on the other virtual computing elements. *Rationale: This is necessary to update the virtualisation environment (e.g. due to IT-sec patches) during runtime of the FS.*	**Not applicable, further standardisation needed**
R080	The Virtualization Environment shall provide (backward) compatibility at the configuration interface. *Rationale: A new version of the virtualisation environment shall not have any impact on the FS related configuration data. The configuration of the virtual machine (resources, communication interfaces, etc.) shall not change.* *Remark: In the MPC, the changes in the actual interface would be adapted in the Platform Management.*	**Not applicable, further standardisation needed**
R081	The Virtualization Environment shall provide the diagnostic interface to monitor the virtual computing element. *Rationale: The diagnostic will allow to monitor the health of virtual computing elements and may detect SW crashes and enable automatic recovery.* *Remark: This is connected to the Platform Management in MPC.*	**Not applicable, further standardisation needed**
R082	The Virtualization Environment shall provide detailed predictive diagnosis about the health state of the physical computing element(s). *Rationale: The predictive diagnosis will allow to monitor the health of physical computing element and may detects HW faults earlier.*	**Not applicable, further standardisation needed**
R083	The Virtual Environment shall provide mechanism to ensures the correct deployment of FS compartments. *Rationale: To ensure the safety requirements such as to run each replica of FS compartment on distinct physical computing element.*	**Not applicable, further standardisation needed**
R084	When non-safe software parts are changed, the MPC shall ensure that there is no impact on the safe software parts. *Rationale: Changing resp. updating non-safe parts (e.g., Basic Integrity applications or parts of the RTE/VE, security updates, etc.) must not create any situation where previously reached conclusions on the freedom of interference towards safe software parts are invalidated. Meaning that changing or updating non-safe parts does not lead to re-certification of safe parts.*	**Fulfilled**

APPENDIX B - EVC PORTING EXPERIENCE AND LESSONS LEARNED

The various artefacts developed during the earlier phases were integrated and consolidated in WP36.4, including the TAS platform, FRMCS, diagnostics, applications across different SIL levels, and the CCS Consist Network (CCN). To establish a reference implementation that was as realistic as possible, the subcontractor Clearsy was commissioned to port its EVC on-board system—used exclusively for testing and not certified—to the TAS platform. In addition to the technical implementation work, Clearsy was asked to document its porting experience and to provide a summary report. The following list presents the resulting insights.

- Co-working approach and objectives:
  - Purpose: foster smooth, responsive collaboration between Clearsy and SBB to speed up issue resolution and reduce validation delays.
  - Main aims: ensure quick follow-up on requests, reduce review cycles, and create a shared understanding of requirements and constraints.
- Meetings and cadence:
  - Kick-off: set project objectives, scope, schedule, input needs and contractual items; established cadence of weekly technical meetings and biweekly project-status meetings through end of 2025.
  - Technical meetings: provided fast, direct access to customer decision-makers and experts, enabled co-building of technical solutions, clarified expectations (contractual vs undocumented), and accelerated fixes.
  - Project status meetings: increased transparency about difficulties and risks, reinforced partnership feeling, and demonstrated Clearsy's strong commitment.
- Key positive outcomes of the collaboration style:
  - Increased efficiency due to participant proximity and immediate availability.
  - Reduced back-and-forth and faster validation.
  - Improved final quality through real-time adjustments.
  - Stronger communication, lowering misunderstanding risks.
  - Overall time savings to complete the project.
- Architecture and execution model
  - The demonstrator split the ETCS EVC into a potentially safe core and non-safe applications (network I/O, communications). This architecture (safe core + non-safe handling network/other tasks) fits well with the TAS Platform and would be retained for new developments.
  - The original EVC used multiple parallel threads; for the TAS Platform port they removed parallel execution and consolidated to a single taskset to avoid complexity (queues between tasksets). Recommendation: avoid unnecessary multiple threads/tasksets in the EVC core unless required for performance.
- TAS Platform and safety layer
  - TAS provides a SIL4-capable platform: hardware, Linux/POSIX OS, and a safety layer (voted execution, multiple execution contexts, voted queues) already assessed. This reduces the application's certification scope: developers can focus on application-level verification and safety analysis rather than implementing intra-application redundancy or protecting against wrong execution.

  - The safety layer enforces constraints (safe queues, initialization of memory regions); violations produce low-level, limited feedback which complicates debugging.

- Memory management and recovery
  - Recovery mechanism requires identifying and compacting the recoverable state. Large state variables (e.g., speed curves) must be split into an essential recoverable part (a few megabytes) and a re-computable part to ensure recovery completes within acceptable time (a few minutes).
  - For new development, define allowed recovery time early to bound the maximal recoverable memory region and design data structures accordingly (compact representations for recoverable region).
  - The Platform has sufficient memory overall, but the recoverable memory size impacts recovery duration.
- Porting and application changes
  - Porting required splitting the EVC into core and network I/O, replacing internal communication with safe queues; this was straightforward but required attention to un-initialised memory regions discovered by the safety checks.
  - Some operations (e.g., file openings) had to be moved earlier in the startup process (before a specific point X), but the POSIX-like environment made porting familiar.
- Development and testing environment
  - The provided Docker-based development environment (runtime, compiler, IDE, virtual CEs) closely matched the physical TAS Platform. Applications tested on the virtual platform deployed successfully on physical TAS with minimal issues—strong positive for early development and reduced target debugging.
- Integration and networking limitations
  - IO/recovery: network connectivity recovery is complex because each CE has its own IP; if an IO task on one CE fails/recovers on another CE, the EVC's IP appears to change to external equipment. Transparent failover requires either:
    - Other equipment to support failover (handle IP change), or
    - A platform-side proxy that hides CE-level failover and preserves a stable network endpoint.
  - Implementing multi-CE collaborative IO can work but is not transparent to external systems without such failover mechanisms.
- Maintenance and updates
  - Non-safe binaries can be delivered and updated independently since they communicate via queues; updating non-safe components should not affect the safety core if interfaces/protocols are respected.
  - Platform updates are likely straightforward because applications are standard Linux binaries; major platform updates may require rebuilds, but no fundamental blockers are anticipated.
- Debugging and observability
  - Safety-layer-detected issues produced limited diagnostics, making root-cause analysis painful despite determinism and reproducibility. Emphasized need for improved feedback/diagnostics for safety-layer interactions.
  - Suggest incremental testing after each feature addition; in the project, even incremental additions were large so debugging remained difficult.

