Rail to Digital Automated up to Autonomous Train Operation

D36.1 – Demonstrator Specification

User Stories & Test Cases

Leader/Responsible of this Deliverable: DB

Reviewed: Y

Document status		
Revision	Date	Description
0	17/10/2023	Initial version
1	20/10/2023	Refined version for feedback from partners
2	31/10/2023	Version for TMT review
3	30/11/2023	Additional references to requirements
4	19/07/2024	Added acronyms table and criteria for selection of user stories

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

ACKNOWLEDGEMENT

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.

ABBREVIATIONS AND ACRONYMS

Abbreviation	Definition
ATO	Automatic Train Operation
API	Application Programming Interface
CCN	CCS Communication Network
CCS	Control-Command and Signalling
ETCS	European Train Control System
FRMCS	Future Railway Mobile Communication System
HTTP	Hypertext Transfer Protocol
IAM	Identity and Access Management
MCx	Mission-Critical Services
MDCM	Diagnostics, Monitoring, Configuration and Maintenance
OCORA	Open CCS Onboard Reference Architecture
R2DATO	Rail to Digital and Automated Train Operation
RTE	Runtime Environment
SIL	Safety Integrity Level
TOBA	Telecom On-Board Architecture
TCMS	Train Control and Management System
UIC	Union Internationale des Chemins de fer (International Union of Railways)
WP	Work Package

TABLE OF CONTENTS

TABLE OF FIGURES

1 INTRODUCTION

1.1 PURPOSE OF THIS DOCUMENT

To properly derive findings from the demonstration work, the evaluation of the demonstrator is based on predefined user stories and test cases that ought to be investigated.

References to related system or sub-system requirements shall set the demonstrator user stories in a meaningful and relevant context.

1.2 BACKGROUND

In the work package 36 (WP36) “Onboard Platform Demonstrator” the partners Deutsche Bahn (DB), Ground Transportation Systems (GTS), Kontron (KONTRON), Schweizerische Bundesbahnen (SBB), Siemens Mobility (SMO) and Trafikverket (TRV) cooperate to validate the feasibility of a future-proof onboard IT-platform that is suitable to host safety critical applications.

In the context of R2DATO this work package demonstrates a concrete implementation of the modular computing platform as defined in work package 26 to a Technology Readiness Level (TRL) 5/6 (demonstrated in relevant environment) enhanced by onboard connectivity to a train adapter, FRMCS communication modules and shared services (e.g., diagnostics and maintenance).

The project timeline started in December 2022 (M1) and ends in May 2026 (M42).

Based on different perspectives and motivations, the partners gathered a total of around 100 user stories to be investigated. Out of those the user stories below have been selected as most relevant to be demonstrated in the context of this work package. The selection was based on the following necessary criteria.

- The user story fits to the scope (system under consideration) of the work package as defined in the system definition [1].
- It is deemed technically feasible by the partners to validate the user story with the joint resources appropriated to the work package. This also concerns the availability of the needed hardware and software.
- If validating the user story is depended on external contributions, the risk that those contributions are not available (on time) is sufficiently low.
- Validating the user story fits to the foreseen TRL of a demonstrator.
- The user story is well defined so that it can be clearly validated.

For the later defined user stories, all those criteria have been assessed to be fulfilled at the time of initial evaluation.

1.3 CHAPTER STRUCTURE

The following chapter contains the user stories clustered in logical groups. There is neither a one-to-one relation of those groups to subsystems nor to system capabilities. The groups are just for the purpose of dividing the large number of user stories in more digestible pieces.

Each user story comes with one or more *Reference*s which make them identifiable in internal or external collections of user stories. Each id that starts with EROPD is a reference to the work package internal Jira project. The ids that start with SPT2CE reference user stories defined by the System Pillar Computing Environment domain.

The *Description* explains the user story in a few sentences from the perspective of the demonstrator work package (WP36).

To set the demonstrator user stories in a meaningful and relevant context, the user stories are mapped to *Related Requirements* out of existing sources.

- As source for requirements on the onboard computing platform, work package 26 suggests considering the work of OCORA, namely High-Level Requirements Generic Safe Computing Platform [2] (all the “approved” requirements MSCP-XX, with XX from 1 to 127 including the optional ones). References to requirements from this source start with the prefix MSCP.
- For FRMCS the UIC has standardized related requirements in the FRMCS System Requirements Specification [3] (FRMCS-SRS), the FRMCS Functional Requirements Specification [4] (FRMCS-FRS) and the On-Board FRMCS Functional Requirements Specification [5] (TOBA-FRS). References to requirements from this source start with the prefix given in round brackets after the source respectively.
- The onboard communication network requirements are standardized in UNISIG FFFIS - CCS Consist Network Communication Layers (SUBSET-147) [6]. References to requirements from this source start with the prefix SUBSET-147.
- Monitoring, Diagnostics, Configuration and Maintenance (MDCM) requirements from the OCORA System Requirements Specification - Monitoring, Diagnostics, Configuration & Maintenance subsystem [7] are referenced by the prefix MDCM-SRS-OCORA.

Finally, the *Test Cases* outline first concrete ideas how to investigate a user story. The test cases define a *Task* to which they belong and a *Responsibility* that is derived from the work split agreed upon in the Statement of Work [8]. The test cases will be extended in number and detail during the implementation tasks. This will include detailed descriptions of the test cases together with test data and required source code and configuration.

2 USER STORIES

The user stories below are a collection of topics of concern of very different granularity and with diverse relevance for the project. Not all user stories will be feasible for practical investigation on the physical demonstrator setup. Some may only be subject to a demonstration in a virtual environment, a theoretical study or are omitted with a justification.

For better readability, “Application” is often abbreviated as “App” without any difference in the meaning. The same holds for the abbreviation of “Safety Layer / Runtime Environment” as “RTE” or the usage of “Platform” instead of “Modular Computing Platform Hardware Pool”.

2.1 MODULAR COMPUTING PLATFORM – DEPLOYMENT & ORCHESTRATION

This group of user stories is concerned with running different kinds of applications on the modular computing platform. To test that an application is running properly, of course additional means to monitor it or to communicate with it are necessary. However, those aspects are still listed as different user stories in order not to mix up the objective that is investigated.

