TECHNICAL SPECIFICATION

ISO/TS 4398

First edition
2022-09

Intelligent transport systems — Guided transportation service planning data exchange

Reference number
ISO/TS 4398:2022(E)

Contents

Page

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/TC 204, *Intelligent transport systems*.

Any feedback or questions on this document should be directed to the user’s national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

Introduction

The purpose of this document is to facilitate the planning of railway operations between organizations in the transportation sector (communication between interested parties).

The aim is to provide a common format for necessary railway data exchange between interested parties in the industry during the conceptual, strategic and tactical phases of railway service planning.

Examples of roles of interested parties are:

— railway authorities;

— public transport authorities;

— train operators;

— infrastructure managers;

— rolling stock companies;

— suppliers to the railway sector (rolling stock, signalling systems, etc.);

— consultants.

These roles can be fulfilled by separate entities or by different units within one integrated entity or company.

This document covers various planning aspects, including the following:

a) **Conceptual planning:** Years in advance of construction of the new or improved infrastructure.

b) **Strategic planning:** Usually for new timetable concepts, new rolling stock or improved infrastructure, more than 15 months before implementation of new annual timetable, including:

— tendering process for passenger train operators (calculating tenders);

— feasibility studies for commercial train operators (freight and passengers);

— temporary infrastructure capacity restrictions.

c) **Tactical planning:** Usually for producing a new timetable, typically for construction of the annual timetable, including:

— train path and capacity requests (train operators);

— train path and capacity allocation (infrastructure managers).

During planning there is a continuous need for exchange of (machine readable) information between different stakeholders.

These stakeholders use different applications for their internal processes (runtime calculations, rostering, temporary capacity restrictions, etc). Within large organizations there can also be different applications in use, where effective exchange of information is essential.

A common (standardized) format for exchanging information between different applications will reduce time-consuming manual work and will improve accuracy. Railway Data Exchange (RailDax) is a format standard for exchanging railway data between applications.

The RailDax format is intended to be used by railway and transportation authorities, infrastructure managers and train operators during long-term planning, tendering processes, commercial evaluations and the yearly capacity allocation processes leading to the annual timetable.

RailDax has been developed in parallel with the data exchange language railML 2.5 (Railway Modelling Language 2.5), which is managed by railML.org.[1)]

Figure 1 illustrates the use of RailDax for conceptual, strategic and tactical planning of railway services and operations, leading to an annual timetable. Other exchange formats will be more suited for the exchange of network and timetable data for public (customers') travel planning and ticketing solutions.

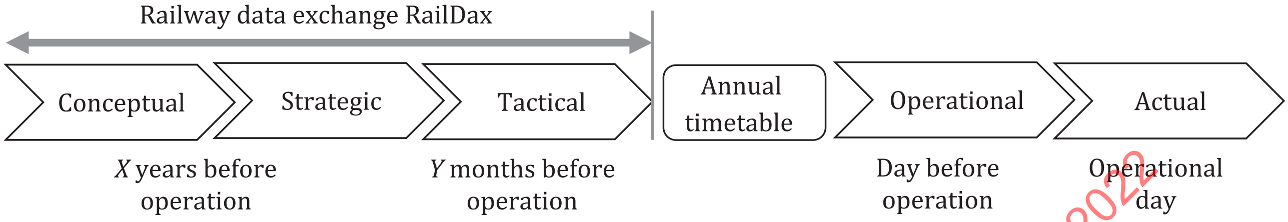


Figure 1 — RailDax as a railway data exchange format for conceptual, strategic and tactical planning

RailDax is not intended as a data exchange format for applications serving the following purposes:

1) asset development and maintenance applications for infrastructure and rolling stock;

2) public travel plan and fare management.

The interface between RailDax and purpose 2) will typically be the annual timetable.

Train operators can be legally obliged to publish in other formats, for example in Europe, the Network Timetable Exchange (NeTEx) for public travel plan and fare management, and Telematics Applications for Passenger services (TAP) and Technical Specifications for Interoperability (TSI) for slot ordering at national access points. The relationship between RailDax and these specifications is explained in Annex A.

When launching the RailDax project it was deemed necessary to base the format on a mature (proven in use) data exchange language and to cover the RailDax use cases. Based on a study, railML 2.x was chosen. For the same reason, RaiDax is developed as a pair to railML version 2.5. The development of possible future revisions of RailDax may be considered to pair with later railML versions or other formats.

1) railML® and the logos of railML.org are copyrighted by railML.org e.V., as they are registered at the European Union Intellectual Property Office as a trademark with the number 12576492. This trademark is provided for reasons of public interest or public safety. This information is given for the convenience of users of this document and does not constitute an endorsement by ISO.