- Certification impact
- Using the TAS Platform reduces certification effort for hardware/OS/safety layer; developers still must perform standard specification, verification, validation, safety analysis and traceability but scoped to the application, ensuring Safety Related Application Conditions of the Platform are met.

Recommended technical improvements for future projects

- Design the application from the start as safe-core + non-safe I/O with clear interfaces and safe queues.
- Limit recoverable state size by design; define allowed recovery time to drive data structure decisions and implement compact representations for recoverable state.
- Avoid unnecessary multithreading / multiple tasksets in the safety core to reduce complexity and inter-task communication overhead.
- Enhance diagnostics and logging exposed by the safety layer (or define richer internal observability) to speed root-cause analysis.
- Use and invest in a high-fidelity virtual development environment to minimize target-side debugging.
- Define and implement a network failover strategy early: either require external equipment to support IP failover or provide a platform-level proxy to keep endpoints stable.
- Establish small, frequent integration/testing increments to localise regressions more easily.
- Keep non-safe components modular so they can be updated independently without impacting the safety core.
- Plan for platform updates: confirm rebuild/test procedures and compatibility checks for major platform runtime updates.

APPENDIX C - REMAINING USER STORIES, TEST CASES AND RESULTS

DOCUMENTATION OF ADDITIONAL USER STORY UND TESTS (CONCERNING NETWORKING)

PRIORITISE SPECIFIC NETWORK TRAFFIC [2.7.7]

Description User Story	As R2DATO WP36, we want to configure different quality of service classes in the network, so that we can demonstrate the prioritisation of specific network traffic.
Annotation	The demonstration of this User Story has been delayed, and the test result will be reported in the next deliverable 36.5.

Setup

This user story will be demonstrated by two test cases. The first test case will demonstrate that the application will fail, when the network and the applications are not properly configured with the appropriate priorities. The second test case will demonstrate that the application continues to work, when the network is configured correctly although the network is flooded with bulk traffic.

The CCN consists of four switches connected in a ring and have the Rapid Spanning Tree Protocol (RSTP) enabled. During normal operation, port X1 of switch 3 blocks the traffic to avoid network loops.

The red link in the figure below, between switch1 and switch2 will be the common critical path for the high priority and bulk traffic.

In the first test case, NONE of the traffic will be marked with any priorities. This results in best effort priority handling for all network traffic on the CCN and therefore causing the application to fail.

The on-board application logic (ETCS_APP) is running in a 2oo3 configuration on all three computing elements (ce0, ce1 and ce2). Network access is realized by the three replicated network bridges (bridge_0, 1 and 2), whereas only one bridge will be active at a point in time.

The communication sessions are monitored by a timeout constrained heartbeat. As soon as the defined timeout is exceeded, the session will be handed over to the next bridge. At startup, bridge_1 on ce1 is active and the handover sequence going from ce1 to ce2 to ce0 and then back to ce1 whenever the communication session is failing.

This behaviour will be used as the indicator for the failing communication due to the overloaded network.