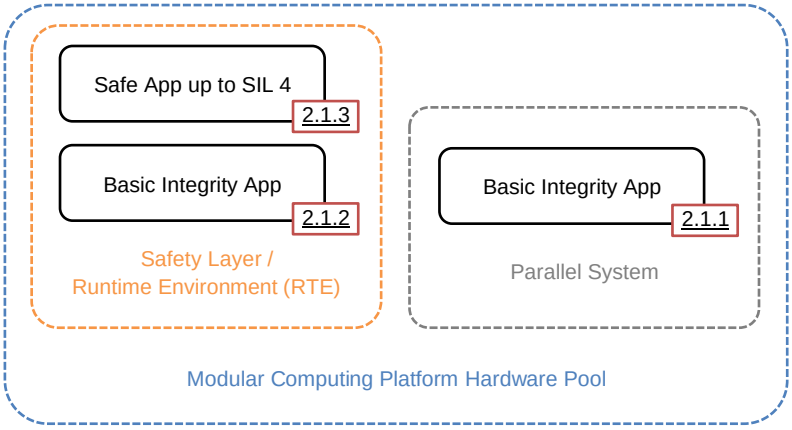

Figure 1: Application Deployment Scenarios

Figure 1: Application Deployment Scenarios shows the different scenarios in which applications of different safety level could be deployed on the modular computing platform hardware pool.

Since the main aim of the modular computing platform is to run multiple applications in parallel on the same hardware, this section also deals with this user story.

Additionally, this section covers user stories around the aspect of keeping the applications running in response to different failures (e.g., hardware or software failures).

The term *platform native* in this chapter refers to an application that was specifically developed to run on the safety layer / runtime environment used in this demonstrator. When an application is described to run “on the safety layer / runtime environment”, it is managed by this runtime environment, while an application that is running “on a parallel system” or called “parallel application” is running by some means on the same hardware but without being managed by the same runtime environment. In the sense of this section, the parallel system on the modular computing platform hardware pool also abstracts a runtime layer from the application with a non-proprietary API, however without providing any support for safe applications.

The safety layer / runtime environment is expected to come with additional services that enable the deployment of safe (up to SIL 4) applications. Those services (e.g., for monitoring, storage, communication, etc.) are referred to in this section by the term *safety services*.

Applications running on systems external to the modular computing platform hardware pool are called “external applications”.

2.1.1 Run a Parallel Basic Integrity Application on Platform

References

EROPD-62

Description

As R2DATO WP36, we want to run a parallel basic integrity application on the modular computing platform hardware pool, so that we can demonstrate a possible virtualisation interface and a possible example of a basic integrity API without safety layer.

Related Requirements

MSCP-112

Test Cases

- Run an application on a single machine of the safe computing platform that can produce the string “hallo world” to some standard output, file or as response to an HTTP request.
  → Task: 36.2
  → Responsibility: GTS
- Run the vehicle diagnostics agent (DIA-VEC) as docker container on a single machine of the safe computing platform.
  → Task: 36.3
  → Responsibility: DB

2.1.2 Run a Basic Integrity Application on Safety Layer / RTE

References

EROPD-11

Description

As R2DATO WP36, we want to run a platform native basic integrity application on the safety layer / runtime environment, so that we can demonstrate an example of a basic integrity API enhanced by access to safety services.

Related Requirements

MSCP-112

Test Cases

- Run a basic integrity application on the safety layer / runtime environment (RTE) that can produce the string “hallo world” to some standard output, file or as response to an HTTP request.
  → Task: 36.2
  → Responsibility: GTS
- Run a communication gate on the safety layer / runtime environment (RTE) that can receive messages from a message queue and relay them to an HTTP request.
  → Task: 36.2
  → Responsibility: GTS

An additional test case is given by the functional application Execute a Simplified ATO Control Loop.

2.1.3 Run a Safe Application (up to SIL 4) on Safety Layer / RTE

References

EROPD-13, SPT2CE-23

Description

As R2DATO WP36, we want to run a platform native safe application (up to SIL 4) on the safety layer / runtime environment, so that we can demonstrate a possible example of a platform independent (PI) API including access to safety services.

Related Requirements

MSCP-21, MSCP-112

Test Cases

- Run a safe “ping-pong” application on the safety layer / runtime environment (RTE) that can send a “ping” message to a message queue and awaits a “pong” message in return.
  → Task: 36.2
  → Responsibility: GTS

  Additional test cases are given by the functional applications described in section 2.10.2 (Execute a Simplified ETCS Control Loop) and section 2.10.3 (Cache Map Data from Digital Register).

2.1.4 Run Multiple Applications

References

EROPD-63, SPT2CE-19

Description

As R2DATO WP36, we want to run multiple combinations of platform native basic integrity and safe (up to SIL 4) applications on the platform as well as the safety layer / runtime environment, so that we can demonstrate the freedom of interference between the deployments.

Related Requirements

MSCP-20, MSCP-36, MSCP-41

Test Cases

- Run a safe application on the safety layer / runtime environment (RTE) and monitor its behaviour in response to artificial system load created by applications in all deployment scenarios depicted in Figure 1: Application Deployment Scenarios.
  → Task: 36.2
  → Responsibility: GTS

2.1.5 Execute Declarative Configuration

References

EROPD-64, (SPT2CE-22)

Description

As R2DATO WP36, we want to configure the deployment of multiple applications of different safety levels (from basic integrity up to SIL 4) on the safety layer / runtime environment, so that we can demonstrate a harmonized configuration.

Related Requirements

MSCP-23, MSCP-38, MSCP-39, MSCP-40, MSCP-62, MSCP-99, FRMCS-FRS-5.3

Test Cases

- Configure the deployment of the “ping-pong” example that involves an application in each deployment scenario depicted in Figure 1: Application Deployment Scenarios.
  → Task: 36.2
  → Responsibility: GTS

2.1.6 Restart Failing Replicas

References

EROPD-41

Description

As R2DATO WP36, we want the safety layer / runtime environment to automatically restart failing replicas of safe applications (up to SIL 4), so that we can demonstrate the resilience in case of spontaneous (not reoccurring) software / hardware failure.

