Acta Polytechnica CTU Proceedings **11**:16–21, 2017
doi:10.14311/APP.2017.11.0016

available online at http://ojs.cvut.cz/ojs/index.php/app

RAILTOPOMODEL AND RAILML 3 IN OVERALL CONTEXT

Adam Hlubuček

CTU in Prague, Faculty of Transportation Sciences, Konviktská 20, Prague, Czech Republic
correspondence: hlubuada@fd.cvut.cz

Abstract.

This paper aims to provide a brief insight into the UIC RailTopomodel and railML initiative. Concerning the railML exchange format, mainly the current form of forthcoming third edition, involving especially the infrastructure schema, based on the RailTopoModel modelling principles, is taken into account. When rewarding, the comparison between railML 3 and the previous railML 2.3 version is given. At the end, the author focuses on selected issues of possible applicability of these tools for the needs of such systems as the automatic train control, automatic train operation, prediction of train etc. If essential, some extensions of the structures are proposed.

Keywords: RailTopoModel, railML, railway infrastructure, information systems.

1. Introduction

RailML [1] is an open source exchange format used in the railway branch to ensure a data transfer among different software applications. The railway markup langue, based on XML, was originally designed to enable data exchange in the scope of timetabling. Later on, the specifications [2] were extended to the subsystems of rolling stocks and infrastructure. In these days, also the schema of interlocking is being developed in relation to oncoming railML 3 release.

Another milestone, as regards the railML 3, is a significant reedition of the infrastructure concept. Whereas the principles of modelling of the railway topology were rather casually defined within the previous versions, railML 3 is supposed to be strongly founded on the UIC RailTopoModel [3] which provides the rules how to describe the infrastructure in predefined way, nevertheless, allowing all kinds of user-specific extensions.

The RailTopoModel initiative was introduced in 2013 [4] in order to create a common standard in railway infrastructure modelling. The generic railway topology model aims to meet multiple needs in the field of railway data exchange, ensured by the means of the railML 3 format. In 2016, the RailTopoModel was released in the form of UIC International Railway Standard 30100 [5].

2. The RailTopoModel Overview

The UIC RailTopomodel [3] is a topological model of railway infrastructure, based on a „connexity graph“. Describing a railway topology, two basic types of elements can be taken into account: the linear elements (e.g. tracks and lines) and nonlinear elements (e.g. operational points). These elements allow describing the infrastructure at different levels of detail, in order to be suitable for different specific purposes. Regarding the “connexity graph”, the elements of both these categories are supposed to be modelled as the nodes, whereas the edges in the node-edge diagram represent the physical connections between these particular interconnected net elements. The linear elements shall have their orientation defined. Therefore, when concerning the (again orientated) relation between two of them, it should be in both cases apparent, whether the beginning or the end of the net element is taken into account. This generic principle is illustrated in the figure 1 considering the tracks to be the nodes.

In order to express the fact, whether a train is able to pass from one net element to another, additional information is necessary to be given. For this reason, the relations between net elements shall be described also by the navigability attribute, besides, concerning the direction of allowable train movement. [5] In this way, a feasible passage through a switch from one track to another can be modelled as well as the line change when passing through an operation control point.

Currently, the model enables to express the topology including positioning and location of net entities to be projected to the topological layer taking into account different views. There are several predefined levels of detail (macro, meso, micro) at which a railway network can be described. A user may utilize (or start with) any of them, depending to his specific use case. Furthermore, he is authorised to create own levels, to be appropriate for his needs (e.g. nano or corridor level). These levels of detail should be vertically interconnectable by the means of aggregation and disaggregation principles. Therefore, when passing from one level to another, sometimes an extra transition level is needed. Except the different views of the topology, the particular level enables to deal with particular entities in different ways. For instance, it allows displaying a certain entity as a section at a detailed level, as a spot at a less detailed level and not to include it to the least detailed level at all. [5–7]

