RailTopoModel
Railway Network Description

A conceptual model to describe a railway network

Kindly supported by
# List of contents

1. EXECUTIVE SUMMARY .................................................................................................................................................... 4  
2. INTRODUCTION ................................................................................................................................................... 5  
   2.1 BACKGROUND .................................................................................................................................................. 5  
   2.2 OBJECTIVES .................................................................................................................................................... 7  
   2.3 SCOPE ....................................................................................................................................................... 8  
3. METHODOLOGY ........................................................................................................................................ 9  
   3.1 MODEL REQUIREMENTS ......................................................................................................................................... 9  
   3.2 FEASIBILITY STUDY ........................................................................................................................................ 9  
   3.3 OVERVIEW .................................................................................................................................................. 10  
   3.4 PREDEFINED DESCRIPTION LEVELS .......................................................................................................... 11  
4. CONCEPTS ........................................................................................................................................... 15  
   4.1 CORE ELEMENTS ......................................................................................................................................... 15  
   4.2 REFERENCING ............................................................................................................................................. 18  
   4.3 TOPOLOGY ................................................................................................................................................ 18  
   4.4 FINALISATION OF THE ELEMENTS ........................................................................................................... 19  
   4.5 BUSINESS ................................................................................................................................................ 19  
   4.6 FINALISATION OF THE MODEL .................................................................................................................... 24  
   4.7 UML DIAGRAM ......................................................................................................................................... 24  
   4.8 REFERENCE AND POSITIONING SYSTEMS .............................................................................................. 29  
5. CONCLUSIONS ........................................................................................................................................ 33  
   5.1 INTEGRATED OVERVIEW ...................................................................................................................................... 33  
   5.2 FUNCTIONALITY UNDER DEVELOPMENT ................................................................................................... 33  
   5.3 NEXT STEPS ............................................................................................................................................... 33  
6. APPENDIX ........................................................................................................................................ 34  
   6.1 GLOSSARY ............................................................................................................................................. 34  
   6.2 ACRONYMS ............................................................................................................................................. 34  
   6.3 UML - UNIFIED MODELLING LANGUAGE - NOTATION .................................................................................. 34  
   6.4 AGGREGATION (LINK BETWEEN LEVELS OF DETAIL) ............................................................................. 37  
   6.5 DISAGGREGATION ...................................................................................................................................... 37  
7. FREQUENTLY ASKED QUESTIONS .............................................................................................................................. 54  
   7.1 COMPARISON WITH OTHER INITIATIVES, NORMS AND STANDARDS ......................................................... 54  
   7.2 UML MODEL VS EXCHANGE FORMAT, DATABASE AND SOFTWARE DESIGN ........................................ 55  
   7.3 RAILTOPOMODEL AND RAILML .............................................................................................................. 56  
REFERENCES .................................................................................................................................................. 57  
COPYRIGHT ......................................................................................................................................................... 57
### Authors

---

#### REVISIONS

<table>
<thead>
<tr>
<th>Version</th>
<th>Author</th>
<th>Date</th>
<th>Page / Paragraph</th>
<th>Commentary</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>RTM workgroup</td>
<td>20.12.2013</td>
<td>Complete document</td>
<td>First publication of the document (Basic Concepts for modeling of rail topology)</td>
</tr>
<tr>
<td>0.2</td>
<td>RTM Workgroup</td>
<td>28.07.2014</td>
<td>Complete Document</td>
<td>Final modeling, including most infrastructure concepts to support main railways activities</td>
</tr>
<tr>
<td>0.3</td>
<td>RTM Workgroup</td>
<td>07.10.2014</td>
<td>Complete Document</td>
<td>Corrections and extended explanations in some paragraphs</td>
</tr>
<tr>
<td>1.0</td>
<td>RTM Workgroup</td>
<td>27.04.2015</td>
<td>Complete Document</td>
<td>Enhance readability, moved &quot;Predefined Description Levels&quot; into Chapter “Methodology”; moved &quot;Aggregation&quot; and &quot;Disaggregation&quot; into Chapter “Appendix”, resolved minor spelling issues</td>
</tr>
</tbody>
</table>
1. Executive summary

The advantages and benefits associated with the adoption and exploitation of a standardised data definition model and data exchange format are widely recognised within many industries.

The rail sector is no exception in the recognition of this; to this end there is drive and desire within the industry to develop and implement a functional and scalable data modelling and exchange format in order to drive and sustain growth and innovation, as well as streamlining everyday business.

To date, a number of specialised formats have been created for both rail and non-rail focused purposes, such as RINF (Register of Infrastructure - railway infrastructure description), ETCS (European train Control System) and INSPIRE (Infrastructure for Spatial Information in the European Community). The design of each format primarily attempted to address only the specific requirements inherently relating to that given discipline. As such, this may dictate that the potential for the transference, reuse, or adaptation of the data format is either limited or non-existent.

Ideally, a single standardised data exchange format would exist, which would:

- Cut duplication of efforts and encourage more collaboration,
- Prevent laboured and repetitive developments in IT,
- Reduce lengthy IT project development phases,
- Enable innovation, and
- Improve compatibility by reducing overlap and redundancy issues.

The RailTopoModel is a logical model to standardise the representation of railway infrastructure-related data. Together with railML®, which defines the schema for the exchange of data, it will revolutionise the transfer of data. This will help the rail sector become a competitive market with the fast and efficient exchange of data between companies, with their industrial suppliers, or with railway regulators and other authorities.

The RailTopoModel depicts the network topology, i.e. the arrangement of the iron network and associated infrastructure data, and its relationship with objects and events located along the network. Understanding the topology is essential, as it defines how the network is connected and how objects and events depend upon each other.

The RailTopoModel is the backbone which defines how objects data should be modelled and described, aggregated, stored and visualised. Once fully implemented, it will be able to:

- Connect, locate and visualise the topological relationship of the iron network and the assets located along it;
- Describe the characteristics of the network, its components and events, according to various attributes at multiple scales: from the micro view of the tracks up to the macro view of lines and corridors;
- Support evolutions related to generic or specific business needs (traffic management, interlocking…).

The RailTopoModel has been devised by the UIC (International Union of Railways) European Rail Infrastructure Masterplan (ERIM) Task Force and the railML® consortium. These innovative
collaborators bring together expertise from a diverse range of backgrounds, including industry and academia. They were able to bring to the fore knowledge of:

- Railway principles for the successful translation of fundamental railway concepts into the model. It is pivotal to ensure the model accurately represents the rail system and guarantees the safe operation of infrastructure and rolling stock. Concepts from a range of fields were included, such as:
  - Asset and information management; and
  - Engineering disciplines such as track, energy and signalling.
- IT principles for the effective exchange of data.
- Mathematical principles for the creation of a model which is capable of representing a simplified abstract view of reality at the required level of detail.

The integration of these concepts makes a model of the rail network extremely powerful. It supports and complements multiple modelling activities undertaken on a daily basis in the rail industry, and yet is simple for technical and non-technical users to interpret. The RailTopoModel has the potential to form the backbone of data transfer across Europe. It will standardise and encourage the development of new applications that can utilise accurate data made readily available by RailTopoModel and railML®.

The adoption of the RailTopoModel as an international railway standard is scheduled for 2015. In the meantime, the present document provides insight into the model, its current state and its possible evolution.

2. Introduction

2.1 Background

One of the greatest challenges for today’s railway sector is to establish a format and mechanism to transfer data both internally across an organisation and externally between organisations. This has arisen from the lack of a standardised data exchange format and a single industry wide approach.

To date there has been little coordination or consensus within the railway community over a standard for the exchange of data. Thus multiple standards have been developed for specific purposes, each with their own data definition (model) and file format.

