

ERJU FP2 WP27 Digital Register Specification, Development, and Implementation

D27.3 – Updated requirements and specification of Digital Register

Due date of deliverable: 15/10/2025

Actual submission date: 13/10/2025

Leader/Responsible of this Deliverable: Danny Kreyenberg, DB InfraGO AG

Reviewed: Y

Document status		
Revision	Date	Description
01	15/08/2025	First issue of updated set of requirements for review within WP27
02	15/09/2025	Intermediate Version after internal WP review
03	13/10/2025	Final version for TMT review
04	18/10/2025	Clean version without comments for TMT Review
05	22/01/2026	Clean version after implementing JU Comments

Project funded from the European Union's Horizon Europe research and innovation programme		
Dissemination Level		
PU	Public	x
CO	Confidential, restricted under conditions set out in Model Grant Agreement	

Start date: 01/12/2022

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
Harish Narayanan	Nextrail / DB InfraGO	Author
Ibtihel Cherif	Nextrail / DB InfraGO	Author
Bazant Lubor	AZD	Reviewer
Xiaolu Rao	SBB	Reviewer
Benedikt Wenzel	Nextrail / DB InfraGO	Reviewer

Disclaimer

The information in this document is provided “as is”, and no guarantee or warranty is given that the information is fit for any particular purpose. The content of this document reflects only the author's view – the Joint Undertaking is not responsible for any use that may be made of the information it contains. The users use the information at their sole risk and liability.

EXECUTIVE SUMMARY

This deliverable, part of the Europe's Rail Joint Undertaking (ERJU) Flagship Project 2 (FP2) under Work Package 27 (WP27), presents the updated requirements and specifications for the Digital Register (DR)—an integral system within the R2DATO initiative. The DR is designed to centralize and manage static infrastructure data for configuration of railway systems such as:

- Automatic Train Operation Transactor (ATO-TA)
- Traffic Management System (TMS)
- Plan Execution (PE)
- Moving Block System (MBS)
- Absolute Safe Train Positioning (ASTP)
- Perception On-Board (PER-OB)
- Automatic Processing Module On-Board (APM-OB)
- Automatic Driving Module (ADM)

Key Enhancements in D27.3

- **Refined Scope:** The DR now focuses on the results of data preparation and configuration of systems rather than the preparation process itself. Responsibilities for data import, validation, and compilation are formally assigned to the Data Preparation, not the DR.
- **Logical Architecture:** Introduces a modular architecture with components like:
 - Maintenance and Data Management (MDM)
 - Repository Trackside (REP-TS)
 - Digital Register Transactor (DR-TA)
 - Digital Register On-Board (DR-OB)
- **Trackside Interface Definitions:** Detailed logical interfaces between DR components and consuming systems (e.g., TMS, MBS, ATO-TA) are specified based on EULYNX SMI Interfaces for Configuration Data distribution and activation.
- **Scenario-Based Requirements:** Scenarios (e.g., Activation, Loading, powering on, border crossing, coupling/uncoupling trains) are used to define system behaviour and requirements for both trackside and on-board systems.
- **On-Board Communication Protocols:** Establishes handshake, session management, and data exchange protocols between DR-TA and DR-OB, based on Subset 126 and FFFIS standards.

ABBREVIATIONS AND ACRONYMS

ADM	Automatic Driving Module
API	Application programming interface
APM	Automatic Processing Module
ASTP	Advanced Safe Train Positioning
ATO	Automatic Train Operation
CCS	Command, Control, and Signalling
CM	Configuration Manager
CONEMP	Concept of Employment
DB	Deutsche Bahn
DR	Digital Register
EDT	Enabling Digital Technologies
ERJU	Europe's Rail Joint Undertaking
ETCS	European Train Control System
FFFIS	Form Fit Function Interface Specification
GoA	Grade of Automation
HTTPS	Hyper Text Transfer Protocol
IM	Infrastructure Manager
IP	Innovation Pillar
JU	Joint Undertaking
MBS	Moving Block System
MDM	Maintenance and Data Management
NL	Non-Leading
OB	On-Board
OC	Object Controller
OPC-UA	Open Platform Communications – Unified Architecture
PE	Plan Execution
PER	Perception
PREP	Preparation
pTAN	Provider Transaction Access Number
R2DATO	Rail to Digital and Automated Train Operation
REP	Repository
REST	Representational State Transfer
RU	Railway Undertaking
SCA	Safe Configuration Authority
SCF	Safe Configuration Function
SCI	Standard Communication Interface
SERA	Single European Railway Area
SH	Shunting
SMI	Standard Maintenance Interface
SW	Software
SysC	System Capability
TA	Transactor
TAN	Transaction Access Number
TCCS	Transversal CCS
TMS	Traffic Management System
TRN	Train Running Number
TS	Trackside
TSI	Technical Specification for Interoperability
TT	Track Train
WP	Work Package

TABLE OF CONTENTS

Acknowledgements.....	2
Report Contributors.....	2
Executive Summary	3
Abbreviations and Acronyms	4
Table of Contents.....	5
List of Figures	7
List of Tables.....	7
1 Introduction	8
1.1 Definitions	8
1.2 Summary of Updates compared to D27.1	11
1.2.1 General changes	12
1.2.2 General remarks.....	12
1.2.3 Updates related to Scope of DR.....	12
1.2.4 Updates related to System Capabilities and System Functions.....	13
1.2.5 Updates related to actor functions.....	15
1.2.6 Updates related to exported constraints.....	15
2 Logical Architecture.....	16
2.1 Logical Component Definition	16
2.1.1 Maintenance and Data Management (MDM)	16
2.1.2 Repository Trackside (REP-TS).....	17
2.1.3 Digital Register Transactor (DR-TA)	17
2.1.4 Digital Register On-Board (DR-OB)	17
2.2 Logical Actor Definition	17
2.2.1 Moving Block System	17
2.2.2 Plan Execution System	17
2.2.3 Automatic Train Operation Transactor	17
2.2.4 Traffic Management System	17
2.2.5 Absolute Safe Train Positioning System	17
2.2.6 Perception On-Board.....	17
2.2.7 Automatic Driving Module	17
2.2.8 Automatic Processing Module On-Board	18
2.2.9 Configuration Manager.....	18
2.2.10 Data Preparation	18
3 Interfaces	18
4 Process description.....	19
4.1 Scenarios	20

4.1.1	Data Distribution, loading, and activation to Trackside systems	20
4.1.2	Data distribution to On-Board Systems	31
4.1.3	Messages	37
5	Exported constraints	38
5.1	Data Preparation	38
6	References	39

LIST OF FIGURES

Figure 1: Logical Architecture of DR.....	16
Figure 2: Overview configuration process (for infrastructure-related Configuration Data).....	19
Figure 3 Scenario: Start distribution of Configuration Data.....	21
Figure 4 Scenario: Loading of Configuration Data.....	22
Figure 5 Scenario: Activation of Top-Level Configuration Data.....	23
Figure 6 Scenario: Deactivation of Configuration Data with version switch.....	25
Figure 7 Scenario: Activation of Configuration Data.....	26
Figure 8 Scenario: Confirmation	28

LIST OF TABLES

Table 1: Terms and Definitions.....	8
Table 2: Updates related to Scope of DR	13
Table 3: Updates related to System Capabilities.....	14
Table 4: Logical Interfaces Overview.....	18
Table 5 Scenario description: Start distribution of Configuration Data	21
Table 6 Scenario description: Loading of Configuration Data.....	22
Table 7 Scenario description: Activation of Top-Level Configuration Data.....	23
Table 8 Scenario description: Deactivation of Configuration Data with version switch.....	25
Table 9 Scenario description: Activation of Configuration Data.....	26
Table 10 Scenario description: Confirmation	28

1 INTRODUCTION

This document, *D27.3 – Updated Requirements and Specification of the Digital Register*, is a key deliverable within Work Package 27 (WP27) of the Europe's Rail Joint Undertaking (ERJU) Flagship Project 2 (FP2). It builds upon the foundational work established in *D27.1* and reflects the evolution of the Digital Register (DR) concept based on insights gained through the System Pillar and the ongoing activities of WP27.

The Digital Register is a central enabler within the R2DATO (Rail to Digital and Automated Train Operation) initiative. It is designed to provide a harmonized, reliable, and standardized source of static infrastructure data to a wide range of railway systems.

This updated specification reflects a shift in focus from defining the internal processes of data preparation to emphasizing the reliable distribution, activation, and consumption of Configuration Data across both trackside and on-board systems. The document introduces a refined logical architecture, updated system capabilities, and detailed interface definitions to support this transition.

The updates presented in *D27.3* are the result of collaborative work between the System Pillar Transversal CCS Domain and the Cluster Enabling Digital Technologies (EDT), ensuring alignment with broader ERJU objectives and interoperability standards. The deliverable also incorporates lessons learned from practical implementation scenarios and feedback from stakeholders across the railway sector.

In the chapters that follow, the document outlines the updated scope of the Digital Register, defines its logical components and interfaces, and details the scenarios and requirements.

To support these, this deliverable includes 2 annexes to it namely,

1. Annex 1 - Track-Train Communication Session Management Principles [1]
2. Annex 2 - Digital Register Transactor and Digital Register On-Board FFFIS Application Layer [2]

1.1 DEFINITIONS

This section provides precise definitions for key terms and concepts used throughout the document. It ensures clarity and consistency in understanding the technical language and system components described in the specification.

Table 1: Terms and Definitions

Term	Definition
Configuration Data	The (CCS/TMS) Configuration Data refers to a conglomerate of different Configuration Data required for CCS/TMS systems. These can be broadly classified as Application Configuration Data and System Configuration Data. CCS/TMS Configuration Data is provided via the configuration interface to the CCS/TMS Systems. The Configuration Data is assumed static within a version and changes occur only when there is a version change or increase, opposite to dynamic data, which may change within a configuration version of the system.