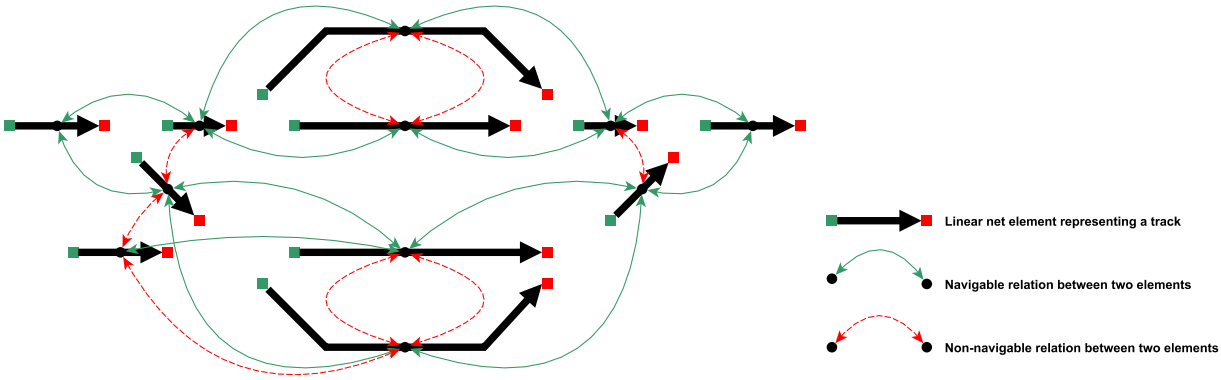

FIGURE 1. An application of the RailTopoModel principles on an example of a railway station with five simple turnouts and one single slip modelled by the means of positioned net relations.

Later on, the model together with railML 3 should be extended so as to enable digital continuity for infrastructure life cycle and operation supporting BIM development and using railML 3 for web services as well. It should be able to provide a unique continuous network description, not only taking into account past, current and future states, but also involving the project management aspect of the lifecycle dimension. [6] Concerning the objects being modelled, expressing their validities (active / non active), variants (alternative current states) and versions (states evolving over time) should be possible. [4]

3. RailTopoModel Packages

3.1. Base

Currently, the RailTopoModel consists of four packages. Concerning the UML class diagram designed to express its structure, there is a base network domain consisting of the base object, network, level network and network resource data classes, defining the general overview of the network being modelled, including the modelling principles used. The classes of the other subsystems are mostly some kind of extensions of the network resource class, eventually of the base object class itself. The base object class specifies the fundamental attributes to be common for all the derived XML elements (in order to express its unique ID, name and validity). As regards the ID, especially the UUID should be used. [5, 8]

3.2. Topology

The topology package is especially based on the net elements and relations which are modelled by the means of node-edge diagram. The aggregation principle (of net elements) is managed within the topology domain as well. Particular net elements can be grouped to larger units, creating new net elements, which are supposed to be used to express the topology at less detailed level. For instance, several tracks (linear net elements in the scope of micro-topology) can be aggregated in order to form a station area, further regarded to be an operational point (a nonlinear element in the scope of macro-topology). Nevertheless, the RailTopoModel defines the data classes using the most general approach (not yet specifying the particular elements of infrastructure). [5, 8]

3.3. Positioning

The positioning package enables to define several positioning systems (linear as well as geometric – both the geographical and these ones used to display particular entities on a monitor) as well as the corresponding coordinates. To a particular net element, several of these positioning systems can be associated, nevertheless every positioning net element involves own intrinsic positioning system, whose coordinates takes values in the range of <0;1> referring to a relative position within the element. [5, 8] In order to be able to express the intrinsic position in absolute terms, the attribute of length related to the oriented net element would have to be specified in addition.

3.4. Net Entities

The last subsystem deals with net entities. The net entities represent these object and characteristics of the infrastructure, which can be connected to the undelaying topology layer using positioning systems. Differently from the previous railML versions, enabling only to link these object to a one certain track forming its data substructure, the RailTopoModel (as designed for railML 3) allows them to be handled separately without relation to the topology (for instance to describe an asset, which has not been fitted to the infrastructure yet). Closer specification of these net entities is provided by the railML 3 infrastructure schema. Together with the entities, their location in relation to net elements and positioning systems is managed. An entity location can be a spot location, linear location or area location, while an entity is allowed to have several locations (including both different and the same types). [5, 8]

