Project funded under the Shift2Rail Joint Undertaking
Grant Agreement n°881826

LinX4Rail

Project Title:	**System architecture and Conceptual Data Model for railway, common data dictionary and global system modelling specifications**
Starting date:	2019-12-05
Duration in months:	36
Grant agreement no:	881826

MS6: Preliminary input on CDM for use case developments in WP4

Report	
Responsible partner	**BTSE**
Revision:	**1.0**
Document Status:	**Released**
Actual date:	**April 29th, 2021**
Dissemination Level:	**PU**

Table of Contents

1 History of changes

N°	Changes	Leader
1.0	First release	BTSE/SMO

2 Authors

		Details of contribution
Author(s)	Neil Freeman (BTSE) Stefan Wegele (SMO)	First draft
Contributor(s)	Airy Magnien (UIC)	Review
	Fabrizio Cosso (STS) Guiseppina Formatto (STS) Christian Esposito (STS)	Review
	Pierre Etienne Gautier (SNCF)	Review
	Jorgen Strandberg (BTSE)	Review

3 List of abbreviations

Abbreviation	Definition
ATO	Automatic Train Operation
CA	Consortium Agreement
CT	Cooperation Tool
EA	Enterprise Architect
EC	European Commission
ETCS	European Train Control System
GA	Grant Agreement
CBM	Condition based maintenance
CDM	Conceptual Data Model
CIM	Computation Independent model
ILX	Interlocking
JSON	JavaScript Object Notation
MS	Milestone
OMG	Object Management Group
PIM	Platform Independent Model
PSM	Platform Specific Model
RSM	Rail System Model
RTM	RailTopoModel
S2R	Shift2Rail
SQL	Structured Query Language
TAF	Technical Specification for Interoperability relating to Telematics Applications for Freight Services
TAP	Technical Specification for Interoperability relating to Telematics Applications for Passenger Services
UML	Unified Modeling Language
WP	Work Package
XMI	XML Metadata Interchange
XML	Extensible Markup Language

4 Executive Summary

LINX4RAIL MS6 “An internal report containing very first modelling framework to be used, and very first model structure including information layers is available for WP4” is detailed in this document. WP3 will produce a modelling framework based on existing core source projects identified as [EULYNX] data preparation model, [IFCRail], [railML3], [X2RAIL4 model], and [RailSystemModel] in a machine-readable format as per Grant Agreement [GA 881826]. This document is the first step in describing how these existing data models can co-exist as a federation of models and how the new data generated from the CDM project can be handled. There are a number of different sub task areas addressed, initially focusing on, but not limited to; Source Model overlapping, domain designation, Model structure, Common modelling rules, CDM tooling and model interfaces. These have been discussed within the WP3 GitLab forum and also during Technical Workshops. A detailed list of these issues is available there.

Additional to the model foundation deliverables, TAF/TAP source projects should be considered as input during the CDM concept extension phase, as well as creation of a concept to describe the rules to be followed when making extensions to the CDM.

This document will also aid to define the data flow process between the source projects as an input to the final target Project output.

Currently there is the start-up of Linx4rail-2 Project which will build on the concepts created in Linx4rail-1 CDM. Close collaboration between the two Projects will be carried out to ensure the convergence of the two Project workstreams.

5 Background and objectives

5.1 Background

The railway sector is currently acting in a fragmented way and in silos corresponding most often to physical or functional subsystems or use cases, and the different owners/managers of the overall infrastructure at regional/national level, without global extensive view or full control of the global system involved by rail operations.

With the progress of digitization, analogue devices based on relays were progressively substituted by digital ones, and it is more and more evident that enabling the communication between already existing digital tools for various subsystems can be beneficial. What is missing is an efficient, automated and standardized way for these integrated and interplaying systems to act as one ecosystem: sharing, integrating, identifying, correlating and exploiting the right data at the right time.

In order to meet these challenges, the objective of LinX4Rail is to develop and promote, a common Functional Rail System Architecture for the Rail sector, as requested by EC [GA 881826] page 24. It will be supported by a widely adopted Shift2Rail Common Data Model that, with the commitment of the Shift2Rail members, will establish the standard for interactions between legacy and new systems, thus ensuring sustainable interoperability between systems.

The ambition of LinX4Rail is to achieve a comprehensive approach for the CDM, as a models' federation, global system modelling specification and the strategy for implementation of technological breakthroughs.

The technical work within WP3 focuses on creating a CDM specification and formal definitions covering modelling requirements from S2R projects and external initiatives.

5.2 WP3 Objectives according to [GA 881826]

Objectives of WP3 are specified in the Grant Agreement [GA 881826] on page 17:

- *Creation of S2R-CDM (Conceptual Data Model) as a consistent federation of data railway system models, based on existing ones coming from RTM, EULYNX, railML and IFC.Rail and not only railway related ones, such as the ones from the OGC Sensor Web Enablement suite for the diagnostic data and processes and so on.*
- *Evolution and enrichment of CDM according to other S2R project requirements*

During the implementation of these objectives the WP3 is planning to produce three deliverables at the very last day of the project (s. Figure 1):

- **Deliverable 3.1: CDM Specification of the first version of CDM, including:**

- References to participating models
- Delimitations of the responsibility domain of each model
- Description of model structure, including information layers and interfaces between models
- Common modelling rules
- Requirements on CDM compliant implementations
- References to business use cases
- A set of UML diagrams extracted from the CDM definition

- **Deliverable 3.2 The complete CDM model in machine readable form, e.g. XMI, RDF**

- **Deliverable 3.3 Public CDM specification report with the high-level description:**
  - The description of overall model structure, including information layers and interfaces between models, Common modelling rules and Requirements on CDM compliant implementations.

5.3 WP3 Timeline

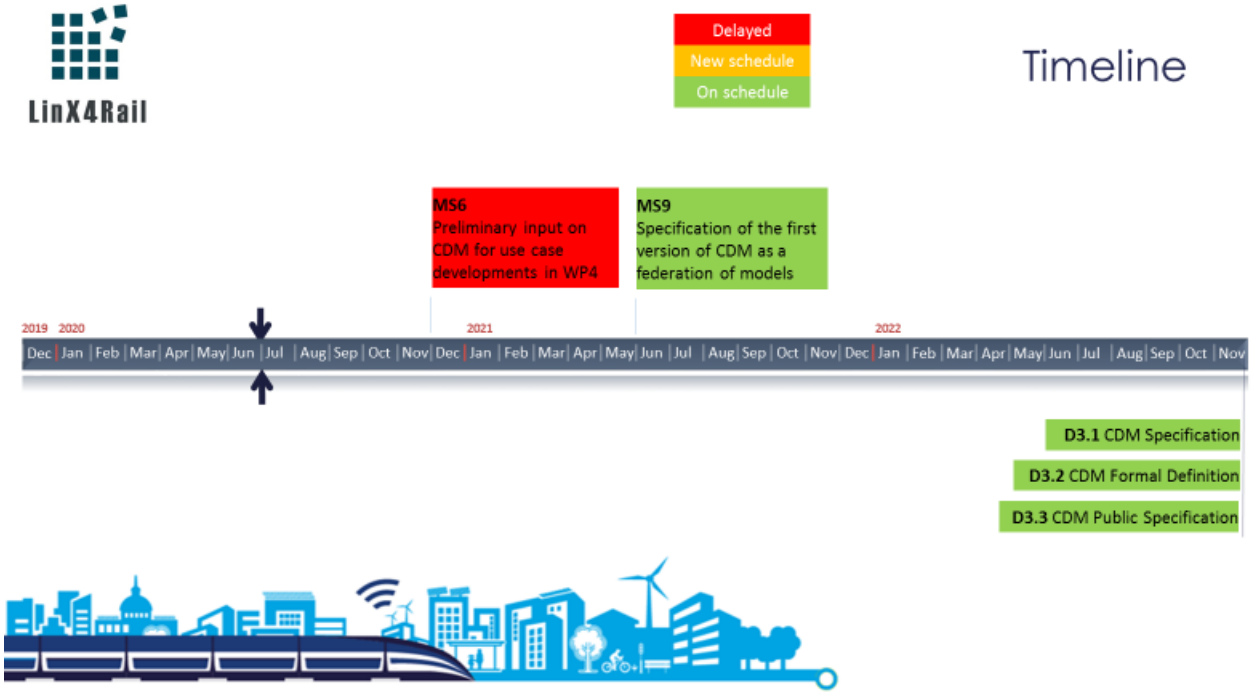

Figure 1. Timeline of WP3.

