Ref. Ares(2026)7036683 - 15/07/2026

Rail to Digital automated up to autonomous train operation

D30.3 – ERTMS L3 trackside with moving block formalisation

(report accompanying video demonstrator D30.3)

Due date of deliverable: 28/02/2026
Actual submission date: 26/01/2026

Leader/Responsible of this Deliverable: TRV

Reviewed: Y

Document status		
Revision	Date	Description
01	03/09/2025	Created preliminary draft of the report
02	12/09/2025	Revised version of the preliminary report
03	13/01/2026	First draft of the video demonstrator report
04	23/01/2026	Finalised the video demonstrator report for TMT review
05	26/01/2026	Finalised the video demonstrator for TMT review

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

Start date: 01/12/2022 Duration: 39 months

Interne

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
Arne Borälv	Trafikverket	Author

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.

COVER PAGE

This document accompanies deliverable D30.3 “ERTMS L3 trackside with moving block formalisation” in the framework of WP30 of Flagship Project 2 (R2DATO).

D30.3 consists of a video-based narrative to present a software-based prototype demonstrator aimed at supporting **early, model-based validation of requirements** for evolving railway signalling system architectures. The purpose of the video is to explain the motivation, modelling approach, and overall structure of the software demonstrator, and to illustrate key ideas using selected visual examples. This document accompanies D30.3, by providing the slide deck supporting the video-based narrative.

D30.3 Demonstrator video link:
https://vimeo.com/1157632290/d6bc1ebdf6?share=copy&fl=sv&fe=ci

Overview

As ETCS trackside systems evolve over time—driven by harmonisation, replacement of legacy solutions, architectural changes, and the introduction of new technologies—there is a growing need for **objective, digital analysis** of requirements prior to construction. Experience from prior projects shows that requirement issues are often identified late, rely heavily on manual judgement, or are difficult to validate systematically using existing modelling approaches.

The core contribution of D30.3 is a **proposed level of abstraction** for reasoning about evolving railway signalling systems-of-systems. The approach builds on an object model–based methodology previously applied to fixed-block interlocking systems and generalises it to support more dynamic, configurable, and heterogeneous contexts. A central design objective is to balance **expressivity and verifiability**, retaining as much as possible of the validation benefits of the original methodology while extending its applicability.

At the foundation of the demonstrator is an **ontology-based modelling framework**. An upper ontology defines a disciplined universe of types covering architectural, design, and global concepts, while a lower ontology introduces domain-specific concepts for ETCS trackside. From this ontology, a generic object model is generated, providing a **common semantic foundation** for expressing requirements, behavioural models, test cases, and safety properties. Behavioural models are defined separately and operate on instantiated object model data through a controlled interface.

Domain specialisation includes modelling abstractions central to railway signalling analysis, such as train-centric representations, path-based movement constraints, forbidden paths expressing interlocking-like safety conditions, and explicit modelling of non-nominal situations. Interaction between subsystems is represented using a message-based abstraction described conceptually.

The demonstrator has been instantiated using a set of real infrastructure data examples provided by Trafikverket. Static configuration and infrastructure data, expressed in RDF, are translated into a graph-based representation using LCF and used to instantiate the object model. From the instantiated model, executable test-based validation and formal verification activities have been performed to assess **feasibility and soundness** of the modelling approach. Validation scope is intentionally limited and reflects the prototype nature of the demonstrator.

Taken together, D30.3 presents a coherent proposal for a pragmatic abstraction level that supports early, model-based requirements validation for evolving railway signalling systems, while maintaining a clear distinction between modelling abstractions, partial implementations, and future extensions.

Interne

1 SLIDE DECK

The remainder of this document provides the supporting slide deck used in the video narrative.

D30.3 Demonstrator – Modelling Techniques for Requirements Validation

Europe's Rail Project R2DATO – Software-based prototype demonstrator (Deliverable D30.3)

- Scope: ETCS Level 3 trackside concepts, including moving-block principles
- Focus
  - Early validation of requirements
  - For evolving railway signalling system architectures
- This video
  - Motivation and core modelling approach
  - Demonstrator architecture
  - Validation evidence, maturity, reuse