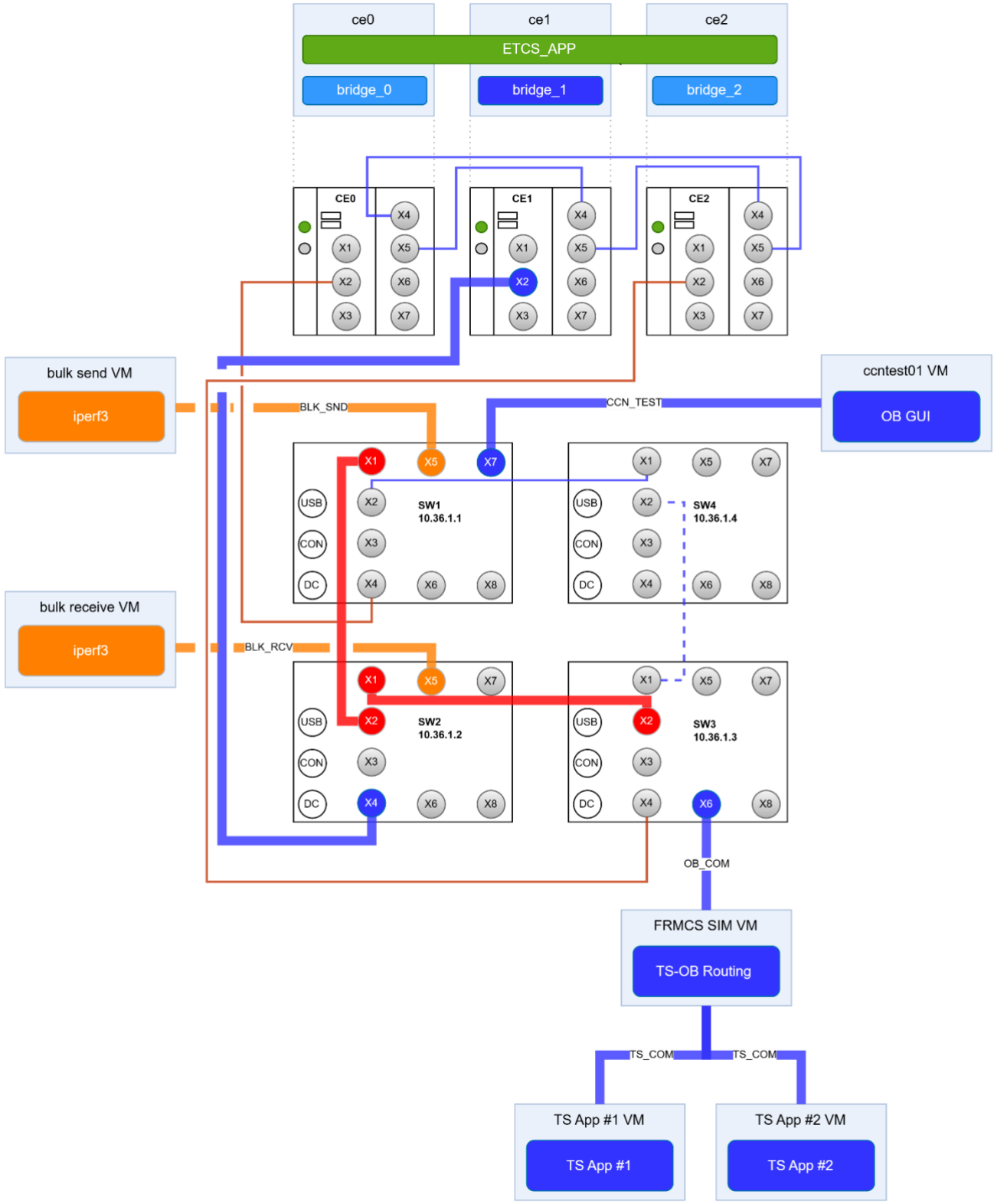
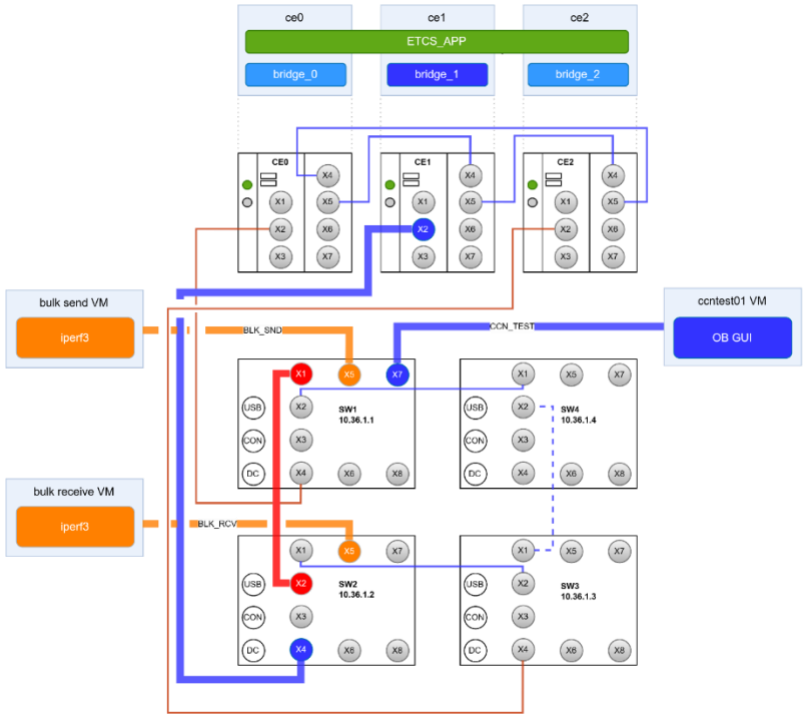


