RailTopoModel - Railway infrastructure topological model
International Railway Solution to be classified in volumes of UIC

III

Application:

With effect from 1st September 2016
All members of the International Union of Railway

Record of updates:

September 2016  First issue

Warning

No part of this publication may be copied, reproduced or distributed by any means whatsoever, including electronic, except for private and individual use, without the express permission of the International Union of Railways (UIC). The same applies for translation, adaptation or transformation, arrangement or reproduction by any method or procedure whatsoever. The sole exceptions - noting the author’s name and the source - are “analyses and brief quotations justified by the critical, argumentative, educational, scientific or informative nature of the publication into which they are incorporated”.


© International Union of Railways (UIC) - Paris, 2016

Printed by the International Union of Railways (UIC)
16, rue Jean Rey 75015 Paris - France, September 2016
Dépôt Légal September 2016
ISBN 978-2-7461-2513-1
The International Railway Solution

The International Railway Standards (IRS) are structured in a General Part and in some eventual Application Parts.

The General Part is valid worldwide, while the Application Parts are valid for a specific railway application, based on a geographical or on a service implementation.

The eventual Application Parts may thus be added according to the current needs of the Railway Community.

Structure of the International Railway Standard:

IRS 30100 RailTopoModel - Railway infrastructure topological model

General Part

Application Part(s) : none
Foreword

The RailTopoModel Project, led by UIC with major contributions from several Railway Infrastructure Managers and Industrials companies, aims to define a universal description of railways business objects, independent of usages (usage-agnostic), structured in layers (topology, referencing, infrastructure, signalling, ..., project life cycle), and open to future evolutions. The RailTopoModel Project aims to cover progressively the complete Railways Business Objects Model.

IRS 30100 RailTopoModel is intended to be used in all business processes dealing with the design, construction, operation and maintenance of a railway network. The IRS 30100 is the foundation for quick, unambiguous and error-free data storage and data exchange inside and between these business processes.

The RailTopoModel abstracts the underlying, necessary concepts in the form of a UML2.0 class diagram.

An important part of these concepts is supported by a generic model description of the railway topology, in such a way that it applies to any aggregation level in which a railway network may be represented.

Consequently all objects are in abstract terms (classes) directly or indirectly related or connected to the topology of their appropriate aggregation level. Besides physical railway constituents, objects also refer to several kinds of characteristics of a railway. Also, the positioning of objects and instances of classes is covered by different kinds of positioning methods.

The overview below (not following UML conventions) introduces the main objects and dimensions of RailTopoModel.
functional coverage

RailTopoModel: Functional coverage

Fig. 1 - Functional coverage of the RailTopoModel

Additional explanations, application examples, documentation and wiki about RailTopoModel (current version and further developments) may be found on the web, following this link: http://www.railtopomodel.org.
Summary

The present standard, IRS30100, complements ISO 191xx series standards by specifying semantics and providing functionalities that are relevant to railway systems.

The present standard is intended to facilitate the implementation of infrastructure management information systems. It includes natively the geographic dimension, and therefore fulfils, inter alia, the requirements of the INSPIRE Directive, when these requirements apply to railway infrastructure. No current ISO 191xx series standard deals specifically with the challenges posed by consistent, scale-independent railway infrastructure data modelling, since these ISO standards stipulate at a higher level.

The present standard deals with semantics close to EN 28701:2012 "Identification of fixed objects in public transport" (see Normative references - page 4), namely fixed objects such as infrastructure, or events such as works. EN 28701:2012, however, is multimodal, and addresses the transport infrastructure mainly from the point of view of passenger information and timetable management. While the semantics are close, and can fairly easily be linked, the present standard aims at a wider usage (asset management and operational planning and management as well) in a narrower field of application.

Field of application:

- The IRS 30100 RailTopoModel describes a framework of concepts, to support the description of railway infrastructure, starting from the iron network and including business objects: network topology, infrastructure elements, their description, referencing and positioning, their behaviour, etc.

- The RailTopoModel should be used especially when there is a need to describe the network (structure and topology) at various levels of detail, depending on intended usage and on data availability. It is especially put the RailTopoModel at use when infrastructure data is expected to be used by various stakeholders for purposes not precisely known in advance, e.g. for network design and maintenance, traffic scheduling, and traffic management. The description can be as general as corridors; it can be detailed at line level, track level, down to physical components such as switches, lineside signals, or balises. An unlimited set of properties can be attached to component classes, for purposes such as conformity assessment, technical characteristics, life cycle data, including economic aspects, etc.
Normative references


The RailTopoModel is based on the following norms resp. standards:

- Unified Modeling Language (UML), V2.4.1 (4)
- ISO 19148:2012, Geographic information - Linear referencing
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CER</td>
<td>Community of European Railway and Infrastructure Companies (Community of European Railways, n.d.)</td>
</tr>
<tr>
<td>EIM</td>
<td>European Rail Infrastructure Managers (European Rail Infrastructure Managers, n.d.)</td>
</tr>
<tr>
<td>ERA</td>
<td>European Railway Agency (now EUAR: European Union Agency for Railways)</td>
</tr>
<tr>
<td>ERIM</td>
<td>European Railway Infrastructure Masterplan (at UIC) (ERIM-Project: Publication of the UIC Railway Topology Model, 2014)</td>
</tr>
<tr>
<td>ETCS</td>
<td>European Train Control System (European Train Control System)</td>
</tr>
<tr>
<td>EU</td>
<td>European Union</td>
</tr>
<tr>
<td>GPS</td>
<td>Global Positioning System</td>
</tr>
<tr>
<td>IM</td>
<td>Infrastructure Manager</td>
</tr>
<tr>
<td>INSPIRE</td>
<td>Infrastructure for Spatial Information in the European Community (About INSPIRE)</td>
</tr>
<tr>
<td>LRS</td>
<td>Linear Referencing System</td>
</tr>
<tr>
<td>OP</td>
<td>Operational Point (RINF concept)</td>
</tr>
<tr>
<td>railML®</td>
<td>Railway Markup Language (1)</td>
</tr>
<tr>
<td>RINF</td>
<td>Register of Infrastructure (European Register for railway network Infrastructure) at ERA (Recommendation on Specification of RINF)</td>
</tr>
<tr>
<td>SOL</td>
<td>Section Of Line (RINF concept)</td>
</tr>
<tr>
<td>UIC</td>
<td>International Union of Railways (Union internationale des chemins de fer, n.d.)</td>
</tr>
<tr>
<td>UML</td>
<td>Unified Markup Language (general-purpose modelling language) (Unified Modeling Language)</td>
</tr>
</tbody>
</table>
RailTopoModel -
Railway infrastructure topological model