5.4 Expectations from WP4 and other S2R projects on LinX4Rail outcome

The WP4 took as demonstrators in LinX4Rail demonstrators planned in other S2R projects, e. g. In2Smart-2. Other S2R projects to be covered by the LinX4Rail project are:

- Impact-2 WP7 (Integrated Mobility)

- X2Rail-4 WP9 (TMS demonstrators TRL4), WP10 (TMS demonstrators TRL6)
- Fine-2 WP11, WP12, WP13, WP14, WP15

All these projects develop various demonstrators based on a common communication platform called "Integration Layer" (IL) [X2Rail-2 D6.1]. The IL defines a set of API to data access based on publish-subscribe pattern as well as prescribes to use a Canonical Data Model for message-schemata. At the low-end, the IL-clients send and receive the CDM-Data in JSON, XML and Protobuf format. To a allow a full interoperability, the schemata in these three languages must be aligned. This can be achieved by using a common Platform Specific Model for message-schema generation.

To summarize:

- Conceptual Data Model (CDM) is useless for the mentioned demonstrators, as there are hundreds of different methods to specify message schemata out of the Conceptual Data Model, each developer would take its own – the applications would be incompatible.

- A Platform Independent Model (PIM) (depending on definition of this term) is also useless for the demonstrators, as it does not provide sufficient unique-ness of the message-generation.

- A Platform Specific Model (PSM) containing sufficient details for unique generation of message-schemata is the expected outcome of LinX4Rail by other S2R projects.

Expected areas of the PSM are defined in the deliverables [X2Rail-2 D6.1], [Fine2 D11.2]. In general these are:

- Infrastructure data sufficient
  - for precise runtime calculation incl. functional elements (signals, stations etc.)
  - to cover ATO ETCS SS 126
  - for graphical representation in TMS applications
- Rolling stock
- Timetable management
- Danger goods
- Possession management
- Etc.

5.5 Integration of WP3 into Linx4rail project

As described in the previous section, according to the GA WP3 shall create a consistent Conceptual Data Model based on selected source models. That means: using UML-Source-Models to create an aggregated UML-Model called CDM and publish the results in three deliverables. The demonstrators in several S2R projects (LinX4Rail-WP4 and X2Rail-4, Impact-2 etc.) require a FFFIS specification of the models in form of message-schemata or at least a Platform Specific Model usable for message-schema generation.

Conceptual Data Model cannot be used for message schema generation, as it lacks composition hierarchy, which is platform specific: an SQL-based model requires other containers compared to NoSQL data model. Therefore, to be usable for the S2R demonstrators in Linx4rail-WP4, X2Rail-4, Fine-2, Impact-2 and external project Optima, the CDM must be extended with the composition hierarchy and partially restructured. This restructured UML-model can be used for generation of message schemata. In the discussions it was decided to give to these models following names:

- CDM is a Conceptual Data Model, which is Platform Independent model (PIM) in OMG-definition. It is structured according to the SOLID concept [SOLID].
- A UML class diagram with composition hierarchy supporting NoSQL data representation for efficient messaging, as required by the S2R demonstrators is called Platform Specific Model (PSM). There are many PSMs possible – here a specific one is considered.
- Schemata generated out of the PSM are called according to the target language:
    - JSON schema
    - Protobuf schema
    - XML schema.

The demonstrators need PSM to generate schemata on their own, or the message schemata to ensure interoperability.

During the Advisory Board Meeting the responsibility for creation of the PSM was assigned to WP3 with support of WP4 (s. Figure 2).

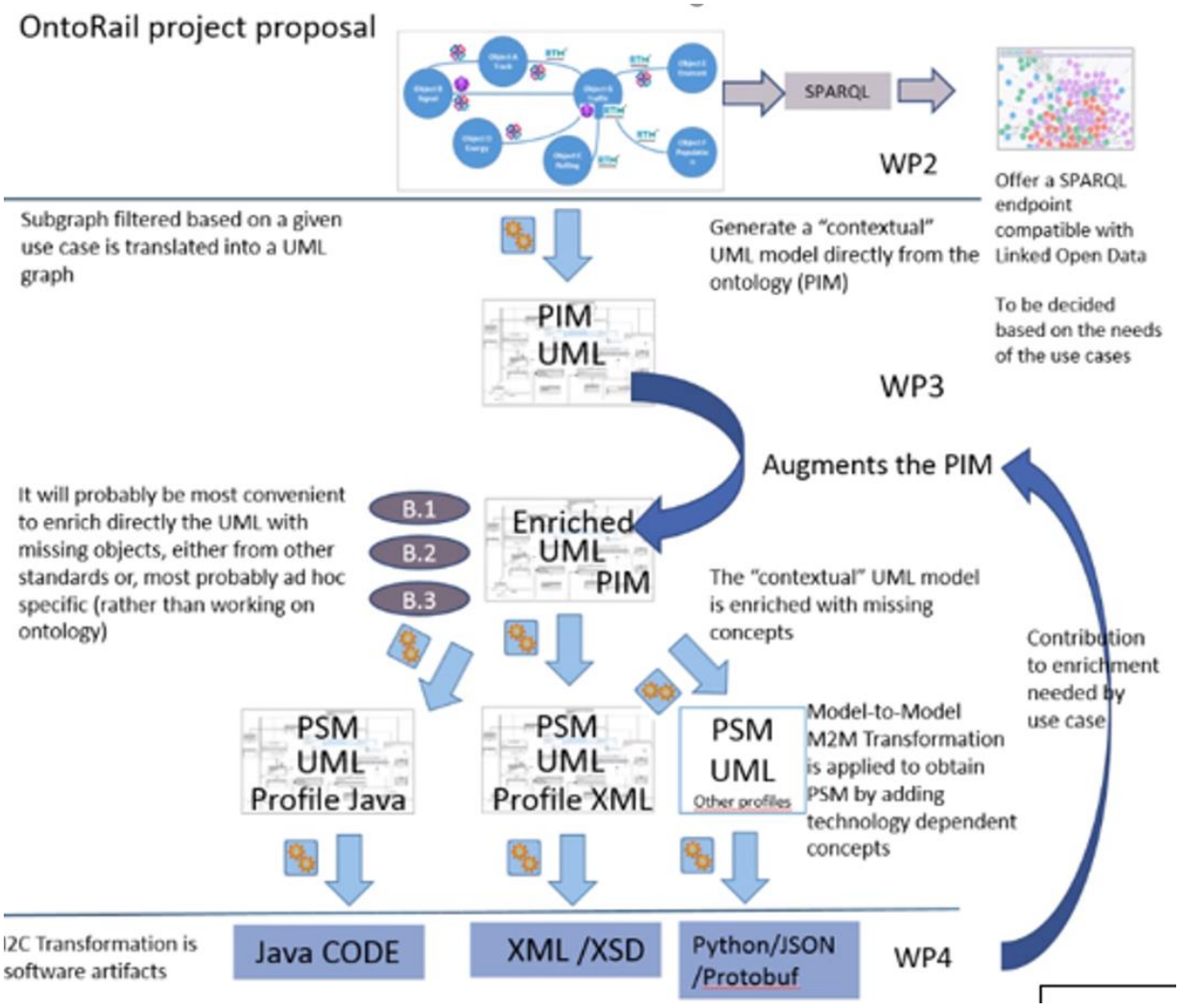

Figure 2. New project structure as decided during the advisory board meeting.