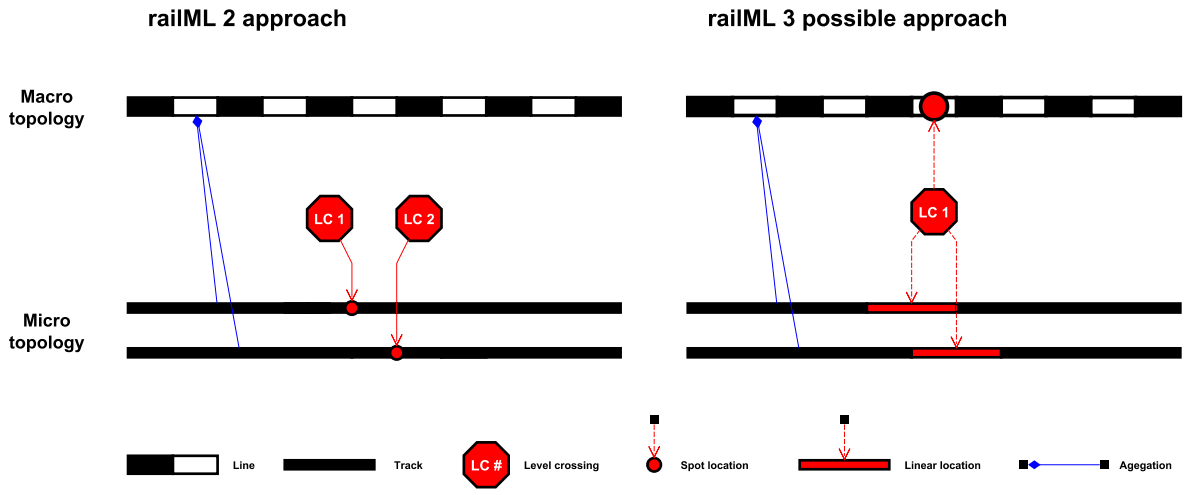

Figure 2. Illustration of different approaches in locating of entities in the scope of railML 2 and railML 3.

4. railML Evolving Over Time

Currently, railML 3 is available for testing purposes only. In 2016, the railML 3.0.is4 alpha version was released containing xsd files (defining allowable structure of particular XML documents) especially regarding the RailTopoModel schema (introduced with the railML 3 release) and the infrastructure schema. [8] The railML 3.0 releases are designed for possible railML 3 users, who are asked to compare it to their needs and to formulate a "use case" defining what kind of data they intend to exchange.

These "use cases" [9] serve as the basis for further development of the railML 3 infrastructure schema. By means of that approach, the creators want to avoid accumulation of such data items which no one needs. The railML 3 infrastructure schema, as regards the 3.0.is4 release, consists of following subsections: the topology, the coordinates, the operation, the geometry, the assets and the infrastructure parameters (while the last one was removed with the 3.0.5 version). [8]

4.1. Topology

Concerning the previous railML versions, [10] topology information is stored within the particular track elements using mutual connectors situated at the track ends and track begins or within switch points. The points defining a switch are allowed to be situated both at the track begin and at a certain distance from the track begin. Such approach makes the model ambiguous, as it is not clear how to segment the tracks, not to mention confusing manner in which railML 2 describes the branching tracks of a switch. [7, 10]

Starting from the third version, the topology is considered separately and based on the abovementioned principles of RailTopoModel. Concerning the micro level, a connection between tracks is allowed to be situated at the track begin or at the track end only, which admits the values (expressed in relative units) of either 0 or 1. [8] Among others, lengths of tracks are not necessary to be given in order to describe "the pure topology" properly. The same approach may be used for any other level of detail. Nevertheless, there are still some questions under discussion within the railML community. RailML 3 should support precious description of the track geometry. It has not been decided yet, whether the geometry will be considered separately or if it should depend on the topology. [6]

4.2. Positioning Systems and Coordintes

In railML 3, the positioning systems are managed in the scope of the coordinates subsection.

Previous versions are only able to express position from the beginning of a track completed with information of absolute position and geographical coordinates. These are expressed as the attributes or subelements of the elements representing the object being placed. [10] The absolute position should represent traditional mileage, which indicated values are not guaranteed to correspond to real distances. It happens for historical reasons (due to reconstructions of railway lines) and it is caused by different lengths of parallel tracks in horizontal curves and in the rail branching areas. In railML 2, there is a mileage change element enabling to express mileage jumps and overlaps being a part of the track topology assigned to a particular track.