Intelligent transport systems — Guided transportation service planning data exchange

1 Scope

This document specifies an open, XML-based data format which enables an efficient and unambiguous exchange of static information concerning the operational functionality of the infrastructure, rolling stock and timetable of a track-bound transportation system. The main objective is to enable heterogeneous railway applications to communicate with each other.

The purpose of the data format is to facilitate common (integrated) planning of track-bound operations between organizations in the transportation sector.

Railway Data Exchange (RailDax) serves as a data exchange format between applications used for railway service planning: connecting information about infrastructure, rolling stock and timetable basics which are necessary for capacity management and timetable planning.

This document describes infrastructure and rolling stock from an operational perspective. To achieve this, the infrastructure and rolling stock is described with a clearly defined meaning from an operational perspective.

RailDax will typically be used by railway authorities, train operators, infrastructure managers and suppliers to the railway industry for communication between applications serving the use cases listed in Clause 7 in this document.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/TS 14812, *Intelligent transport systems — Vocabulary*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/TS 14812 and the following apply.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at https://www.electropedia.org/

3.1
RailDax file
railway data exchange file

Note 1 to entry: The RailDax file is constructed according to the principles of this document.

3.2
infrastructure
tracks, switches, engineering structures (bridges, tunnels, etc.), platforms, zones of access (including the needs of persons with reduced mobility), safety and protective equipment

[SOURCE: railML RailGlossary[10]]

3.3
infrastructure capacity
potential to schedule train paths requested for an element of infrastructure for a certain period

[SOURCE: European Directive 2001/14/EC[5]]

3.4
capacity
maximum number of trains which can be planned to move in both directions over a specified section of track in a 24-hour period

[SOURCE: railML RailGlossary[10]]

3.5
guided transportation
track-bound public transport, such as railway, metro and tram systems

3.6
border
location where a line or frontier area separates the railway network between legal jurisdictions, e.g. countries, regions, tariff zones, infrastructure managers, project areas, etc.

[SOURCE: railML RailGlossary[10]]

3.7
train path
infrastructure capacity needed to run a train between two places over a given time-period

[SOURCE: European Directive 2001/14/EC[5]]

3.8
train route
course of a train over the tracks through its defined stations

Note 1 to entry: The train's route usually spans its entire service line from first to last operational point (station).

3.9
speed change
change of speed on a given location on the track

[SOURCE: railML RailGlossary[10]]

3.10
signal route
specific directional course over the tracks from one signal to the next for a train's safe passage

3.11
mileage
classical location framework in railways, given in form of a kilometre value

[SOURCE: railML RailGlossary[10]]

3.12
network
entire railway infrastructure owned or managed by an infrastructure manager

[SOURCE: European Directive 2001/14/EC[5]]

3.13
network statement
statement which sets out in detail the general rules, deadlines, procedures and criteria concerning the charging and capacity allocation schemes

Note 1 to entry: The statement shall also contain such other information as is required to enable application for infrastructure capacity.

[SOURCE: European Directive 2001/14/EC[5]]

3.14
timetable
schedule listing the times at which certain events, such as arrivals and departures at a transport station, are expected to take place

Note 1 to entry: The timetable defines all planned train and rolling-stock movements which will take place on the relevant infrastructure during the period for which it is in force.

[SOURCE: railML RailGlossary[10]]

3.15
train operator
railway undertaking
public or private undertaking, the principal business of which is to provide services for the transport of goods and/or passengers by rail with a requirement that the undertaking ensures tractionNote 1 to entry: This also includes undertakings which provide traction only.

[SOURCE: European Directive 2001/14/EC[5] modified — Preferred term “train operator” added.]

3.16
infrastructure manager
any body or undertaking that is responsible in particular for establishing and maintaining railway infrastructure

Note 1 to entry: This may also include the management of infrastructure control and safety systems. The functions of the infrastructure manager on a network or part of a network may be allocated to different bodies or undertakings.

[SOURCE: European Directive 2001/14/EC[5]]

3.17
overlap
section beyond a stop signal, or a stopping point in a continuous signalling system, which must be kept clear to avoid the risk of collision should a train inadvertently run past the signal or the stopping point

[SOURCE: IEC 60050-821:2017, 821-01-21]

3.18
product
item being offered by a train service, passenger or freight

3.19
line
list of railway tracks between two major operational points used for regular railway operation

[SOURCE: railML RailGlossary[10]]

3.20
station
area defined by one or more station boundaries, where an exit main signal on a remote-controlled line, or a manually given signal on a line without remote control, signals whether the next block is free

Note 1 to entry: A station is a place where operational trains can begin and end operations.

[SOURCE: railML RailGlossary[10]]

3.21
switch
unit of track comprising of two fixed rails (stock rails) and two moveable rails (switch rails) used to direct vehicles from one track to another track

Note 1 to entry: The term “point” is sometimes used for this concept.