General Part
1 - RailTopoModel high level concepts

1.1 - Introduction

The ultimate goal of RailTopoModel is to propose a universal representation of a railway network and associated events, to support and facilitate business development within the rail sector.

For this purpose, RailTopoModel is based on a graph model, as far as topology is concerned.

The first objective is to ensure that the model supports current and future railway business needs. To achieve this, the model fulfils the following criteria:

- The Model provides a topological representation of the iron network which is fully connected and can be visualised schematically. It supports the display of track locations at any detail level, from corridors down to tracks.

- The Model enables data to be aggregated and disaggregated, while managing connections between detail levels (or "scales"), to make sure that data consistency is retained across all scales.

- The Model allows permitted routes to be identified, based on network topology and other available information such as events (track possessions), power supply characteristics, signalling assets, etc.

- The Model supports multiple referencing systems, thus ensuring consistency during transformation from one referencing system to another one. Primary examples are:
  - Linear referencing – using mileposts and "rail addresses";
  - Positioning - using geographic reference systems;
  - Screen (schematic) coordinates.

- The model defines and locates "point", "linear" and "areal" entities, i.e.:
  - Points or nodes, such as any installation and equipment or event, etc.;
  - Lines or edges, such as speed limits, slopes, platforms etc.;
  - Areal objects, such as track circuits, tunnels, stations, etc.

This model is designed to be enriched progressively with new concepts to support business usages as they evolve.
1.2 - Previous work

The foundation principles of RailTopoModel are based on previous work which is documented, inter alia:

- in the paper “A multi scalable model based on a connexity graph representation” presented at the 12th International Conference on Computer System Design and Operation in the Railways and other Transit Systems, COMPRAIL 2010, Aug 2010, Beijing, China [5];

- in the feasibility study “UIC RailTopoModel and data exchange format” [2].

1.3 - Related work

The RailTopoModel is one important basis for the further development of railML®. railML® is an open-source, XML-based data exchange format for IT usages in railways. railML® is developed and maintained by the railML.org initiative. railML.org develops new versions of railML®, starting with railML® 3.0, that are based on RailTopoModel.

The current state of development of railML® 3.0 is described on the railML.org website [1].

1.4 - Fundamental principles

Managing and operating a railway network, for both traffic management and maintenance activities, leads to some specific challenges for IT systems and their algorithms for problems like:

- ensuring consistency in a business process spread over multiple departments working at both line and track levels;

- supporting a shared view between traffic management and works planning over time, from design to operation;

- making capacity planning and control-command share a common view on interlocking, from design to operation, including simulations.
First trials, proof on concepts, and operational developments based on those concepts, prove that the limitations encountered with traditional "monolithic" infrastructure descriptions can be solved using RailTopoModel.

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

- **What**: the semantics of assets, i.e. what is installed in the field.
- **Where**: the location of assets.
- **How**: the connections and dependencies between assets.
- **When**: the life cycle of assets.
- **Why**: the business rules which dictate how the infrastructure behaves and how it is operated.

The "what", "where", "how" and "when" are the building blocks of this Model, the description of structuring objects, independently from any usage; i.e. these are the independent foundation layers on which the Model is built.

The "why" drives the Model application, and helps utilise the information in the foundation layers to identify how the network is operated. Those rules will be populated step by step, driven by particular business project plans; these could be, for instance, rules and behaviours for interlocking.

### 1.5 - Modelling principles

RailTopoModel is based on Graph Theory, as far as topology is concerned.

Topology is only logical and therefore independent of any physical or technical items used to represent it. Topology does not for instance assume that the network described is a road network or a railway network.

Traditionally, a railway network would be represented, at track level, with "nodes" being switches and "edges" being tracks. This is not in line with graph theory, since resources are identified, in the traditional representation, as nodes or edges: In graph theory all resources are nodes, and only nodes; edges represent relations between nodes, and only relations. The network topology must therefore be described as a graph in the following manner:

![Fig. 3 - From the usual view to the graph model](image-url)
In the graph model of topology, all the nodes and edges of the above usual interpretation are instead derived from a single, so-called "NetElement" class. The NetElements A, B, C are related with each other by the edges 1, 2, 3 that define their connections. A railway graph on any level is in principle a directed graph, even though in most cases it is assumed that rail connections can be used in two directions. The outcome is an accurate railway network functional description. For that reason, in the context of topology, the physical switch device is not considered.

This principle, which is very different from the usual schematic presentation of a railway network, is consequently applied in the Model.

Actually, from a graph point of view, NetElements are nodes, and their mutual relations are edges. Now regardless of whether we have to deal with (for example) a section of line or an operational point, we can assign any characteristic to it (as an instance of the class NetEntity). This is what a connexity graph (see point 1.2 - page 9) is all about.

1.6 - Multilevel architecture

1.6.1 - Overview over levels

One purpose of the Model is to provide a standard network description with various levels of detail, following common railway practice and recent, sector-wide applications such as RINF. Those different "views" of a given network are linked by aggregation rules.