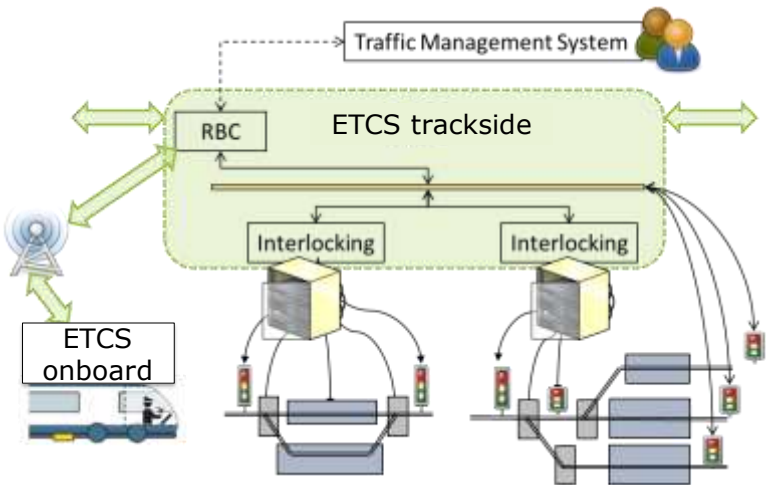
Author: Arne Borälv (R2DATO)

Project R2DATO grant agreement No: 101102001

1

Context of Demonstrator for ETCS Trackside

- System of Systems
  - Traffic management system
  - ETCS onboard
  - **ETCS trackside**
    - RBC
    - Interlocking
  - Trackside object controllers



2

Interne

System Evolution & Architectural Context

- Key drivers for long-term evolution of ETCS trackside include:
  - Harmonization
  - Replacement of legacy solutions
  - Evolving architectures and interfaces
  - Introduction of new technologies (e.g., Automatic Train Operation, moving block)
- This creates a clear need to
  - Assess and evaluate options objectively
    - Architectural options
    - Migration options
    - Technology options
  - Validate requirements for new technologies

3 R2DATO | GA 101102001

Digital Analysis Perspective

- Infrastructure managers need to:
  - Assess the impact of system migrations under consideration
  - Compare alternative architectural options
- This requires:
  - Realistic digital models of the signalling system and its environment
  - Useful analysis constructs to represent and analyse dynamic behaviour
  - Modern, digital processes for requirements validation

4 R2DATO | GA 101102001

Interne

Motivation: Problems Observed in Practice & Lessons Learnt

- Problems observed in practice
  - Vague concepts in requirements
  - Informal models not usable for V&V
  - Late discovery of requirement issues
  - Formal models too implementation-close

The Validation Gap

- There is a lack of methods that support:
  - Rigorous validation of requirements prior to production phases
  - Objective analysis that does not rely primarily on manual judgement
- What is missing is an approach that:
  - Scales to evolving architectures and system-of-systems contexts
  - Is precise enough to support executable testing and formal verification
  - Is abstract enough to avoid having to make premature implementation decisions

Core Idea of Demonstrator

- Define the conceptual system domain as an ontology
- Derive an object model as common foundation
- Express requirements and behaviour in terms of the object model

Ontology	Defines the conceptual system domain
Object model	Shared vocabulary/interface
Requirements	Statements using that vocabulary
Behavioural model	Executable model using that vocabulary
V&V	Check behaviour vs. requirements

1

Ontology Structure

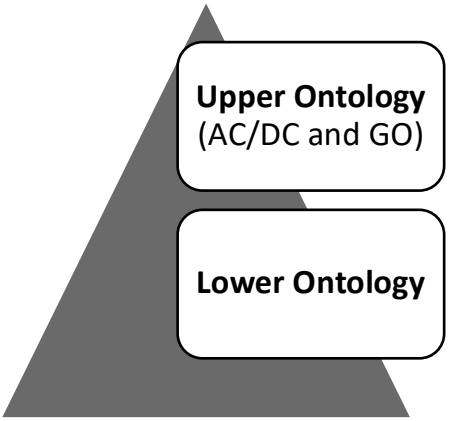
Upper ontology: universe of general types

- Architecture Concepts (AC)
- Design Concepts (DC)
- Global Objects (GO)

Lower ontology: domain-specific concepts

- Specialization of types in the upper ontology

- AC: Containers, communication links, factories
- DC: Design-level and behavioural constructs
- GO: Track topology primitives (node, edge, segment)



Disciplined ontology structure

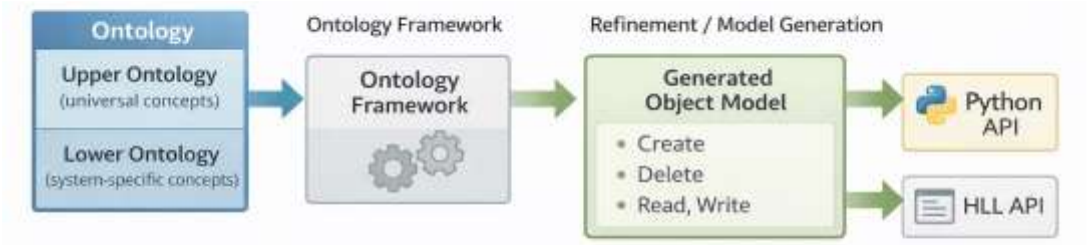
- Class-based type hierarchy (single inheritance)
- Attributes defined locally or imported via *mixins*, with orthogonal attribute *specialisations*