Configuration Management Process	The Configuration Management Process captures all concrete activities along the complete configuration management chain: from the Trackside Consuming System realisation (by the Supplier(s)), to the integration into a specific System Instance (by the Integrator(s)), to the distribution to the field (by the Operator), and the actual activation (going into operation) in the field (it includes also on a train). The Configuration Management Process is a solution agnostic term that collects all activities needed for the Configuration Management.
System Configuration Data	The System Configuration Data refers to the static data set required to configure systems with primary information before being put into operation. These data are elaborated as Parameter Data
Parameter Data	Parameter Data define the system Configuration Data required for national and supplier-specific operative environments. A notable example of such data are ETCS national values. ETCS national values may be required for migration purposes and shall be replaced by SERA standardised values in the target system.
Application Configuration Data	The Application Configuration Data refers to use case-specific data for the Consuming Systems for a specific application. These can be detailed as Infrastructure data i.e., Track edges, Track geometry, Track properties, Segment Profiles, etc. and Vehicle data i.e., Braking and Traction efforts, Rolling coefficients, etc.
Infrastructure Data	Infrastructure Data is a detailed digital representation of the railway network that contains all infrastructure related information necessary for planning and performing railway operations, such as infrastructure characteristics, location and details of Field Elements, etc. The Infrastructure Data is static and remains unchanged until intended infrastructure updates occur.
Vehicle Data	Vehicle Data is a detailed definition of the static train/vehicle characteristics i.e., Braking and Traction efforts, rolling coefficients, of the train, wagons, locomotives used for railway operations
Software Configuration Data	The Software Configuration Data refers to executable binary files from suppliers which are usually required to install software on the systems. Software Configuration Data is provided via configuration interface to the CCS/TMS Systems.
Black Box Data	SW Configuration Data content format that is supplier specific. The payload is not standardized for this case, therefore it is "black box" and being distributed as non-transparent payload.
Binary Executable File	A Binary Executable File is a platform-specific file that contains compiled code in a machine-readable binary format, which can be directly executed by a computer's processor
Operational Data	Operational Data refers to real-time information generated from daily operations and activities related to the functioning of railway systems, reflecting the current status of operations (e.g. locked position of a switch). Operational data is exchanged between CCS/TMS systems via Standard

	Communication Interfaces (SCI-xx). Note: while Operational Data can be related to infrastructure or vehicle Configuration Data, it is clearly separated from Configuration Data.
Dependency tree	The dependency tree is established during the Configuration Data preparation phase and represents the hierarchical relationships between subsystem's Configuration Data, indicating which subsystem's Configuration Data depends on other Configuration Data.
Loading	In the loading phase of the configuration update process, new Configuration Data are transferred to the Trackside Consuming System before the actual activation takes place. This step ensures that all Trackside Consuming Systems have the necessary data in advance, minimizing downtime and reducing the risk of operational disruption. The background loading is supported and designed to run without restrictions, meaning it does not interfere with operations or require immediate activation.
Safe Top-Level Configuration Data Activation	<p>The Safe Top-Level Configuration Data Activation phase is a step defined within the configuration update process to ensure the controlled and secure activation of new configurations in the trackside system. Its primary purpose is to guarantee that the transition to a new configuration is both intentional and verifiable, with clear traceability and accountability.</p> <p>This phase introduces a Configuration Manager triggered activation mechanism that requires validation through a ConfirmationCode, thereby preventing unintended or unauthorized configuration changes.</p>
Deactivation	<p>The deactivation of Configuration Data is step defined within the configuration update process and executed prior to the activation of a new configuration version. Its main objective is to ensure that all Configuration Data requiring a version switch are reliably taken out of operation across all relevant subsystems (e.g., TMS, PE, MBS). A critical design principle in this phase is the top-down deactivation order, which reflects the underlying nature of Configuration Data dependencies.</p> <p>Additionally, this process includes the use of protected Transaction Authorization Numbers (pTANs) and shared secrets to ensure that only authorized subsystems can process deactivation requests, maintaining full traceability and security compliance.</p>
Activation	<p>The Activation phase is a critical part of the configuration update process. It ensures that each trackside consuming system is correctly installed, verifiable, and safely integrated into the system before it becomes operational. The primary goal of this phase is to transition Configuration Data from a preloaded state into an installed and confirmable state.</p> <p>A central aspect of this phase is the enforcement of bottom-up activation, which follows the dependency structure defined in the dependency tree. Because higher-level Configuration Data depend functionally and structurally on lower-level ones, activation must start from the bottom of the dependency tree. This order guarantees that each Configuration Data is only activated</p>

	<p>once all its dependencies are already active, ensuring a coherent and stable configuration baseline throughout the system.</p> <p>During activation, product-specific installation processes are triggered for each trackside consuming system. To maintain security and data integrity, post-installation hashes are calculated to verify that the expected Configuration Data has been installed as intended. These hashes are protected using generated Transaction Authorization Numbers (TANs), which prevent replay attacks and guarantee freshness of installation proof.</p> <p>At the end of the activation phase, the Configuration Data is marked as installed and awaits confirmation.</p>
Confirmation	<p>The Confirmation phase serves as the final safeguard in the configuration update process, ensuring that only correctly installed and validated Configuration Data are permitted to enter operational use. Its main purpose is to verify that the activation and installation process was successful and that the installed Configuration Data exactly matches the pre-approved and preloaded version expected.</p> <p>This phase provides an integrity check by comparing post-installation hashes (generated during the activation phase) with the expected values that were calculated during Configuration Data preparation. For safe Configuration Data, these values are securely transmitted and validated using cryptographic methods, including TAN-based protection mechanisms. This ensures that mismatches caused by unauthorized modifications are prevented.</p> <p>Furthermore, the Confirmation phase facilitates the provision of OperationTokens.</p>
OperationTokens	<p>The OperationTokens are required for each Configuration Data to transition into an operational state. These tokens act as proof that the Configuration Data has been correctly installed, validated, and authorized to run. Only after the token has been securely stored on the corresponding Trackside consuming system can the Configuration Data be executed during runtime. This mechanism guarantees that no unverified or tampered Configuration Data can be operational.</p>
DistributionJob	<p>A DistributionJob defines the time and conditions when one or more CCS deployments with the same homologation receive and activate their configuration update. The DistributionJob is defined within a “distribution-job.json” document which references the Top-Level Configuration Data that has recursive dependency tree.</p> <p>The schema.json for the DistributionJob document is defined in[4]</p>

1.2 SUMMARY OF UPDATES COMPARED TO D27.1

D27.1 is a prerequisite for D27.3, ergo this section summarizes the major updates and changes made since the release of previous deliverable D27.1. It highlights refinements in scope, system capabilities, and design decisions based on recent insights and stakeholder feedback.

1.2.1 General changes

- Data and Map Data from [5] have been replaced with Configuration Data. For definitions, see chapter 1.1
- SysC1: Generate Configuration Data has been removed. See 1.2.3 for rationale
- SysC2: Provide Configuration Data to Trackside and SysC3: Perform Activation of Configuration Data in Trackside have been combined into SysC1: Activate Configuration Data. This is done to keep the primary capabilities for configuration process consistent to System Pillar specifications.
- SysC4: Deploy Domain Data to On-Board has been renamed as SysC2: Provide Configuration Data to On-Board.
- IM Data System has been renamed to Data Preparation. This is due to several outstanding open points raised in the framework of System Pillar TCCS Domain with regards to division of responsibilities between IM and Supplier to prepare Configuration Data.
- DR Data Manager has been renamed to Configuration Manager. This is done to keep the primary actors responsible for configuration process consistent to System Pillar specifications.
- Functions have not been defined anymore, and focus is kept to requirements for the relevant logical components identified in chapter 2.1
- Automatic Train Protection On-Board is no more an actor to Digital Register On-Board. This has been removed due to the rationale and decisions made in the R2DATO Digital Register Working Group.

1.2.2 General remarks

- The SMI interfaces to trackside systems defined in this requirement specification only cater for infrastructure related configuration data. Other kinds of configuration data like firmware, vehicle related data are not in scope of this requirement specification.
- Mapping between Interfaces and System Capabilities is provided in respective chapters defining the trackside and On-Board topics. This is done to show the link between capabilities
- Scenario descriptions given in chapters 4.1.1 and 4.1.2 are to be considered as requirements, in addition to individual requirements defined in chapters 4.1.1.7, 4.1.1.8, 4.1.1.9, and 4.1.2.3.
- References [1] and [2] are provided as Annex 1 and Annex 2 to this document and are to be considered while reading this requirements specification.
- The scenarios and data packets defined in [1] and [2] use the terminology Repository On-Board. Hereby, this term is replaced with Digital Register On-Board for use in this document. This is done to be compliant with the R2DATO Grant Agreement and Architecture defined in D27.1.

1.2.3 Updates related to Scope of DR

The content here corresponds to the scope defined in chapter 3.1 in [5]

Table 2: Updates related to Scope of DR

Scope Item	Update	Rationale
Import received Engineering Input Data that are generated and provided by external engineering processes into a standardised data model format as Engineering Data. Note: This standardised data model format is to be aligned with SD1 System Pillar	The responsibility for this function lies outside the DR-TS scope and is formally specified as an exported constraint managed by the Data Preparation	The actual preparation process and validation of engineering data, incl. the selection of a suitable toolchain, remains within the scope of respective IMs, RUs and suppliers. Rationale for the updated DR scope:
Aggregate and Validate Engineering Data (including Parameter Data) based on rules checks/acquisition data.	The responsibility for this function lies outside the DR-TS scope and is formally specified as an exported constraint managed by the Data Preparation	1. Keeping the focus of the DR system under consideration on the required <i>results</i> of the data preparation process, and not on the processes related to data preparation.
Compile validated Engineering Data into consuming system specific Configuration Data using generic transformation rules.	The responsibility for this function lies outside the DR-TS scope and is formally specified as an exported constraint managed by the Data Preparation	2. Avoiding the complexity of handling different, stakeholder-specific input data sources and emerging complexities during data aggregation from different sources. 3. Be open to different migration paths from today's engineering process situations, which are very specific (e.g. regarding the state of digitisation/optimisation, work split with supplier, and working modes) to an improved process resulting in standardised outputs.