Depending on business needs or maturity level, data can be entered and used at any scale in the RailTopoModel, without lower levels of detail if these are not required, but ensuring consistency with future evolutions at other levels.

Implementations of the Model should include at least one of the following levels. Other levels may be defined and used that may be intermediate, or more detailed, or less detailed.
The RailTopoModel itself does not mandate particular description levels. The analysis of potential use cases brought forward three description levels of high significance for railway IT systems.

As the model describes a generic network, and as every detailed level shares the same concepts, an unspecified number of other levels may be derived according to the requirements of the respective use case.

### 1.6.2 - Micro level

This level defines the network in a way very close to the physical level as commonly viewed, as illustrated below:

![Fig. 4 - Micro level sample](image)

At micro level, the non-linear elements (defined in section 6.2.5) are the switch points, the network borders, maybe some administrative points (ownership boundaries), buffer stops.

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
<th>Use cases / examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Micro</td>
<td>Large scale Detailed information at track level. Basis = Switches or buffer stops that are connected by tracks.</td>
<td>ETCS, Interlocking, maintenance, asset (lifecycle) management</td>
</tr>
<tr>
<td>Meso</td>
<td>Intermediate scale Functional information at track level. Basis = Operating points that are connected by one or more tracks.</td>
<td>Visualise and process capacity properties of Sections of Lines. Capacity properties are directly linked with the number of tracks.</td>
</tr>
<tr>
<td>Macro</td>
<td>Small scale Minimal track level information. Basis = Operating points connected with each other via single connections (one or more tracks).</td>
<td>Network of railway lines and stations. Timetabling information.</td>
</tr>
</tbody>
</table>
The linear elements (defined in point 2.2.4 - page 29) are the tracks connected to, rather than connecting, those non-linear Elements.

1.6.3 - Meso level

The Meso level brings the description of the tracks between the operational points of the network into focus.

![Meso level sample](image)

Fig. 5 - Meso level sample

The non-linear elements are the Operating Points (OP = stations, yards, junctions, boundaries), and linear elements are the tracks connecting Ops.

1.6.4 - Macro level

The Macro level aims to describe the network at regional or national level, with the non-linear elements being the boundaries and the major OP’s, while the linear elements are the sections of lines connecting those points.

![Macro level sample](image)

Fig. 6 - Macro level sample
1.6.5 - Nano level and other levels

The nano level could be described as a properly attributed surveyor’s map, including topological properties of the rail network in the finest possible granularity.

Typically, this level will be built starting from the micro level, by using "switch templates". Conversely, topological properties of the Micro level can be automatically produced from the Nano topology using RTM aggregation. In case the detailed nano level information does not exist, it is possible to add navigability information manually to the track edges at micro level.

Use cases for the Nano level would include interlocking and asset management, for instance.
1.6.6 - Aggregation principle

The following chart shows the principle for aggregation from tracks (micro) to OPs (meso), then Sections of Lines (macro):

![Aggregation example](image)

Fig. 8 - Aggregation example

1.7 - Packages and main elements

1.7.1 - General

The RailTopoModel is described in UML notation.

Similar to UML, the modelling concepts of the RailTopoModel are grouped within packages. A package consists of a collection of tightly coupled modelling concepts. Each package deals with a specific aspect of the model.

1.7.2 - Packages

The RailTopoModel representation in UML consists of four packages: Base, Topology, Positioning Systems, and Net Entities. These packages are depicted using specific colour codes.
## Table 2: RTM Packages

<table>
<thead>
<tr>
<th>Package</th>
<th>Colour code</th>
<th>Main element(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base</td>
<td>Grey</td>
<td>Network, LevelNetwork</td>
</tr>
<tr>
<td>Topology</td>
<td>Yellow</td>
<td>NetElement, Relation, CompositionNetElement</td>
</tr>
<tr>
<td>Positioning</td>
<td>Green</td>
<td>PositioningSystem, IntrinsicCoordinate</td>
</tr>
<tr>
<td>Systems</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Net Entities</td>
<td>Light blue</td>
<td>LocatedNetEntity, EntityLocation</td>
</tr>
</tbody>
</table>

![Diagram of RTM Packages](image)
1.7.3 - Model overview

Note: not all classes are represented in the following class diagram. The diagram follows simplified UML conventions: full arrows represent generalizations, links with diamonds represent associations (possibly aggregations or compositions). A few classes (not part of the Model) have been added for illustrative purposes. This figure as a whole is not normative.

![Class diagram](image)

Fig. 9 - RTM Class diagram

The packages and classes are described into more detail under chapter 6, following conventions detailed under point 1.8.

1.8 - Conventions for package and class description

All concepts of RailTopoModel are depicted as UML classes. Each RailTopoModel concept is described in a sub-section of point 2 - page 20.

The names of the RailTopoModel concepts are enclosed in double quotation marks like "NetworkResource".

Description items consist of:

1. Definition
2. Diagram with current class and its attributes including inherited attributes
3. Description of associations
4. Table with own attributes (inherited attributes may not be shown)
5. Class diagram containing all classes up to the root class and all associated classes

6. List of classes that are derived from the current class.

A sample diagram representing a class and its attributes is shown below:

![Sample class representation](image)

The upper compartment of the UML-Class rectangle contains the name of current class in the centre ("PositionedRelation"). If the current class has a parent class, then the name of the parent class is found in the upper right corner of the upper compartment ("Relation").

Attributes of the current class are found at the top of the second compartment, mentioning the name of the attribute (e.g. "positionOnA") and the type of the attribute (e.g. "Usage").

Attributes of the parent class (or parent classes of the parent class, up to the root class), if shown, are also contained in the second compartment below the name of class they belong to (e.g. "BaseObject"). Those inherited attributes are also shown with their name ("validFrom") and their type ("Date").