1

Interne

From Ontology to Object Model

- The **generic object model** is generated from the ontology, in two languages:
  - Python: for instantiation, and executable validation (based on test)
  - HLL: for formal verification (formal model in Many-Sorted First Order Logic, MSFOL)



9

R2DATO | GA 101102001

Object Model as a Common Semantic Foundation

- The object model is shared semantic foundation for:
  - Configuration data requirements
  - Design/behaviour models
  - Test cases
  - Safety requirements
- Expected benefits
  - Automated V&V
  - Reduced manual reinterpretation
  - Convergence across validation activities



10

R2DATO | GA 101102001

Interne

Builds on Proven Methodology

- Previously applied for fixed-block interlocking systems
- Automated testing and formal verification
- Successful prior application

- Previous work characteristics
  - Single subsystem context
  - Fixed-block object model
  - Object model as common reference for requirements

- Demonstrator D30.3 adopts this methodology, but generalizes it to enable broader context

Generalization of Methodology

- Enable application in broader context:
  - Moving block principles
  - Without prescribing specific operational solution
  - Configurable architectures and architectural options

- Extensions required to support:
  - Dynamic object types (runtime creation and deletion)
  - Configurable system architectures (evolution and change)
  - Richer set of path types (e.g., moving block paths)
  - Explicit system-of-systems interaction (message-based)

- Generic support for:
  - Object lifecycle management (via factories)

Object/Attribute Semantics and Object Lifecycle

- Object types:
  - Classified as static or dynamic
  - Dynamic objects may be created and deleted at runtime
  - Object existence is modelled explicitly (attribute 'active')
- Attributes:
  - Categorised as static, dynamic, or static for incarnation (fixed during object's lifetime)
  - All attributes explicitly typed
- Lifecycle management:
  - Object creation and deletion handled by behavioural models
  - Generic support provided via factories
  - Messages are dynamic objects allocated to communication links

13 R2DATO | GA 101102001

Demonstrator Architecture Overview

- The demonstrator is structured around:
  - An ontology defining system concepts, providing a shared conceptual reference for stakeholders
  - A generic object model generated from the ontology
  - Instantiation of the object model, from static configuration and infrastructure data
- These elements provide the basis for:
  - Behavioural modelling
  - Requirements validation, based on executable test and formal verification

14 R2DATO | GA 101102001

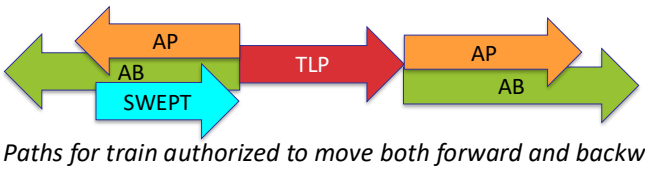
Interne

Static Configuration and Global Objects

- Global Objects (GO) represent:
  - Static, globally shared elements of the system
  - Track topology primitives: nodes, edges, and segments
- Global Objects:
  - Are instantiated from static configuration and infrastructure data
  - Used in normalized representation for areas and paths with static extent
  - Provide a stable reference for behaviour validation using specific applications

Lower Ontology with Train-Centric System Model

- The lower ontology defines concepts for ETCS trackside
- Inspired by the Moving Block Specification (Shift2Rail TD2.3), together with
  - Some typical interlocking concepts
  - Concept 'Forbidden path' (several usage purposes)
  - Concept 'Non-Nominal' (memorize non -nominal input/situations)
  - Train-centric modelling, for train movement-related requirements using "moving block" paths



Paths for train authorized to move both forward and backward

- *TLP: Train Location Path*
- *AB: Authority Base*
- *AP: Authority Path*

Interne

Visual Validation of Train Paths (Demonstration)

- Show
  - a **graphical visualization**
  - of **train movement**
  - over an **instantiated topology**
  - with **paths highlighted**

Separation of Object Model and Behaviour Models

- Object model:
  - Defines object types and their attributes
  - Provides a minimal, precise modelling interface

- Behavioural models:
  - Defined separately from the object model
  - Defined per container (subsystem)
  - Manipulate objects via factories