1.2.4 Updates related to System Capabilities and System Functions

The concrete updates and deltas compared to D27.1 are summarised in this section:

Table 3: Updates related to System Capabilities

System Capability	Function Title	Update	Rationale
SysC1: Generate Configuration Data	SysF1: Initiate Import	The responsibility for this capability and corresponding functions lies outside the DR-TS scope and is formally specified as an exported constraint managed by the Data Preparation in chapter 5	See Rationale in Table 2: Updates related to Scope of DR
	SysF2: Import and Enrich Engineering Data		
	SysF3: Validate Engineering Data		
	SysF4: Handle validation issues		
	SysF5: Compile Configuration Data		
	SysF6: Handle compilation issues		
	SysF7: Initiate publishing		
SysC2: Provide Configuration Data to Trackside	SysF8: Maintain Configuration Data	Configuration Data distribution and activation for the trackside system will be executed in accordance with the System Pillar's standard configuration update process, using the SMI interface. As defined by the System Pillar, the process also includes an additional deactivation step prior to activation, ensuring the previous configuration is safely and	The distribution of software Configuration Data and software updates via the SMI interface shares key characteristics with the distribution of Configuration Data. Both require a reliable provisioning process across multiple consuming systems, ensuring data integrity, version compatibility, and synchronized activation to deploy new versions effectively. Therefore, the same principles applied to Configuration Data distribution shall be leveraged for software and Configuration Data. Reusing the System Pillar process and standardized interface enhances harmonization across the entire CCS system.
	SysF9: Transact Configuration Data		
	SysF10: Transact Configuration Data version acknowledgement		
SysC3: Perform Activation of Configuration Data in Trackside	SysF11: Provide Configuration Data authorisation		
	SysF12: Transact Configuration Data activation request		
	SysF13: Authorise activation of Configuration Data		

		systematically cleared before the new configuration is applied. Ergo, it is now redefined as part of the chapter 4.1.1. The definitions part of D27.1 [5] are obsolete.	
SysC4: Deploy Configuration Data to On-Board	SysF14 Perform Map Data relevancy check	Configuration Data for on-board system will be executed in accordance with the Track Train Communication Principles [1] defined by the R2DATO Working Group. Ergo, it is redefined as part of chapter 4.1.2. The definitions part of D27.1 [5] are obsolete.	The goal R2DATO Digital Register Working Group was to define harmonised needs on the Digital Register for the On-Board applications. Ergo, the old concepts and specifications from previous years are deemed obsolete.
	SysF15 Receive On-Board Map Reference Data trigger		
	SysF16 Maintain Configuration Data		
	SysF17 Validate On-Board Map Data		
	SysF18 Activate Map Data		
	SysF19 Deactivate Map Data		

1.2.5 Updates related to actor functions

All actor functions namely from ActF1. to ActF23. are obsolete due to updates coming from System Pillar specifications as well as R2DATO Digital Register Working Group.

1.2.6 Updates related to exported constraints

All exported constraints namely from ExpCon1. to ExpCon9. are obsolete due to updates coming from System Pillar specifications as well as R2DATO Digital Register Working Group.

2 LOGICAL ARCHITECTURE

This chapter presents the logical architecture of the Digital Register, detailing its modular structure and the interaction between trackside and On-Board components.

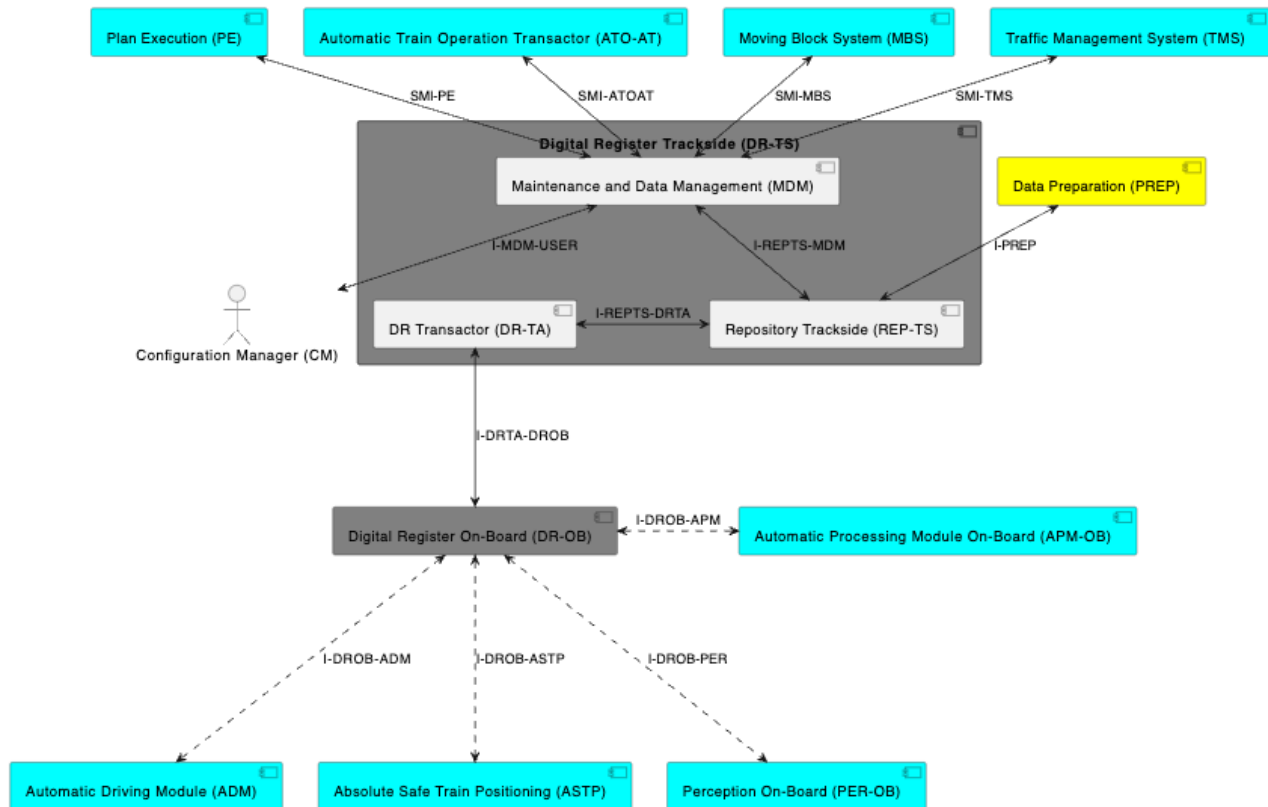


Figure 1: Logical Architecture of DR

According to DR's functional responsibilities, on the trackside, the Digital Register Trackside is broken down into Maintenance and Data Management (MDM), DR Transactor (DR-TA), and Repository Trackside (REP-TS). On the On-Board side, the Digital Register On-Board (DR-OB) remains a black box and provides interfaces to the consuming applications like ASTP, APM-OB, ADM, and PER-OB.

The new logical components implicate new logical interfaces between the different components of DR-TS. These corresponding components and interfaces are described in the next chapters.

The interfaces marked in dotted lines are not part of the specification but have been defined here to depict the complete picture considering the needs of Digital Register on the On-Board side.

2.1 LOGICAL COMPONENT DEFINITION

2.1.1 Maintenance and Data Management (MDM)

The Maintenance and Data Management is the component responsible for handling the distribution and activation of Configuration Data to the trackside consuming systems. This component includes the service function configuration (SCF), which is responsible for providing Configuration Data on trackside consuming systems consistently according to an explicit dependency tree. The configuration process runs through different steps, such as Preloading, Deactivation of incompatible (safe) Configuration Data, Activation (Installation) and Commit. This service function includes a small

safe function called SafeConfigurationAuthority (SCA) that ensures the safe Deactivation of incompatible Configuration Data.

The distribution process is triggered using a 'Publish Trigger' by the Configuration Manager.

2.1.2 Repository Trackside (REP-TS)

The Repository trackside component is a repository to store and maintain the Configuration Data. Repositories could access other repositories to add missing items recursively. For this reason, different repositories can exist on the Configuration Manager level (IM/RU) storing the Infrastructure Data or Vehicle Data, and it could also be on the level of the integrator/supplier storing Software Configuration Data.

2.1.3 Digital Register Transactor (DR-TA)

A Transactor is a trackside "service" hosting the access to one or several trackside functions delivering data to be transmitted to On-Board consumers. Digital Register Transactor publishes the Segment Profiles and Map Area to DR-OB.

2.1.4 Digital Register On-Board (DR-OB)

The On-Board logical component responsible to request Segment Profiles and Map Area from DR-TA for the On-Board applications like Perception, Automatic Driving or ASTP.

2.2 LOGICAL ACTOR DEFINITION

2.2.1 Moving Block System

Refer to chapter 4.3.1 in [5]

2.2.2 Plan Execution System

Refer to chapter 4.3.1 in [5]

2.2.3 Automatic Train Operation Transactor

Refer to chapter 4.3.1 in [5]

2.2.4 Traffic Management System

Refer to chapter 4.3.1 in [5]

2.2.5 Absolute Safe Train Positioning System

Refer to chapter 4.3.1 in [5]

2.2.6 Perception On-Board

Refer to chapter 4.3.1 in [5]

2.2.7 Automatic Driving Module

Refer to chapter 4.3.1 in [5]

2.2.8 Automatic Processing Module On-Board

Refer to chapter 4.3.1 in [5]

2.2.9 Configuration Manager

An institution operating/using a system through its life-cycle, in this context it is usually an Infrastructure Manager or Train Operator/Owner.

2.2.10 Data Preparation

The Data Preparation is responsible for preparing and validating the Configuration Data required by the consuming systems. This system encompasses the contributions of two types of stakeholders:

Infrastructure Manager: Provides infrastructure-related Configuration Data, such as track topology.

Supplier: Delivers application-specific Configuration Data, such as hardware configurations or software parameters for onboard or wayside components.

3 INTERFACES

This chapter defines the logical interfaces between the Digital Register Trackside components and external systems.

Table 4: Logical Interfaces Overview

Id No.	Name of Interface	Interface Endpoints
1.	I-DRTA-DROB	DR-TA and DR-OB
2.	I-MDM-CM	Configuration Manager and MDM
3.	I-PREP	Data Preparation and REP-TS
4.	I-REPTS-DRTA	REP-TS and DR-TA
5.	I-REPTS-MDM	REP-TS and MDM
6.	SMI-MBS	MDM and MBS
7.	SMI-TMS	MDM and TMS
8.	SMI-PE	MDM and PE
9.	SMI-ATOAT	MDM and AT

4 PROCESS DESCRIPTION

This section provides an overview of the process involved in distributing the (infrastructure-related) Configuration Data to trackside systems and on-board systems.

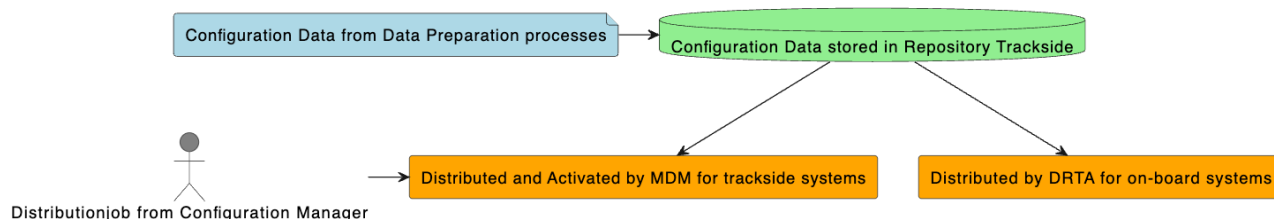


Figure 2: Overview configuration process (for infrastructure-related Configuration Data)

The configuration starts with the data preparation phase, including product-specific amendments and final compilation of Configuration Data. The compiled data resulting from the data preparation phase is stored in corresponding repositories represented in the subsystem Configuration Repository and made available for the Configuration Management Process to distribute and activate the data in the trackside consuming systems. For the on-board consuming systems, the process is well defined in [1].