Refer to Feasibility Study:

The consequences of this have been:

- Laboured and repetitive developments in IT,
- Long project lead times, and
- Incompatibility between different standards which has prevented the development of transformation software in a competitive market.

As such, each data model and format cannot be used for other purposes. Examples include formats designed for RINF, INSPIRE, ETCS projects, etc. Figure 1 illustrates the current situation in multiple countries.
The vision for a standardised data exchange process requires a number of components:

- A logical model: to describe the topological relationship of infrastructure objects and their attributes.
- An exchange format: to represent objects within a model as structured data, typically in text format with a defined schema.
- An adaptor / translator: to restructure data from one format to another.
  - Translators can be used to convert the output of platform specific data to a standardised format which can then be shared more readily with other applications.

Together these components will provide a data exchange tool that can facilitate the efficient transfer of data within the rail sector. They will allow users to exchange tabular and geographical data related to all aspects of the rail sector from infrastructure description and status, interlocking and routes, timetabling and traffic control etc. using a standardised format.

Considering the work done by the railML initiative project in co-operation with this modelling work, there are currently two products available to facilitate the exchange of data in the domain of railway infrastructure.

<table>
<thead>
<tr>
<th>Logical model</th>
<th>The RailTopoModel is a generic railway data model designed to support current and future business needs. It is particularly useful for:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>- Engineering activities – mainly based on installations and components, and</td>
</tr>
<tr>
<td></td>
<td>- Circulation activities – mainly based on routing and scheduling.</td>
</tr>
</tbody>
</table>

| railML        | railML 3 is the latest evolution of the format created by railML.org. RailML 3 was specifically developed to compliment the RailTopoModel. |

Thus, railML can be viewed as the first benefit of RailTopoModel. Figure 2 summarises the role that RailTopoModel and railML would play when fully integrated with existing systems.
Investing in a standardised railway data exchange format will provide multiple benefits for the sector, including:

- Improved data quality,
- More efficient business performance,
- Streamlined and re-usable development,
- Integrated IT systems, and
- Return on investments.

Detailed information about railML can be found on the railML website at http://www.railml.org/.

### 2.2 Objectives

This document provides a brief overview of the RailTopoModel and the rationale behind its creation.

The aim of this work is to:

- Transform the physical view of the railway (the reality on the ground) into a model with sound mathematical foundations, which accurately represents the rail network.
- Define the specification for a fully scalable logical data model which, as a first operational deliverable, will form the base for a common exchange format.
2.3 Scope

The emphasis of this document relates to RailTopoModel, and will explore how the railway infrastructure model can be used to formulate and manage data which describes the network topology and installations with a spatial dimension.
3. Methodology

3.1 Model requirements

The ultimate goal is to propose a standardized infrastructure master model which supports a common representation of a railway network and events, and facilitates the exchange of data within the rail sector.

For this purpose, UIC proposes the use of a graph topological model, as such a model is commonly used to display networks for a range of sectors, including the railways. One of the main reasons that such a model has been so widely adopted is that it is systemic, i.e. it is independent of any particular use or application. This choice guarantees sustainability and scalability, meaning it can evolve as business needs change. It also ensures the integrity, quality and dimension of data is not compromised due to the usages and evolutions.

The first objective is to ensure that this model supports the railway’s business needs, today and tomorrow. In order to achieve this, the model must fulfil the following criteria:

- Provide a topological representation of the iron network which is fully connected and can be visualised schematically. It must display the track location at the most detailed level and be able to view the connections which exist at other scales (levels of detail) such as line and corridor.
- Enable data to be aggregated and disaggregated, to ensure consistency is retained across all scales.
- Allow permitted routes to be identified, based on network topology and the location and rules of signaling assets etc.
- Support multiple referencing systems, ensuring consistency during transformation. Primary examples include:
  - Linear referencing – using mileposts and ‘rail addresses’,
  - Geographic Positioning Systems (GPS),
  - Screen (schematic) coordinates,
  - Etc.
- Locate point and linear entities, including:
  - Points / nodes, such as any installation and equipment or event etc.
  - Lines / edges, such as speed limits, slopes, platforms etc. (attributes which are the same along a linear feature).
  - Areal objects, such as track circuits, tunnels etc.

Finally, it is important to future proof the model. In other words this model is designed to be enriched progressively, per layer, with new concepts to support business usages as they evolve.

3.2 Feasibility study

Before launching the RailTopoModel project, a feasibility study was performed as part of the ERIM project. The results were presented at the UIC in Paris on 17 September 2013 and then published – http://documents.railml.org/science/201213_UIC_RailTopoModel_DraftDec13.pdf

The output of this work was a schema for an ‘off the shelf’ network model which describes the topology and basic elements of a railway’s iron network, and related assets such as track, signals etc. The model, or graph, should be designed to be independent of any particular usage, and can therefore be used for multiple applications.
The vision is for this model to be adopted by users from across the rail sector to design future applications, and as a first deliverable to support the exchange of data within and between organisations.

The study found that it was indeed achievable, and put together a road map for successful implementation.

### 3.3 Overview

The aim of any modelling approach is to create an abstract representation of reality. Put simply, it needs to enable users to understand the following:

- **What**: the ‘physical view’ of what can be viewed in the field.
- **Where**: the location of assets.
- **How**: the connections between neighbouring assets.
- **When**: the life time of assets
- **Why**: the business rules which dictate how the infrastructure operates.

The ‘where’ and the ‘how’ comprise the building blocks to this model, i.e. they are the independent foundation layers on which the model is built. The ‘why’ drives the model’s application, and helps utilise the information in the foundation layers to identify how the network operates.

Together this data provides users with a greater understanding of the ‘what’, and gives them a system overview of the network, especially when the aggregation tools are utilised to understand how the network behaves at different scales, namely:

<table>
<thead>
<tr>
<th>Description</th>
<th>Use cases / examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nano</td>
<td>Very large scale – description of the switch design</td>
</tr>
<tr>
<td>Micro</td>
<td>Large scale – track level information at the highest level of detail. Basis = Switches or buffer stops that are connected by tracks</td>
</tr>
<tr>
<td>Meso</td>
<td>Intermediate scale – reduced track level information. Basis = Operating points that are connected by one or more tracks.</td>
</tr>
<tr>
<td>Macro</td>
<td>Small scale – minimal track level information. Basis = Connected operating points with one connection (one or more tracks).</td>
</tr>
</tbody>
</table>

These common scales are proposed to provide a standard network description with varying levels of detail following common railway practices, and recent sector wide applications such as RINF. They are linked by aggregation rules.

Depending on business needs and/or maturity level, data can be entered at any scale in the RailTopoModel, without lower levels of detail if they’re not required, but ensuring consistency with future evolutions at other levels.
The relationships between all these drivers is summarised in Figure 3, and section 4 looks at each in more detail.

Figure 3: conceptual overview of RailTopoModel and its functionality

3.4 Predefined Description Levels
As said above, there are four predefined levels of detail. As the model describes a generic network and as every detailed level shares the same concepts, an infinite number of other levels may be derived.

The four predefined levels of details were identified as being the most common anchor points to publish data.

3.4.1 Nano Level
The Nano Level identifies what happens inside a switch or crossing. It is where the distinction may be made between different types of switches.

For example, a single switch crossing may be modelled this way in Nano level:

Figure 4: Single switch example

While a double switch crossing may be this
Typically, this level will be built starting from the micro level, by using “switch templates”.

### 3.4.2 Micro Level

This level defines the network in the nearest way to the physical level as commonly viewed.

**Figure 6: Micro level sample**

Its nonlinear elements are the Switch points, the Network borders, maybe some administrative points (ownership boundaries), buffer stops, ... 

Its linear elements are the rails connecting those NonLinear Elements.

We will call those rail sections track edges.

### 3.4.3 Meso Level

Describes the tracks in the network.