Interne

Conceptual Model for Message-Based Interaction

- All information exchange between subsystems is modelled explicitly, using messages
- Messages:
  - Form the primitive interaction layer
  - Are dynamic objects allocated to communication links
- Message types are derived from the ontology and a configuration
  - E.g., configure the message types used between RBC and ETCS onboard
- Messages support:
  - Static interfaces (configuration-style exchange)
  - Dynamic interfaces (runtime interaction involving dynamic objects)
- This interaction model is conceptual and only partially realized

19 R2DATO | GA 101102001

Primitive Execution Model (Behavioural Models)

- Behavioural models operate by:
  - Processing relevant input messages
  - Updating objects within the container
  - Sending output messages to the environment

20 R2DATO | GA 101102001

Interne

Instantiation Activities

- The demonstrator has been instantiated using a handful of examples from Trafikverket
  - Static configuration and infrastructure data, expressed in RDF
- Automated translation:
  - Translates RDF data into a graph-based representation (LCF)
  - Instantiates Global Objects and other trackside objects from infrastructure data

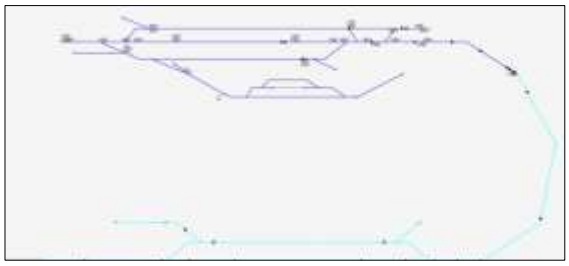


Figure 1: Visualized (part of) test data: Botniabanan

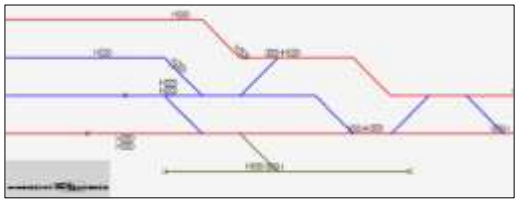
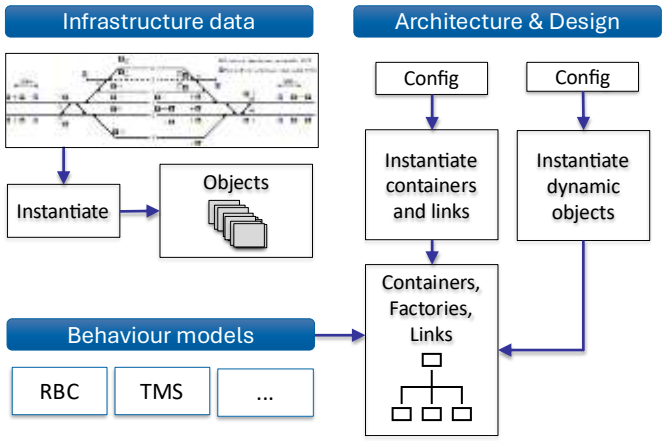


Figure 2: Visualized (part of) test data: Alingsås

Creating V&V Model With Demonstrator



Execute System of Systems model
- Execution-based test (Python)

Verify formal requirements (HLL)
- Configuration data requirement
- Consistency requirements
- Critical properties for subsystem
- Hazard at system of systems level

Animate execution traces

Interne

Validation Activities

- Validation performed so far includes:
  - Performance and capacity-related aspects
  - Efficiency of static data representations
  - Configuration data and consistency requirements
  - Formal modelling trade-offs (design for verifiability)
- Scope of validation:
  - Focused on feasibility and soundness of the approach
  - Limited, intentional scope

Maturity, Limitations, and Reuse Potential

- Demonstrator maturity
  - Prototype demonstrator, developed to explore modelling techniques
  - Instantiated and exercised using real infrastructure data examples, including larger cases
- Current limitations:
  - Behavioural models are intentionally abstract
  - Validation activities performed so far are limited in scope
  - Message-based interaction is only partially realized
- Reuse potential:
  - Modelling approach independent of behavioural models (signalling principles)
  - Applicable to ETCS variants, CBTC, and similar systems
  - Provides a basis for analyzing system-of-systems architectures and migration options

Interne

Key Takeaways and Closing

- A proven object model–based methodology is extended through ontology-based modelling
- Requirements for evolving signalling system architectures can be validated prior to construction
- Proposes a pragmatic level of abstraction for analysing evolving railway signalling systems-of-systems

- Alignment with Europe's Rail:
  - Supports early, model-based validation of new technologies
  - Enables analysis of system-of-systems architectures

- Outlook:
  - Explore AI-assisted authoring of behavioural models (e.g., behaviour trees) from natural-language specifications, constrained to use the object model interface
  - An associated report deliverable (D30.4) will follow later in the spring