Based on planned activation times from TMS or RU inputs, the user (Configuration Manager level) generates a distribution job. The distribution job contains all the needed attributes, including “When” and “What” to update and the dependency tree.

The distribution job contains these main files:

File	Description
configuration.json	The primary configuration descriptor. Contains metadata, dependencies, and optional payload references.
pConfigurationSafe.jwt	The signed and encrypted version of the configurationSafe.json. Used only for Safe (safety related and SIL1-4) configurations.
<payload filename>	The transparent or non-transparent payload file.

The Configuration Manager starts the distribution job on the level of MDM, leading to the preloading of configuration over configuration interface (SMI) in consuming systems. The top-level safe configuration activation is then started after receiving the command from the Configuration Manager, which starts the phase of deactivation of incompatible configurations and safe configuration, and is switched to safe state for configuration update (e.g. complete shut down / taking out of operation): this step is planned to be triggered by the Safe Configuration Authority of the MDM. Finally, the activation is started, and a confirmation is needed to ensure the successful activation of the updated configuration.

In the context of the DR system, the dependency hierarchy is defined as follows: TMS is at the top level, followed by PE, and then MBS. This means that during deactivation, the process starts with TMS, continues with PE, and ends with MBS. Activation follows the opposite sequence, starting with MBS and ending with TMS.

In addition, for this use case, the payload data includes infrastructure-related Configuration Data required for trackside and on-board system. The configuration.json and pConfigurationSafe.jwt shall describe this dependency.

4.1 SCENARIOS

This section presents scenarios that illustrate how the Digital Register supports Configuration Data distribution and activation. It includes real-world use cases and defines system behaviour under different conditions.

4.1.1 Data Distribution, loading, and activation to Trackside systems

This chapter covers the scenarios and requirements for the system capability SysC1: Activate Configuration Data

The scenarios presented in this chapter are based on the generic configuration update process defined by the System Pillar Task 2 – Transversal CCS Domain, as described in the document TCCS CONEMP Configuration and Diagnostics [4]. Additionally, scenarios involving the SMI interface are derived from the EULYNX SMI interface specifications outlined in documents [6] and [7], as published in Baseline 4 Release 3. These scenarios also consider the planned updates by the System Pillar for SMI v3, which aim to fully align with the TCCS Configuration Logical Concept [5] as part of the migration path.

While the System Pillar defines a generic and architecture agnostic configuration update process, this document specifies how the generic process is concretely applied to the DR system. It describes the specific use case of distributing and activating both infrastructure data and vehicle data to Trackside Consuming Systems within the DR context.

4.1.1.1 Start distribution of Configuration Data

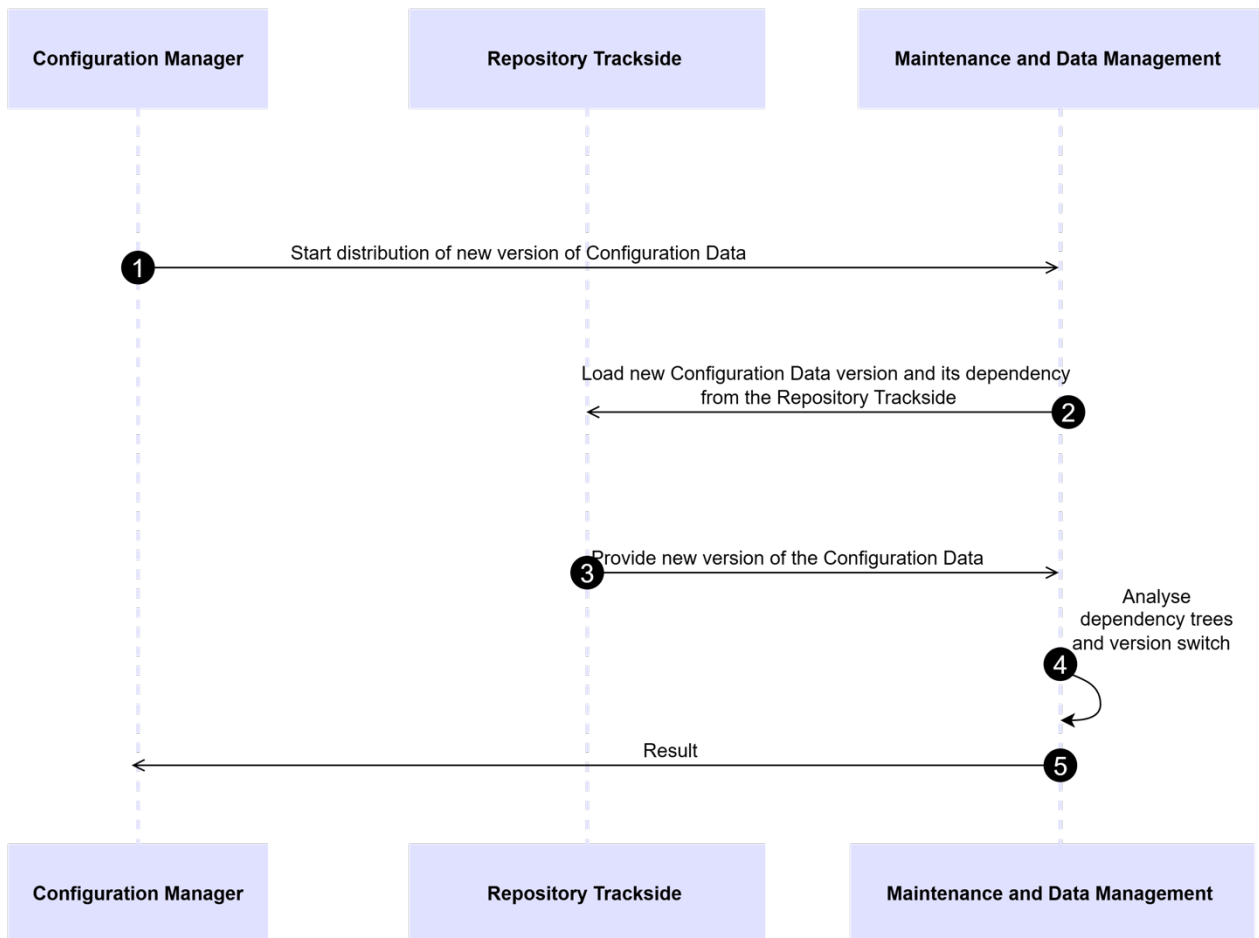


Figure 3 Scenario: Start distribution of Configuration Data

Table 5 Scenario description: Start distribution of Configuration Data

Scenario Condition	Scenario Description
Nominal case	<p>Precondition: The Configuration Data must be prepared and available in the Trackside Repository.</p> <p>The corresponding Distribution Job must be created and accessible within the Trackside Repository.</p> <p>A shared secret in place for safe Consuming Trackside system must be provisioned in MDM.</p> <p>A secret for top level Configuration Data activation must be provisioned in MDM and data preparation.</p> <p>A shared secret and unique ID for Consuming Trackside system must be provisioned in Consuming Trackside System.</p>

1. Configuration Manager initiates the start of distribution for the new Configuration Data version.
2. MDM loads the version of Configuration Data according to the distribution job from the Repository Trackage.
3. Repository Trackage provides the version of the Configuration Data according to the version needed.
4. MDM analyses the dependency trees and version switch according to the distribution job.
5. MDM provide feedback to Configuration Manager

Postcondition: MDM has the correct version of Configuration Data and ready to start preloading in the Trackage Consuming Systems.

4.1.1.2 Loading of Configuration Data

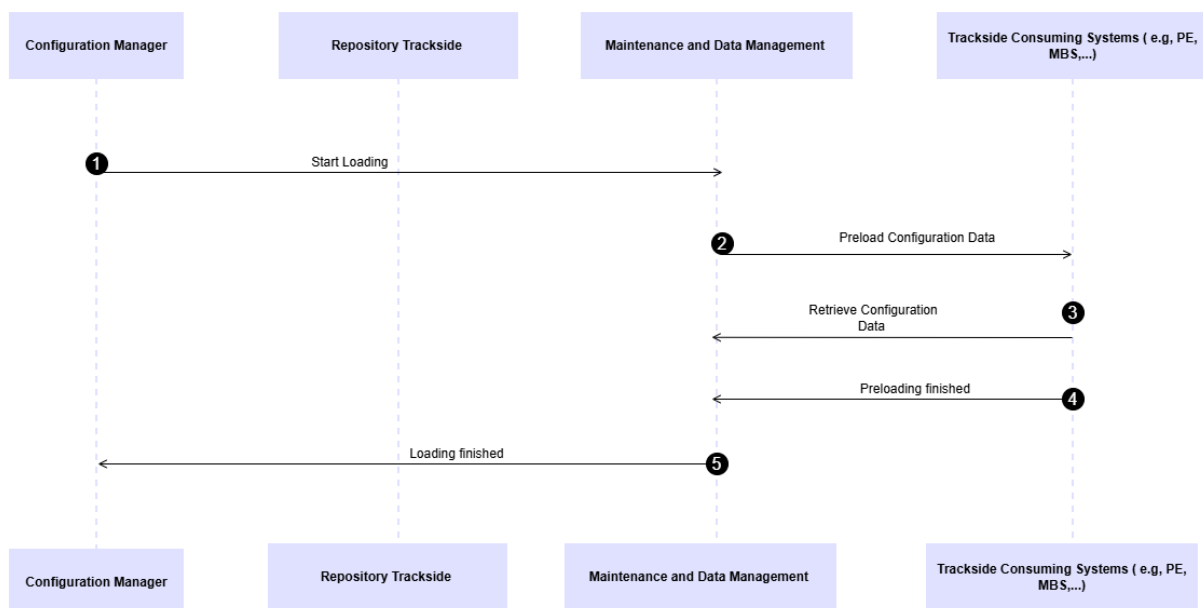


Figure 4 Scenario: Loading of Configuration Data

Table 6 Scenario description: Loading of Configuration Data

Scenario Condition	Scenario Description
Nominal case	<p>Precondition: MDM has the correct version of Configuration Data and ready to start loading to the Trackage Consuming Systems.</p> <ol style="list-style-type: none"> 1. Configuration Manager initiates the Loading of new Configuration Data. 2. MDM requests starting the Loading of new Configuration Data.