If the current class possesses a parent class or an associated class then a second diagram containing all classes up to the root class and the associations is included.
Fig. 11 - Example of a class diagram containing all classes up to the root class

The second diagram type shows the full context of the current class within the model. The current class itself is depicted with the colour of the respective package. All generalizations (up to the root class) are shown. All associations of the current class and all associations of parent classes are shown. Attributes are always shown in the second compartment of the class they belong to.
2 - Package and class description

2.1 - Package: Base

2.1.1 - General

The package “Base” is centred on the classes “Network” and “LevelNetwork”.

An instance of “Network” can be considered as a set of description levels (“LevelNetwork”) and a set of “NetworkResource” instances.

“NetworkResource” depicts the building blocks of a “Network”, mainly

- For Topology: “NetElement” (nodes), “Relation” (edges);
- For Net Entities: “NetEntity” (e.g. physical assets, or speed limits), “EntityLocation” (their location), with the class “AssociatedNetElement” providing the link between assets and Topology;
- The relevant positioning system, referred to via “AssociatedPositioningSystem”.

Fig. 12 - Base package overview
An instance of "LevelNetwork" depicts a specific description level, as described under point 1.6 - page 11. The concept of description level is central to the goal of computational efficiency.

2.1.2 - BaseObject

The base class "BaseObject" defines four properties shared by most objects in the RailTopoModel.

```
class BaseObject
  - id : UUID
  - name : String
  - validFrom : Date
  - validTo : Date
```

Fig. 13 - BaseObject

Table 3 : BaseObject (Attributes)

<table>
<thead>
<tr>
<th>Attributes</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>UUID</td>
<td>Unique identifier. It is recommended to use an UUID whenever possible. For easy adaptation of existing systems, any other unique identifier is permitted.</td>
</tr>
<tr>
<td>name</td>
<td>String</td>
<td>Natural designation of the object.</td>
</tr>
<tr>
<td>validFrom</td>
<td>Date</td>
<td>Point in time where the object is available for usage for train operations (if empty, then the object is valid till the validTo date).</td>
</tr>
<tr>
<td>validTo</td>
<td>Date</td>
<td>Point in time where the object is no longer available for functional usage (if empty, then the object is valid since the validFrom date).</td>
</tr>
</tbody>
</table>
2.1.3 - Network

The class "Network" defines the network being considered. It includes all resources that compose it (all Levels included), inter alia the topological, structural and positional properties exhibited by any railway network.

The class "Network" is derived from "BaseObject".

A "Network" is described in at least one "LevelNetwork". A "Network" may be described in more than one "LevelNetwork", typically for different levels of detail.

Whenever an instance of "Network" is removed, all related "NetworkResource" instances and all related "LevelNetwork" instances are removed.

Fig. 14 - Network

Fig. 15 - Network (Neighbourhood)
2.1.4 - LevelNetwork

The class "LevelNetwork" defines a consistent "view" of a Network at a certain level of granularity. An instance of this class therefore includes all resources that are required to define the corresponding level (e.g. micro/track, or macro/line).

The class "LevelNetwork" is derived from "BaseObject".

A "LevelNetwork" belongs to exactly one "Network".

Fig. 16 - LevelNetwork

Fig. 17 - LevelNetwork (Neighbourhood)
2.1.5 - NetworkResource

Every object of the network is qualified as a resource. The class "NetworkResource" defines this concept.

Fig. 18 - NetworkResource

The class "NetworkResource" is derived from "BaseObject".

A "NetworkResource" belongs to exactly one "Network".

Fig. 19 - NetworkResource (Neighbourhood)
2.2 - Package: Topology

2.2.1 - General

The Topology package essentially applies Graph theory concepts to the RailTopoModel. Nodes are conceptually embodied by "NetElement", and edges by "Relation". "CompositionNetElement" is the class, derived from "NetElement" that will ultimately allow the assembly of nodes into bigger nodes, and zooming in and out from one level to another.

Fig. 20 - Topology package overview
2.2.2 - NetElement

The class "NetElement" defines the base member of topology in a connectivity graph of a network (at any level).

```
class NetElement

::BaseObject
- id :UUID
- name :String
- validFrom :Date
- validTo :Date
```

Fig. 21 - NetElement

The class "NetElement" is derived from "NetworkResource".

Each "NetElement" takes part in one or many "Relation" with other "NetElement" instances.
2.2.3 - **CompositionNetElement**

The class "CompositionNetElement" carries the generic concept of topological aggregation. It defines a topological element that aggregates some other topological element from another level (e.g. a macro element aggregates micro elements).
The class "CompositionNetElement" is derived from "NetElement".

Whenever an instance "CompositionNetElement" is removed, all related "ElementPartCollection" instances are removed.

Fig. 24 - CompositionNetElement (Neighbourhood)
2.2.4 - LinearElement

The class "LinearElement" defines "PositioningNetElement" instances that are one-dimensional.

```
class LinearElement
```

```
:: BaseObject
- id : UUID
- name : String
- validFrom : Date
- validTo : Date
```

Fig. 25 - LinearElement

The class "LinearElement" is derived from "PositioningNetElement".

```
Fig. 26 - LinearElement (Neighbourhood)
```
The classes "Trail" and "SectionOfLine", shown on the overall simplified class diagram [see point 1.7.3 - page 17], are derived from "LinearElement". These two classes are not part of the RailTopoModel; they are provided here for illustration purposes.

- A "Trail" is an uninterrupted track between two adjacent switches, or between a switch and an adjacent buffer stop. "Uninterrupted" means that there are no other switches in that connection. Therefore, the class "Trail" can represent nodes at Micro level [see point 1.6.2 - page 12], according to Graph theory and to the modelling principles presented under point 1.5 - page 10.