Let's look on model examples on each level to evaluate the differences.

5.5.1 Model examples

Conceptual data model according to the SOLID principle can be seen in EULYNX specification.

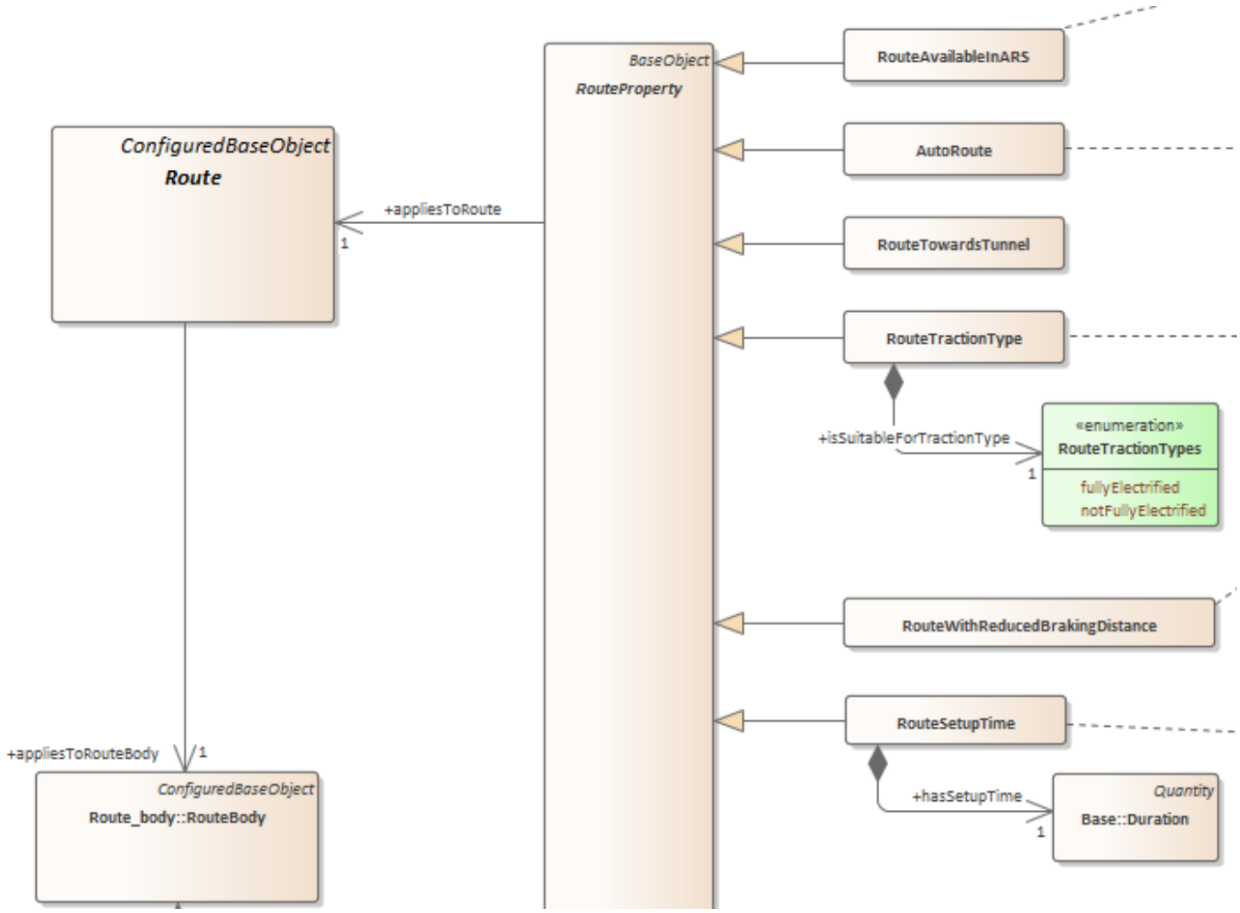

Figure 3. EULYNX specification of the Route.

As visible from the Figure 3, instead of putting Route-properties inside of the Route, as would be expected in case of messages, the single RouteProperties are dedicated objects, having their own life-cycle and loosely referencing the Route they describe. This approach has several advantages:

- No optional attributes inside of the Route are required.
- The definition of container classes is shifted to the PSM-level, as the container-hierarchy depends on the selected data management architecture.
- IM-specific Routes can be created by deriving and extending of single RouteProperties.

In this respect it is a model of concepts, not directly usable for messaging.

Now, lets look at the Platform Specific Model expected by the X2Rail-4 WP9 demonstrators. As it does not support inheritances, it requires leaf-classes from the CDM-inheritance hierarchy. Here is an example, how a leaf-class a Route for DB-IM could be extracted from the Figure 3 CDM into PSM.

```
{
    "name": "DBRoute",
    "attrs": [
      {"intId": 1, "name": "id", "dataType": "string", "key": true},
      {"intId": 2, "name": "body", "composition": "RouteBody"},
      {"intId": 3, "name": "availableInARS", "dataType": "boolean"},
      {"intId": 4, "name": "routeTractionType", "enumType": "TractionType"},
      {"intId": 5, "name": "withReducedBrakingDistance", "dataType":
"boolean"},
      {"intId": 6, "name": "setupTime", "dataType": "uint32", "units":
"msec"}
     …
```

```
    ]
  }
```

Hence, the PSM contains specific datatypes, supports references and aggregations, does not support inheritances, defines string and integer ids of all attributes.

5.5.2 The way from source UML to CDM

A parallel discussion took place about the integration of Ontologies into the creation of the Conceptual Data Model. In fact, Conceptual Data Model can be created:

- Directly out of the source UML-models
- Over the intermediate step:
    - source UML-Models into Ontologies
    - filtering out of Ontologies into the Conceptual Data Model.

The consensus was, that the way over Ontologies provides a set of advantages:

- The concepts from the source models are linked together to identify overlaps
- Additional associations between concepts can be integrated
- A dictionary integration is built-in.

Therefore, it was decided to follow the Over-Ontology-Approach until the end of the project. The drawback of this solution is, that a consistent CDM would be ready at the very late stage of the project, so that the S2R-Demonstrators will not be able to use it.

Therefore, it was decided to follow in parallel the “short-term” solution, by directly creating the CDM out of the source UML-models. To create this solution WP3 needs a technical tool, called “Modelling Framework” described in the next section.

6 Initial modelling framework

According to the Grant Agreement [GA 881826] page 19, the internal report shall contain "a very first modelling framework to be used".

In this chapter several aspects of the modelling framework will be discussed:

- Introduction
- Requirements on the framework
- Decision process for model filtering
- Enterprise Architect implications
- Versioning of the modelling framework
- Long-term solution
- Open questions.