Within railML 3, the issue is addressed in more complex way, following the RailTopoModel principles again. A user is allowed to define different linear and geometric positioning systems using line, geometric and screen coordinates. The mileage irregularities are resolved by the means of binding different linear positioning systems. Regarding the intrinsic positioning system, inseparably associated to given net element, as already mentioned, the position is measured in relative values from 0 to 1). [6, 8, 10]

4.3. Net Entities: Operational Entities, Geometry Entities and Assets

The most of particular subsections of the railML 3 infrastructure schema deal with located net entities, to be fit to the topological layer. These are the operational entities, geometry entities and assets (in railML 3.0.is4 mostly representing technical facilities of railway and with 3.0.5 version renamed to functional assets in order to be distinguished from physical assets).

As regards railML 3.0.is4, there are following entities included, while users can add any others (not to be a part of the common basis) to meet their needs. [6, 8] In terms of further releases, the lists are expected to be extended, based on the "use cases". [9]

- OPERATIONAL ENTITIES
  - ▷ Line
  - ▷ Operational point
  - ▷ Border
  - ▷ Speed profile
  - ▷ Speed restriction
  - ▷ Track
- GEOMETRY ENTITIES
  - ▷ Horizontal curve
  - ▷ Gradient curve
  - ▷ Superelevation curve
  - ▷ Track geometry point
- ASSETS
  - ▷ Bridge
  - ▷ Buffer stop
  - ▷ Derailer
  - ▷ Isolation rail joint
  - ▷ Level crossing
  - ▷ Platform
  - ▷ Platform edge
  - ▷ Service section
  - ▷ Signal
  - ▷ Stop post
  - ▷ Switch
  - ▷ Train detection element
  - ▷ Train protection element
  - ▷ Tunnel

For comparison, there are two major groups of entities (the track elements group and the ocs elements group) in railML 2 creating data substructures of each track element. Their location within their parent track shall be given by means of the position attribute, expressing the distance from the beginning of a track in metres. In the case of linear character of the entity being placed, railML 2 uses two different approaches how to represent it; either to determine the spots where the data change (e.g. the radius changes, speed changes...), or to determine the spot location of the object itself, possibly completed with the attribute of length (e.g. the platform edges, bridges, level crossings...). In some cases, it is essential to determine the considered direction. [10]

The railML 2 way of description, indeed, sometimes implies data inconsistency. The data could be misunderstood. The fact that all entities are strictly allocated to one specific track may cause some difficulties concerning the infrastructure description. For instance, how to define and describe track circuits when only their borders can be marked out? Structural object as bridges, tunnels and level crossings often belong to more than one track. In the scope of railML 2, every affected track needs to involve an extra instance of related data classes, although the original real world object is always the same (as seen in the figure 2). [7, 10]

RailML 3, building on the RailTopoModel principles, offers solutions of these issues. Not only that it enables spot, linear and aggregated location of net entities, but it also allows them to be multiply located (see again the figure 2). Locating of all the net entities is managed in the same way, no matter what kind of entity it is. [8]

Every XML element representing a net entity can contain unlimited amount of spot, linear and aggregated location elements involving location spots and linear sections expressing different locations of the entity. Regarding the aggregated location, it can be expressed both by spots and linear sections defining an area within the railyard. The location can be also represented by point and linear types of coordinates. Their definition is taken from the GML geometry schema. It means that the net entities are already not necessarily assigned to one specific track. [7, 8]

The XML elements describing the entities of the same type are included in common container elements within relevant subsection. In some cases, more complex substructure is used (especially as regards the signals, nevertheless, some of their items could be possibly managed within the interlocking schema). Concerning the assets, their operating time, constriction time and blocked time is possible to be expressed, in addition. Prospectively, a link of these functional assets with physical assets to be a part of inventory management is intended to be ensured. [6, 8]

4.4. Infrastructure Paramteres

The last category, also enabling to describe characteristics of railway infrastructure, used in railML 3.0.is4 is the subsection of infrastructure parameters. The following (again extendable) list of the parameter classes is given, as regards the 3.0.is4 release. [8]

- INFRASTRUCTURE PARAMETRES
  - ▷ Track gauge
  - ▷ Speed electrification
  - ▷ Weight limit
  - ▷ Train protection
  - ▷ Clearance gauge
  - ▷ Train radio
  - ▷ Track alignment

In railML 2, the most of these parameters can be expressed in two different ways: through infrastructure attribute groups (to be assigned to a certain track) or as track elements (changes within a track). [10]