- Similarly, a "SectionOfLine", being a line section between two adjacent Operational Points, would be an important class of nodes to be used at Macro level [see point 1.6.4 - page 13].

2.2.5 - NonLinearElement

The class "NonLinearElement" defines "PositioningNetElement" instances with no dimensions (spots).

```plaintext
class NonLinearElement

<table>
<thead>
<tr>
<th>PositioningNetElement</th>
</tr>
</thead>
<tbody>
<tr>
<td>NonLinearElement</td>
</tr>
</tbody>
</table>

::BaseObject
- id :UUID
- name :String
- validFrom :Date
- validTo :Date
```

Fig. 27 - NonLinearElement

The class "NonLinearElement" is derived from "PositioningNetElement".
Note:

The classes “OperationalPoint” and “NetLimit”, derived from “NonLinearElement”, are not part of the RailTopoModel. These are provided as two examples of classes that may represent nodes (in the sense of Graph theory) at macro level.

2.2.6 - ElementPartCollection

The class "ElementPartCollection" defines the collection of Net elements to be aggregated into the higher level NetElement (Generic class).
The class "ElementPartCollection" is derived from "NetworkResource". An "ElementPartCollection" instance belongs to exactly one "CompositionNetElement" instance.

Fig. 30 - ElementPartCollection (Neighbourhood)
2.2.7 - OrderedCollection

The class "OrderedCollection" is a subclass of ElementPartCollection, dedicated to ordered NetElements (required to build a route).

![Diagram of OrderedCollection]

Fig. 31 - OrderedCollection

<table>
<thead>
<tr>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>sequence</td>
</tr>
</tbody>
</table>

Sequence of the child element within the ordered collection

The class "OrderedCollection" is derived from "ElementPartCollection".
2.2.8 - UnorderedCollection

The class "UnorderedCollection" is a subclass of ElementPartCollection that is dedicated to unordered NetElements (bulk list without need for routes).

The child "NetElement" instances possess no ordering property.

The class "UnorderedCollection" is derived from "ElementPartCollection".
2.2.9 - Relation

The class "Relation" defines the connexity relation between two NetElements in the connexity graph of the network.

The class "Relation" is derived from "NetworkResource".

In a functional railway network, each instance of "Relation" typically brings together two "NetElement" instances. "Relation" can be seen as the base class to define edges in the sense of Graph theory.
2.2.10 - PositionedRelation

The class "PositionedRelation" is a subclass of "Relation", defining an oriented relation between exactly two PositioningNetElements.

Fig. 36 - Relation (Neighbourhood)

Fig. 37 - PositionedRelation

The class "PositionedRelation" is derived from "Relation".
One connected "NetElement" is designated code "A", the other connected "NetElement" is designated code "B".

Table 5: PositionedRelation (Attributes)

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Navigability</th>
<th>AB</th>
<th>BA</th>
<th>Both</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>navigability</td>
<td>AB</td>
<td>it is possible to move a train from NetElement &quot;A&quot; to NetElement &quot;B&quot;. It is not possible to move it from NetElement B to NetElement A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>positionOnA</td>
<td>Usage</td>
<td>0</td>
<td>the &quot;Relation&quot; is using the start of NetElement A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>positionOnB</td>
<td>Usage</td>
<td>0</td>
<td>the &quot;Relation&quot; is using the start of NetElement B</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>the &quot;Relation&quot; is using the end of NetElement A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>the &quot;Relation&quot; is using the end of NetElement B</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5: PositionedRelation (Attributes)

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Navigability</th>
<th>AB</th>
<th>BA</th>
<th>Both</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>navigability</td>
<td>AB</td>
<td>it is possible to move a train from NetElement &quot;A&quot; to NetElement &quot;B&quot;. It is not possible to move it from NetElement B to NetElement A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>positionOnA</td>
<td>Usage</td>
<td>0</td>
<td>the &quot;Relation&quot; is using the start of NetElement A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>positionOnB</td>
<td>Usage</td>
<td>0</td>
<td>the &quot;Relation&quot; is using the start of NetElement B</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>the &quot;Relation&quot; is using the end of NetElement A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>the &quot;Relation&quot; is using the end of NetElement B</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5: PositionedRelation (Attributes)

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Navigability</th>
<th>AB</th>
<th>BA</th>
<th>Both</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>navigability</td>
<td>AB</td>
<td>it is possible to move a train from NetElement &quot;A&quot; to NetElement &quot;B&quot;. It is not possible to move it from NetElement B to NetElement A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>positionOnA</td>
<td>Usage</td>
<td>0</td>
<td>the &quot;Relation&quot; is using the start of NetElement A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>positionOnB</td>
<td>Usage</td>
<td>0</td>
<td>the &quot;Relation&quot; is using the start of NetElement B</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>the &quot;Relation&quot; is using the end of NetElement A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>the &quot;Relation&quot; is using the end of NetElement B</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Fig. 38 - PositionedRelation (Neighbourhood)
2.2.11 - PositioningNetElement

The class "PositioningNetElement" defines a NetElement requiring at least one Positioning System, with orientation (carried by IntrinsicCoordinate).