6.1 Introduction

As defined in the GA the objective CDM shall be created as a federation of existing source models RailSystemModel (RSM), EULYNX, RailML and IFC Rail (BIM). The first delivery of the S2R-Conceptual Data Model (CDM) shall contain only the data from source models without any additional content. Overlapping concepts in the source-models shall be identified and prevented in the target S2R-CDM. In meantime the list of the source models has been extended by the model proposed by X2Rail-4 project.

Objective of the modelling framework is to enable this approach (s. Figure 4).

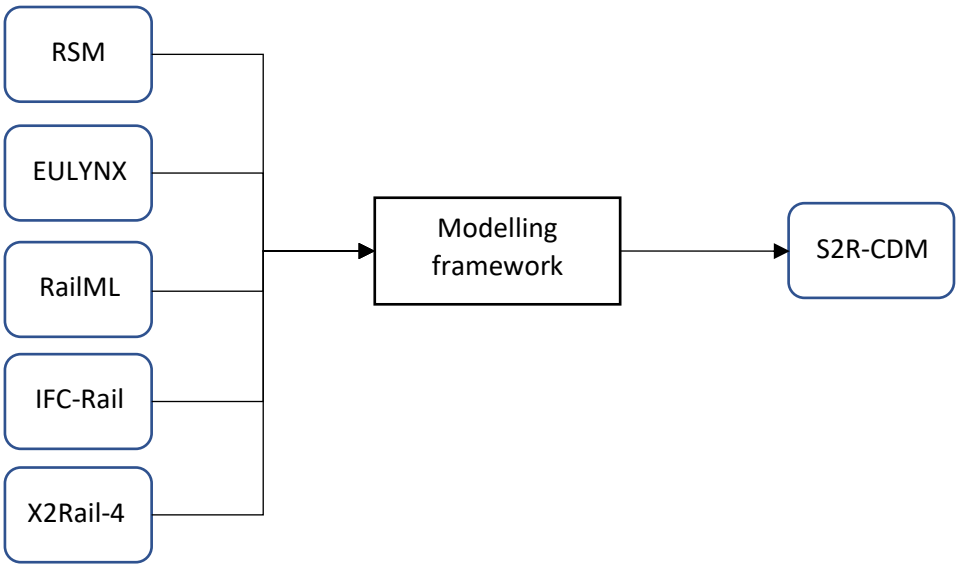

Figure 4. Modelling framework responsibility.

In the context of demonstrators in WP4 and X2Rail-4, the single Conceptual Data Model (PIM) shall be transformed to Platform-Specific and application-specific Models (PSMs) to be used by

the Integration Layer (s. e.g. [X2Rail-2 D6.3]). Same would apply to other projects such as OPTIMA (open call).

The transformation from CDM to PSM shall be, to the greatest extent possible, governed by user choices (selection of relevant packages) and transformation rules, in view of automated or assisted transformation.

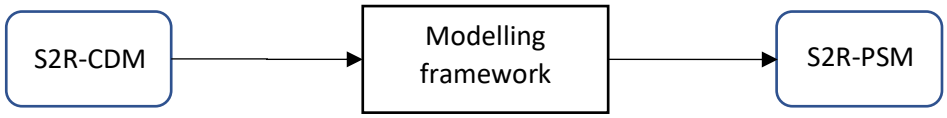

Figure 5. Using the same modelling framework for creation of S2R-PSM.

6.2 Requirements on the modelling framework

There are several requirements to be fulfilled by the modelling framework listed in Table 1.

Table 1. Requirements on the modelling framework.

ID	Reason	Requirement
1	The S2R-CDM shall be consistent for references.	The modelling framework shall ensure that all references inside the model point to the existing elements. References to external (source) models are prohibited.
2	The modelling framework shall ensure, that primitive attribute types in the classes come from an agreed set of primitives. The CDM contains over 1000 classes and it is responsibility of the modelling framework to map from the source-model-primitives (XML-specific, C-specific…) to the agreed set. The	S2R-CDM shall be consistent for agreed base types.
3	It is assumed that the source models contain over 1000 classes. Manual operations on them prevents scalability. The modelling framework shall automate the conversion process to a large extent.	The modelling framework shall be scalable.

4	The source models follow their own development life cycle, so new versions will appear several times a year.	The modelling framework shall allow automated versioning of the generated model (CDM & PSM).
5	Use cases are coming from several demonstrators in WP4 and from X2Rail-4, Fine-2, Impact-2 projects. It must be possible to validate Use-Case coverage by the model.	The modelling framework shall allow linking between Use-Cases and the UML-Model e.g. by means of tag-values.
6	The development, testing, learning and maintenance of the modelling framework requires a considerable amount of time and budget. To keep it low, the modelling framework shall be reusable.	The same framework shall be applicable for conversion of S2R-CDM into S2R-PSM.
7	The target model shall contain references to the source classes/enumerations/attributes as well as the authors.	The modelling framework shall integrate tracking information from the source models into the generated UML model, e.g. by means of tags.
8	As the source models are already using the UML-Tool Enterprise Architect, it was decided, that S2R-CDM shall be maintained on the Enterprise Architect Cloud.	The modelling framework shall put the final model into Enterprise Architect.
9	It is assumed, that in case of overlapping between the source models decisions against some source models shall be taken. They must be formally collected and archived.	The modelling framework shall establish a formal decision process on selection of classes/enumerations/attributes from the source models.
10	The reasons for decisions shall be annotated and archived.	The modelling framework shall allow to annotate reasoning for the modification decisions.
11	The long-term solution foresees to use ontologies for “filtering” of the S2R-CDM. Making the decisions twice: once for the short-term and one for the long-term solutions is too expensive. Therefore, the	The modelling framework shall preserve the decisions made for the usage in a long-term solution based on ontologies.

	artefacts containing the decisions shall be usable for filtering of the ontologies, by being readable and parsable by a general purpose programming language (e.g. Python).	

In this context the "formal decision process" means, that all steps of the process are persisted in a schema-based parsable document and can be automatically re-applied to the source models at any point in time and to future versions.

6.3 Technical architecture of the modelling framework

Based on the listed requirements the modelling framework was structured according to Figure 6. The process is divided into two steps:

- A manual step of decision-making considering current version of the S2R-CDM and comparing it to the source Use-Cases.
- An automated step implementing the decisions made in the first step of the process.

As follows from the figure the modelling process is iterative: the decision process is applied until the resulting S2R-CDM fulfils the Use-Cases covering the Requirements 7, 9.

The results of decision-making process are captured in a formal representation in files, containing references to the Use-Cases and reasoning. The decision-files will be managed by a versioning system, covering Requirements 5, 10.

The decision implementing script will be applied to any new set of the source models, generating a new version of the S2R-CDM, covering requirement 3.

The decision implementing scripts will be managed by the version control system covering the requirement 4.

The decision script shall apply validation checks, ensuring consistency of the resulting model and implementing the requirements 1 and 2.