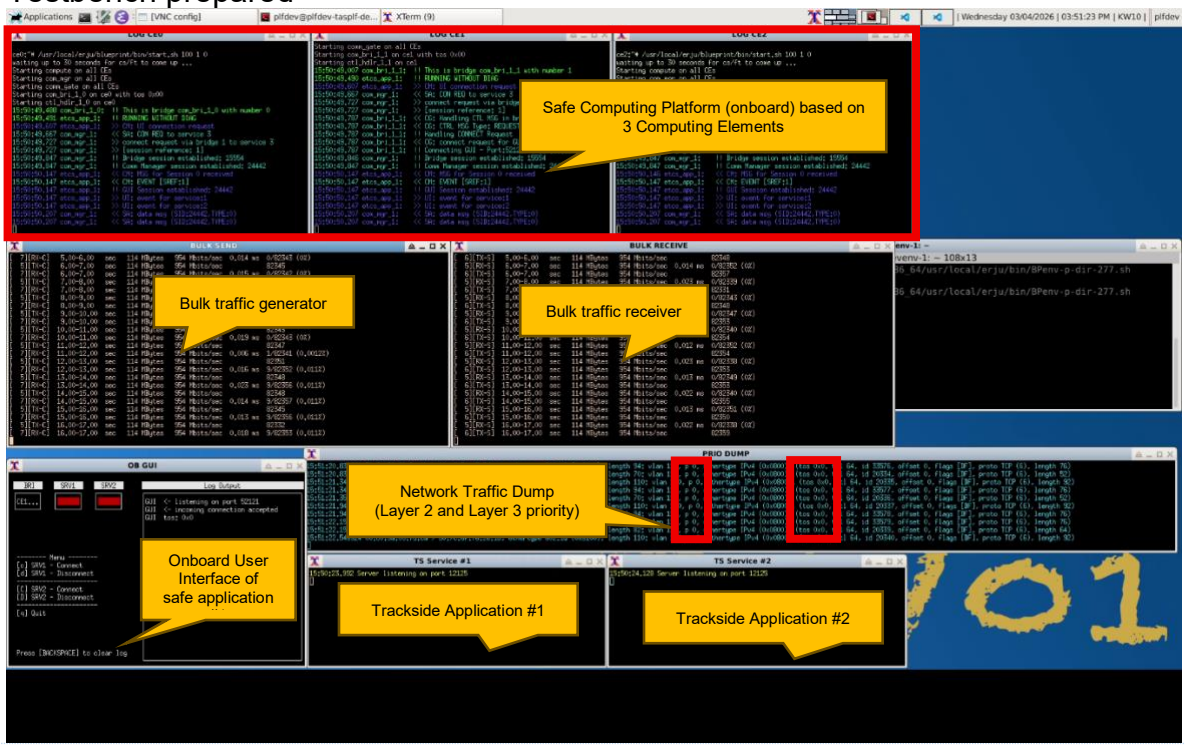
Figure 14: User Story 2.7.7 Setup

Test Case Title	**Effect of bulk traffic without priority-based separation**
Test Case ID	2.7.7-A
Description	Demonstrate that bulk traffic on the same physical Network as the application is running **does impact** the behaviour and functionality of the system. For this test case, the setup does not include the communication to the trackside applications, because the setup with the on-board user interface (OB GUI) provides already the required evidence. 
Test Environment	• Modular Platform (TAS Platform) • CCS Communication Network (CCN) • Trackside application simulations running as VM on the lab server • User interfaces (OB GUI) for on-board application • on-board application (etcs_app_1) running on the Modular Platform
Preconditions	Set up Test environment in a manner that runs long enough for this test case.
Test Steps	1. Prepare testbench 2. Start testbench components and monitoring a. Start trackside application(s) b. Start on-board user interface (OB GUI) c. Start the on-board safe application (etcs_app_1)

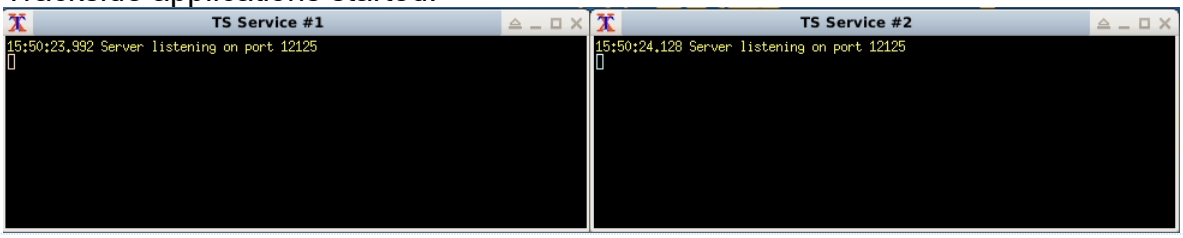
	d. Start bulk traffic receiver and sender
	3. Monitor connection timeout due to the overloaded network
Test Data	None.
Expected Result	During this test, the application will fail due to a communication timeout and then switch over to the next communication bridge.

Actual Result

1 Testbench prepared

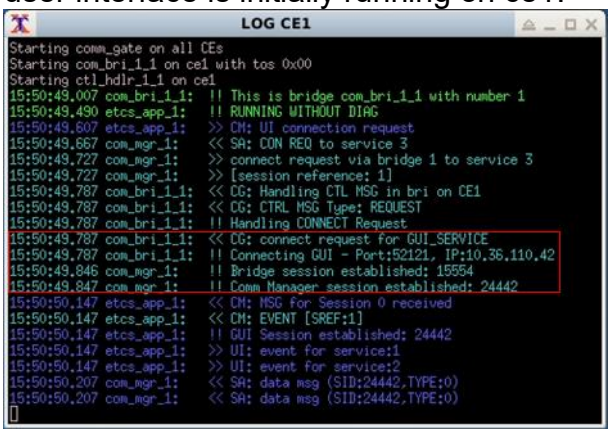


2a Trackside applications started.

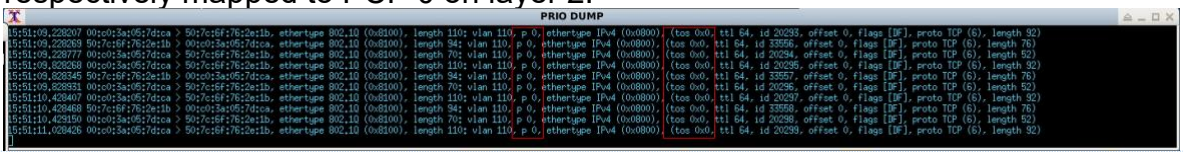


2b On-board user interface started without priority (best effort, TOS 0x0)

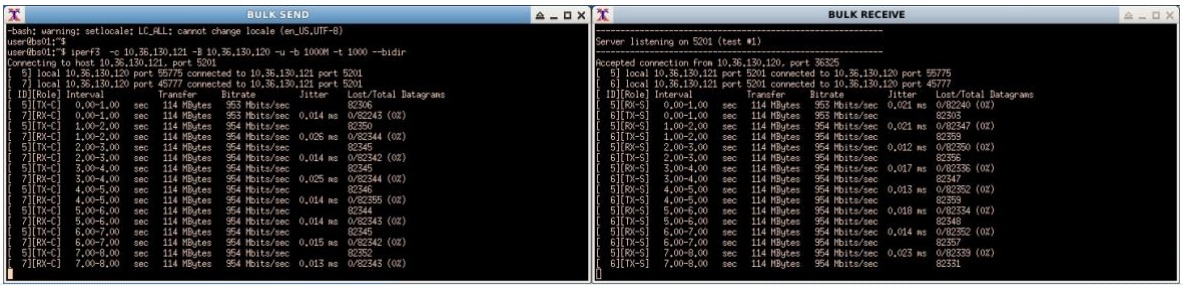
2c The on-board application is ready and connected to the user interface. The log output of ce1 is shown below, because the active bridge connecting to the on-board user interface is initially running on ce1.