Whenever an instance of "PositioningNetElement" is removed, all related "AssociatedPositioningSystem" instances are removed.
2.2.12 - AssociatedNetElement

The class "AssociatedNetElement" defines topological structures and location information in relation between "NetElement" instances and in relation between one "NetElement" instance and location information for "NetEntity" instances.

```
class AssociatedNetElement
- intrinsCoordBegin double
- intrinsCoordEnd double
- keepsOrientation boolean
```

The class "AssociatedNetElement" has no specific parent class.
<table>
<thead>
<tr>
<th>Attributes</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>intrinsicCoordBegin</td>
<td>double</td>
<td>Start location of the &quot;NetEntity&quot; instance in relation to the &quot;PositioningNetElement&quot; which is used for positioning within the network.</td>
</tr>
<tr>
<td>intrinsicCoordEnd</td>
<td>double</td>
<td>End location of the &quot;NetEntity&quot; instance in relation to the &quot;PositioningNetElement&quot; which is used for positioning within the network.</td>
</tr>
<tr>
<td>keeps Orientation</td>
<td>boolean</td>
<td>Child LinearElement keeps same Orientation as parent LinearElement 0 (false) : Orientation is not relevant 1 (true) : Orientation is relevant</td>
</tr>
</tbody>
</table>
2.2.13 - OrderedAssociatedNetElement

The class "OrderedAssociatedNetElement" defines the ordered sequences of "AssociatedNetElement" instances which together describe the complete structure of a "LinearLocation" instance.
Table 7: OrderedAssociatedNetElement (Attributes)

<table>
<thead>
<tr>
<th>Attributes</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequence</td>
<td>Int</td>
</tr>
</tbody>
</table>

Fig. 44 - OrderedAssociatedNetElement (Neighbourhood)
2.3 - Package: Positioning Systems

2.3.1 - General

The “Positioning Systems” package offers a catalogue of positioning methods, which currently fall into three categories: intrinsic, linear and geographic. The network topology may use any of these, only the intrinsic positioning being mandatory.

Fig. 45 - Positioning systems package overview
2.3.2 - PositioningSystem

The class "PositioningSystem" defines the generic concept of a positioning system.

![Diagram of PositioningSystem class](image1)

**Fig. 46 - PositioningSystem**

The class "PositioningSystem" is derived from "BaseObject".

![Diagram showing PositioningSystem and its neighbors](image2)

**Fig. 47 - PositioningSystem (Neighbourhood)**
2.3.3 - LinearPositioningSystem

The class "LinearPositioningSystem" defines a "PositioningSystem" where a "line of reference" together with a single number allows a location within a railway network to be defined.

In railway business a "line of reference" is very often represented with a line number or a track number together with a start mileage and an end mileage.

Note: RailTopoModel makes no assumption about the nature of the "line of reference".

The class "LinearPositioningSystem" is derived from "PositioningSystem".
Whenever an instance of "LinearPositioningSystem" is removed, all related "LinearAnchorPoint" instances are removed.

Table 8: LinearPositioningSystem (Attributes)

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>linearReferencingMethod</td>
<td>Method for linear referencing</td>
</tr>
<tr>
<td>LrsMethod</td>
<td>absolute</td>
</tr>
<tr>
<td></td>
<td>relative</td>
</tr>
<tr>
<td></td>
<td>interpolation</td>
</tr>
<tr>
<td>startMeasure</td>
<td>Value for measurement at the beginning of the &quot;LinearPositioningSystem&quot;</td>
</tr>
<tr>
<td>endMeasure</td>
<td>Value for measurement at the end of the &quot;LinearPositioningSystem&quot;</td>
</tr>
<tr>
<td>units</td>
<td>Units for measurement (e.g. kilometre, metre, mile)</td>
</tr>
</tbody>
</table>
2.3.4 - LinearAnchorPoint

The class "LinearAnchorPoint" defines an ordered set of named points within a "LinearPositioningSystem", which are used to transform between LRS based locations suitable for field work and locations using intrinsic coordinates. Each point contains an LRS measure and the distance to next "LinearAnchorPoint" instance.

This information allows the mapping of LRS locations to intrinsic locations.

The class "LinearAnchorPoint" is derived from "NetworkResource".

A "LinearAnchorPoint" belongs to exactly one "LinearPositioningSystem".

Table 9: LinearAnchorPoint (Attributes)

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>anchorName</td>
<td>String</td>
<td>Name of the &quot;LinearAnchorPoint&quot; instance which is unique within the given &quot;LinearPositioningSystem&quot;</td>
</tr>
<tr>
<td>measure</td>
<td>double</td>
<td>Measure of the Anchor Point within the given &quot;LinearPositioningSystem&quot;</td>
</tr>
<tr>
<td>measureToNext</td>
<td>double</td>
<td>Basis for modified interpolation of location in the interval up to the next &quot;LinearAnchorPoint&quot; of the given &quot;LinearPositioningSystem&quot;.</td>
</tr>
</tbody>
</table>
Fig. 51 - LinearAnchorPoint (Neighbourhood)
2.3.5 - GeometricPositioningSystem

The class "GeometricPositioningSystem" defines schematic, geographic or geodetic Coordinate Reference Systems which are used to position "NetElement" instances or "NetEntity" instances. In the context of RailTopoModel, "GeometricPositioningSystem" instances are used to support the transformation between intrinsic locations and geometric coordinates.

![Fig. 52 - GeometricPositioningSystem](image)

The class "GeometricPositioningSystem" is derived from "PositioningSystem".

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>crsDefinition</td>
<td>String</td>
<td>Coordinate Reference System</td>
</tr>
</tbody>
</table>

![Fig. 53 - GeometricPositioningSystem (Neighbourhood)](image)
2.3.6 - PositioningSystemCoordinate

The class "PositioningSystemCoordinate" defines the generic concept of a coordinate in a positioning system that is used to specify locations for "NetEntity", "PositioningNetElement", and all other objects of the network. These coordinates are either expressed as "GeometricCoordinate", or "LinearCoordinate", or any future type of coordinate.

Fig. 54 - PositioningSystemCoordinate

The class "PositioningSystemCoordinate" has no specific parent class.

Fig. 55 - PositioningSystemCoordinate (Neighbourhood)
2.3.7 - IntrinsicCoordinate

The class "IntrinsicCoordinate" defines a coordinate which is used to specify locations in reference to "NetElement" instances. An intrinsic coordinate may have an arbitrary real number in interval [0,1] of associated "PositioningSystemCoordinate" instances. 0 and 1 correspond to the extremities of the element.