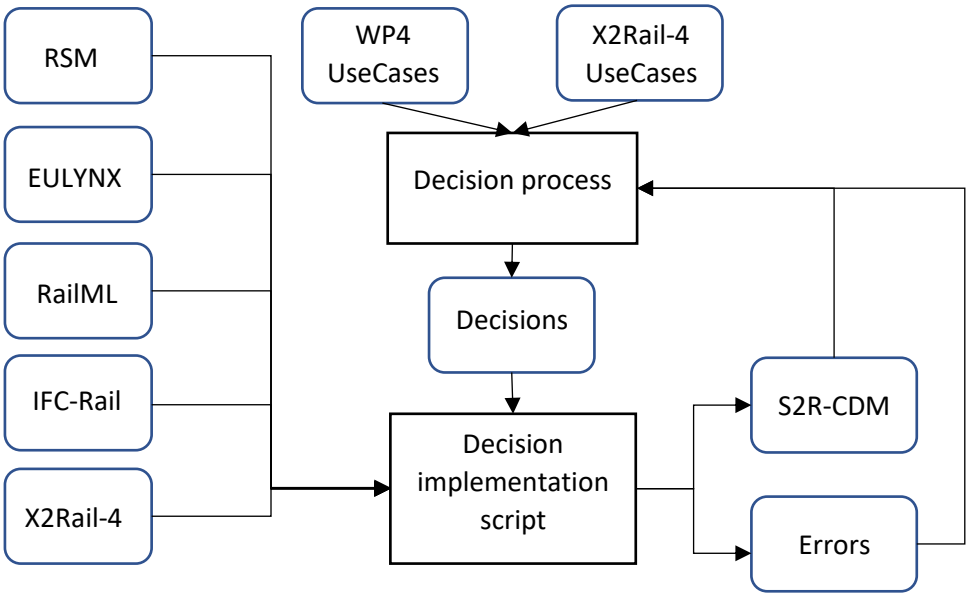

Figure 6. Application of the modelling framework for generation of CDM (SOLID PIM) from the source models.

The described process shall be applicable for other input model (s. Figure)

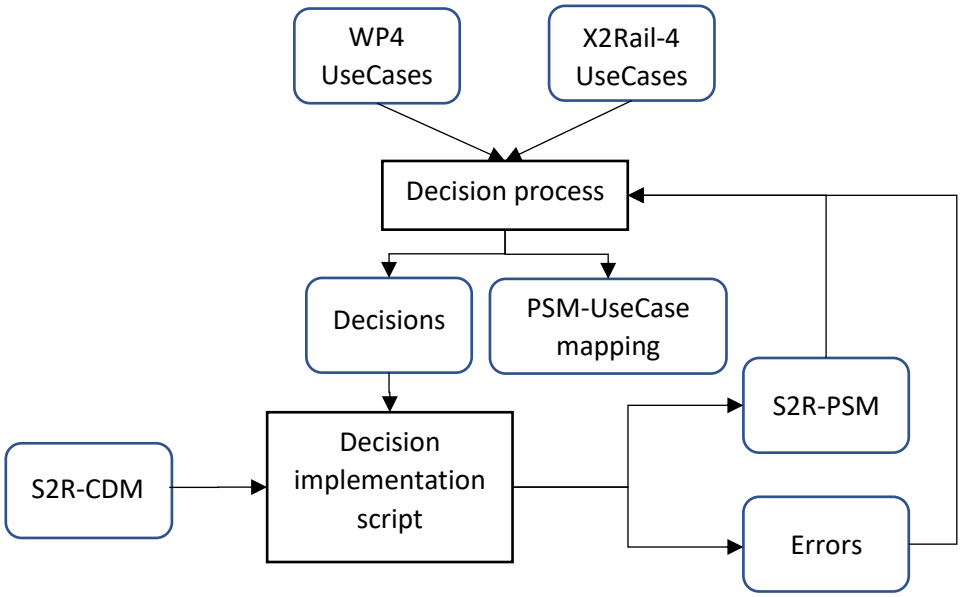

Figure 7. Application of the modelling framework for generation of S2R-PSM.

The decisions taken in the two processes are quite different:

- In the first case:
  - which class to take from which source model
  - which attributes must be extracted according to SOLID principle
  - which container classes must be removed to allow Platform specific models
- in the second case:
  - which attributes must be grouped into the class
  - which container classes are needed

- which base-datatypes shall be used for each attribute

6.4 Decision process

The decision process comprises atomic decisions on:

- Structure of the target model (packages of packages of packages …)
- Allocation of the source-model-classes and enumerations to the packages
- Modifications of the classes, associations, attributes, and enumerations.

The decisions are captured in a JSON-file with the following structure (s. Figure).

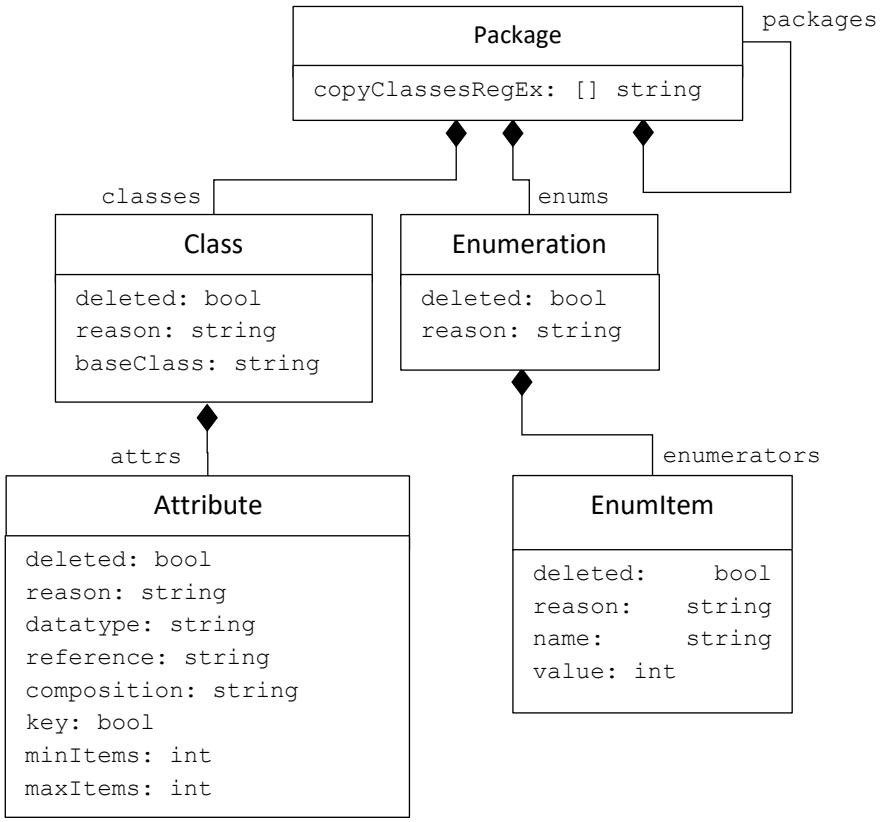

Figure 8. Class diagram for decisions-capturing.

Example of the decision file (consider this as a proof-of-concept for the modelling framework, not as already agreed decisions):

```
{
  "packages": {
    "Common": {
     "packages":
     {
      "Base": {
         "copyClassesRegEx": ["RSM/Common/Base/*"]
      },
      "Topology": {
         "copyClassesRegEx": ["RSM/Common/Topology/*"],
```

```
       "classes": {
         "NetElement": {
          "attrs": {
            "relations": {
           "deleted": true,
           "reason": "to prevent double linking from PosisionedRelation"
            }
          }
         },
         "PositioningNetElement": {
          "deleted": true,
          "reason": "Creating of 1001 maps of NetElements using 1001 PositioningSystems is a
 separated domain: reverse dependency in this case."
       },
       "LinearElement": {
         "baseClass": "NetElement",
         "attrs": {"length": {"reference": "RSM.Common.Base.Length"}}
       },
       "NonLinearElement": {"baseClass": "NetElement"},
       "PosisionedRelation": {
         "attrs": {
          "elementA": {"reference": "LinearElement", "reason": "Only LinearElements have Start/End
to be connected."},
          "elementB": {"reference": "LinearElement", "reason": "Only LinearElements have Start/End
to be connected."}
         }
       }
     }
   }
 }
}}}
```

The captured decisions are implemented in two simple steps:

1. As the first step in a package the source-model-classes are copied according to the list of “copyClassesRegEx”. As source of the classes different source models can be used.
2. On the list of copied files, the following “classes” and “enums” modifications are applied:
    a. If deleted = true, the class or attribute is removed. The reason must be defined.
    b. If an attribute occurs in the list, it will overwrite all its parameters in the target model. If the attribute was not defined in the source model, it will be created.