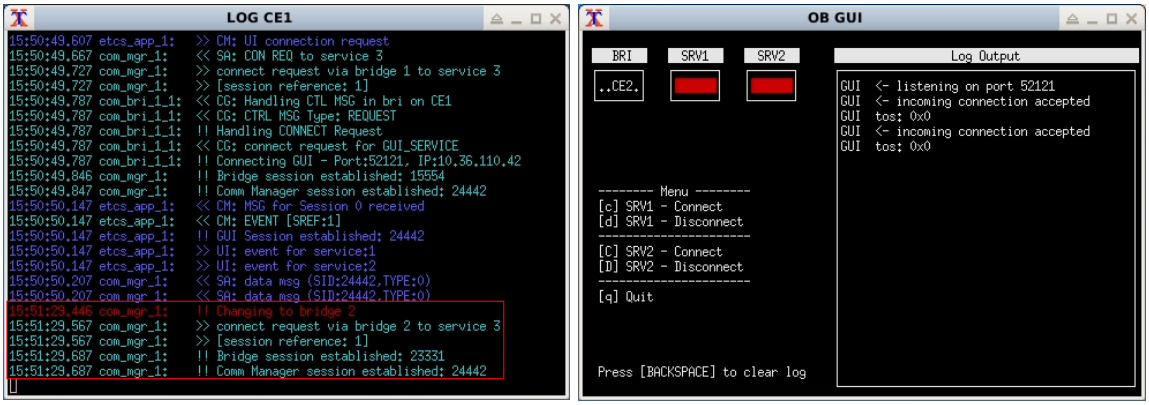
Communication is established with TOS 0x00 (Type Of Service) on layer 3 respectively mapped to PCP 0 on layer 2.



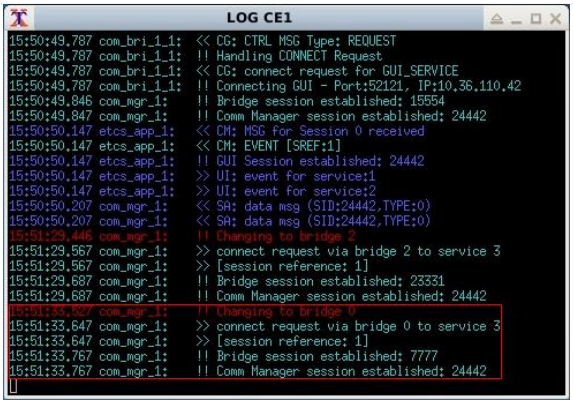
2d Bulk traffic generator started, generating bi-directional traffic to flood the 1Gb/s backbone between switch 1 and switch 2.



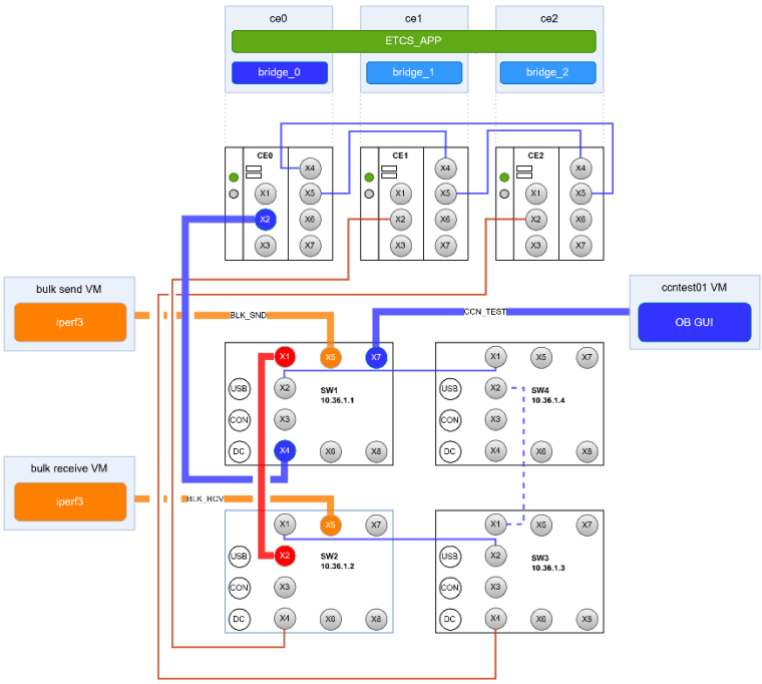
3 The connection to the on-board user interface is handed over to the bridge running on ce2.