**Figure 7: Meso level sample**

Its NonLinear Elements are the Operating points (stations, yards, boundaries) and its linear elements are the tracks connecting those elements, which will be called SectionOfTracks

### 3.4.4 Macro Level
The Macro Level aims to describe the network at Regional or National level, with the NonLinear elements being the Boundaries and the major OP’s while the Linear elements are the section of lines connecting those points.

3.4.5 Navigability view

As said above, this network carries the information on how the railway network behaves as a mathematical network allowing routing operations. One navigability view may be defined for every level, but as this view may be computed by aggregation, only the lowest level view has to be explicitly defined.

However, when transmitting data about a single level, the navigability view for this level has also to be transmitted.

3.4.6 Other Levels

While those four levels are predefined, the model allows to build an infinite number of other levels, or an infinite number of paths to aggregate from one level to the other.

For example, the European High speed lines form a network which can be aggregated from the macro network of the European countries.
Figure 10: Aggregation Paths

- Figure 10: Aggregation Paths shows how to navigate along a predefined path: Micro is defined in function of Nano; As shown in 0, the aggregation mechanism takes place in two steps. First, we split elements into elements parts, then we combine the different element parts in upper level NetElements.

  The virtual intermediate level of “ElementParts” in this illustration is called “X for Y”, shown in light green.

- Meso is defined on Micro,
- Macro is defined on Meso.

However, while those four predefined levels are quite common among IMs, it is still possible to start from other network definitions, using different segmentation conventions (for example, defining the most basic network with the signaling blocks, not the iron network). It is also possible to split from the main path by defining networks using different segmentations (for example, a grid/intergrid structure).

While these networks do not follow the predefined path, they still can be expressed as networks using the RailTopoModel structure.
4. Concepts

The analysis of different approaches in network modeling confirms the mathematical graph as the most open and sustainable model to describe a network. It is also the case for the railway network and supports the requirements of the sector for both description and usages of a railway network.

4.1 Core elements

The first thing we have to do is to list the elements comprising the core of the network. Each of those elements must have some basic characteristics: an unique identifier, a name and a validity range. So we begin by creating a generic class containing those data.

According to the view level the network is composed of

- parts of rail, switches, buffer stops, … or
- parts of tracks and operating points, or
- parts of lines and major operating points or
- …

Each of those being uniquely identified (ideally with an UUID, but any unique national identifier is allowed) and given a temporal validity range.

This validity range is the first step in time dimension and versioning management, which will be extended in future versions of this model.

As the model aims to be valid through all detail levels, a neutral name has to be given to those elements forming the network: the “NetElements”.

![Diagram showing class hierarchy](image)

Figure 11: Model part 1

The common point between all those levels is that we have to consider two types of elements:

- Linear elements, that have a direction (a start point and end point) and length,
- Non-Linear Elements, which do not have a clear orientation, or length. In our model, we will consider all those non-linear elements as points.

Up to now, our model makes no distinction between those two types of elements, because they share so many characteristics.
It means that, if we see this piece of network in the reality:

![Figure 12: Sample piece of network](image)

We will, up to this point model it that way:

![Figure 13: Network element view](image)

A mere list of objects (symbolised here by dots).

This way of thinking (considering the linear and nonlinear elements as the same kind of objects) will prove especially useful when we will talk of aggregation on chapter 7.

These figures are valid throughout all the levels. They can represent either:
- The inside of a switch (Nano level)

![Figure 14: Nano level example](image)
- a switch (viewed as a single functional piece: Micro level)
- a station (viewed as a single functional place: Operating Point at Macro level)

(For B = Belgium):

**Figure 15**: Micro level example

**Figure 16**: Macro level example

Superseded and replaced by IRS30100
Most of the samples found in this document will be based on the Micro level. The same principles are always applicable to other levels, predefined or not.

4.2 Referencing

Now that we have listed the elements, we can move on to our next goal, which is to provide a way to place items on these elements. A Referencing System.

As there are a great many referencing method available, and that we will have to deal with many sources that are using different types of units or reference point, we have chosen a unitless and universal method as basic structure. However, we will later on add means to use any other referencing method, as long as the reference system is described.

For linear elements (which have a start, end and length) we will express the position as the distance from the start to the application point. To avoid conflicts between different unit types, this distance will be expressed as a percentage of the length of the element. For example, something right in the middle of the element would receive a coordinate of 0.5.

For non-linear elements, the position will always be at the “start” of the element (coordinate 0, as the element has no length).

It implies that, in our model, an item may be located anywhere “along” a linear element (coordinate ranging from 0 to 1) but only “at” a nonlinear element (coordinate ranging from 0 to 0… that is 0)

---

1 See Reference and positioning systems
As we have a universal reference system, any object can be attached to the network with a coordinate “Element, Position”. However, if this reference system is univocal, it is unreadable by a human. A conversion to more widely used coordinate systems will be discussed in the annex to this document.

In terms of class diagram, we extend the Generic NetElement class with a class in which the intrinsic coordinate system is defined. The only role of this class is to know that the NetElement is oriented (linear) or not (we will discuss the role of the CompositionNetElement class later on).

While universal and deterministic (no two different locations on the network can point to the same coordinate – no single location on the network can have two coordinates), this referencing system is strictly logical and not human-usable. The link to Geographical/Geometrical or the more widely used Mileage system will be discussed later in this document.

4.3 Topology

Now that we have listed all the (basic) components of our network, and given them their own referencing system, we can continue with their interaction. Understanding that interaction, will enable us to use those components as a network.

So, if we see the network as a graph, we must express that NetElements are related to other NetElements (as most of the objects in this model, this Relation object has an identifier and a validity range).
Two Elements sharing the same relation are somehow linked.

Looking back at Figure 13: Network element view, our piece of network becomes:

With the relations drawn in black.
This drawing can be "read" as:

- B is related to 1, 2 and 3
- 1 is related to A and B
- ... 

But that’s not enough:
When we deal with a linear object, this relation may occur at both end of the element. In order to determine which end is linked to the other element, we will expand the relation with a class hosting the position where the relation occurs. Therefore, we will use the existing intrinsic positioning system.
According to the model, a relation between two elements may only happen at an element extremity. So, we force the value of the position of the relation to 0 or 1 (begin or end).

So, if we look at the relation “R1” on Figure 23, Its attributes will be:
- ElementA : element named A
- PositionOnA : 0 (non-linear element)
- ElementB : element named C
- PositionOnB : 0 (start of the element)

If we look at the relation “R2” on the same Figure, its attributes will be:
- ElementA : element named C
- PositionOnA : 1 (end of the element)
- ElementB : element named B
- PositionOnB : 0 (non-linear element)

We now have a graph expressing the connectedness of the elements, which means we describe that each element is related (is connected to) to one or several others.

But that is not enough to carry routing information.
Let’s look back to the model, and imagine it represents some sections of rail connected by a switch, as in Figure 21.

If we add navigability information to the relations, it will show:

![Figure 24: Navigability between elements](image)

Meaning:
- It is possible to go from A to 1 (and vice versa)
- It is possible to go from 1 to B
- It is possible to go from B to 3
- It is possible to go from B to 2
- ...