3. Each Trackside Consuming System loads the new Configuration Data version
4. Each Trackside Consuming sends to MDM “Loading finished” once the loading of new Configuration Data is finished
5. MDM provide feedback to Configuration Manager

Postcondition: Trackside consuming system has preloaded the new version of Configuration Data and ready for Activation

4.1.1.3 Activation of Top-Level Configuration Data

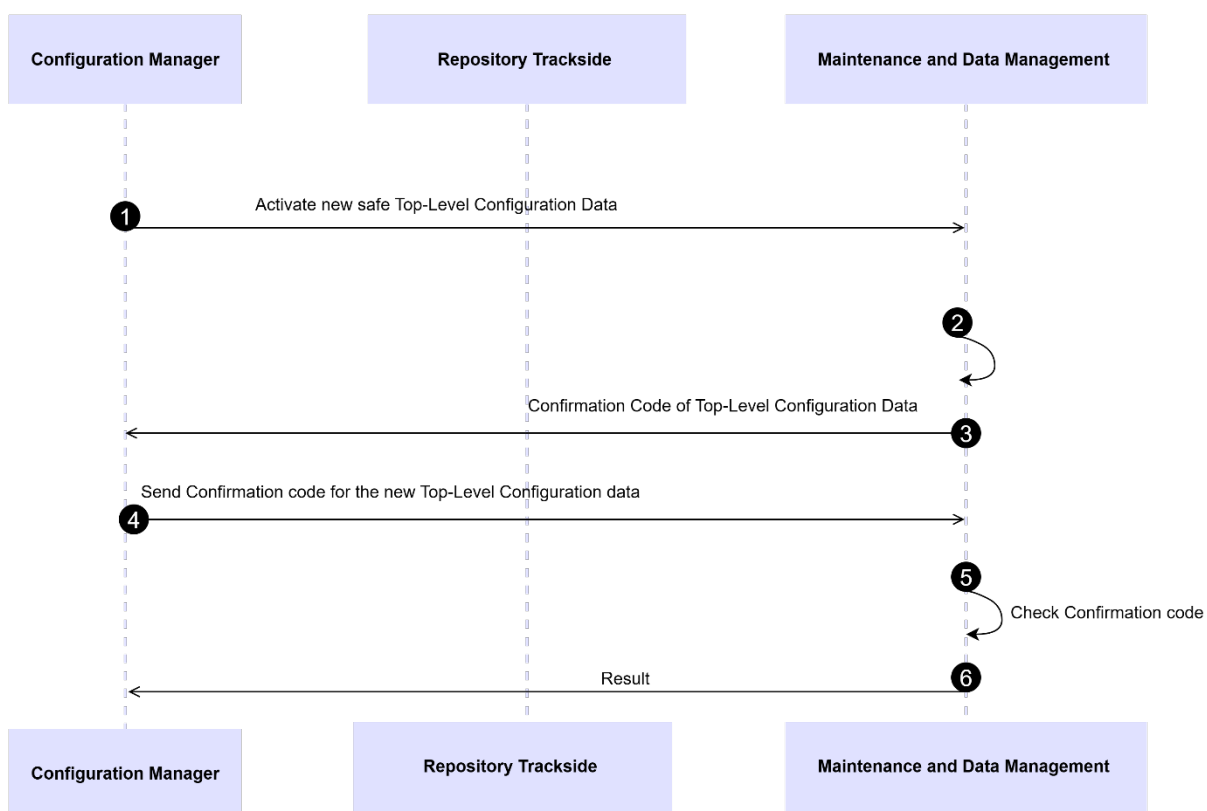


Figure 5 Scenario: Activation of Top-Level Configuration Data

Table 7 Scenario description: Activation of Top-Level Configuration Data

Scenario Condition	Scenario Description
Nominal case	<p>Precondition: The loading of the new Configuration Data has been successfully completed, and the data is ready for activation.</p> <ol style="list-style-type: none"> 1. Configuration Manager initiates the process by sending a request to activate the new safe Top-Level Configuration Data corresponding to a preloaded distribution-job to the MDM.

2. MDM validates the configuration integrity by checking the aggregated configuration hashes. Following successful validation, it generates a ConfirmationCode.
3. ConfirmationCode is shown to the Configuration Manager, who must manually verify it. This fingerprint ensures that the activation process can only proceed with Configuration Manager awareness and consent.
4. If Configuration Manager decides to proceed he/she sends the ConfirmationCode for the new Top-Level Configuration Data.
5. MDM validates the ConfirmationCode and analyzes the configuration dependency tree. It determines which Trackside system require a version update and identifies the necessary post-installation hashes. To ensure consistency, it temporarily blocks the issuance of OperationTokens until all relevant components of the new Configuration Data are fully activated.
6. MDM reports the status to the Configuration Manager.

Postcondition: The new safe Top-Level Configuration Data is activated, with all dependencies updated and verified.

4.1.1.4 Deactivation

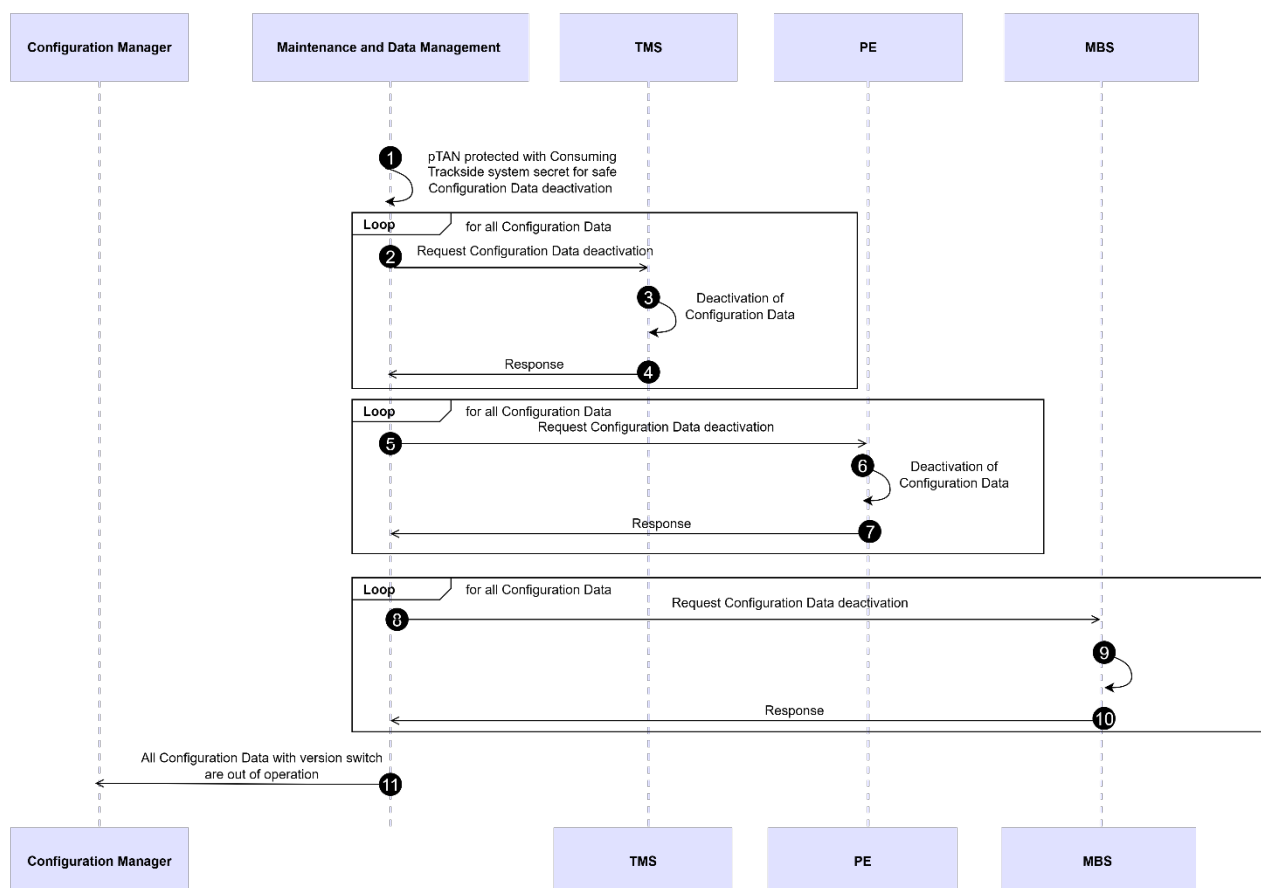


Figure 6 Scenario: Deactivation of Configuration Data with version switch

Table 8 Scenario description: Deactivation of Configuration Data with version switch

Scenario Condition	Scenario Description
Nominal case	<p>Precondition: The new safe Top-Level Configuration has been confirmed for activation, and all affected Configuration Data requiring version switching has been identified. The system has verified which subsystems (TMS, PE, MBS) contain incompatible configurations that must be deactivated.</p>
Loop: For each Configuration Data in TMS	<ol style="list-style-type: none"> MDM initiates the deactivation phase by generating protected deactivation tokens (pTANs) for all identified Configuration Data. The pTANs are generated using shared secrets to ensure secure and authorized deactivation requests. MDM sends a deactivation request including the corresponding pTAN to TMS. TMS validates the request, processes the deactivation internally. TMS confirms the result back to MDM.
Loop: For each Configuration Data in PE	<ol style="list-style-type: none"> MDM sends a deactivation request including the corresponding pTAN to PE. PE validates the request, processes the deactivation internally. PE confirms the result back to MDM.
Loop: For each Configuration Data in MBS	<ol style="list-style-type: none"> MDM sends a deactivation request including the corresponding pTAN to MBS. MBS validates the request, processes the deactivation internally. MBS confirms the result back to MDM.
	<p>11. Once all targeted Configuration Data across all subsystems have been successfully deactivated, MDM confirms that all Configuration Data with version switch are now out of operation.</p> <p>Postcondition: All Configuration Data with incompatible configuration versions have been deactivated. The system is now ready to proceed with the version switch and activation of the new Configuration Data version.</p>