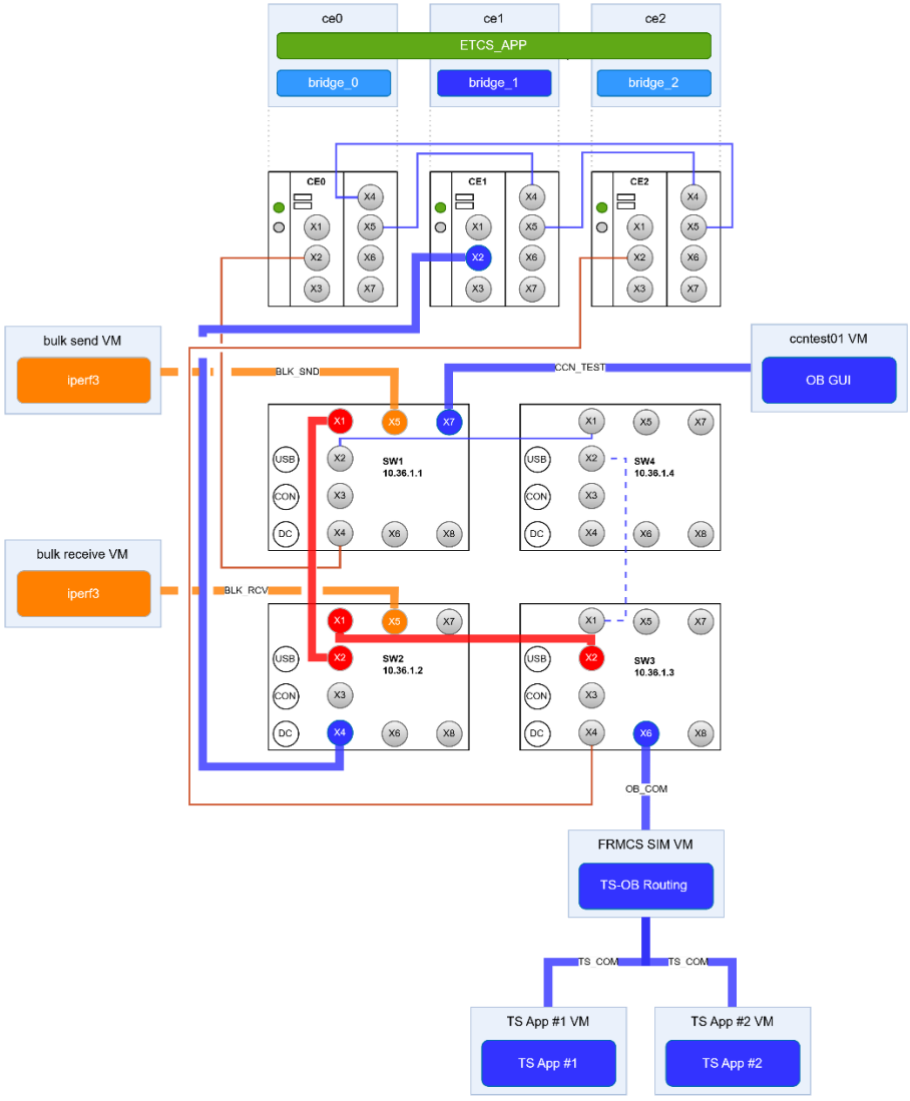
Since the communication between ce2 and the on-board user interface uses the same (overloaded) link, the connection is again handed over to the next bridge running on ce0.



Finally, the bridge running on ce0 will remain stable, since the on-board user interface and the active bridge running on ce0 are connected to the same switch and therefore not impacted by the generated bulk traffic.