We can express these relations more compactly in the form of a matrix:

```
<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>1</th>
<th>B</th>
<th>2</th>
<th>C</th>
<th>3</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

The physical relations of Figure 24 are stored according to the RailTopoModel. Optionally they can be presented in a matrix view:

- “Physical” relations matrix

The physical view is not very well suited for a specific class of usecases like routing, timetabling or simulation.

With physical relations matrix alone, it is impossible to tell that, through B, it is not allowed to go from 3 to 2, as there is no direct relation between 3 and 2.

In fact, all the connections to and from a node are always navigable, as it means that we can “enter” the node from there.

So, to this “Physical” level, we have to associate a “Functional”, or routing level describing how the elements act together as a routable network, telling us how to cross the nodes.

As we want to know the relations between the objects **through** one node, we will only describe the interactions between the elements related to that node. We would then see the relations between the following elements:

---

*Superseded and replaced by IRS30100*
Many existing datasets covering the physical view contain information objects which are not essential for routing algorithms. These objects increase the address space as well as the runtime of algorithms. The table below shows the most common nodetypes of railway networks at the micro description level.

<table>
<thead>
<tr>
<th>Micro level nodetypes</th>
<th>Number of neighbours</th>
<th>Type of neighbours</th>
</tr>
</thead>
<tbody>
<tr>
<td>bufferstop</td>
<td>1</td>
<td>trackEdge</td>
</tr>
<tr>
<td>borderpoint</td>
<td>1</td>
<td>trackEdge</td>
</tr>
<tr>
<td>switch</td>
<td>3</td>
<td>trackEdge, adjacent switch or crossing</td>
</tr>
<tr>
<td>Double Diamond Crossing</td>
<td>4</td>
<td>trackEdge, adjacent switch or crossing</td>
</tr>
<tr>
<td>Single Diamond Crossing</td>
<td>4</td>
<td>trackEdge, adjacent switch or crossing</td>
</tr>
<tr>
<td>crossing</td>
<td>4</td>
<td>trackEdge, adjacent switch or crossing</td>
</tr>
<tr>
<td>turntable</td>
<td>depending on construction</td>
<td>trackEdge</td>
</tr>
<tr>
<td>Transfer table</td>
<td>depending on construction</td>
<td>trackEdge</td>
</tr>
</tbody>
</table>

All those elements above can be removed from the dataset provided that the linear elements (pieces of track between two nodes aka trackEdge) carry the necessary attributes for navigation.

In the RailTopoModel the navigability information can be stored in an optimal way at the PositionedRelation between the edges of the respective level.

If there exists a connected dataset with the NetElements and Relations below the current description level, navigability can be deduced from this information. The micro level data without navigability information together with nano level data containing the topological structure of each switch, crossing, turntable or transfer table allow to transfer navigability information to the micro level. After removal of the superfluous nodes the resulting subset of micro level data consists only of trackEdges with the relevant navigability information which in turn is relevant to routing algorithms.

In case the detailed nano level information does not yet exist, it is also quite easy to add navigability information manually directly to the trackEdges.

The RailTopoModel provides the necessary structures to produce a condensed relation matrix out of the detailed physical relations matrix. If there exists a detailed nano level description of each switch connected to the micro level – as is the case in some pilot projects of early adopters of the RailTopoModels – then full automation is achievable.

The same principle holds true between the macro and the micro level. This again is a direct consequence of the elegant recursive nature of the RailTopoModel.
Meaning:
- 1 is related to 2, and this relation is navigable,
- 1 is related to 3, and this relation is navigable,
- 2 is related to 1, and this relation is navigable,
- 2 is related to 3, and this relation is NOT navigable (I),
- ...

As we describe the relations through the nodes, the nodes themselves will not be encompassed in this description, resulting in a network with only linear elements.

While this routing view exists for all detailed levels, it only has to be described at the lowest level, as the other routing views are a direct result of aggregation.

However, when transmitting data for only a specific level, both physical and routing views for this level have to be included.

We have now created a network of connected elements. It is modeled in a way where graph theory algorithms are applicable, and we can locate items on it.

So bridging the gap between the two worlds of "railway builders" and "railway operators" is possible and a secure and correct flow of information between all information systems becomes a reality.

4.4 Finalisation of the Elements

Now that we have defined the core elements of our network we can finally differentiate between the linear and non-linear elements which can be further extended to suit a particular level's need.
There are 4 predefined levels, each containing its set of Linear and NonLinear objects. For example, at the macro-level, the linear objects will be the sections of lines, and the NonLinear elements will be the major operating points.

4.5 Business

4.5.1 Objects and entities

4.5.1.1 Concept explication

This layer allows the user to project objects and entities on the topology, by referencing topology elements. Three types of generic entities, that can be located on the network, were identified:

1. Spot entity: Happens at a point (e.g.: signals, buffer stops, ETCS balise, etc.)
2. Linear entity: Happens along a path (e.g.: platform, speed limit, ballast renewal, etc.)
3. Areal entity: Happens on a sub network (e.g.: catenary cases, track circuits zones, switches, stations, tunnels, bridges, etc.)
One entity can have more than one location.

### 4.5.1.2 Functional view

Figure 29: Object types

### 4.5.1.3 Comments

To define all types of entity, a generic class is defined: **NetEntity**. This class defines a general relation with a shared location to define the position on the network. The location of each entity on the network is defined using the intrinsic reference of net element. Thus, three types of entity are defined:

**Spot entity**

A spot entity can be either a nonlinear element (happens "in" the element) or an entity along a section of rail/section of line at a certain distance. If it happens IN the element, it encompasses the whole nonlinear element. It should be described as track edge X at 0 (or 100%), meaning it happens at the elements centroid, but only on this particular track edge.

If it happens ALONG a track edge at a certain distance, it is necessary to defined whether this entity happens directly on the track edge, or alongside the track edge; and if the entity occurs in both directions or only in one.

These types of entities are located on the network by the type of class: **SpotLocation**. It is possible to have more than one location for a punctual entity:

- Multiple location (on multi-track) on the network for the same level (micro,…).

- Multiple location because the entity is located on different levels.

**Linear entity**

A lot of entities can be seen as paths.
To describe a path location, a start point, an end point and an ordered list of every encompassed element, is needed.

The start and end points are point locations, thus either a nonlinear element or a point along a track edge.

The path should be a topological path, meaning that at each nonlinear element only one exit direction can be taken.

If the start- or end point is a nonlinear element, this element is fully encompassed within the path.

The correct ordering of the list should be checked against the topology definition of the network. This could be the task of a validator. The orientation of the path on each track edge can be deduced from the order of the crossed nonlinear elements.

This type is located by the type of class: **LinearLocation**. The linear location, in this case, is defined by an ordered set of net elements with a position (in intrinsic reference) for the first and last element.
Areal entity (SubNetwork)

We call it an areal object with an analogy to the geometric object, but it defines in fact a subgraph (a continuous fraction of the network).

It is defined in the same way, by listing the ElementParts member of the location.

If a nonlinear element is included in a collection, then the whole element is included. If only one track edge is concerned, then the entity should be referenced to the track edge at position 0% (or 100%) to show it happens only on this track edge. It must be checked that the objects included in the areal location form a connected subgraph (no independent parts) and that no object is listed twice.

This type is located by the type of class: AreaLocation. The area location, in this case, is defined by a set of net elements with a pair of positions (beginning and end in intrinsic reference) for all elements.
4.6 Finalisation of the model

In order to finalise the model some classes are added:

- **Network**: This class defines the railway network. It allows us to define the graphs of different levels of description of the network.
- **LevelNetwork**: Groups all the elements (and relations, reference systems...) of a given level. The same element, relation, etc. may be valid for more than one level.
4.7 UML Diagram

The complete RailTopoModel is available at http://www.railml.org/.
4.8 Reference and positioning systems

4.8.1 Concept explication