6.5 Implications from Enterprise Architect

The Enterprise Architect (EA) is a state-of-the-art tool covering all aspects of UML-Modelling as well as SysML, code generation etc. In the context of this project only a small part of the EA is used:

- Definition of packages/classes/attributes/connections
- Creation of class diagrams with external figures and annotations.

The EA provides a central repository on the cloud, so that a cooperative work can take place very efficiently.

The EA provides an export into an OMG-standard UML-XMI file, unfortunately losing any comments assigned to the model. This drawback prevents using of this export/import capability for model transformation.

As the solution for this problem the integrated Scripting of EA is used by the modelling framework. EA supports JScript, JavaScript and VBasic.

Unfortunately, the runtime environment of the Scripting is quite slow, so that one run of the decision implementation script (s. Figure 9) for RSM-only takes over 12 minutes. Therefore, it would be quite time consuming for running iterative decision process.

Another drawback is difficulty in representation of the classes/attributes. The tree-view shows only list of classes with their attributes (s. Figure 9).

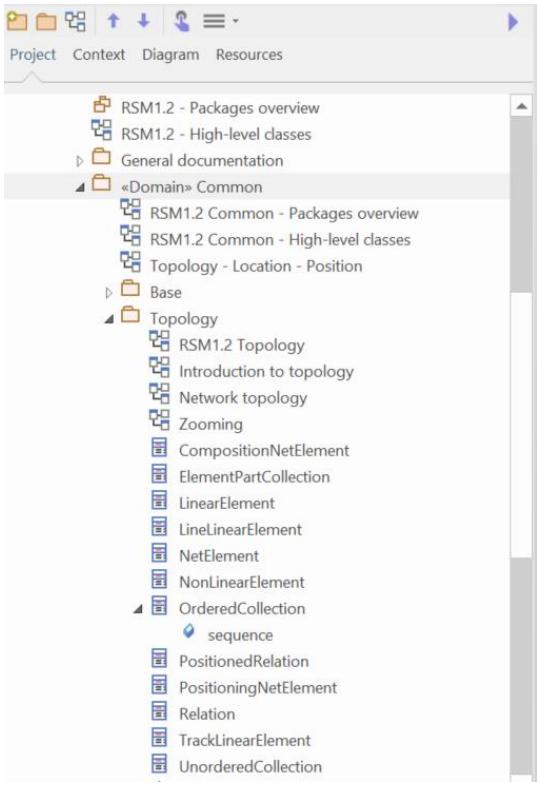

Figure 9. Tree-view of the model in EA.

The linking information is partially visible by clicking on the context menu „Properties“ at each class, which is useless (s. Figure 10).

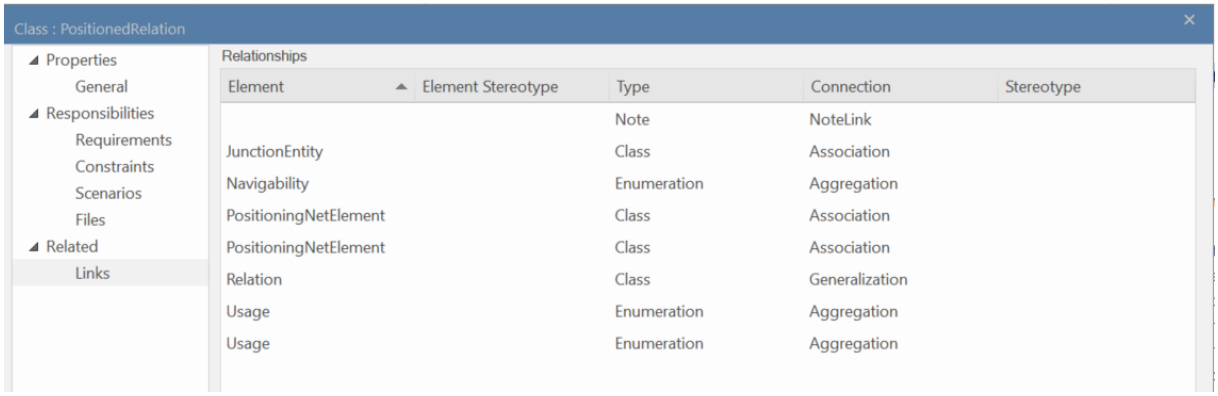

Figure 10. Connection representation for the class RSM.Common.Topology.PositionedRelation. Missing information on links.

The links are shown, but it is still unclear;

- where they start and where they end
- the naming of associations and aggregations are not provided.

To evaluate the resulting model all classes must be placed into diagrams together with all linked classes, which is currently a manual process.

For our use case we take 5 source models with over 2000 classes, apply filtering and copying of the classes during about 20 minutes. To validate the resulting model more than 200 diagrams must be created and layout.

Therefore, it was decided to adjust the architecture of the modelling framework to the Enterprise Architect in the following way (s. Figure 11). The JSON representation of the source and target models is quite similar to the decision-JSON-file, except the attributes “deleted” and “reason” are omitted. This keeps the learning curve flat – only one “format” must be understood by the user of the modelling framework.

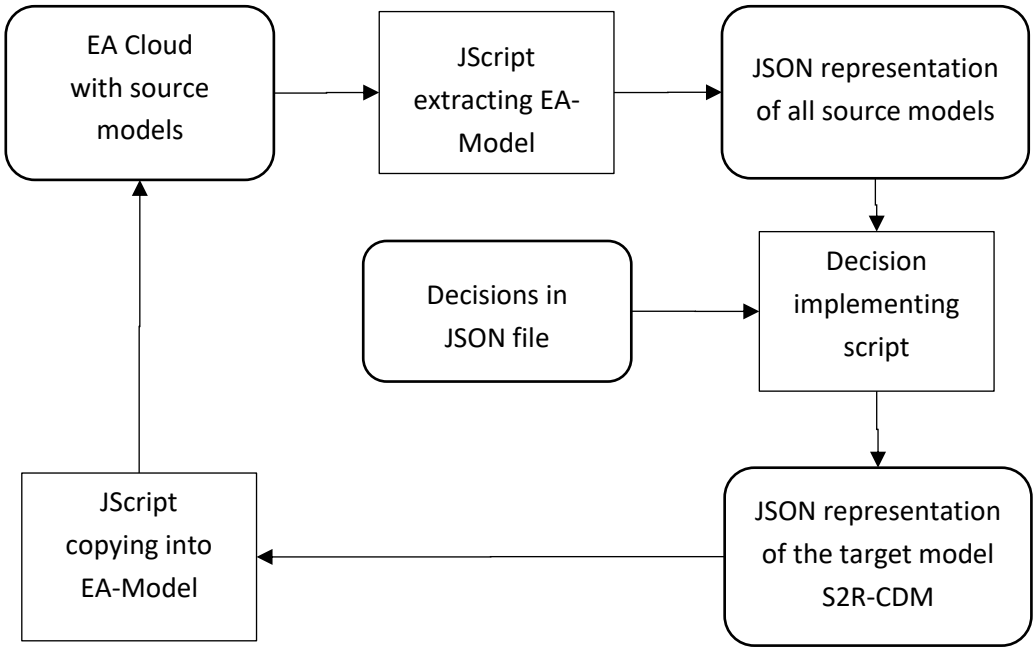

Figure 11. Model extraction artefacts around Enterprise Architect.