Related Requirements

MSCP-58, MSCP-91, MSCP-108, MSCP-118

Test Cases

- Simulate software / hardware failures by running and monitoring an application that can deliberately bring the replica states out of sync.
  → Task: 36.2
  → Responsibility: GTS

2.1.7 React to Hardware Failure

References

EROPD-65

Description

As R2DATO WP36, we want the safety layer / runtime environment to detect consistently failing replicas of safe applications (up to SIL 4), so that we can demonstrate the detection and reaction of persistent (reoccurring) hardware failure.

Related Requirements

MSCP-91

Test Cases

- Simulate software / hardware failures by running and monitoring an application that can consistently bring the state of one replica out of sync or unplugging one of the used hardware nodes.
  → Task: 36.2
  → Responsibility: GTS

2.1.8 Failover to Cold Backup Hardware

References

EROPD-66, SPT2CE-26

Description

As R2DATO WP36, we want the safety layer / runtime environment to shift load to additional backup hardware, so that we can demonstrate the short-term mitigation of persistent (reoccurring) hardware failure.

Related Requirements

MSCP-37, MSCP-92, MSCP-94

Test Cases

- Unplug one of the used hardware nodes from the modular computing platform while monitoring the remaining nodes, including a yet unused spare node.
  → Task: 36.2
  → Responsibility: GTS

2.1.9 Replace Failing Hardware

References

EROPD-67

Description

As R2DATO WP36, we want the safety layer / runtime environment to integrate replaced hardware of the exact same kind, so that we can demonstrate the long-term mitigation of persistent hardware failures.

Related Requirements

MSCP-22, MSCP-92, MSCP-94

Test Cases

- Restart one of the used hardware nodes of the modular computing platform while monitoring the remaining nodes.
  → Task: 36.2
  → Responsibility: GTS

2.2 Modular Computing Platform – Modularity

2.2.1 Change to Different Hardware

References

EROPD-42, SPT2CE-25

Description

As R2DATO WP36, we want the safety layer / runtime environment to integrate replaced hardware of a different kind, so that we can demonstrate the upgrade of obsolete hardware.

Related Requirements

MSCP-18, MSCP-19, MSCP-26

Test Cases

- Replace one of the used hardware nodes of the modular computing platform with hardware of different properties while monitoring the remaining nodes.
  → Task: 36.2
  → Responsibility: GTS

2.2.2 Change to Different Platform or Safety Layer / RTE

References

EROPD-43, SPT2CE-21, SPT2CE-24

Description

As R2DATO WP36, we want to exchange the platform or safety layer / runtime environment (RTE), so that we can demonstrate the portability of existing applications.

Related Requirements

MSCP-17, MSCP-112, MSCP-113, MSCP-121

Test Cases

n/a (no second source available to this work package)

2.2.3 Use Diverse Hardware

References

ERODP-68, SPT2CE-27

Description

As R2DATO WP36, we want the safety layer / runtime environment to run on diverse hardware, so that we can demonstrate future upgrade / migration scenarios.

Related Requirements

-

Test Cases

n/a (no diverse hardware available to this work package)

2.3 Modular Computing Platform – Communication

A central aspect in the execution of (safe) applications is the ability of the applications to communicate among each other. In this work package the focus especially lies on the abstraction of the communication channel from the business logic of the applications. This abstraction is supposed to be achieved by functions of the modular computing platform or the safety layer / runtime environment. While the used communication pattern may depend on the deployment scenario of the application, it ought to be transparent to the application in which replication it is deployed or where the communication counterpart is deployed.

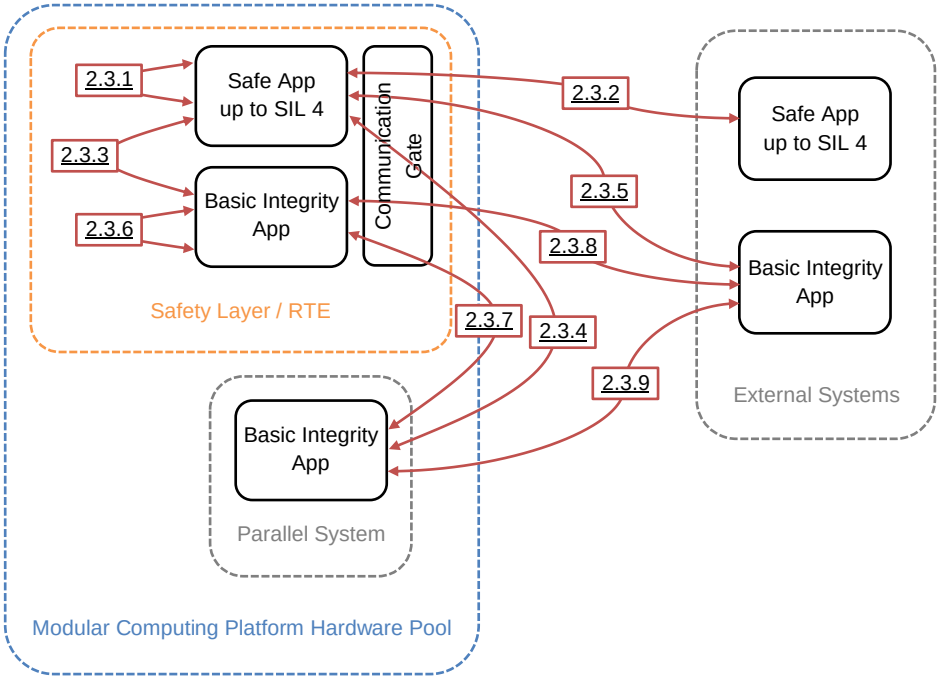

Figure 2: Communication Scenarios

The investigated communication scenarios are depicted in Figure 2: Communication Scenarios.

2.3.1 Communicate Safe App on RTE ↔ Safe App on RTE

References

EROPD-17

Description

As R2DATO WP36, we want a platform native safe application (up to SIL 4) to communicate with another platform native safe application (up to SIL 4), both on the safety layer / runtime environment, so that we can demonstrate a safe communication via the paradigm of flows (as defined in [2]).

Related Requirements

MSCP-44, MSCP-76, MSCP-78, MSCP-79, MSCP-81, MSCP-101, MSCP-102, MSCP-106, MSCP-115, MSCP-116, MSCP-117, MSCP-123

Test Cases

To be specified in 36.2.