[SOURCE: railML RailGlossary[10]]

3.22
tunnel
covered, horizontal passageway for railway transport through or under an obstruction

[SOURCE: railML RailGlossary[10]]

3.23
bridge
structure built for the explicit purpose of spanning and providing passage for railway transport over a gap or a barrier, i.e. a river, chasm, road, lake etc.

[SOURCE: railML RailGlossary[10]]

3.24
derailer
fixed device which, when placed on the rail, derails the wheels of a vehicle, and serves to protect a converging line

[SOURCE: railML RailGlossary[10]]

3.25
stop signal
signal at route exit where a train is usually required to stop

[SOURCE: railML RailGlossary[10]]

3.26
balise
wayside transmission unit that uses the magnetic transponder technology

Note 1 to entry: Its main function is to transmit or receive signals through the air gap. The balise is a single device mounted on the track, which communicates with a train passing over it.

[SOURCE: railML RailGlossary[10]]

3.27
balise group
one or more balises that on a higher system level together create a quantity of information related to the location reference in the track, the direction of validity of data, and train protection information

Note 1 to entry: The single balises form together a functionality that is described in the balise group.

Note 2 to entry: This is the location in the track where spot transmission occurs. The telegrams transmitted by all the balises of a group form a track-to-train message.

[SOURCE: railML RailGlossary[10]]

3.28
level crossing
location where railway and other traffic types cross each other at the same level (for example, without overpass or underpass)

Note 1 to entry: Level crossings may be technically secured or non-technically secured. Technically secured level crossings must have gates, barriers, traffic lights or other means of securing.

[SOURCE: railML RailGlossary[10]]

3.29
rolling stock
collective term for the rail fleet

Note 1 to entry: This term is sometimes used for one vehicle.

Note 2 to entry: It describes all the vehicles that are used on a railway track. It usually includes both powered and unpowered vehicles, for example locomotives, hauled passenger vehicles and freight vehicles (coaches and wagons), diesel units, electric units and service stock. The term is sometimes used to refer only to non-powered vehicles, thus excluding locomotives. The term contrasts with fixed stock (infrastructure), which is a collective term for the track, signals, stations and buildings etc. necessary to operate a railway.

[SOURCE: railML RailGlossary[10]]

3.30
vehicle roster
list or plan showing turns of duty for vehicles (rolling stock) in an organization

3.31
train
movement of a single vehicle or a number of coupled vehicles/units operating on a guided ground transport system

3.32
train part
basic part of a train with the same characteristics such as formation, and that is constant during an operating period

Note 1 to entry: The train part includes the actual information regarding the path of the train as a sequence of operation or control points together with the corresponding schedule information.

4 Abbreviated terms

ERA European Union Agency for Railways

IS infrastructure

NeTEx network timetable exchange

OCP operational control point

RailDax railway data exchange

railML railway markup language[9]

RINF Register of Infrastructure

NOTE 1 This is the main tool for describing the static rail network characteristics and capabilities as required by Directive (EU) 2019/777 on rail Interoperability.[11]

RS rolling stock

TAF telematics applications for freight services

TAP telematics applications for passenger services

TSI technical specifications for interoperability

NOTE 2 TSI define the technical and operational standards which are required to be met by each subsystem or part of the subsystem in order to meet the essential requirements and ensure the interoperability of the railway system of the European Union.

TT timetable and vehicle rostering

TVD track vacancy detection area

UC use case

UML unified modelling language

XML extensible markup language[8]

5 Modelling concepts

5.1 General

This clause describes modelling of infrastructure and timetable concepts as connections of infrastructure and timetable elements.

Information about and mapping with other systems is included in Annex A.