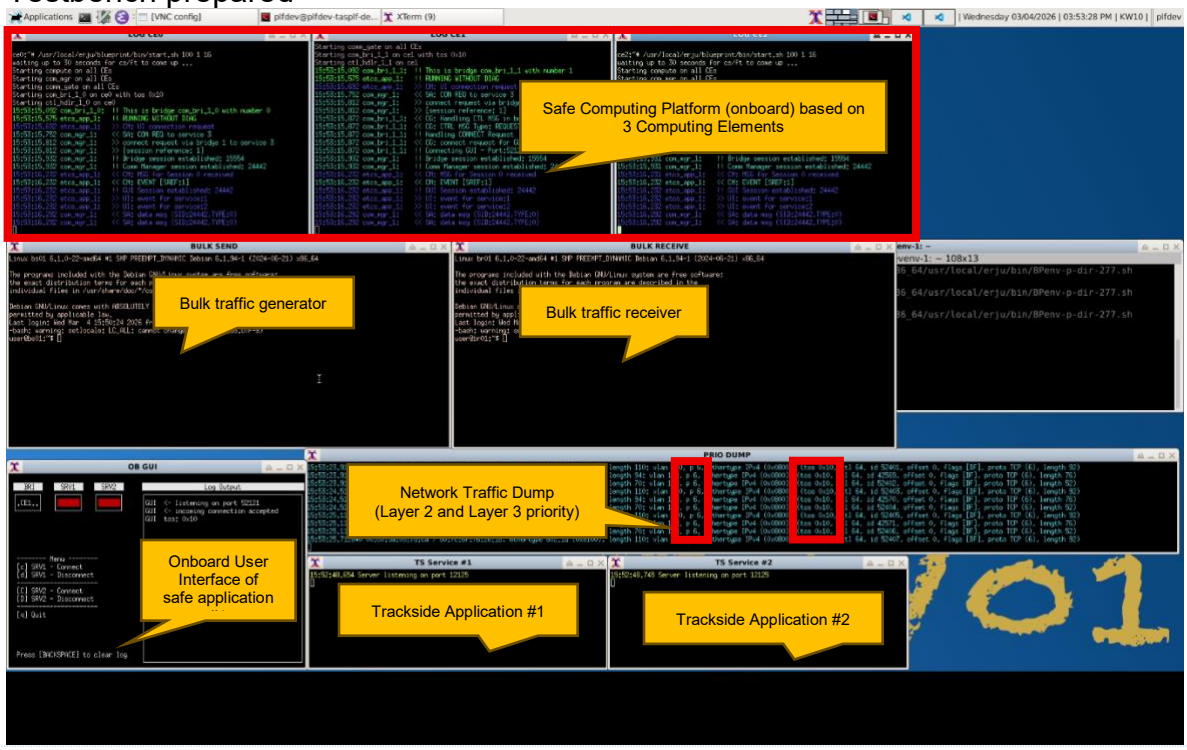
Status	Pass
Comments	none
Attachments	

Test Case Title	**Effect of bulk traffic with priority-based separation**
Test Case ID	2.7.7-B
Description	Demonstrate that bulk traffic on the same physical Network has **no impact** on the behaviour and functionality of the system. The setup from testcase 2.7.7-A has been extended to include also additional prioritized communication sessions between the on-board application and the two trackside applications. 
Test Environment	• Modular Platform (TAS Platform) • CCS Communication Network (CCN) • Trackside application simulations running as VM on the lab server • User interfaces (OB GUI) for on-board application • on-board application (etcs_app_1) running on the Modular Platform
Preconditions	Set up Test environment in a manner that runs long enough for this Test Case.

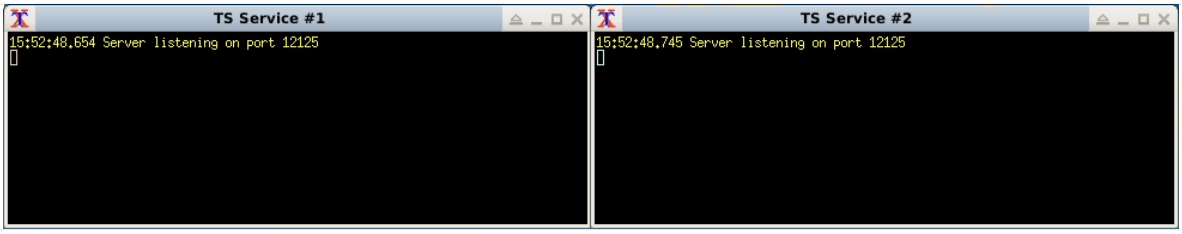
Test Steps	1. Prepare testbench 2. Start testbench components and monitoring a. Start trackside application b. Start on-board user interface (OB GUI) c. Start the on-board safe application (etcs_app_1) d. Start bulk traffic receiver e. Establish connection between on-board and trackside application #1 f. Start bulk traffic sender g. Establish connection between on-board and trackside application #2
	3. Demonstrate, that the application remains stable while bulk traffic is applied to the common physical network.
Test Data	None.
Expected Result	During the full test, the system must continue to perform its intended functions, without any detectable issues or degradation.

Actual Result

1 Testbench prepared

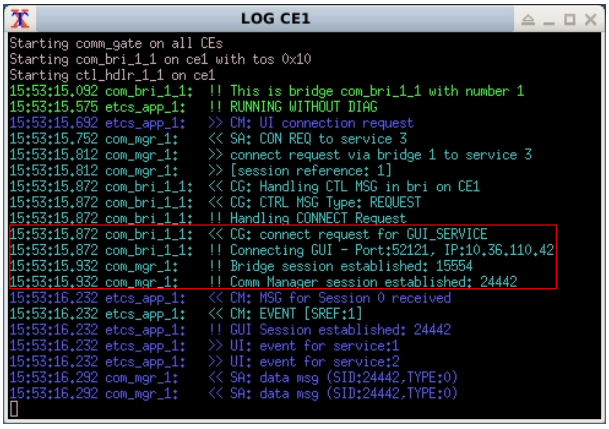


2a Trackside applications started.



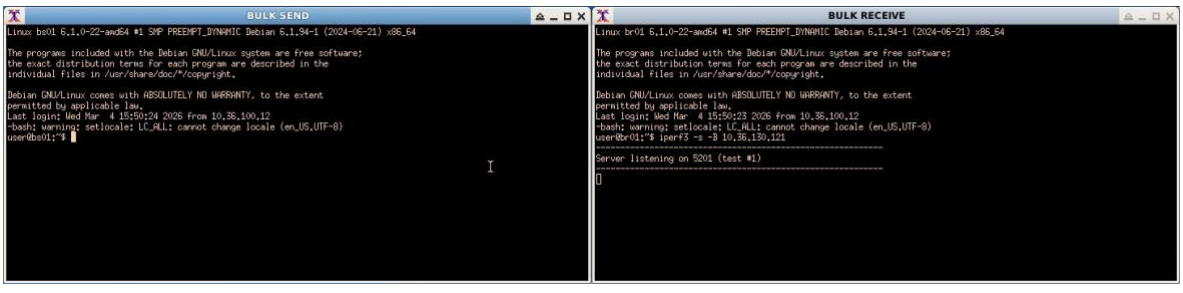
2b On-board user interface started with TOS 0x10 (LOW_DELAY)

2c The on-board application is ready and connected to the user interface. The log output of ce1 is shown below, since it is running the active bridge connected to the on-board user interface.

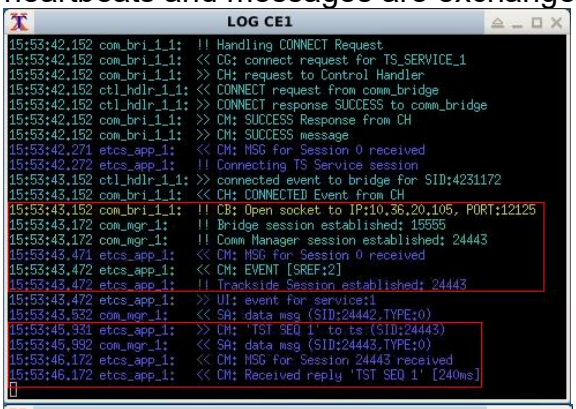


Communication is established with TOS 0x10 (Type Of Service) on layer3 respectively PCP 6 on layer 2.