Fig. 56 - IntrinsicCoordinate

The class "IntrinsicCoordinate" has no specific parent class.

Table 11: IntrinsicCoordinate (Attributes)

<table>
<thead>
<tr>
<th>Attributes</th>
<th>intrinsicCoord</th>
<th>double</th>
<th>Location in reference to the chosen NetElement, given as value in the interval from 0 to 1.</th>
</tr>
</thead>
</table>

An "IntrinsicCoordinate" belongs to exactly one "AssociatedPositioningSystem".

An "IntrinsicCoordinate" may have multiple associated traditional coordinates. This is required for transformation between intrinsic locations and traditional locations.
2.3.8 - LinearCoordinate

The class "LinearCoordinate" defines a location in reference to a given "LinearPositioningSystem".

![Diagram of LinearCoordinate](image-url)
The class "LinearCoordinate" is derived from "PositioningSystemCoordinate".

### Table 12: LinearCoordinate (attributes)

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lateralOffset</td>
<td>double</td>
<td>distance perpendicular to the &quot;line of reference&quot;</td>
</tr>
<tr>
<td>measure</td>
<td>double</td>
<td>position at the &quot;line of reference&quot; (possibly adjusted to local anomalies using &quot;LinearAnchorPosition&quot;)</td>
</tr>
<tr>
<td>verticalOffset</td>
<td>double</td>
<td>Height above the &quot;line of reference&quot; at the position defined by &quot;measure&quot;.</td>
</tr>
</tbody>
</table>

*Fig. 59 - LinearCoordinate (Neighbourhood)*

### 2.3.9 - GeometricCoordinate

The class "GeometricCoordinate" defines one coordinate using a "GeometricPositioningSystem" as reference system. Depending on the properties of the coordinate system used, a coordinate consists of cartesian or spherical values. In case of 2D coordinate systems, the attribute z is undefined.

*Fig. 60 - GeometricCoordinate*
The class "GeometricCoordinate" is derived from "PositioningSystemCoordinate".

Table 13 : GeometricCoordinate (Attributes)

| Attributes | x  | double | x value of cartesian coordinate, longitude of spherical coordinate |
|           | y  | double | y value of cartesian coordinate, latitude of spherical coordinate |
|           | z  | double | z value of cartesian coordinate, altitude of spherical coordinate |

Fig. 61 - GeometricCoordinate (Neighbourhood)
2.3.10 - AssociatedPositioningSystem

The class "AssociatedPositioningSystem" defines the relation between a "PositioningNetElement" instance and a "PositioningSystem" instance.

The associated set of "IntrinsicCoordinate" together with the related "PositioningSystemCoordinate" instances define the translation parameters between "IntrinsicCoordinate" based locations, and locations based on external coordinates ("LinearLocationCoordinate" or "SpotLocationCoordinate") using "LinearPositioningSystem" or "GeometricPositioningSystem" as a coordinate system.

![Fig. 62 - AssociatedPositioningSystem](image.png)

The class "AssociatedPositioningSystem" is derived from "NetworkResource".

Any "AssociatedPositioningSystem" belongs to exactly one "PositioningNetElement".
Fig. 63 - AssociatedPositioningSystem (Neighbourhood)
2.4 - Package: Net Entities

2.4.1 - General

The “Net Entities” package defines the classes that allow to structure the functional description of the considered network, beyond the mere scope of topology. Net entities are the functional images of physical objects (such as bridges or tunnels, signals or level crossings, tracks and switches), or even immaterial objects (such as speed limits or radio coverage area).

The “Net Entities” package is structured in such way that these objects are associated with their location, and locations are themselves associated with the topology elements described above.

Fig. 64 - Net Entities package overview
### 2.4.2 - NetEntity

"NetEntity" is a generic parent class for all information that can be associated with the network considered. Information may be, for instance: tunnels, signals, level crossings, track circuits, speed limits, etc.

![NetEntity Diagram](image)

**Fig. 65 - NetEntity**

### 2.4.3 - LocatedNetEntity

The class "LocatedNetEntity" is a parent class for information that can definitely be localized, which is the case of most infrastructure-related objects.

![LocatedNetEntity Diagram](image)

**Fig. 66 - LocatedNetEntity**

Note: this class has been introduced for semantic clarification, as one may expect "UnlocatedNetEntities" to also be introduced in the future. Possible derived classes, as shown in the Model overview (see point 1.7.3 - page 17), would be "StructureNetEntity", "SignallingNetEntity", "DressingNetEntity", etc. The RailTopoModel user may create such classes, according to use cases. Further class definitions, resulting from common use cases, may be added to the present Standard in the future.

"LocatedNetEntity" is a generic docking station for all relevant domain information which can be located in the context of the network in question.
Fig. 67 - LocatedNetEntity (Neighbourhood)
2.4.4 - EntityLocation

The class "EntityLocation" defines topological and positional location information for "NetEntity" instances.

The class "EntityLocation" is derived from "NetworkResource". One instance of "EntityLocation" belongs to exactly one "LocatedNetEntity" instance. When one of the "LocatedNetEntity" instances is removed, the instance of "EntityLocation" could continue to exist.
2.4.5 - SpotLocation

The class "SpotLocation" defines point location information for "LocatedNetEntity" instances in reference to one "PositioningNetElement" instance.

Table 14 : SpotLocation (Attributes)

<table>
<thead>
<tr>
<th>Attributes</th>
<th>ApplicationDirection</th>
<th>normal</th>
<th>the located object is valid in the direction of the LinearElement</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationDirection</td>
<td>ApplicationDirection</td>
<td>reverse</td>
<td>the located object is valid in the reverse direction of the LinearElement</td>
</tr>
<tr>
<td></td>
<td></td>
<td>both</td>
<td>the located object is valid in both directions</td>
</tr>
</tbody>
</table>
2.4.6 - SpotLocationIntrinsic

The class "SpotLocationIntrinsic" defines additional information in respect of intrinsic positioning for a "SpotLocation" instance.