2.3.2 Communicate Safe App on RTE ↔ External Safe App

References

EROPD-14, EROPD-15, EROPD-38, EROPD-44, SPT2CE-28

Description

As R2DATO WP36, we want a platform native safe application (up to SIL 4) on the safety layer / runtime environment to communicate with an external safe application (up to SIL 4), so that we can demonstrate a safe communication via the paradigm of flows (as defined in [2]) that leaves the modular computing platform via a communication gate.

Related Requirements

MSCP-44, MSCP-51, MSCP-76, MSCP-78, MSCP-79, MSCP-81, MSCP-101, MSCP-102, MSCP-106, MSCP-115, MSCP-116, MSCP-117, MSCP-123, SUBSET-147-8.4.8.1

Test Cases

See section 2.1.3 (Run a Safe Application (up to SIL 4) on Safety Layer / RTE).

2.3.3 Communicate Safe App on RTE ↔ Basic Integrity App on RTE

References

EROPD-16

Description

As R2DATO WP36, we want a platform native safe application (up to SIL 4) to communicate with a platform native basic integrity application, both on the safety layer / runtime environment, so that we can demonstrate a communication via the paradigm of flows (as defined in [2]).

Related Requirements

MSCP-44, MSCP-76, MSCP-78, MSCP-79, MSCP-81, MSCP-101, MSCP-102, MSCP-106, MSCP-115, MSCP-116, MSCP-117, MSCP-123

Test Cases

See Run a Safe Application (up to SIL 4) on Safety Layer / RTE.

2.3.4 Communicate Safe App on RTE ↔ Parallel Basic Integrity App

References

ERODP-69

Description

As R2DATO WP36, we want a platform native safe application (up to SIL 4) on the safety layer / runtime environment to communicate with a parallel native basic integrity application on the modular computing platform hardware pool, so that we can demonstrate a communication via the paradigm of flows (as defined in [2]) that leaves the safety layer / runtime environment via a communication gate.

Related Requirements

MSCP-44, MSCP-76, MSCP-78, MSCP-79, MSCP-81, MSCP-101, MSCP-102, MSCP-106, MSCP-115, MSCP-116, MSCP-117, MSCP-123

Test Cases

To be specified in 36.2.

2.3.5 Communicate Safe App on RTE ↔ External Basic Integrity App

References

ERODP-15

Description

As R2DATO WP36, we want a platform native safe application (up to SIL 4) on the safety layer / runtime environment to communicate with an external basic integrity application, so that we can demonstrate a communication via the paradigm of flows (as defined in [2]) that leaves the modular computing platform via a communication gate.

Related Requirements

MSCP-44, MSCP-51, MSCP-76, MSCP-78, MSCP-79, MSCP-81, MSCP-101, MSCP-102, MSCP-106, MSCP-115, MSCP-116, MSCP-117, MSCP-123

Test Cases

To be specified in 36.2.

2.3.6 Communicate Basic Integrity App on RTE ↔ Basic Integrity App on RTE

References

ERODP-36

Description

As R2DATO WP36, we want a platform native basic integrity application to communicate with another platform native basic integrity application, both on the safety layer / runtime environment, so that we can demonstrate a communication via the paradigm of flows (as defined in [2]).

Related Requirements

MSCP-44, MSCP-76, MSCP-79, MSCP-101, MSCP-102, MSCP-115, MSCP-116, MSCP-117, MSCP-123

Test Cases

To be specified in 36.2.

2.3.7 Communicate Basic Integrity App on RTE ↔ Parallel Basic Integrity App

References

EROPD-70

Description

As R2DATO WP36, we want a platform native basic integrity application on the safety layer / runtime environment to communicate with a parallel basic integrity application on the modular computing platform hardware pool, so that we can demonstrate a communication via the paradigm of flows (as defined in [2]) that leaves the safety layer / runtime environment via a communication gate.

Related Requirements

MSCP-44, MSCP-76, MSCP-79, MSCP-101, MSCP-102, MSCP-115, MSCP-116, MSCP-117, MSCP-123

Test Cases

To be specified in 36.2.

2.3.8 Communicate Basic Integrity App on RTE ↔ External Basic Integrity App

References

EROPD-12

Description

As R2DATO WP36, we want a platform native basic integrity application on the safety layer / runtime environment to communicate with an external basic integrity application, so that we can demonstrate a communication via the paradigm of flows (as defined in [2]) that leaves the modular computing platform via a communication gate.

Related Requirements

MSCP-44, MSCP-51, MSCP-76, MSCP-79, MSCP-101, MSCP-102, MSCP-115, MSCP-116, MSCP-117, MSCP-123

Test Cases

To be specified in 36.2.

2.3.9 Communicate Parallel Basic Integrity App ↔ External Basic Integrity App

References

EROPD-37

Description

As R2DATO WP36, we want a parallel basic integrity application on the modular computing platform hardware pool to communicate with an external basic integrity application, so that we can demonstrate a generic communication that leaves the modular computing platform.

Related Requirements

-

Test Cases

To be specified in 36.2.

2.3.10 Communicate Independent of RTE Instance

References

EROPD-39

Description

As R2DATO WP36, we want to deploy two communicating applications – unchanged – either on the same or on different instances of the safety layer / runtime environment, so that we can demonstrate the location transparency of the communication model.

Related Requirements

MSCP-93

Test Cases

To be specified in 36.2.

2.3.11 Communicate Independent of Replication

References

EROPD-40

Description

As R2DATO WP36, we want to deploy two communicating applications – unchanged – with different replication configuration on the safety layer / runtime environment, so that we can demonstrate the replication transparency of the communication model.

Related Requirements

MSCP-81

Test Cases

See section 2.1.3 (Run a Safe Application (up to SIL 4) on Safety Layer / RTE).

2.4 FRMCS

2.4.1 Communicate over FRMCS Transparent to the Application

References

EROPD-23, EROPD-45, EROPD-58

Description

As R2DATO WP36, we want an application on the modular computing platform (and an application on the safety layer / runtime environment) to communicate over FRMCS to a trackside entity, so that we can demonstrate an FRMCS communication that is transparent to the application.