The localization of an object or entity on the railway network is based on a reference system. The first version of the model includes the three most common reference systems in the railway sector:

- **Linear positioning system:**
  localization expressed in a linear reference system, generally a coordinate along a predefined axis.

  ![Linear positioning system diagram]

  It could be read as “at coord X on the line Y” (being aware that the line Y is composed of many elements)

- **Intrinsic positioning system:**
  The coordinate system associated with each element. In order to be the most generic possible the value 0 is assigned to the start and 1 to the end for a linear element on the network. Non linear elements don't need the notion of a number between 0 and 1 since all the segments correspond to the same place on the network: the non linear element itself (= 0).

  ![Intrinsic positioning system diagram]

  It could be read as “At coord X after the start of the element”

- **Geometric positioning system:**
  (Geometrical coordinates)

  ![Geometric positioning system diagram]

  It could either be a projected X,Y,Z coordinate, a λ,ϕ,h geodetic coordinate or even an X,Y,Z schematic plan coordinate.

  It could be read as “at coord X,Y,Z in the system [EPSG:xxxx] [01/06/2014]”

---

5 The length of the element could also be used as “end coordinate”, but we would then have to describe the “intrinsic coordinate system” (units, source, timeframe,…) in order to be able to merge different datasets.

6 The exact content of the definition string of the geometric system is not yet definitive.
The model integrates a link between the linear positioning system and the intrinsic positioning system. It supports the "conversion" of an element position associated with a number between 0 and 1 to a kilometric point on its reference system.

4.8.2 Comments

The classes we will use to define the location and positioning systems are:

- **PositioningSystem**: This class defines the general concept of a Reference System.
- **LinearPositioningSystem**: This class defines a Linear Referencing System. It defines a starting and an ending coordinate.
- **LinearAnchorPoints**: this class describes the reference points in the linear reference system (Milestones, anomaly points...) and their characteristics.
- **LinearCoordinate**: This class defines the localization expressed in a Linear Reference System (LRS).
- **GeometricPositioningSystem**: This class defines a Geometrical Reference System, so it allows to localise a resource with his geometrical coordinates \((x,y,z)\) or \((\lambda,\phi,h)\).
- **GeometricCoordinate**: This class defines the localization expressed in a geometrical (or geographical) Reference System, so it defines the coordinates \((x,y,z)\) or \((\lambda,\phi,h)\).
- **PositioningSystemCoordinate**: This class represent a coordinate in either a geometric or linear reference system.
- **IntrinsicCoordinate**: This class allows to associate an intrinsic coordinate to another coordinate, either geographic or linear.
- **AssociatedPositioningSystem**: This class allows to group couples of coordinates to define the translation parameters between an external (geometric or linear) coordinate system and the element's intrinsic coordinate system.
5. Conclusions

5.1 Integrated overview
This document has introduced a number of concepts that are utilised in the creation of the RailTopoModel.
The complete UML diagram found at http://www.railml.org/ combines all concepts and shows how the
features within the model interact.

5.2 Functionality under development
The model will be progressively enriched to support business usages.

The next version will include
- Multiple times (horizons, creation and modification dates, validity dates, …)
- Projects, versions and life cycle management
- Requirements for Signalling and Interlocking
- …

Following versions will also include:
- Operational events
  - Traffic Management (commercial routes, …)
  - Asset Management
- …

5.3 Next steps
This version of the document should give answers to the requirements of most general usages.
This work has been made a priority to fulfill the needs for railML Organization to enable them to
develop the next version of the standard exchange format railML 3.0.
The future enhancements will follow the requirements and priorities of the community. The short term
enrichments will be driven by the Interlocking-signalling requirements to support route design and
programming including programming of equipements.

Your comments and requests are welcome to drive the future roadmap of RailTopoModel.
6. Appendix

6.1 Glossary

The following table gives the definitions of major terms used in this document:

<table>
<thead>
<tr>
<th>Term</th>
<th>Explanation / Definition</th>
<th>Comments</th>
<th>source/reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Balise</td>
<td>A balise is an electronic beacon or transponder placed between the rails of a railway as part of an Automatic Train Protection (ATP) system.</td>
<td></td>
<td><a href="http://wiki.railml.org/index.php?title=IS:balise">http://wiki.railml.org/index.php?title=IS:balise</a></td>
</tr>
<tr>
<td>Buffer stop</td>
<td>A buffer stop (or bumper) is a device to prevent railway vehicles from going past the end of a physical section of track.</td>
<td></td>
<td><a href="http://wiki.railml.org/index.php?title=IS:bufferStop_trackBegin">http://wiki.railml.org/index.php?title=IS:bufferStop_trackBegin</a></td>
</tr>
<tr>
<td>Canonical data model</td>
<td>A common-data-exchange-model.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Catenary</td>
<td>Overhead wiring.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class</td>
<td>A class is an extensible template for creating objects, providing initial values for state (member variables) and implementations of behavior (member functions, methods, …)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Crossing</td>
<td>A &lt;crossing&gt; defines parameters of diamond crossings. A diamond crossing is defined as a level junction of two tracks without the possibility of changing between these tracks.</td>
<td></td>
<td><a href="http://wiki.railml.org/index.php?title=IS:crossing">http://wiki.railml.org/index.php?title=IS:crossing</a></td>
</tr>
<tr>
<td>Derailer</td>
<td>A derailer is a device used to prevent fouling of a rail track by unauthorized movements of trains or unattended rolling stock. It works (as the name suggests) by derailing the equipment as it rolls over or through the derail.</td>
<td></td>
<td><a href="http://wiki.railml.org/index.php?title=IS:derailer">http://wiki.railml.org/index.php?title=IS:derailer</a></td>
</tr>
<tr>
<td>Edge</td>
<td>Edge classes represent orientated links between nodes. An edge has two relations with nodes, From and To.</td>
<td>E.g. track edge</td>
<td></td>
</tr>
<tr>
<td>Event</td>
<td>In linear referencing an event represents the geographic location of a point or linear feature expressed as a measure along a line.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Graph</td>
<td>A graph is a representation of a set of objects where some pairs of objects are connected by links.</td>
<td>A graph is called “not oriented” as long as the information on the direction of the connection between two elements is not given.</td>
<td><a href="http://wiki.railml.org/index.php?title=IS:levelCrossing">http://wiki.railml.org/index.php?title=IS:levelCrossing</a></td>
</tr>
<tr>
<td>Level crossing</td>
<td>A &lt;levelCrossing&gt; defines a crossing between a railway track and a non railway track (e.g. road, path).</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Line</strong></td>
<td>Is a continuous chain of sections of line and operational points, starting and ending at an operational point. It can also be just one section of line with an OP at each end.</td>
<td>A line is an aggregation of continuous interconnected OPs and SOLs. Thus it can be seen as an area-entity, which possibly have common characteristics.</td>
<td>RINF_EIM-CER Key Notions V2013-6</td>
</tr>
<tr>
<td>----------</td>
<td>-------------------------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td><strong>Linear Element</strong></td>
<td>Element with a length</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Linear referencing</strong></td>
<td>A method for identifying a relative position or measure along a measured linear feature.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Link</strong></td>
<td>See track edge.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Macro</strong></td>
<td>Macro level means that overall railway network defined by sections of line and operational points.</td>
<td>The macro topology is the graph representation of the railway network on the level of operational points and sections of line. In this graph, operational points are nodes and sections of line are edges.</td>
<td>RINF_EIM-CER Key Notions V2013-6</td>
</tr>
<tr>
<td><strong>Meso</strong></td>
<td>Intermediate level between the macro and micro level. It is a generalised view of the micro level.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Micro</strong></td>
<td>Micro level represents the rail network on the level of switches and buffer stops and their connecting rail sections.</td>
<td>The micro topology is the graph representation of the rail network on the level of switches and buffer stops and their connecting track edges. In this graph, switches or buffer stops are nodes and track edges are edges.</td>
<td>RINF_EIM-CER Key Notions V2013-6</td>
</tr>
<tr>
<td><strong>Model / Data model</strong></td>
<td>A data model explicitly determines the structure of data</td>
<td></td>
<td><a href="http://en.wikipedia.org/wiki/Data_model">http://en.wikipedia.org/wiki/Data_model</a></td>
</tr>
<tr>