6.6 Long term solution

In a long-term solution, it is planned to extract the S2R-CDM out of ontologies, located on a dedicated framework outside of EA-Cloud. It is assumed, that the artefacts of the modelling framework can be reused in WP2 according to Figure 12 covering the Requirement 11.

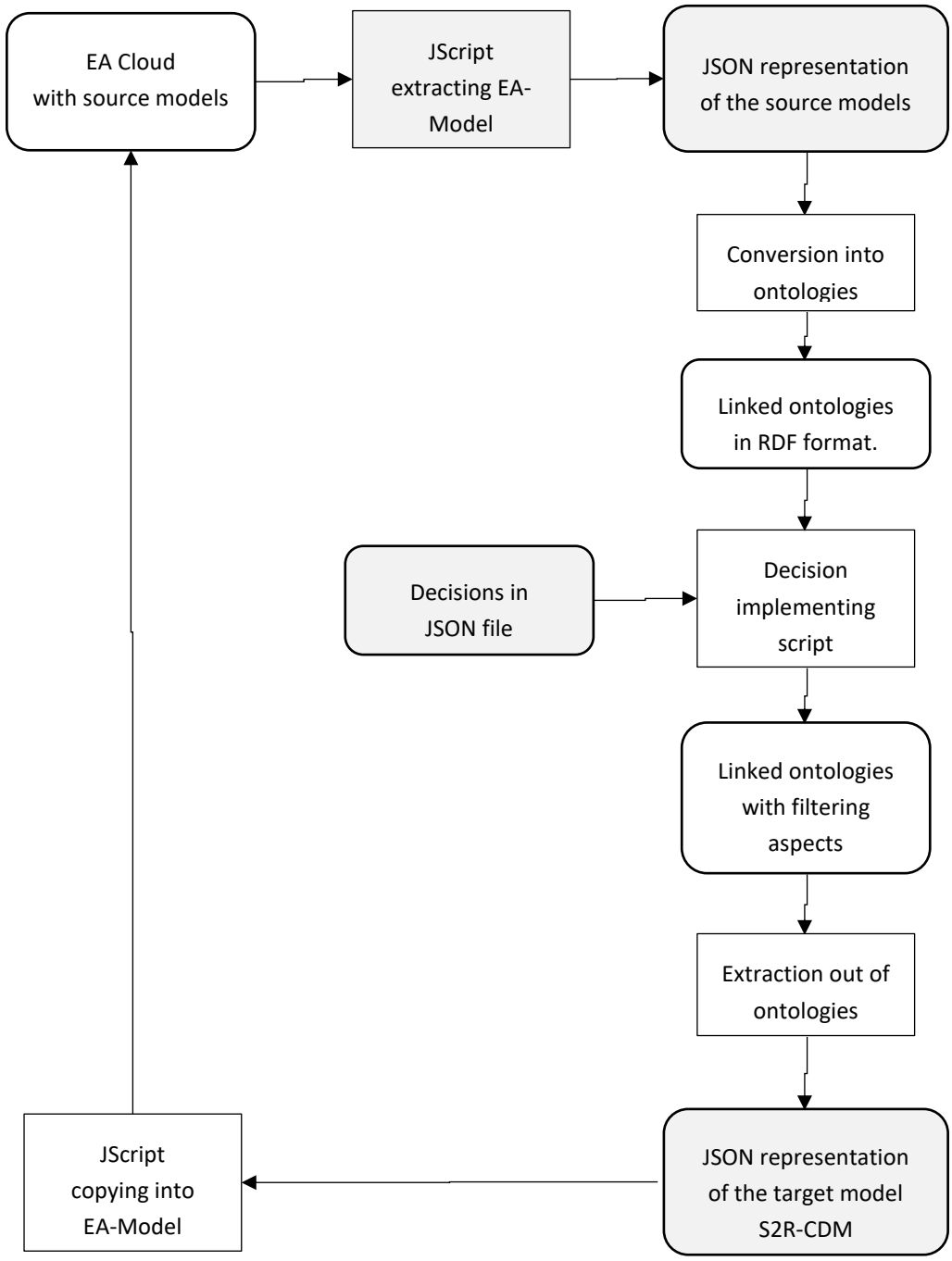

Figure 12. Reuse of shadowed artefacts in WP2 for the long-term solution.

To enable the long-term solution while establishing the short-term solution, the linkage between UML class diagram language elements (classes, associations, packages…) and their "image" in the web ontology language (OWL) shall be investigated at an early stage with WP2. Such linkage may involve:

- A suitable usage of UML (e.g. consistent usage of unambiguous role names in associations);
- A limited usage language extensions (stereotypes, tagged values).

6.7 Versioning of the modelling framework

The artifacts of the modelling framework are managed by the version management system Gitlab.

The directory contains:

- decisions: various decision files in json format.
- scripts: JScript and Python scripts comprising modelling framework.
- sources: source models in json format extracted from the EA-cloud by the script ea2json.js
- target: target models generated from the source models by the script applyDecisions.py

Every partner interested in the development shall be able to understand the algorithms and setup a development environment on a normal PC.

6.8 Open questions

As the modelling framework is still under development some questions are not yet evaluated:

- How to integrate “domain documentation” from the source models, which refers to source-model-classes preserving Requirement 1 (the S2R-CDM shall be self-contained). This is probably possible after developing of a JScript copying the diagrams.

7 Preliminary model structure of S2R-CDM

7.1 Introduction

Although the S2R-CDM is built out of the source models, it allows to setup its own model structure. This puts the source models into the competitive state, where the best fit can be selected to implement the desired S2R-CDM-structure.

The term “model structure” means the Tree of Packages containing Classes and Dependency relations between Classes crossing the packages limits. The Dependency relations can be inheritance, aggregation, and composition relations.

The S2R-CDM the Packages shall build a Tree-hierarchy, which prohibits cyclic dependency relations. This simplifies quite strongly the data management and reusability of the packages.

The term Package is used in UML-notation and can be considered as synonym to the terms Module, Layer.

In S2R-CDM the package tree follows the intrinsic modules in the railway domain, which are already identified by the source models, which “cut” the railway domain in quite similar packages.

Higher level packages deal with dimensions that are common to several domains (time, geography, measure, units, and other general engineering concepts).

In the S2R-PSM the package structure can be slightly modified to cope with the platform – e.g. if the platform supports only one level of packages, the classes from the sub-packages are put into the parent package.

7.2 The preliminary S2R-CDM package structure

The preliminary S2R-CDM package structure is shown in Figure 13. To keep the figure readable the expected Dependencies between the Packages are hidden. In general, they are expected to go from the bottom to the upper layers.