4.1.1.5 Activation

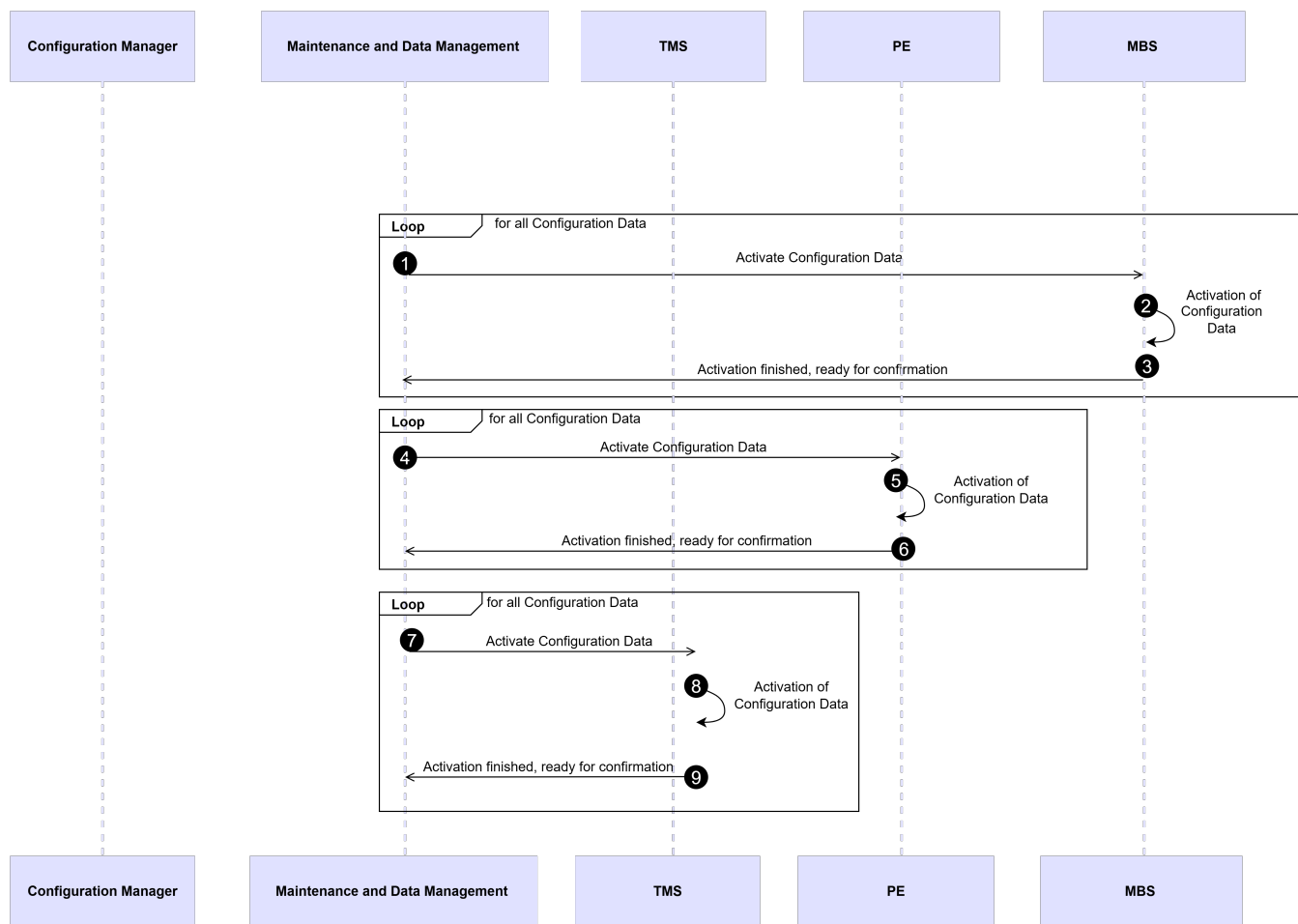


Figure 7 Scenario: Activation of Configuration Data

Table 9 Scenario description: Activation of Configuration Data

Scenario Condition	Scenario Description
Nominal case	Precondition: All Configuration Data with a version switch have been deactivated
Loop: For each Configuration Data in MBS	<ol style="list-style-type: none"> MDM sends request to start the installation of the corresponding Configuration Data to MBS MBS processes the activation request internally, it installs the relevant Configuration Data, generates a unique TAN to protect the installation hash, calculates the post-installation hash and sets the activation and confirmation states accordingly. MBS sends activation (activation finished) and confirmation (ready for confirmation) states to MDM.

**Loop: For each
Configuration Data
in PE**

4. MDM send request to start the installation of the corresponding Configuration Data to PE
5. PE processes the activation request internally, it installs the relevant Configuration Data, generates a unique TAN to protect the installation hash, calculates the post-installation hash and sets the activation and confirmation states accordingly.
6. PE sends activation (activation finished) and confirmation (ready for confirmation) states to MDM.

**Loop: For each
Configuration Data
in TMS**

7. MDM send request to start the installation of the corresponding Configuration Data to TMS
8. TMS processes the activation request internally, it installs the relevant Configuration Data, generates a unique TAN to protect the installation hash, calculates the post-installation hash and sets the activation and confirmation states accordingly.
9. TMS sends activation (activation finished) and confirmation (ready for confirmation) states to MDM.

Postcondition: All Configuration Data targeted for activation have been installed successfully. Trackside consuming systems have computed and securely protected post-installation hashes for verification. All activated Configuration Data are now in a “Ready for Confirmation” state.

4.1.1.6 Confirmation

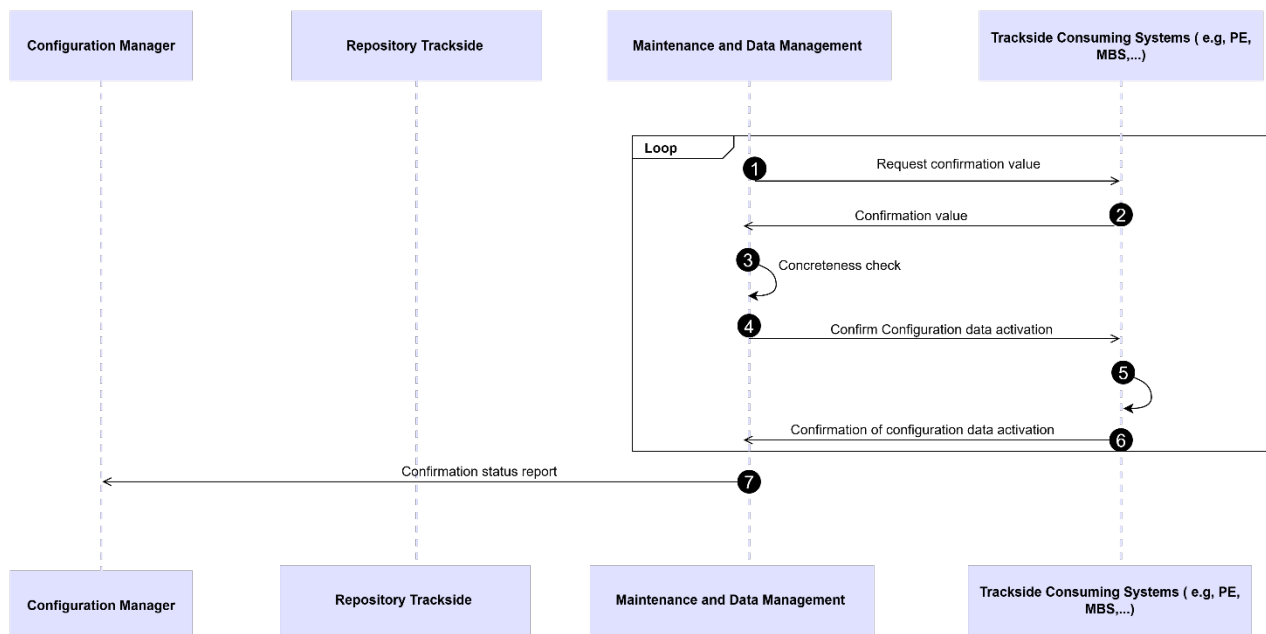


Figure 8 Scenario: Confirmation

Table 10 Scenario description: Confirmation

Scenario Condition	Scenario Description
Nominal case	Precondition: All Configuration Data have been successfully activated
Loop: For each activated Configuration Data	<ol style="list-style-type: none"> MDM requests confirmation values (e.g., protected SafetyHash, protected TAN, or NonSafetyHash) for each activated Configuration Data from the corresponding Trackside Consuming System. Each Trackside Consuming System responds by providing the confirmation value (e.g., protected SafetyHash, protected TAN, or NonSafetyHash) MDM validates the confirmation values against the pre-calculated expected values. MDM, upon successful validation, sends the protected OperationToken to the Trackside Consuming System to confirm the activation and the start of operational state of the Configuration Data.

5. Trackside Consuming System internally verifies the token, sets its confirmation state, and stores the token for future operation checks.
6. Trackside Consuming System send the confirmation status to MDM.
7. MDM reports the overall confirmation result to the Configuration Manager, completing the process.

Postcondition: All confirmed Configuration Data have securely verified post-installation integrity and now hold a valid OperationToken. They are ready for operational use.

4.1.1.7 Requirements for MDM

This Chapter covers the functional requirements for the MDM.

4.1.1.7.1 Functional Requirements

- 4.1.1.7.1.1 The MDM shall implement functionality to fulfill the distribution requirements as described in the sequences in chapter 4.1.1.
- 4.1.1.7.1.2 The MDM shall map each Trackside Consuming System to its specific Configuration Data version.
- 4.1.1.7.1.3 the MDM shall analyse and compare different versions of Configuration Data dependency trees to determine which Configuration Data require updates during a Top-Level Configuration Data version change.
- 4.1.1.7.1.1 The MDM shall temporarily suspend the distribution of operation tokens during the safe Top-Level Configuration Data phase, until all dependencies associated with the new configuration are fully activated.
- 4.1.1.7.1.2 The MDM shall activate and confirm all Configuration Data undergoing a version change.
- 4.1.1.7.1.3 The MDM shall implement the functionalities for Loading, Activation of Top-Level Configuration, Deactivation, Activation and Confirmation for Configuration Data according to the scenarios in chapter 4.1.1.
- 4.1.1.7.1.4 The MDM shall use a Create-only operation to insert newly retrieved configurations into the local repository. It must never modify or delete existing versions but maintain versioned instances for rollback and traceability.
- 4.1.1.7.1.5 The MDM shall support central orchestration of configuration updates. Local preloading may be supported, but all distribution processes must be controlled and initiated from a centralized MDM instance.
- 4.1.1.7.1.6 The MDM shall recursively retrieve all referenced Configuration Data configuration.json documents and store them in the local configuration repository to ensure readiness for processing and distribution.

- 4.1.1.7.1.7 The MDM shall support recursive dependency resolution for all dependencies listed in the configuration.json of the Configuration Data. The MDM must resolve dependencies until no further nested dependencies are referenced.
- 4.1.1.7.1.8 The MDM shall store the dependencies in the local configuration repository (Create operation) once these dependencies are identified and retrieved (READ operation).
- 4.1.1.7.1.9 If a dependency is declared with "configurationSafetyRelevance": "SAFE", the MDM must check that the corresponding configurationSafe.json exists and that its configurationSafeHash is valid and linked in the parent configuration. Missing or invalid safe dependencies shall block the inclusion of such a Configuration Data into the runtime deployment plan.
- 4.1.1.7.1.10 The MDM shall verify the authenticity and integrity of the Configuration Data.
- 4.1.1.7.1.11 The MDM shall verify the authenticity and integrity of the distribution job.
- 4.1.1.7.1.12 The MDM shall check before initiating a distribution job, for each Configuration Data used in the Dependency Tree its presence in a Blacklist. If a Configuration Data corresponds to one referred in said Blacklist, the MDM shall mark this Configuration Data as [Blacklisted] and not authorise its preloading. All subdependencies of this Configuration Data shall be removed from the distribution job.
- 4.1.1.7.1.13 The MDM shall begin from the top of the dependency tree and works its way down for the Deactivation. The higher-level Configuration Data, which depends on other Configuration Data, is taken out of operation first. Then, moving down the tree, dependent Configuration Data are progressively deactivated.
- 4.1.1.7.1.14 The MDM shall enforce that updates are activated bottom-up, following the dependency order. A higher-level Configuration Data must only be activated when all its dependencies have been successfully activated.
- 4.1.1.7.1.15 The confirmation from the MDM to the Trackside Consuming System can hold a lease policy for each of the Configuration Data. The lease policy might hold conditions based on operational status like: onReboot, onReconnect, onMaintenance, inDepot (geofencing).
- 4.1.1.7.1.16 The MDM shall handle a fallback situation identically to a new distribution job referencing an unchanged older version of the Configuration Data.