<td><strong>NetElement</strong></td>
<td>The expression of a linear event in the model.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Node</strong></td>
<td>A point at the start or end of a line. Nodes connect linear features and ensure that the topology is correct. E.g. switches and buffer stops</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Non-Linear Element</strong></td>
<td>Element without dimension (point, operational point, etc.)</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Object</strong></td>
<td>An entity, which can be either spatial or non-spatial, whose characteristics within the model are defined by its class.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Operational Point</strong></td>
<td>An operational point is any location for train service operations, where train services can begin and end or change route, and where passenger or freight services are provided. An operational point may be any location where the functionality of basic parameters of a subsystem changes.</td>
<td></td>
<td>RINF_EIM-CER Key Notions V2013-6</td>
</tr>
<tr>
<td><strong>Platform edge</strong></td>
<td>A <code>&lt;platformEdge&gt;</code> defines the border line between platform and railway track, where passengers are meant to board and unboard the train. A platform edge is always connected with exactly one railway track.</td>
<td></td>
<td><a href="http://wiki.railml.org/index.php?title=IS:platformEdge">http://wiki.railml.org/index.php?title=IS:platformEdge</a></td>
</tr>
<tr>
<td><strong>Rail section</strong></td>
<td>See track edge.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Section Of Line</td>
<td>A section of line is the uninterrupted connection by rails between two adjacent operational points.</td>
<td>A section of line can be made of more than one track in parallel.</td>
<td>RINF_EIM-CER Key Notions V2013-6</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------------------------------------------------------</td>
<td>-----------------------------------------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>Service section</td>
<td>A <code>&lt;serviceSection&gt;</code> defines the border line between a railway service area and a railway track, where the service for the train takes place.</td>
<td>A service section is always connected with exactly one railway track.</td>
<td><a href="http://wiki.railml.org/index.php?title=IS:serviceSection">http://wiki.railml.org/index.php?title=IS:serviceSection</a></td>
</tr>
<tr>
<td>Siding</td>
<td>A siding is a track, not necessarily demarcated like a track, which is not used for operational routing of trains.</td>
<td></td>
<td>RINF_EIM-CER Key Notions V2013-6</td>
</tr>
<tr>
<td>sub-graph</td>
<td>A graph “H” is a subgraph of graph “G” if every graph component of H also belongs to G.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>switch</td>
<td>A switch (or point) is a constituent part of the rail network with which the physical bifurcation of rails is realised.</td>
<td>Several subtypes, each with their own characteristics, can be distinguished.</td>
<td>RINF_EIM-CER Key Notions V2013-6</td>
</tr>
<tr>
<td>switch</td>
<td>A <code>&lt;switch&gt;</code> defines a standard switch with three connections as an infrastructure element. A railroad switch, turnout or (set of) points is a mechanical installation enabling railway trains to be guided from one track to another, such as at a railway junction or where a spur or siding branches off.</td>
<td></td>
<td><a href="http://wiki.railml.org/index.php?title=IS:switch">http://wiki.railml.org/index.php?title=IS:switch</a></td>
</tr>
<tr>
<td>topology</td>
<td>The rules which define the relationship between adjoining features which ensure data integrity is achieved.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>track</td>
<td>A track is (part of) a trail demarcated by two valid track boundaries and named by a unique track code within an operational point or section of line.</td>
<td>See track edge.</td>
<td>RINF_EIM-CER Key Notions V2013-6</td>
</tr>
<tr>
<td>track</td>
<td>A <code>&lt;track&gt;</code> represents one of possibly multiple tracks (i.e. “pair of rails”) that make up a line.</td>
<td>The <code>&lt;track&gt;</code> and its child elements contain all information about the track’s topology and the trackside elements associated with that track.</td>
<td><a href="http://wiki.railml.org/index.php?title=IS:track">http://wiki.railml.org/index.php?title=IS:track</a></td>
</tr>
<tr>
<td>track edge</td>
<td>A rail-section is the direct rail connection between 2 adjacent switches or between a switch and an adjacent buffer stop. (Remark: in previous versions of this document the notion “trail” was used instead)</td>
<td>On a rail-section no interruption by another switch occurs, but there can be several kinds of objects on it like for example signals, insulated joints, derail blocks, balises etc.</td>
<td>RINF_EIM-CER Key Notions V2013-6</td>
</tr>
</tbody>
</table>
6.2 Acronyms

The following table gives the definitions of major terms used in this document:

<table>
<thead>
<tr>
<th>Definition of acronyms used throughout this document</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>CER</td>
<td>Community of European Railways Companies</td>
</tr>
<tr>
<td>EIM</td>
<td>European Rail Infrastructure Managers</td>
</tr>
<tr>
<td>ERIM</td>
<td>European Railway Infrastructure Masterplan (at UIC)</td>
</tr>
<tr>
<td>IM</td>
<td>Infrastructure Manager</td>
</tr>
<tr>
<td>INSPIRE</td>
<td>European directive to create a EU spatial data infrastructure enabling the sharing of environmental spatial information</td>
</tr>
<tr>
<td>OP</td>
<td>Operational Point (RINF concept)</td>
</tr>
<tr>
<td>RFC</td>
<td>Railway Freight Corridor</td>
</tr>
<tr>
<td>SOL</td>
<td>Section Of Line (RINF concept)</td>
</tr>
<tr>
<td>UIC</td>
<td>International Union of Railways</td>
</tr>
</tbody>
</table>

6.3 UML - Unified Modelling Language - Notation


UML consists of a large body of concepts and definitions and is positioned as a "general-purpose modeling language". UML also allows domain specific and task specific extensions. Available tools may use slightly different graphical notations.
Therefore it seems appropriate to provide some basic information of how UML is applied within the RailTopoModel project.

The RailTopoModel has been designed to be a "conceptual model to describe a railway network". Using UML terms RailTopoModel can be considered as a "Platform Independent Model" (PIM).

The model is expressed using the UML diagram type "Class Diagram". The concepts are visualised with "class" symbols.

A concept can contain properties which are visualised as "attributes". Concepts are related to other concepts. There are different categories of possible relationships.

**UML definitions:**

The definitions below are taken from the UML standard document.

Source: [http://www.omg.org/spec/UML/2.4.1/](http://www.omg.org/spec/UML/2.4.1/)

**Class:** The purpose of a class is to specify a classification of objects and to specify the features that characterize the structure and behavior of those objects.

**Classifier:** A classifier is a classification of instances - it describes a set of instances that have features in common. In the context of the RailTopoModel the terms "class" and "classifier" may in most cases be used synonymously.

**Attribute:** axiomatic term, used in the context of an UML-Class definition to specify a typed property owned by a classifier.

---

**Class:**

<table>
<thead>
<tr>
<th>BaseObject</th>
</tr>
</thead>
<tbody>
<tr>
<td>- id : tID</td>
</tr>
<tr>
<td>- name : String</td>
</tr>
<tr>
<td>- validFrom : Date</td>
</tr>
<tr>
<td>- validTo : Date</td>
</tr>
</tbody>
</table>

**Classifier**

**Attributes**

Example for Class:

The concept of a "BaseObject" is modelled as an UML-Class "BaseObject" and four attributes. Information shown for each Attributes consists of a name ("id", "name", "validFrom", "validTo") and a type (tID, String, Date).

**Association:** An association specifies a semantic relationship that can occur between typed instances. It has at least two ends represented by properties, each of which is connected to the type of the end.
Example: An instance of the class “NetElement” is related to 1 or more (1..*) instances of the class “Relation”

**Aggregation:** An aggregation is a binary association which represents a whole/part relationship.

Example: An instance of the Class “EntityLocation” is part of one or more instances of the class “NetEntity”

**Composite aggregation:** Composite aggregation is a strong form of aggregation that requires a part instance to be included in one composite at a time. If a composite is deleted, all of its parts are normally deleted with it.
Example: An instance of the Class “AssociatedPositioningSystem” is part of one instance of the class “PositioningNetElement”

**Generalization:** A generalization is a binary relationship that relates a specific Classifier to a more general Classifier. The specific Classifier inherits all attributes and relevant relationships of the more general Classifier.

Example: the classes “AssociatedPositioningSystem” and “PositioningSystem” inherit the attributes “id”, “name”, “validFrom” and “validTo” from the class “BaseObject” via class “NetworkResource” (Generalization1) and from “NetworkResource” via Generalization2 and Generalization3.
The class “LinearPositioningSystem” inherits via Generalization4 both the attributes “id”, “name”, “validFrom” and “validTo” indirectly from “BaseObject” and the unqualified relationship (association) positioningSystem to the class “AssociatedPositioningSystem” directly from “PositioningSystem”.