The railML 3.0.is4 approach is closer to the first attitude mentioned. The infrastructure parameters are managed in rather different way than the net entities. Every net element (as be a part of the topologic layer) can refer to unbounded number of infrastructure parameters representing specific track conditions. Besides, it means that the parameters are valid for the relevant net entity (e.g. a track or line section) as a whole. Except they cannot be positioned, the related XML elements are handled in the same manner as the net entities. [8]

Apparently, there was an effort to remove the redundancy. Unfortunately, it would mean that these global parameters cannot be changed within a particular net element. Indeed, in some cases it is not essential; however, for example, as regards the electrification, it could cause serious difficulties. For that reason, based on discussions within the railML community, [11] the infrastructure parameters subsection was cancelled with railML 3.0.5 (released in March 2017) and the entities included was moved to the functional assets group (thereby becoming net entities). At the same time, the network location was introduced in order to represent the intended function of the cancelled subsection. [8]

5. Selected Application Issues

The railML 3 exchange format is supposed to find application in interconnection of such IT systems as infrastructure registers, maintenance systems, evidences of assets, construction projects, visualisation of railway infrastructure (georeferenced documents, GIS applications, schematic track plans, longitudinal sections), simulations of railway operation, calculations of energy consumption, capacity planning, information systems for operative control, predictions of train running, automatic routing, automatic train control, automatic train operation and many others. Apparently, it is big challenge for developers to meet all the multiple needs. Certainly, the alpha release does not cover all the desired aspects, as surely as it will be gradually complemented reflecting particular use cases. [9]

5.1. railML for ATC?

As the infrastructure description used by railML is essentially based on real track lengths, not only to use absolute mileage which can be misleading, railML seems to be very useful tool to provide data for such applications that operate with actual distance travelled by a train. Among other applications, it is essential for the ERTMS / ETCS [12] as an automatic train control system using balises to be reference points for measuring the distance travelled.

The balises and balise groups are included in the railML 2 specifications, nevertheless, railML 3 have not been saturated with such entities yet. Perhaps, it is caused by the unresolved issue in which way to perform referencing between the balises and balise groups elements. [8, 10]

Except the balise locations, other additional data are needed, including locations of important objects, speed restrictions, slopes, track conditions (change of traction systems, powerless sections, non stopping areas, sound horn commands...) road suitability parameters (loading gauge, traction system, axle load category...). Many of them have already been included into the railML 3 infrastructure schema, nevertheless, some of them should be added or revised. [8, 11]

5.2. Transfer Times Planning

Concerning predictions of train running (and in some cases capacity planning as well), not only the railway infrastructure itself must be taken into account, but also some related facts are important. For instance, when ensuring correspondences among trains, transfer times are often critical. Among others, they depend on parameters of the paths connecting the platform edges. When there are platforms with level access, the transferring passengers are influenced by passing trains (and conversely).

This fact shall be taken into account when assigning the platform edges and tracks to the trains as well (for instance, not to allow a passing train run over the foot level crossing being used by passengers). Not to mention the vital importance of the foot level crossing location register for such application as the automatic train operation because of the need to ensure human safety. To cope with these cases, the model and the exchange format would need to be significantly extended.

6. Proposal of a RTM Based Passenger Movement Model

The abovementioned issue is more complex, as the passenger routes (connecting particular platforms together and with station building and other important locations in a station) consist of different elements (e.g. platform ramps, stairways, lifts, underpasses, overpasses, foot level crossings, footpaths). All these facilities could be expressed using railML 3 as net entities (although the railML 3 alpha release, for instance, enables to express the existence of a lift on a platform by the means of its attribute only).

These platform accessories and related facilities should be interconnectable and describable by such attributes expressing their usability by persons with reduced mobility and other important parameters so as to enable searching suitable routes connecting particular “passenger access nodes” (in the sense of significant spots assigned to platforms, station building and other net entities). Even such situations as a lift being out of service should be taken into account.