The class "SpotLocationIntrinsic" is derived from "SpotLocation".
Table 15: SpotLocationIntrinsic (Attributes)

| Attributes       | intrinsicCoord | double                  | Location in reference to the chosen NetElement given as value in the interval from 0 to 1. |
|------------------|----------------|-------------------------|

**Fig. 73 - SpotLocationIntrinsic (Neighbourhood)**
2.4.7 - SpotLocationCoordinate

The class "SpotLocationCoordinate" defines the relation between a "SpotLocation" and "PositioningSystemCoordinate".

![Fig. 74 - SpotLocationCoordinate](image)

**Table 16 : SpotLocationCoordinate (Attributes)**

<table>
<thead>
<tr>
<th>Attributes</th>
<th>ApplicationDirection</th>
<th>normal</th>
<th>the located object is valid in the direction of the LinearLocation</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationDirection</td>
<td>ApplicationDirection</td>
<td>reverse</td>
<td>the located object is valid in the reverse direction of LinearLocation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>both</td>
<td>the located object is valid in both directions</td>
</tr>
</tbody>
</table>

The class "SpotLocationCoordinate" is derived from "SpotLocation".
Fig. 75 - SpotLocationCoordinate (Neighbourhood)
2.4.8 - LinearLocation

The class "LinearLocation" defines location information with a startpoint and an endpoint for "LocatedNetEntity" instances in reference to one or more "PositioningNetElement" instances. The set of associated "PositioningNetElement" instances is ordered.

![LinearLocation Diagram]

The class "LinearLocation" is derived from "EntityLocation".

Table 17: LinearLocation (Attributes)

<table>
<thead>
<tr>
<th>Attributes</th>
<th>ApplicationDirection</th>
<th>Normal</th>
<th>Reverse</th>
<th>Both</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationDirection</td>
<td>ApplicationDirection</td>
<td>normal</td>
<td>reverse</td>
<td>both</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the located object is valid in the direction of the LinearLocation</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>the located object is valid in the reverse direction of LinearLocation</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>the located object is valid in both directions</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Fig. 77 - LinearLocation (Neighbourhood)
2.4.9 - LinearLocationCoordinate

The class "LinearLocationCoordinate" defines the relation between a "LinearLocation" and "PositioningSystemCoordinate" instances.

![Class Diagram for LinearLocationCoordinate]

Fig. 78 - LinearLocationCoordinate

The class "LinearLocationCoordinate" is derived from "LinearLocation".
Fig. 79 - LinearLocationCoordinate (Neighbourhood)
2.4.10 - AreaLocation

The class "AreaLocation" defines a set of "AssociatedNetElement" instances which together represent an area of interest. Each "AssociatedNetElement" instance contains attributes which designate the extent of the related "PositioningNetElement" instance using intrinsic coordinates.

The class "AreaLocation" is derived from "EntityLocation".

Whenever an instance of "AreaLocation" is removed, all related instances of "AssociatedNetElement" are removed as well.

Fig. 80 - AreaLocation
Fig. 81 - AreaLocation (Neighbourhood)
3 - Conformance

3.1 - General

RailTopoModel, as a conceptual data model, describes essential structural properties of possible implementations of railway IT systems (named “Systems” in the present section), when these Systems contain or handle information about railway infrastructure.

Conformance is defined in the present section so as to ensure, inter alia:

- the computational efficiency and scalability of the System;
- the compatibility of the System with railML®, when relevant.

To be declared conformant to the RailTopoModel, Systems must present the features required below ("shall"). Recommended ("should") and optional ("may") features are also mentioned, for the sake of clarity.

Conformant Systems:

- may include all RailTopoModel concepts, or a subset of these concepts;
- may extend the RailTopoModel, e.g. with additional packages and classes;
- shall not alter the concepts provided under the present IRS and their relations, irrespective of whether these concepts are required, recommended, or optional, except for the cases described below.

3.2 - Referencing techniques

Linear net elements shall be used as a reference system.

Locations shall be stored using intrinsic positioning.

Transformation between intrinsic positioning and linear references (if used) shall be supported.

Transformation between intrinsic positioning and geometric positioning (if used) shall be supported.

3.3 - Description Levels

At least one of the widely used description levels (Macro, Meso, Micro) level description should be supported.

Additional description levels may be implemented.
3.4 - Navigation

The system shall provide topological navigation inside and between different description levels.

Interaction between objects of different description levels shall be supported.

A conformant System may only have one level. In such case, the aggregation mechanisms need not be developed.

3.5 - Identification of objects

If a new System is created, all objects derived from “BaseObject” will, by design, inherit a unique identifier in the shape of a UUID (Leach, 2005).

For legacy Systems being brought, partially or progressively, in conformance with the RailTopoModel, other identifiers may be used instead.
Appendix A - RailTopoModel complete class diagram
Warning

No part of this publication may be copied, reproduced or distributed by any means whatsoever, including electronic, except for private and individual use, without the express permission of the International Union of Railways (UIC). The same applies for translation, adaptation or transformation, arrangement or reproduction by any method or procedure whatsoever. The sole exceptions - noting the author's name and the source - are "analyses and brief quotations justified by the critical, argumentative, educational, scientific or informative nature of the publication into which they are incorporated".


© International Union of Railways (UIC) - Paris, 2016

Printed by the International Union of Railways (UIC)
16, rue Jean Rey 75015 Paris - France, September 2016
Dépôt Légal September 2016

ISBN 978-2-7461-2513-1