Example: this diagram contains identical semantic information as the diagram before in respect to the two classes “AssociatedPositioningSystem” and “LinearPositioningSystem”. The inherited attributes and associations are shown by the respective class symbols.

6.4 Aggregation (Link between levels of detail)

6.4.1 Concept explanation

As we have seen the definition of a complete network is not enough. We have to deal with several view levels, and there must be coherence between them. We also want to keep exactly the same logic (and thus the same model) throughout the different levels.

To ensure the coherence between levels, every level is built relative to another level. The levels are built by aggregating (or dividing) elements of the base level.

From a practical point of view, it means that we will group elements to make a simpler, more manageable network.

For example, if we have this part of network

![Figure 30: Micro level sample](image)

It is all very well to make accurate paths. But, if we want to share timetables, the public isn’t interested in all the different paths that can be taken.

It will be communicated under the form

![Figure 31: Macro level sample](image)
Station A, departure time, station B arrival.

Internally, the network management teams are also interested in intermediate levels, such as this one:

![Diagram showing stations A, C, B and links between them]

*Figure 32: Meso level sample*

Each of those representations form their own network, and aggregation is the way we will manage the links between those networks.

We will put links between the higher level and the lower levels.

![Diagram showing aggregation principle]

*Figure 33: Aggregation principle*

We define what is encompassed in station A, in every section of track, ...

Of course, information linked to lower level must also be transmitted to the higher levels. By aggregating, we will lose details, but we will be able to calculate the aggregated values according to rules.

These may be simple rules, such as

- Aggregated value = minimum of individual values
- Aggregated values = sum of individual values

Or complex ones, such as

- Most permissive value on every possible individual paths between two elements linked to the considered element.

All those rules have to be defined once and for all, for any type of information or entity attached to the network.

It will be the job of all the relevant workgroup to define those rules, for the objects or entities they define.

The client receiving the information is expected to use it at the most relevant level. There is no sense in computing switch occupation conflicts at the international level. You will have to switch back to the most accurate level. As we will see further in this document, the proposed model contains four predefined levels,

- Nano (inside of switches and crossings, running track)
- Micro (Track edges, switches, bufferstops)
- Meso (Track edges, OPs)
- Macro (Line sections, major OPs)
In addition to those levels, a routing view of the lowest level has to be given to describe the navigability on the network. Should no routability information be given, every movement will be considered possible (which may well be the case if we only describe the macro level).

There is an infinite number of possible uses for the network definition. **Thus the model allows for the creation of an arbitrary number of other levels**, each one a network in itself.

As they all share the same model, there are no technical problems when defining them, but it is important not forget that:

- the links between levels have to be laid and the aggregation rules have to be defined,
- that it will become confusing to manage data attached to these levels and
- that parallel lines of levels (levels derived/ aggregated from the same base level) may not be compatible.

So sticking as most as possible to a minimum number of levels would be a good option (see 3.1).

Dividing an element into smaller parts is also possible. Again, sets of rules must be identified to manage information that will be transmitted to the level below, such as:

- Constant value
- Value proportional to the length of the element part regarding the other element,
- ...

### 6.4.2 Modeling

To model the relation between levels, we define a further extension to the NetElement class: The “CompositionNetElement”, which describes the fact that a NetElement is composed of parts of elements of the level below.

How does it work?

Let’s take part of a network as an example (at the most basic level):

![Figure 34: Aggregation: Base network](image)

We want to define “Operating Points” on it to use it on a more generalised application:

![Figure 35: Aggregation: Target network](image)

The model sees the original network this way (each dot represents a NetElement):

7 To ease reading, we will keep the symbology where a green dot is a NonLinear NetElement and a blue dot is a Linear Element, although for the aggregation mechanisms, we will only consider them as NetElements, regardless of their type.
The target OP’s have known boundaries that happen to be in the middle of the tracks, so we have to split those tracks first to identify what falls inside the boundaries and what stays outside.

We will now benefit from the fact that an object may be split. For each element we have to split, we create two (or more) element parts, one from 0 to split point, the other from split point to 1 (or to next split point), effectively forming two(or more) new elements on another level.

In terms of the model view, it amounts to this: from one element, we make two (or more).

The resulting network will be depicted as follows (Each dot is an “ElementPart”, those in orange are the new ElementParts resulting from a split, those in blue and green ElementParts keeping the totality of their parent element).
It is now that considering linear and non-linear elements in the same way will come in handy: The graph theory gives us tools to aggregate nodes of a graph. As we only see “NetElements” as nodes (and not linear elements as links and punctual elements as nodes), we can easily aggregate those elements to form a new network:

Figure 41: Aggregation: Step 2 – Fusion (model view)

Back to our previous symbology (where the blue dots represent linear and green dots nonlinear) the green rectangles shows the elements that will be aggregated to obtain our final network:

Figure 42: Aggregation: Step 2 result – Target network (model view)

Which is the model view of the Figure 35: Aggregation: Target network

Figure 43: Aggregation result: Target network
In terms of modeling, we have this structure:

Where the compositionNetElement are the NetElements of .
Each of those elements is composed of one or more collection (the blue and green boxes in )
Each of those collections contain Element parts (the dots of ) and those parts are defined by 1 or two
coordinates of the parent element.
Mark that there are two types of collections defined in the model: Ordered and unordered.
The OrderedCollection defines a collection that has to be a continuous string of elements, without
orientation changes\(^8\). It allows us to compute a length and orientation.

\[^8\] The orientation of the Part needs not be the same as the orientation of the parent. The parameter KeepsOrientation allows to reverse the element in order to keep a homogeneous orientation for the collection.
For example, on this piece of network:

This red collection is an OrderedCollection: it has a start, an end and it is possible to compute a length.

The more generic UnorderedCollection only checks if all the elements are connected, but no length or orientation may be deduced.

This purple collection is an UnOrderedCollection.

It may have more than one start, more than one end and more than one length.

### 6.5 Disaggregation

It is likely most levels will be built up, from nano to macro, but it is also possible to build downwards. For example, if you have the data to do so, or if you assume that the nodes are to be disaggregated using “Templates” (a default station layout, a standard switch layout…).

This procedure seems to be very useful as a system function for capacity planning or for the fast configuration of simulation datasets in situations of inadequate or outdated base documentation.

This assumes we have the level of detail for the network\(^9\) as shown below, and that we have to build a lower level:

![Figure 45: Base network](image)

In the model view this will translate into:

---

\(^9\) Let’s assume that, for the following example, you know your network as a whole on the meso level and wish to build a micro representation for capacity planning, using a standard station layout.
If this is your “Template” OP:

It becomes:

Then, if you “fuse” the adjacent linear elements (if needed)

and obtain this:

Superseded and replaced by IRS30100
The resulting model view will be as follows:

![Figure 51: Target network](image)

The aggregation is done in two steps: first the elements are decomposed into “element parts” and then the parts are fused together to create new elements. There is thus very little difference between aggregation and disaggregation, the main difference is the purpose of the operation. While aggregation tends to simplify the network, and thus lower the number of elements, disaggregation tends to raise this number by adding detail to the network.

There are two ways of interpreting the disaggregation:
Either we see it as building a higher level, or a lower level, depending on how the created level will be linked to the existing level.

If we see it as a **Higher** level, it means that we define the objects of the new level as a function of the objects of the original level.

If we see it as a **Lower** level, it means that we add the information to the original Elements level of how they are defined as a function of the new level.

Both have advantages and disadvantages.