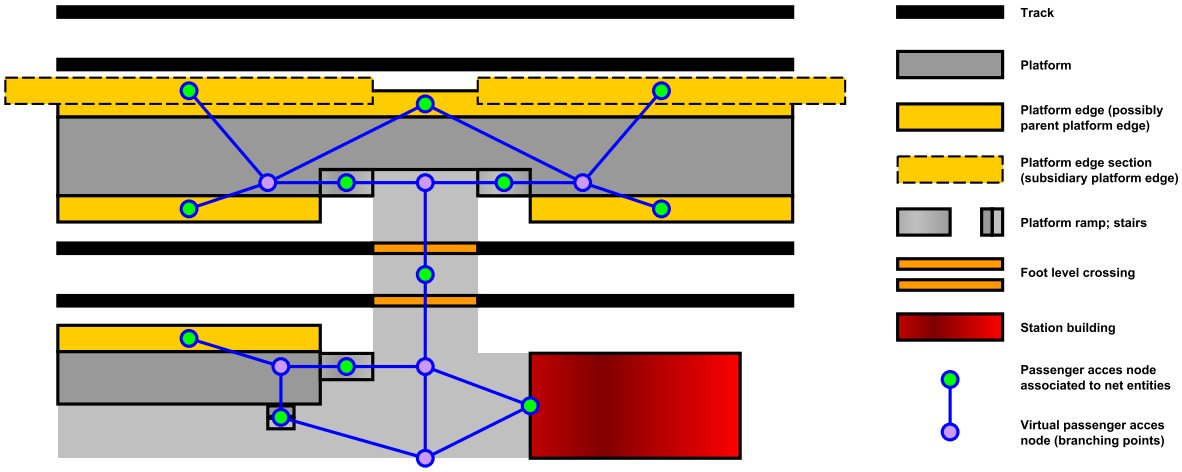

FIGURE 3. An author's proposal of RailTopoModel application in the scope of passenger routing issue exemplified for the case of a railway station equipped with platforms with level access.

There are several possible solutions how to meet these needs. One of them is to apply RailTopoModel again, this time not to connect pieces of railway infrastructure but the abovementioned "passenger access nodes", henceforth to be considered as a new sort of the net elements (see the figure 3).

This approach, in general, would also enable to model emergency access roads to railway tunnels, for instance, and to cope with other related issues.

7. CONCLUSIONS

RailTopoModel and the corresponding data exchange format railML 3 have a great opportunity to become very useful tools in describing railway infrastructure and ensuring the rail data sharing. Involving also the schemas of timetable, rolling stocks and prospectively interlocking, the railML language covers almost the whole range of the railway branch.

Although the railway system is based on relatively similar principles across the whole Europe, there are many nationally specific details that should be addressed in a coordinated manner in order to make the constantly evolving railML system universal enough to enable different software application to consistently communicate with each other. To make the system as usable as possible, the potential users should formulate their "use cases" [9] and be in contact with the railML community creating the specifications. Then they should be able to influence the final form of the railML 3 first public release in order to be suitable for their needs.

Nevertheless, the very special matters can be solved individually by the means of user-specific extensions. Within this document, some of possible extensions were outlined as well.

REFERENCES

[1] railML. *railML* [online]. `https://www.railml.org/`.

[2] railML. *The railML subschemas* [online]. `https://www.railml.org/en/user/subschemes.html`.

[3] railTopomodel. *RailTopoModel* [online]. `http://www.railtopomodel.org//`.

[4] UIC. *Feasibility Study UIC RailTopoModel and data exchange format* [pdf], 2013. `http://www.railtopomodel.org/files/download/RailTopoModel/270913_trafIT_FinalReportFeasibilityStudyRailTopoModel.pdf`.

[5] UIC. *UIC International Railway Standard* [pdf], 2016. `http://www.railtopomodel.org/files/download/RailTopoModel/270913_trafIT_FinalReportFeasibilityStudyRailTopoModel.pdf`.

[6] railML, UIC. *7th RailTopoModel and 30th railML Conference*, Paris, 2016.

[7] V. P. Kolmorgen, C. Rahmig. *railML/RTM Explanation Session*, Prague, 2017.

[8] railML. *railML schema version 3.0.is4* [xsd], 2016.

[9] railML. *Use Cases* [online]. `https://www.railml.org/en/user/use-cases.html`.

[10] railML. *railML schema version 2.3* [online], 2016. `https://www.railml.org/files/download/schemas/2016/railML-2.3/documentation/railML.html`.

[11] M. Karlsson, C. Rahmig. *[railML3/alpha] Modelling of track conditions* [online], 2016. `http://www.railml.org/forum/index.php?t=msg&th=474&start=0&`.

[12] ERTMS. *The European Traffic Management System* [online]. `http://www.ertms.net`.