Related Requirements

SUBSET-147-3.1.1.3, FRMCS-FRS-11, FRMCS-SRS-6.4.1, TOBA-FRS-7.1

Test Cases

To be specified in 36.3.

2.4.2 Reuse FRMCS Platform Functions

References

EROPD-31

Description

As R2DATO WP36, we want multiple applications on the modular computing platform (and applications on the safety layer / runtime environment) to communicate over FRMCS to a trackside entity, so that we can demonstrate the reuse of FRMCS platform functions.

Related Requirements

-

Test Cases

To be specified in 36.3.

2.4.3 Integrate an Application with FRMCS over Loose Coupling

References

EROPD-25, EROPD-47

Description

As R2DATO WP36, we want to implement an OBapp client for an application on the modular computing platform (and an application on the safety layer / runtime environment) to enable communication over FRMCS to a trackside entity, so that we can demonstrate a “loose coupling” communication model where the MCx client is integrated in the FRMCS gateway.

Related Requirements

FRMCS-SRS-7.1.5.9, TOBA-FRS-7.7.2.8

Test Cases

To be specified in 36.3.

2.4.4 Failover to Redundant FRMCS Platform Functions

References

EROPD-28, EROPD-32

Description

As R2DATO WP36, we want an application on the modular computing platform (and an application on the safety layer / runtime environment) to communicate over redundant FRMCS gateways to a trackside entity, so that we can demonstrate the increase in availability in failover scenarios.

Related Requirements

TOBA-FRS-8.1

Test Cases

To be specified in 36.3.

2.4.5 Host FRMCS Onboard Services on the Modular Computing Platform

References

EROPD-24, EROPD-29, EROPD-30, EROPD-46,

Description

As R2DATO WP36, we want to split onboard FRMCS TOBA functions, with parts of the functions hosted on the same hardware pool as the modular computing platform and decuple the modem hardware (e.g., with OBrad if available) via the CCN (also analysing the hardware effort of the different hosting scenarios), so that we can demonstrate the modularity of the TOBA solution.

Related Requirements

FRMCS-SRS-7.1, TOBA-FRS-5.2.1.5, TOBA-FRS-7.9

Test Cases

n/a (OBrad specification not yet available)

2.5 MONITORING, DIAGNOSTICS, CONFIGURATION AND MAINTENANCE (MDCM)

As part of the shared services the Monitoring, Diagnostics, Configuration and Maintenance (MDCM) is communicating with onboard applications deployed in different scenarios. The following user stories cover those communication flows are shown in Figure 3: Diagnostics Data Flow.

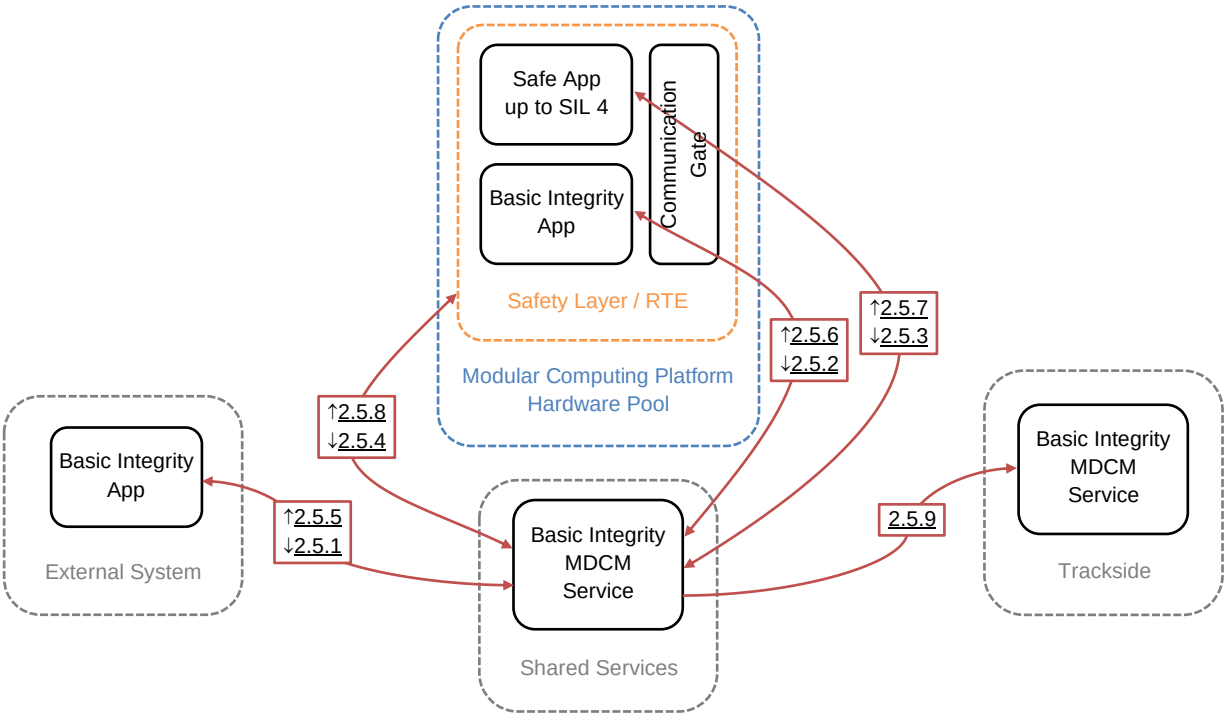

Figure 3: Diagnostics Data Flow

The communication path to access vehicle data is covered by a separate user story, which due to the needed abstraction of the vehicle as well as additional aggregation agents does not fit in the generic diagram.

2.5.1 Collect Diagnostics Data from External Basic Integrity App

References

EROPD-50, EROPD-59

Description

As R2DATO WP36, we want to collect diagnostics data from an external basic integrity application through a standardized interface, so that we can demonstrate a harmonised diagnostics service on the modular computing platform.

Related Requirements

MDCM-SRS-OCORA-8216

Test Cases

To be specified in 36.3.

2.5.2 Collect Diagnostics Data from Basic Integrity App on RTE

References

EROPD-50

Description

As R2DATO WP36, we want to collect diagnostics data from a basic integrity application on the safety layer / runtime environment through a standardized interface, so that we can demonstrate a harmonised diagnostics service on the modular computing platform.