6.5.1 Higher level view

Disaggregation of a higher level amounts to creating an alternative path. In other words creating another “Branch” in the tree of level definitions. For example, you may want to divide your network following a station/open track rule - the “predefined path” - for path calculation purposes, and divide it into track circuits for train tracking and control.

![Diagram showing different aggregation paths](image)

*Figure 52: Different aggregation paths*

The track circuits have to be defined as a collection of parts of the elements of the base network.

As the model allows an infinite number of those branches, this is not a problem.

![Diagram showing aggregation paths with alternative path](image)

*Figure 53: Different aggregation paths (principle)*
Where it may lead to issues is if several of those alternative paths want to rejoin the main trunk.

Once again, one Element may be defined differently for different functions across the various levels which are allowed.

There is a risk that data properties calculated along different branches may lead to inconsistencies after the branches rejoin the trunk.

In the previous example, we could have defined the function of macro level both in terms of the Station/open track definition and as a collection of track circuits.

Both path give a valid network (in that it obeys the model’s rules), but the two results may not always be identical.
6.5.2 Lower level view

If you are building downwards, then you are in fact creating multiple “roots” from the trunk, adding a new definition of the elements to the “base” level. It is important to be careful and ensure that the multiple definitions result in exactly the same element at the base level.

It also means that, an element at any level may have several definitions, depending on what level is considered as the source, and that a routing level has to be built at the end of each “root”. As the roots never rejoin the trunk (else, they would be built upwards) there is no real issue concerning the data coherency. As long as the client is conscious from which root he takes the data.

For example, a station may have several definitions, whether viewed from the point of a signalman, a passenger operator.

Signalman’s view – all the tracks and switches are important

Operator view – the platforms and the yard are the main elements in this view.
The two definitions may be considered equivalent if there is an agreement of the boundaries, meaning that what is inside and outside stays the same (geographically or functionally).
7. Frequently Asked Questions

*This chapter will be progressively enriched with the answers to your questions. Do not hesitate to ask.*

7.1 Comparison with other initiatives, norms and standards

7.1.1 Transmodel

Transmodel and RailTopoModel are both Conceptual Models dedicated to transportation, infrastructure and services. The main difference between these two models is the business and functional domain of coverage.

**Transmodel** is a European standard data model for public transport, designed to cover multiple transportation means (bus, tramway, trains,...) in terms of interoperability, and the places where they meet each-other (e.g. stations, cities, towns, villages). The aim is to support operation, and more precisely schedule and plan journeys.

**RailTopoModel** has been developed for the specific needs of the railway sector, to precisely and consistently model network topology, rail infrastructure, and all railway objects and events, at any level of granularity (track, line, corridors,...).

7.1.2 Inspire

INSPIRE is a European directive which aims to create spatial data infrastructure for the European Union (EU). This will enable the sharing of environmental spatial information among public sector organisations and better facilitate public access to spatial information across Europe.

A European Spatial Data Infrastructure will assist in policy-making across boundaries. Therefore the spatial information considered under the directive is extensive and includes a great variety of topical and technical themes.

INSPIRE is based on a number of common principles:

- Data should be collected only once and kept where it can be maintained most effectively.
- It should be possible to combine seamless spatial information from different sources across Europe and share it with many users and applications.
- It should be possible for information collected at one level/scale to be shared across all levels/scales; detailed for thorough investigations, general for strategic purposes.
- Geographic information needed for good governance at all levels should be readily and transparently available.
- It should be possible to easily find what geographic information is available, how it can be used to meet a particular need, and under which conditions it can be acquired and used.
- Within INSPIRE, topology is handled in the data specification implicitly rather than explicitly, with the main reason to keep the model as simple as possible, but with the expectation that most applications will use the network data within a topological environment. There is therefore a prerequisite for “implicit topology”, where the data provided must be sufficiently clean and capable of automated topological construction within a user’s application. This concept is framed with the specific requirements, including data quality information.

RailTopoModel will ensure the minimum consistency with INSPIRE specifications, including geographical positioning and dimensions.
7.1.3 Main norms and standards in transportation

**IFOPT**: An approved Technical Standard for location referencing in public transport.

**SIRI**: The Service Interface for Real time Information covers transit communications between centres, and centre’s transit vehicles. SIRI provides traveller information on real-time transit vehicle location, predicted transit vehicle arrival/departure time, and predicted transit trip travel time.

**TransXChange**: UK national XML based data standard for the interchange of bus route and timetable information between bus operators, the Vehicle and Operator Services Agency, local authorities and passenger transport executives, and others involved in the provision of passenger information.

**NeTEx**: Network and Ticketing Exchange protocol for communicating timetable and fares details.

**TRIDENT**: TRansport Intermodality Data sharing and Exchange NeTwork. Standard exchange data format for multimodal interoperability between RUs and service providers.

**DJPS**: Exposed interface standard for distributed journey planning systems.

7.2 UML Model vs Exchange Format, database and software design

RailTopoModel, is a universal railway business model, which aims to define railway objects and events in a standard form (UML), to show how they interact with each other, and how they are expected to be used.

As such, it aims to standardise the process for designing any business process, data structure, IT software and data flow in the railway industry.

One of the first deliverables based on RailTopoModel will be an enhanced version of the standard exchange format railML, with the announcement of railML3.0.

Other deliverables will come, as SQL Format and loader, etc.
7.3 RailTopoModel and RailML

The origin of the RailTopoModel initiative came from a small group of EIM/CER representatives involved in the RINF EU project, local ETCS work with industrial partners, requests from the Inspire EU directive, and other similar activities.

In 2012, these actors shared the fact that the whole sector is permanently facing the following two issues:

- Repetitive development of multiple flows of infrastructure data exchange with all kinds of partners
- Difficulties managing both concepts of network routes and infrastructure equipment and characteristics in a unique set of data.

This shared observation, enriched with some local initiatives, has led to the creation of a working group whose aim it was to design a robust model to support these needs, which could evolve over time.

Considering the ambition of this group and the result of that the work would be of benefit to the whole community, it was proposed that the UIC should support this initiative and make it a public repository.

At the same time, this same group of EIM/CER representatives, as contributors to RINF project, proposed ERA to enhance the RINF data model to include route topology in order to support future business use cases to “find the possible routes compatible with my train characteristics”.

In August 2012, during a RINF project meeting, ERA organised a presentation by the railML organisation to propose the railML standard as a possible solution to exchange data in the RINF project.

Collaboration between the RailTopoModel working group, and railML.org which was also facing limitation on its current solution to support network topology stemmed from this meeting.

The collaboration between these two teams will lead to the delivery of a consistent model and data exchange format.

railTopoModel and railML are two separate initiatives that, although complementary, will remain separate:

- RailTopoModel is defined as a public good, designed by the railway community to support their needs in the long term. As such it should and will remain independent of any usage.
- RailML is one use case of RailTopoModel, supported by an open source community, and driven by their interest and priorities.
References

|--------------|--------------------------------------------------------------------------------------------|

Copyright

All images, figures and tables were provided or created during the development of the RailTopoModel by members of the UIC RTM working group. All rights belong to UIC and the contributing people.