4.1.1.8 Requirement for Repository Trackside

This Chapter covers the functional requirements for the Repository trackside.

4.1.1.8.1 Functional Requirements

- 4.1.1.8.1.1 The Repository Trackside shall provide a versioned, immutable, and structured storage and access system for all artifacts related to Configuration Data.
- 4.1.1.8.1.2 The Repository Trackside shall support authorized UPLOAD and GET operations over HTTPS via a REST API. This interface is used by the Data Preparation processes for uploading content, and by the MDM for retrieving Configuration Data and their dependencies during configuration orchestration.

4.1.1.8.1.3 The Repository Trackside shall store each Configuration Data version in its own uniquely addressable directory or repository path based on its identifiers

4.1.1.8.1.4 The Repository Trackside shall contain the following files for each configuration version:

1. “configuration.json”: The primary configuration descriptor. Contains metadata, dependencies, and optional payload references. Used by the MDM to orchestrate distribution trees
2. “pConfigurationSafe.jwt”: The signed and encrypted version of the configurationSafe.json. Used only for SAFE configurations and consumed exclusively by the Safe Configuration Authority (SCA) and must **not** be sent to the Consuming Trackside Systems.
3. “payload filename”: optional and it is the transparent or non-transparent payload file (e.g., firmware, parameter set). The filename is explicitly specified in configuration.json → configurationFile → configurationFileName.

4.1.1.8.1.5 The Repository Trackside shall support GET operations via HTTPS REST endpoints. A client (e.g., the MDM) must be able to retrieve all artifacts (configuration.json, configurationSafe.jwt, payload, signature) associated with a Configuration Data version

4.1.1.9 Requirements for Configuration Manager

4.1.1.9.1.1 The Configuration Manager shall select the Top-Level Configuration Data from the repository trackside and define which systems shall preload and activate a Configuration Data based on the dependency tree provided for a distribution job.

4.1.1.9.1.2 The Configuration Manager in case of rollback situation shall initiate the identical process: a new distribution-job must be created that references an unchanged older version of Configuration Data.

4.1.2 Data distribution to On-Board Systems

This chapter covers the scenarios and requirements for the system capabilities SysC2: Provide Configuration Data to On-Board

The scenarios part of this chapter has been defined by the R2DATO Digital Register Working Group as part of the document Track-Train Communication Session Management Principles [1]. Ergo, this chapter does not define all the scenarios rather only provides references to them. Since the document defined by the working group covers a wide scope of concepts and scenarios, all the scenarios have been listed below along with their corresponding references and relevant remarks related to scope of D27.3. The deliverable Track-Train Communication Session Management Principles will be provided as an Annex 1 to D27.3.

4.1.2.1 Basic Principles

See chapter 4 in [1].

4.1.2.2 Track Train Communication Use Cases

4.1.2.2.1 Powering On and Start of Operation

See chapter 5.2 in [1]. Defines the interactions between DR Transactor and Digital Register On-Board during the powering on and start of operation phase.

4.1.2.2.2 Operating in a fitted trackside area

See chapter 5.3 in [1]. Defines the interactions between DR Transactor and Digital Register On-Board when train is operating inside a fitted trackside area. An area is considered as fitted where there are DR Transactor can deliver Map Area and Segment Profiles for the Trains.

4.1.2.2.3 Crossing the border of a DR Transactor

See chapter 5.4 in [1]. Defines the interactions between DR Transactor and Digital Register On-Board when the train is moving between two DR Transactors and performing handovers.

4.1.2.2.4 Crossing a trackside area border of a Traffic Mgt Transactors

See chapter 5.5 in [1]. Defines the interactions between DR Transactor and Digital Register On-Board when the train is moving between two traffic management transactors and performing handover. This scenario is primarily relevant for ATO related activities, but relevant for D27.3 since it also covers the situation when DR Transactor provides Map Data to ASTP.

4.1.2.2.5 Coupling two Train Consists

See chapter 5.6 in [1]. Defines the interactions between DR Transactor and Digital Register On-Board when two train consists are being coupled. After being coupled the new train consists require for new Map Area and Segment Profiles.

4.1.2.2.6 Uncoupling a Train

See chapter 5.7 in [1]. Defines the interactions between DR Transactor and Digital Register On-Board when train consists are being uncoupled. After being uncoupled the new train consists require for new Map Area and Segment Profiles.

4.1.2.2.7 Change of cab (1 OB device in 1 traction Unit)

See chapter 5.8 in [1]. Defines the interactions between DR Transactor and Digital Register On-Board when cabs are being changed for change in direction of travel.

4.1.2.2.8 Change of cab (2 OB devices in 2 traction Units)

See chapter 5.9 in [1]. Defines the interactions between DR Transactor and Digital Register On-Board when cabs are being changed for change in direction of travel.

4.1.2.3 Requirements for DR-TA and DR-OB

Note: Since the interface I-DRTA-DROB is designed based on the concepts defined in Subset 126, the requirements defined in the next sections have been simply taken over from Subset 126 and modified for I-DRTA-DROB application.

4.1.2.3.1 Conditions for establishing a communication session

4.1.2.3.1.1 The DR-OB shall establish a communication session as soon as all the following conditions are fulfilled:

- a) The Cab is active;
- b) DR-OB has received:

- 1) A valid TRN;
 - 2) An ETCS identity;
 - 3) A valid train length;
- c) ETCS-OB is not in NL Mode or in SH Mode.

4.1.2.3.1.2 The DR-OB shall establish a communication session with an DR-TA identified by an DR-TA Contact Information.

4.1.2.3.1.3 The DR-TA Contact Information shall be unique and composed of NID_C and NID_DR_TA.

4.1.2.3.2 Establishing communication sessions

4.1.2.3.2.1 The DR-OB shall have a preconfigured DR-TA Contact Information corresponding to the DR-TA to be initially contacted.

4.1.2.3.2.2 The DR-OB shall send an DR_HSReq to the DR-TA corresponding to the preconfigured DR-TA Contact Information including the list of ATO system versions that it supports.

4.1.2.3.2.3 The DR-TA shall answer with an DR_HSAck or with an DR_HSRej.

4.1.2.3.2.4 If no feedback (DR_HSAck or DR_HSRej) has been received after DR-OB has sent an DR_HSReq to DR-TA and DR-OB repeats the DR_HSReq, this repetition shall be done 30 seconds later, at the earliest.

4.1.2.3.2.5 When DR-OB receives an DR_HSAck, the communication session shall be considered established.

4.1.2.3.2.6 The DR_HSAck sent by the DR-TA to DR-OB shall include:

- a) DR-TA Identifier;
- b) ATO system version that will be used by the DR-TA for the DR-OB/DR-TA communication;
- c) Timeout for DR-TA response;

4.1.2.3.2.7 **Note:** If there is no ATO major version supported by both DR-OB and DR-TA, the DR-TA will not send an DR_HSAck, but an DR_HSRej asserting that there is no compatible TT System versions.

4.1.2.3.2.8 **Note:** The values of the “Time-out for DR-TA response” is DR-TA application specific.

4.1.2.3.2.9 If DR-TA answers with an DR_HSRej, the DR_HSRej packet shall mention the following possible reasons for rejection:

- a) DR-OB and DR-TA versions are not compatible;
- b) Another DR-TA is in charge (DR-TA Contact Information provided);
- c) DR-TA in charge is unknown (DR-TA Contact Information not provided).

4.1.2.3.2.10 The DR-TA shall perform the version compatibility check only if it is in charge of the DR-OB.

- 4.1.2.3.2.11 If the DR_HSRej indicates that DR-OB and DR-TA are not compatible, DR-OB shall stop trying to open a communication session until the TRN has changed.
- 4.1.2.3.2.12 If the DR_HSRej indicates that DR-TA in charge is not known, DR-OB shall attempt to re-establish connection with the pre-configured DR-TA Contact Information.
- 4.1.2.3.2.13 If the DR_HSRej indicates that another DR-TA is in charge, the DR-TA Contact Information sent back in the DR_HSRej is corresponding to the DR-TA which is currently in charge of controlling the train. In this case, the DR-OB shall release the communication and shall establish a new communication session with the DR-TA Contact Information transmitted in the DR_HSRej packet and start a new handshake process.
- 4.1.2.3.2.14 **Note:** TRN is also considered as changed when ETCS-OB reports that the set of ETCS Operational Data has changed to “Valid”.

4.1.2.3.3 Hand-over between two adjacent DR-TA

- 4.1.2.3.3.1 When the train already has an open communication session with an DR-TA and it approaches the border of the area controlled by this DR-TA (called ‘handing-over DR-TA’), the DR-TA Contact Information of the adjacent DR-TA (called ‘accepting DR-TA’) is given by the last SP referred to by the current MAPArea.
- 4.1.2.3.3.2 The DR-OB shall recover the DR-TA Contact Information of the Accepting DR-TA from the SP referred to in the current MAPArea Packet.
- 4.1.2.3.3.3 The DR-OB shall establish a communication session with the Accepting DR-TA, by indicating that it is due to a handing over procedure.
- 4.1.2.3.3.4 If the DR-TA receives an DR_HSReq indicating that it is due to a handing over procedure, the DR-TA shall send an DR_HSAck unless the DR-OB and DR-TA versions are not compatible (see 4.1.2.3.2.9 a).
- 4.1.2.3.3.5 Having established communication with the accepting DR-TA on the approach to the border, DR-OB shall maintain communication with the handing over DR-TA until the rear end of the train has left the last SP referred to by the MAPArea which ends at the boundary between the two adjacent DR-TA.
- 4.1.2.3.3.6 Maximum two communication sessions for DR-OB implies that the area covered by one DR-TA shall be large enough that a previous hand over is complete before the next border is approached, i.e., large enough to cover the maximum train length plus the distance travelled while contacting the third DR-TA.
- 4.1.2.3.3.7 If the SP referred to by the current MAPArea does not contain any DR-TA Contact Information when the train is approaching the border of the area controlled by this DR-TA, this means that the adjacent area is not fitted for DR. In this case, the DR-OB will be unable to get MAPArea Packet, and the application On-Board have to operate without MAPArea.
- 4.1.2.3.3.8 For each border of an DR-TA the associated list of SPs in the MAPArea shall include at most one SP containing the DR-TA contact Information, such SP shall be the one which ends at the border and is called “last SP”.