Related Requirements

MSCP-59, MSCP-73, MSCP-84, MSCP-86, MSCP-87, MSCP-111, MSCP-124, MDCM-SRS-OCORA-8216

Test Cases

To be specified in 36.3.

2.5.3 Collect Diagnostics Data from Safe App on RTE

References

EROPD-50

Description

As R2DATO WP36, we want to collect diagnostics data from a safe application (up to SIL 4) on the safety layer / runtime environment through a standardized interface, so that we can demonstrate a harmonised diagnostics service on the modular computing platform.

Related Requirements

MSCP-59, MSCP-73, MSCP-84, MSCP-86, MSCP-87, MSCP-111, MSCP-124, MSCP-127, MDCM-SRS-OCORA-8216

Test Cases

To be specified in 36.3.

2.5.4 Collect Diagnostics Data from Safety Layer / RTE

References

ERODP-71

Description

As R2DATO WP36, we want to collect diagnostics data from the safety layer / runtime environment through a standardized interface, so that we can demonstrate a harmonised diagnostics service on the modular computing platform.

Related Requirements

MSCP-58, MSCP-60, MSCP-84

Test Cases

To be specified in 36.4.

2.5.5 Provide Diagnostics Events to External Basic Integrity App

References

ERODP-72

Description

As R2DATO WP36, we want to provide diagnostics events to an external basic integrity application through a standardized interface, so that we can demonstrate a harmonised diagnostics service on the modular computing platform.

Related Requirements

MDCM-SRS-OCORA-7425

Test Cases

To be specified in 36.3.

2.5.6 Provide Diagnostics Events to Basic Integrity App on RTE

References

ERODP-73

Description

As R2DATO WP36, we want to provide diagnostics events to a basic integrity application on the safety layer / runtime environment through a standardized interface, so that we can demonstrate a harmonised diagnostics service on the modular computing platform.

Related Requirements

MSCP-111, MDCM-SRS-OCORA-7425

Test Cases

To be specified in 36.3.

2.5.7 Provide Diagnostics Events to Safe App on RTE

References

ERODP-74

Description

As R2DATO WP36, we want to provide diagnostics events to a safe application (up to SIL 4) on the safety layer / runtime environment through a standardized interface, so that we can demonstrate a harmonised diagnostics service on the modular computing platform.

Related Requirements

MSCP-85, MSCP-111, MSCP-126, MDCM-SRS-OCORA-7425

Test Cases

To be specified in 36.3.

2.5.8 Provide Diagnostics Events to Safety Layer / RTE

References

ERODP-75

Description

As R2DATO WP36, we want to provide diagnostics events to the safety layer / runtime environment through a standardized interface, so that we can demonstrate a harmonised diagnostics service on the modular computing platform.

Related Requirements

MSCP-60

Test Cases

To be specified in 36.4.

2.5.9 Exchange Diagnostics Data with Trackside Diagnostics Service

References

ERODP-76

Description

As R2DATO WP36, we want to exchange diagnostics data with a trackside diagnostics service over FRMCS, so that we can demonstrate a harmonised diagnostics service on the modular computing platform.

Related Requirements

MDCM-SRS-OCORA-7427, MDCM-SRS-OCORA-7629, MDCM-SRS-OCORA-7631, MDCM-SRS-OCORA-7632, MDCM-SRS-OCORA-8045

Test Cases

To be specified in 36.4.

2.5.10 Collect and Aggregate Diagnostics Data from Vehicle

References

ERODP-59

Description

As R2DATO WP36, we want to collect and aggregate diagnostics data from the vehicle via the TCMS Data Service, so that we can demonstrate the vehicle diagnostics agent (DIA-VEC) as basic integrity application on the modular computing platform.

Related Requirements

MDCM-SRS-OCORA-8220

Test Cases

To be specified in 36.3.

2.6 SOFTWARE UPDATE

Updating onboard software is a complex subject which involves a special process to ensure compatibility of all systems as well as their valid safety assessment. This work package does not focus on the safety related process of updating onboard systems but rather on the technical workflow how to update specific applications or sub-systems.

The software update scenarios in focus are shown in Figure 4: Software Update Scenarios.

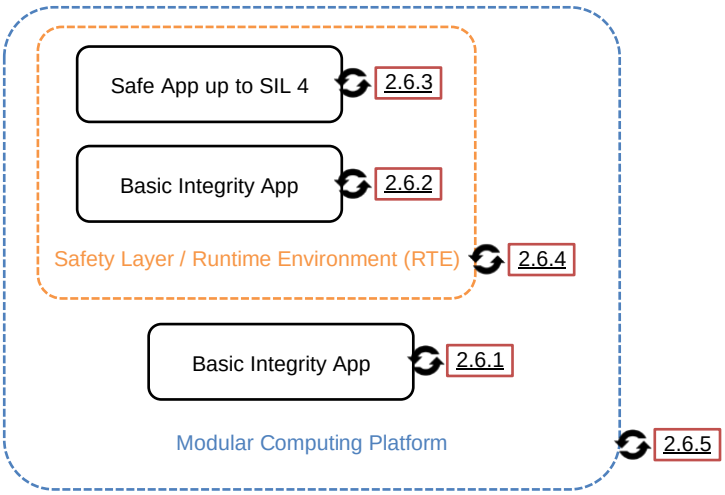

Figure 4: Software Update Scenarios

Additionally, there are user stories concerning the rollback of an (unsuccessful) update and the aspect of updating software over the air.

2.6.1 Update a Basic Integrity Application on Platform

References

ERODP-77

Description

As R2DATO WP36, we want to update a basic integrity application on the modular computing platform, so that we can demonstrate the technical workflow for the software update.

Related Requirements

FRMCS-FRS-5.3, MDCM-SRS-OCORA-8046

Test Cases

To be specified in 36.4.

2.6.2 Update a Basic Integrity Application on Safety Layer / RTE

References

EROPD-78

Description

As R2DATO WP36, we want to update a basic integrity application on the safety layer / runtime environment, so that we can demonstrate the technical workflow for the software update.

Related Requirements

MSCP-64, MSCP-69, FRMCS-FRS-5.3, MDCM-SRS-OCORA-8046

Test Cases

To be specified in 36.4.