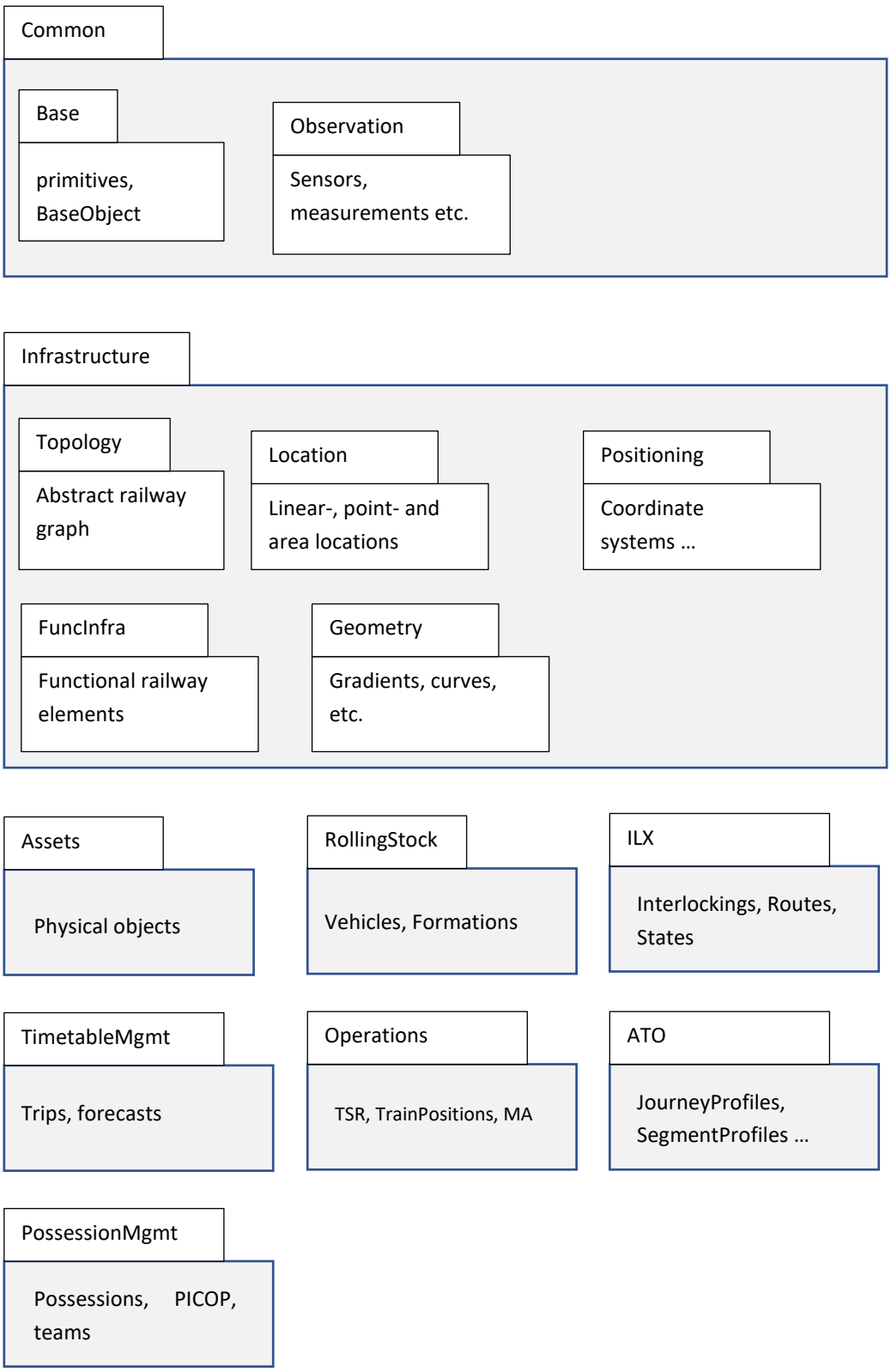

Figure 13. Overview of S2R-CDM packages.

In the Table 2 the mapping of the S2R-CDM-packages to the source models is provided. The source models are listed in the priority sequence: first use classes from the previous source model in the list. As of today (10.02.2021) following availability of the source models on the EA-Cloud is observed:

- RSM available
- X2Rail-4 available
- EULYNX is in preparation to be imported
- IFC-Rail discussions about permission with IFC-project
- RailML discussions with JU about license agreement with RailML.org.

Table 2. Coverage of the S2R-CDM packages by the source models.

S2R-CDM package	Source models	Estimated coverage of the WP4 use-cases by WP3.
Common.Base	RSM	95%
Common.Observations	RSM, SensorML?	50%
Infrastructure	RSM, X2R4, RailML	95%
Assets	IFC Rail	First import expecting
RollingStock	X2R4, RailML	60%
ILX	EULYNX	Not used
TimetableMgmt	X2R4	95%
Operations	EULYNX, X2R4	95%
ATO	X2R4 (ETCS SS 126)	Not used
PossessionMgmt	X2R4	95%

As the output of this work package two separated models are expected:

- S2R-Conceptual Data Model emphasising:
  - Relations between concepts, i.e. associations and inheritance
  - Ignoring container objects required for the data management
  - Following the SOLID principle, especially O – open for extension, but closed for modification.
- S2R-Platform Specific Model
  - Defining a compositional data tree,
  - Introducing container classes required for the NoSQL-Data management,
  - Removing inheritance, to allow serialisations without inheritance support,
  - Introducing additional decisions about specific primitives, e.g.
    - how many bytes shall have the length-attribute of a Vehicle,
    - shall it be real or fixed point-integer,
    - do we consistently use one and only one geographic reference system such as WGS84,

- of what physical unit shall it be [m] [cm] [mm]?

7.3 Next steps

According to the project plan in GA the first version of the S2R-CDM is expected by 5th June 2021. In order to achieve this goal, cooperation activities from the stakeholders are planned.

According to the Grand Agreement following companies decided to participate in CDM-creation:

BTSE, SNCF-R, STS, CAF, TRV, DLR, TD, RLN, SMO, CEIT, DB, NR, ALS.

They shall come to agreements on selection classes/attributes from the source models and capture them in the modelling framework (s. above).

Asked questions:

How is the modelling framework developed and maintained, what are the resources needed for building it, is there specific development needed?

The modelling framework represents a set of scripts together with EA-cloud and EA-clients. The scripts are located on the Gitlab and every partner has access to them and possibility for adjustments. Currently SMO is responsible.

Who will run the process for which it is understood that there are decisions to make?

The decision process for initial CDM is allocated in Task 3.1. The task leader is BTSE, who will organise decision workshops.

In the process of the decision making and validation: who decides (and validates choices)?

The decisions must be taken by representers of WP3-companies: BTSE, SNCF-R, STS, CAF, TRV, DLR, TD, RLN, SMO, CEIT, DB, NR, ALS. Validation of the decisions is done in the context of WP4 in the prototypes. In WP4 we have a subset of WP3-companies: STS, SNCF-R, CAF, TRV, TD, CEIT, DB, NR. In the last deliverable D3.1 with the CDM all WP3-participants will be listed at least as reviewers, who accepted the resulting CDM.

8 References

EULYNX	http://dataprep.eulynx.eu/2020-10/index.htm
GA 881826	GRANT AGREEMENT NUMBER 881826 — IP/ITD/CCA1 — IPX and CCA
IFCRail	https://www.buildingsmart.org/ifc-rail-candidate-standard-is-available-for-review-and-comment/
RailSystemModel	Preliminary version available to named users.
RailML3	https://www.railml.org/en/download/schemes.html downloadable after registration.
SOLID	https://en.wikipedia.org/wiki/SOLID
TAF TSI	https://www.era.europa.eu/content/technical-documents-baseline-250
TAP TSI	https://www.era.europa.eu/content/tap-baseline-140
X2Rail-4 model	shift2rail-cdm.org