4.1.2.3.3.9 The DR-OB shall terminate the communication session with the accepting DR-TA when the handing over DR-TA sends a JP update where the last SP contains deviating DR-TA Contact Information (deleted or modified).

4.1.2.3.4 **Maintaining communication sessions**

4.1.2.3.4.1 A communication session with an DR-TA shall be considered as lost when:

- a) The DR-OB sends a MAPAReq and no MAPArea is received before the end of the “Timeout for DR-TA response” mentioned in the DR_HSAck;
- b) The DR-OB sends a SPReq and no SP is received before the end of the “Timeout for DR-TA response” mentioned in the DR_HSAck.

4.1.2.3.4.2 If DR-OB loses a communication session, the DR-OB shall try to re-establish the communication session using the same DR-TA Contact Information that was used when establishing the session with the DR-TA.

4.1.2.3.5 **Communication session termination**

4.1.2.3.5.1 DR-OB shall terminate the communication session with a given DR-TA when one of the following conditions is fulfilled:

- a) DR-TA sends a termination session request;
- b) The rear of the train has left this last SP referred to by the MAPArea which ends at the boundary of the DR-TA;
- c) Active cab is closed;

4.1.2.3.5.2 When terminating a communication session, the DR-OB shall send a packet SESSTerm to the DR-TA.

4.1.2.3.5.3 **Note:** This SESSTerm packet has not to be acknowledged

4.1.2.3.6 **Packet Exchange requirements**

4.1.2.3.6.1 The I-DRTA-DROB interface shall be interoperable.

4.1.2.3.6.2 The transfer of data between DR-TA and DR-OB shall follow the Track-Train Communication Session Management Principles [1] and Digital Register Transactor and Digital Register On-Board FFFIS Application Layer specification [2].

4.1.2.3.6.3 The DR-TA shall generate the packets required for communication between DR-TA and DR-OB as defined in Digital Register Transactor and Digital Register On-Board FFFIS Application Layer [2].

4.1.2.3.6.4 Any packet received by the DR-TA or the DR-OB shall be ignored if it is considered as not valid.

4.1.2.3.6.5 A packet shall be considered as not valid if one of the following conditions is fulfilled:

- a) Its number is unknown;
- b) Its timestamp is older than the timestamp of a previously received packet with the same number;

- c) Its packet counter has not changed compared to the last previously received packet with the same number;
- d) Any variable uses spare values;
- e) The ETCS Identity provided by the DR-TA does not match the ETCS Identity provided by the ETCS-OB.

4.1.2.3.6.6 In addition to the previous clause, an acknowledgment packet shall be considered as not valid if the timestamp and the packet counter information is not equal to the corresponding values of the packet to be acknowledged.

4.1.2.3.6.7 The DR-TA shall answer to any MAPAReq received from DR-OB with which a Communication session is established.

4.1.2.3.6.8 The DR-OB shall send MAPAReq indicating the Train Position and required Map Area Radius.

4.1.2.3.6.9 The DR-TA shall determine the required MAPArea for the DR-OB according the Train Position AND the required Map Area Radius defined in MAPAReq Packet.

4.1.2.3.6.10 Any “valid” MAPArea sent by DR-TA shall include at least one SP.

4.1.2.3.6.11 When a MAP Area has already been sent to DR-OB, the DR-TA shall be able to update it if necessary by sending a MAPArea with the status “Update”.

4.1.2.3.6.12 When it receives a “MAPArea Update” from the DR-TA, the DR-OB shall discard any previously received Map information.

4.1.2.3.6.13 DR-OB behaviour after receiving a JP depending on the JP Status:

- a) A “Valid” MAPArea specifies that the data sent is up-to-date and corresponds to the latest MAPAReq:
 - 1) The DR-OB shall use the data received from DR-TA.
- b) An “Update” MAPArea specifies that the MAPArea has been updated by the DR-TA modifying or adding information to previously transmitted ones:
 - 1) The DR-OB shall use the data received from DR-TA;

4.1.2.3.6.14 **Note:** A single SPReq could include several SP_ID requests.

4.1.2.3.6.15 The reply of the DR-TA to an SPReq shall indicate the evaluation status of the request:

- a) Valid: the requested SP is included;
- b) Invalid: the requested SP is not available in the DR-TA.

4.1.2.3.6.16 When it receives an SP, the DR-OB shall compare the version number of the SP listed in the MAPArea and the version number of the received one.

4.1.3 Messages

4.1.3.1 SMI

The messages part of the interface SMI between the MDM and the Trackside Consuming System covering the establishment of connection and communication protocol OPC-UA information model are defined in the EULYNX documents [5] and [7] for the current version. In future, a migration to System Interface Description_TCCS-System Interface SMI (v3) as defined in [11] by System Pillar.

4.1.3.2 I-DRTA-DROB

The messages part of the interface I-DRTA-DROB are defined in the document Digital Register Transactor and Digital Register On-Board FFFIS Application Layer [2] as part of the Task 27.4 in Digital Register. The corresponding document will be provided as an Annex 2 to D27.3. Since the said document is defined for a widened scope, the messages and relevant remarks to them will be defined in this chapter.

4.1.3.2.1 Digital Register Handshake Request Packet (DR_DR_HSReq)

See chapter 2.3.1 in [2]. Defines the structure of Handshake request during setting up a communication session between DR Transactor and Digital Register On-Board .

4.1.3.2.2 Digital Register Handshake Acknowledgement Packet (DR_DR_HSAck)

See chapter 2.3.2 in [2]. Defines the structure of Handshake Acknowledgement during setting up a communication session between DR Transactor and Digital Register On-Board .

4.1.3.2.3 Digital Register Handshake Reject Packet (DR_DR_HSRej)

See chapter 2.3.3 in [2]. Defines the structure of Handshake Rejection during setting up a communication session between DR Transactor and Digital Register On-Board .

4.1.3.2.4 Map Area Request Packet (MAPAReq)

See chapter 2.3.4 in [2]. Defines the structure of the Map Area request exchanged between DR Transactor and Digital Register On-Board .

4.1.3.2.5 Map Area Packet (MAPArea)

See chapter 2.3.5 in [2]. Defines the structure of the Map Area exchanged between DR Transactor and Digital Register On-Board .

4.1.3.2.6 Segment Profile Request Packet (SPReq)

See chapter 2.3.6 in [2]. Defines the structure of the Segment Profile Request exchanged between DR Transactor and Digital Register On-Board .

4.1.3.2.7 Segment Profile GoA3/4 Packet Extension (SP)

See chapter 2.3.7 in [2]. Defines the structure of the Segment Profile exchanged between DR Transactor and Digital Register On-Board .

4.1.3.2.8 Digital Register Session Termination Request Packet (DR_SESSTermReq)

See chapter 2.3.8 in [2]. Defines the structure of the Session Termination Request to initiate terminate the communication session between DR Trasactor and Digital Register On-Board

4.1.3.2.9 Digital Register Session Termination Packet (DR_SESSTerm)

See chapter 2.3.9 in [2]. Defines the structure of the Session Termination to terminate the communication session between DR Trasactor and Digital Register On-Board

4.1.3.3 I-PREP

4.1.3.3.1 The I-PREP interface shall be defined as a REST API as defined in SPT2TS-130394 in [8].

4.1.3.3.2 The configuration.json shall follow the structure as defined in SPT2TS-130417 in [8]

4.1.3.3.3 The configurationSafe.json for any safe consuming system shall follow the strucutre as defined in SPT2TS-130432 in [8]

4.1.3.4 I-REPTS-MDM

4.1.3.4.1 The I-REPTS-MDM interface shall be defined as a REST API as defined in SPT2TS-130394 in [8].

4.1.3.4.2 The configuration.json shall follow the structure as defined in SPT2TS-130417 in [8]

4.1.3.4.3 The configurationSafe.json for any safe consuming system shall follow the strucutre as defined in SPT2TS-130432 in [8]

4.1.3.4.4 The distribution-job.json shall follow the strucutre as defined in SPT2TS-130443 in [8].

4.1.3.5 I-MDM-CM

4.1.3.5.1 I-MDM-CM interface shall provide functionalities to the Configuration Manager via local user interface as defined in Eu.MDM.99 chapter in [6].

4.1.3.6 I-REPTS-DRTA

4.1.3.6.1 The I-REPTS-DRTA interface shall be defined as a REST API as defined in SPT2TS-130394 in [8].

4.1.3.6.2 The configuration.json shall follow the structure as defined in SPT2TS-130417 in [8]

4.1.3.6.3 The configurationSafe.json for any safe consuming system shall follow the structure as defined in SPT2TS-130432 in [8]

5 EXPORTED CONSTRAINTS

This chapter specifies the constraints and responsibilities of actors interacting with the Digital Register.

5.1 DATA PREPARATION

5.1.1.1.1.1 The Data Preparation shall create consumer specific Configuration Data based on the structure defined in domain I-PREP by the CCS/TMS Data model [10].

5.1.1.1.1.2 The Data Preparation shall perform plausibility checks on Configuration Data sent to REP-TS

Note: This is done to ensure that the data is consistent and suitable for application in safety / non-safety critical environments. The stringency of rules associated with plausibility checks are based on the safety relevance of the data consumer.

5.1.1.1.1.3 The Data Preparation shall create a dependency tree for all the systems that require a change in Configuration Data version.

6 REFERENCES

This chapter lists all referenced documents, standards, and specifications cited throughout the deliverable.

- [1]. Track-Train Communication Session Management Principles, Revision 07
- [2]. Digital Register Transactor and Digital Register On-Board FFFIS Application Layer, Revision 02
- [3]. EULYNX 20250619 Interface definition and specification SMI Eu.Doc.76 v2.3 (1.A)
- [4]. EULYNX 20250620 Maintenance and data management specification Eu.Doc.18 v4.1 (1.A)
- [5]. EULYNX 20250619 Generic interface and subsystem requirements for SMI Eu.Doc.120 v1.2 (1.A)
- [6]. D27.1 Set of requirements on the Digital Register in R2DATO, Revision 05
- [7]. CCS/TMS Data Model, <https://github.com/StructLab/CCSTMS-Data-Model>
- [8]. System Interface Description_TCCS-System Interface REPO
- [9]. System Interface Description_TCCS-System Interface SMI (v3)