2.6.3 Update a Safe Application on Safety Layer / RTE

References

EROPD-79

Description

As R2DATO WP36, we want to update a safe application (up to SIL 4) on the safety layer / runtime environment, so that we can demonstrate the technical workflow for the software update.

Related Requirements

MSCP-64, MSCP-69, FRMCS-FRS-5.3, MDCM-SRS-OCORA-8046

Test Cases

To be specified in 36.4.

2.6.4 Update the Safety Layer / RTE

References

EROPD-80

Description

As R2DATO WP36, we want to update the safety layer / runtime environment on the modular computing platform, so that we can demonstrate the technical workflow for the software update of the safety and runtime layer.

Related Requirements

MSCP-67

Test Cases

To be specified in 36.4.

2.6.5 Update the Platform

References

ERODP-81

Description

As R2DATO WP36, we want to update the modular computing platform, so that we can demonstrate the technical workflow for the software update of the virtualization layer.

Related Requirements

MSCP-67

Test Cases

To be specified in 36.4.

2.6.6 Rollback a Software / Configuration Update

References

ERODP-82, SPT2CE-30 / SPT2CE-20

Description

As R2DATO WP36, we want to rollback a software / configuration update, so that we can demonstrate the failsafe scenario for a corrupt / incomplete update process.

Related Requirements

MSCP-62, MSCP-68, MSCP-99

Test Cases

To be specified in 36.4.

2.6.7 Perform a Software / Configuration Update over the Air (FRMCS)

References

ERODP-57

Description

As R2DATO WP36, we want to perform a software / configuration update over the air using FRMCS, so that we can demonstrate possibility of a remote update.

Related Requirements

MSCP-62, MSCP-63, MSCP-64, MSCP-66, MSCP-99, FRMCS-FRS-5.3, MDCM-SRS-OCORA-7426

Test Cases

To be specified in 36.4.

2.7 ONBOARD COMMUNICATION NETWORK

The following user stories concerning the onboard communication network are all related to work package 24 and are therefore highly dependent on the timely delivery of the respective deliverables.

2.7.1 Allow Exchange of Data over the Network

References

EROPD-83

Description

As R2DATO WP36, we want to exchange data between onboard components via a standardized onboard communication network, so that we can demonstrate the standardized integration of the modular computing platform.

Related Requirements

SUBSET-147-3.1.1.1, SUBSET-147-8.4

Test Cases

To be specified in 36.3.

2.7.2 Provision of Uninterrupted Connectivity

References

EROPD-84

Description

As R2DATO WP36, we want to provide uninterrupted connectivity between onboard components even if up to a defined number of onboard components fail, so that we can demonstrate reliable communication paths.

Related Requirements

SUBSET-147-8.4.1.3.1, SUBSET-147-8.4.6

Test Cases

To be specified in 36.3.

2.7.3 Allow Management Access to Onboard Components

References

EROPD-85

Description

As R2DATO WP36, we want to have management access to all onboard components remotely or physically, so that can demonstrate the update of configurations or software.

Related Requirements

-

Test Cases

To be specified in 36.3.

2.7.4 Allow Ease of Access to the Network

References

EROPD-86

Description

As R2DATO WP36, we want to allow plug-and-playability of components after an initial identity and access management (IAM) configuration, so that we can demonstrate an easy reconfiguration when a component is added or removed.

Related Requirements

-

Test Cases

To be specified in 36.3.

2.7.5 Supply List of Onboard Network Components

References

EROPD-87

Description

As R2DATO WP36, we want to automatically supply a list of all configured onboard components, so that we can demonstrate maintainable configurations.

Related Requirements

MDCM-SRS-OCORA-7630

Test Cases

To be specified in 36.3.

2.7.6 Provision of Date Time Reference

References

EROPD-88

Description

As R2DATO WP36, we want to have a common date time reference for all onboard components, so that we can demonstrate a shared service crucial for reliable information sharing.

Related Requirements

SUBSET-147-9, FRMCS-FRS-8.2.12

Test Cases

To be specified in 36.3.

2.7.7 Prioritise Specific Network Traffic

References

EROPD-89

Description

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.

Related Requirements

MSCP-80, SUBSET-147-8.4.3.1, SUBSET-147-8.4.3.2, , SUBSET-147-8.4.4

Test Cases

To be specified in 36.3.

2.7.8 Monitor the Network Health Status / Guaranteed Characteristics

References

EROPD-90

Description

As R2DATO WP36, we want to monitor the network health status, so that we can demonstrate how the guaranteed network characteristics can be measured.

Related Requirements

MSCP-80

Test Cases

To be specified in 36.3.

2.8 IT/ OT SECURITY

2.8.1 Realize Authentication and Authorization Mechanisms

References

EROPD-91

Description

As R2DATO WP36, we want to realize authorization and authentication mechanisms for the nodes in the network, so that we can demonstrate a measure to ensure the integrity of the sent data.

Related Requirements

SUBSET-147-8.4.3.3, SUBSET-147-8.4.10.3

Test Cases

To be specified in 36.3.

2.8.2 Change Identity and Access Management Settings

References

EROPD-92, (SPT2CE-29)

Description

As R2DATO WP36, we want to change Identity and Access Management (IAM) settings of the modular computing platform, so that we can demonstrate the possibility to react to security threats.

Related Requirements

-

Test Cases

To be specified in 36.3.

2.8.3 Encrypt all Data on the Network

References

EROPD-93

Description

As R2DATO WP36, we want to encrypt all data that is exchanged between the components on the onboard network, so that we can demonstrate the feasibility of secure communication over grey channels.

Related Requirements

SUBSET-147-8.4.7.2

Test Cases

To be specified in 36.3.

2.8.4 Realize Secure Communication over FRMCS

References

EROPD-33

Description

As R2DATO WP36, we want to realize a secure communication over FRMCS, so that we can demonstrate the feasibility of safe communication over FRMCS as grey channel.

Related Requirements

FRMCS-FRS-11.18, FRMCS-SRS-15.5

Test Cases

To be specified in 36.3.

2.8.5 Separation of Network Zones

References

EROPD-94

Description

As R2DATO WP36, we want to separate network zones and conduits according to the security levels of the applications, so that we can demonstrate the virtual separation of applications of different security levels.