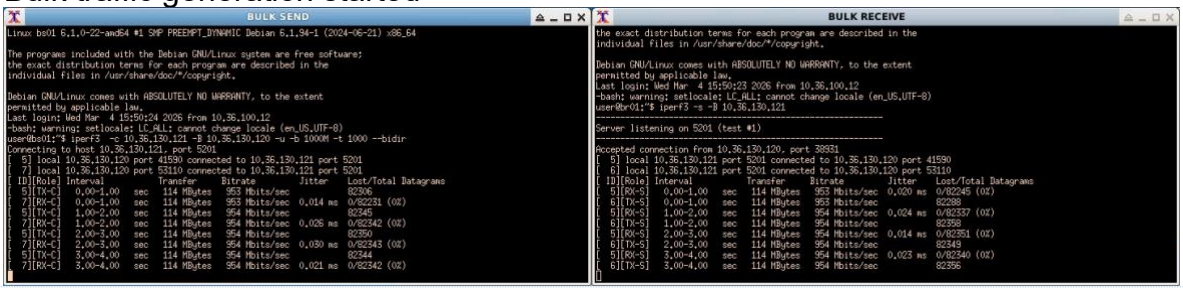
2d Bulk traffic receiver started



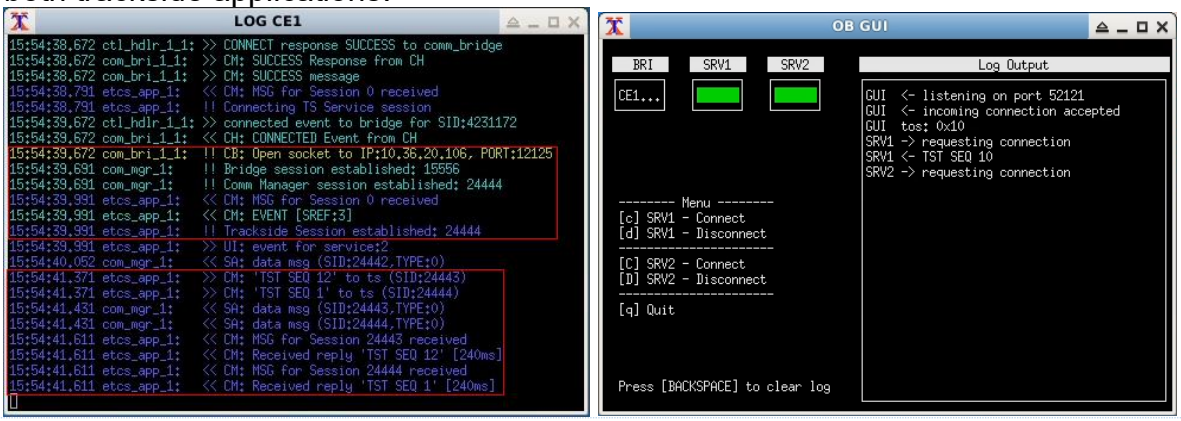
2e The communication session to the trackside application #1 is established, heartbeats and messages are exchanged.

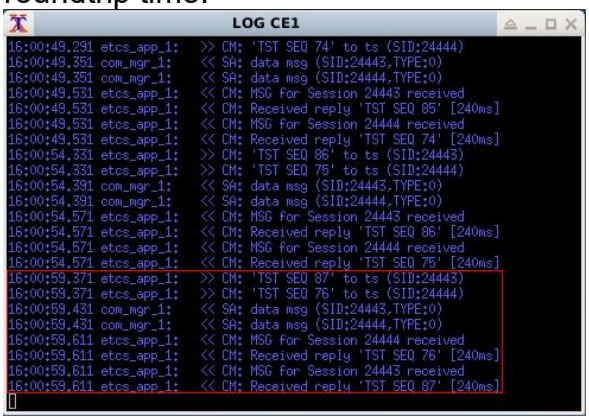
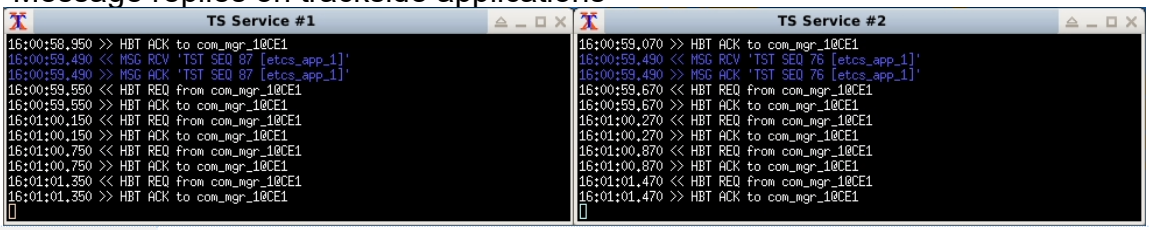


2f Bulk traffic generation started



2g The communication session to the trackside application #2 is established, heartbeats and messages are exchanged between the on-board application and both trackside applications.



3	The connection remains stable using the communication bridge on ce1. All messages, 76 with the trackside application #1 and 86 with trackside application #2, have been successfully exchanged without recognisable impact on the message roundtrip time.  Message replies on trackside applications 
Status	Pass
Comments	none
Attachments	