Related Requirements

SUBSET-147-8.4.3.1, SUBSET-147-8.4.3.2

Test Cases

To be specified in 36.3.

2.8.6 Detect Unusual Network Traffic

References

EROPD-95

Description

As R2DATO WP36, we want to receive notifications when unusual traffic goes through the network, so that we can demonstrate the detection of unusual traffic.

Related Requirements

-

Test Cases

To be specified in 36.3.

2.8.7 Detect Unauthorised Network Device

References

EROPD-96

Description

As R2DATO WP36, we want to receive alarms when unauthorised devices are added to the network, so that we can demonstrate the detection of unauthorized devices.

Related Requirements

-

Test Cases

To be specified in 36.3.

2.9 TRAIN ABSTRACTION

2.9.1 Provide Safe Access Train Hardware via a Functional Vehicle Adapter

References

EROPD-61

Description

As R2DATO WP36, we want to integrate an application on the modular computing platform with a Functional Vehicle Adapter, so that we can demonstrate the abstraction of train hardware for safe applications (up to SIL 4).

Related Requirements

-

Test Cases

n/a (no implementation of a Functional Vehicle Adapter available to this work package; may be replaced by the simulation of a safe I/O)

2.9.2 Provide Access to Train Hardware via a TCMS Data Service

References

EROPD-97

Description

As R2DATO WP36, we want to integrate an application on the modular computing platform with a TCMS Data Service (as defined in https://eurospec.eu/download/tcms-dataservices-1-0), so that we can demonstrate the abstraction of train hardware for basic integrity applications.

Related Requirements

-

Test Cases

Is covered by the test case for Collect and Aggregate Diagnostics Data from Vehicle.

2.10 FUNCTIONAL APPLICATIONS

2.10.1 Execute a Simplified ATO Control Loop

References

EROPD-56

Description

As R2DATO WP36, we want to host a simplified ATO onboard application on the safety layer / runtime environment executing an ATO control loop, so that we can demonstrate a basic integrity application communicating over FRMCS and interacting with the TCMS.

Related Requirements

SUBSET-147-3.1.1.2, SUBSET-147-3.1.1.7 (UNISIG SUBSET-125), FRMCS-FRS-11.5, MDCM-SRS-OCORA-8053, MDCM-SRS-OCORA-8215

Test Cases

- Run a basic integrity ATO onboard application on the safety layer / runtime environment (RTE) that can execute an ATO control loop.
  → Task: 36.4
  → Responsibility: SBB

2.10.2 Execute a Simplified ETCS Control Loop

References

EROPD-55

Description

As R2DATO WP36, we want to host a simplified ETCS onboard application on the safety layer / runtime environment executing an ETCS control loop, so that we can demonstrate a safe application communicating over FRMCS and interacting with the TCMS.

Related Requirements

SUBSET-147-3.1.1.4, SUBSET-147-3.1.1.7 (UNISIG SUBSET-026), MDCM-SRS-OCORA-8044, MDCM-SRS-OCORA-8214

Test Cases

- Run a safe ETCS onboard application on the safety layer / runtime environment (RTE) that can execute an ETCS control loop.
  → Task: 36.4
  → Responsibility: SBB

2.10.3 Cache Map Data from Digital Register

References

EROPD-60

Description

As R2DATO WP36, we want to host an onboard map cache for the digital register on the safety layer / runtime environment, so that we can demonstrate a safe application communicating to a safe trackside entity over FRMCS and using safe persistent storage.

Related Requirements

MSCP-43, MSCP-48

Test Cases

- Run a safe dbs-onboard-map-api application on the safety layer / runtime environment (RTE) that can cache requests to the digital register.
  → Task: 36.4
  → Responsibility: DB

3 TEST CASES

Where possible and feasible the test cases are defined and described in the chapters above and together with the corresponding user stories. Most user stories are on a high abstraction level. The specification and refinement of the test cases will happen together with the corresponding user stories in the implementation tasks 36.2-4.

The refinement of the test cases will base on the work of R2DATO work package 34. Even so their work focusses on testing in the context of the certification process and aims to provide finally a test facility to do virtual certification testing, we might benefit from their deliverables. Of interest for our work package 36 (which will not be certified) are the topics methodology, testing framework, and test bench architecture. All these will be available in M21 (August 2024) from work package 34 and then be evaluated and incorporated or used as appropriate in this work package.

Because work package 36 aims to deliver only a demonstrator (and not a product) the test cases must be tailored accordingly: Main goal is to make sure the defined functionality of modules is properly delivered, and the modules can be used and integrated in the proceeding implementation tasks of the work package by all team members (partners/contributors).

WORKS CITED

[1] ERJU R2DATO WP36, “Deliverable 36.1 - Demonstrator Specification - System Definition,” 2023.

[2] OCORA, “High-Level Requirements - Generic Safe Computing Platform,” 2022. [Online]. Available: https://github.com/OCORA-Public/Publications/blob/master/00_OCORA%20Latest%20Publications/Latest%20Release/OCORA-TWS03-020_Computing-Platform-Requirements.pdf.

[3] UIC, “FRMCS - System Requirements Specification,” 2023. [Online]. Available: https://uic.org/IMG/pdf/3._frmcs_system_requirements_specification_at-7800-1.0_0.pdf.

[4] UIC, “FRMCS - Functional Requirements Specification,” 2023. [Online]. Available: https://uic.org/IMG/pdf/1._frmcs_functional_requirements_specification_fu-7120-1.0_0.pdf.

[5] UIC, “On-Board FRMCS - Functional Requirements Specification,” 2023. [Online]. Available: https://uic.org/IMG/pdf/2._toba_functional_requirements_specification_toba-7510-1.0_0.pdf.

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

[7] OCORA, “System Requirements Specification - Monitoring, Diagnostics, Configuration & Maintenance subsystem,” 2023. [Online]. Available: https://github.com/OCORA-Public/Publications/blob/master/00_OCORA%20Latest%20Publications/Latest%20Release/OCORA-TWS08-030_MDCM-SRS.pdf.

[8] ERJU R2DATO WP36, “Deliverable 36.1 - Demonstrator Specification - Statement of Work,” 2023.