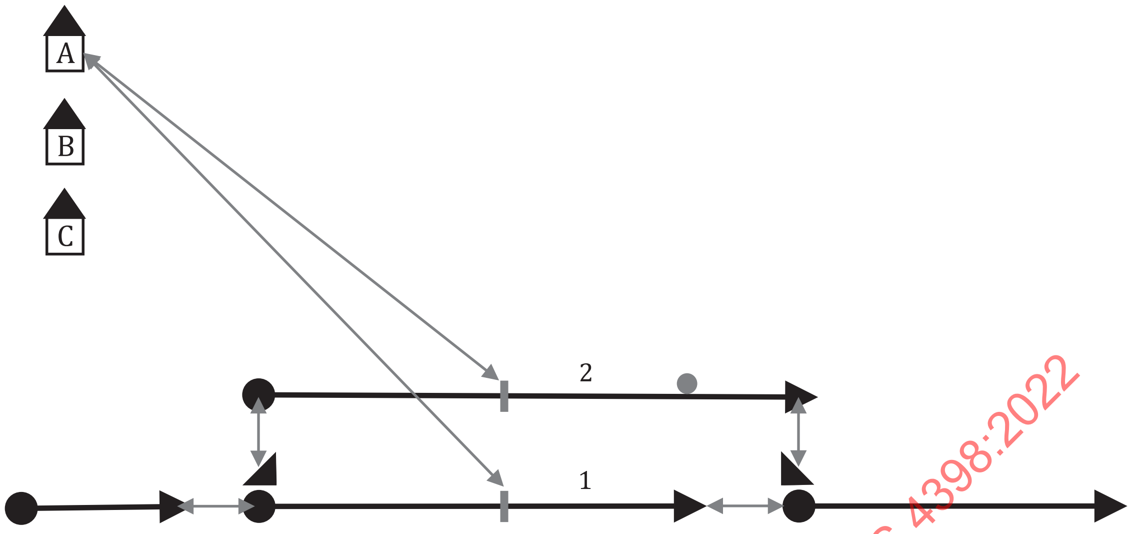
5.2 Concepts infrastructure

5.2.1 Introduction

Railway models can have different aggregation levels. The RailDax model has a microscopic level where `<track>`s are the elementary units (e.g. for infrastructure planning) and a macroscopic level where the `<ocp>`s are the basic elements (e.g. for timetable planning).

As far as the infrastructure elements' references/connections are concerned, all the `<trackElements>` and `<ocsElements>` are objects placed on the on the `<track>`. Their position on the track is specified with the pos attribute, while their mileage on the railway line is given by the absPos attribute.

Railway lines are modelled as ordered sequences of tracks, while operational control points are unordered elements without any reference to their position in the topology. The link between tracks and operational control points (OCPs) is given by `<crossSection>` elements which specify the connection and the position of the station with respect to the track.



Key

- track begin
- track end
- switch
- reference connection
- Cross-section
- operational control point (OCP)
- track element (e.g. a tunnel)

Figure 2 — Connections between `<track>`s and between `<track>`s and `<ocp>`s

5.2.2 Connections between tracks

5.2.2.1 Modelling of a switch

A switch is a connection element, which hierarchically belongs to exactly one track. However, it is connected to one more track in the form of the switch's branch. The switch is situated along the track via its relative position given by the pos attribute. Following the principle of a classic node-edge-model, the switch should only be situated in the track begin (pos = 0) or in the track end (pos = length of track). Thus, a switch always marks a change of track, no matter if the main or the branching way is chosen. Additionally, it is useful to connect the switch element with that track, from which the switch can be travelled facing.

5.2.2.2 Connections at a switch

5.2.2.2.1 General

The main track or "through track" of the switch is modelled as a connection element in the beginning or end of the track, which refers to a connection element in the beginning or end of another track. So, the main track is not included in the list of connections within the switch element, but only the branching tracks are modelled there. Most switches have exactly one branching track, but in case of a three-way-switch, there would be two connection elements in the switch. Considering the connection's attributes, orientation and course, there are four different combinations possible:

1) **Incoming** branch from **left;**

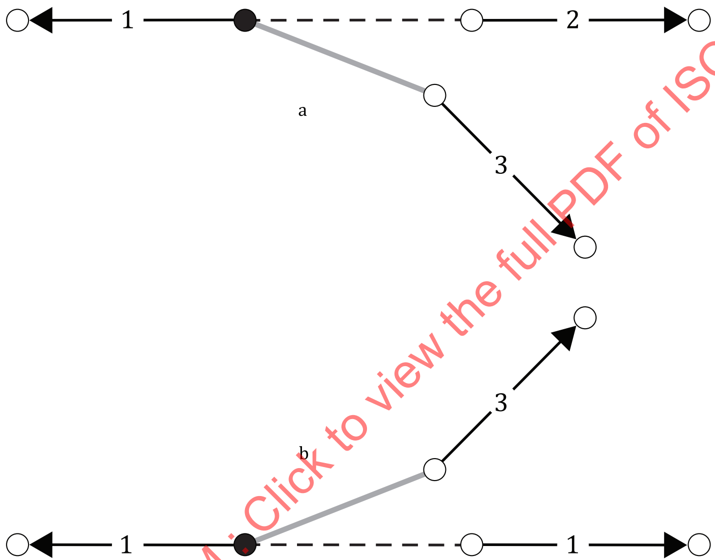
2) **Incoming** branch from **right;**

3) **Outgoing** branch to the **left;**

4) **Outgoing** branch to the **right.**

5.2.2.2.2 Incoming branch

Figure 3 shows a switch with an incoming branch track. The switch element is situated in the beginning of track 1. The main course of the switch continues with the beginning of track 2. Track 3 is connected with the branching course, which is of orientation “incoming” from course “left” or “right” related to the orientation of track 1.



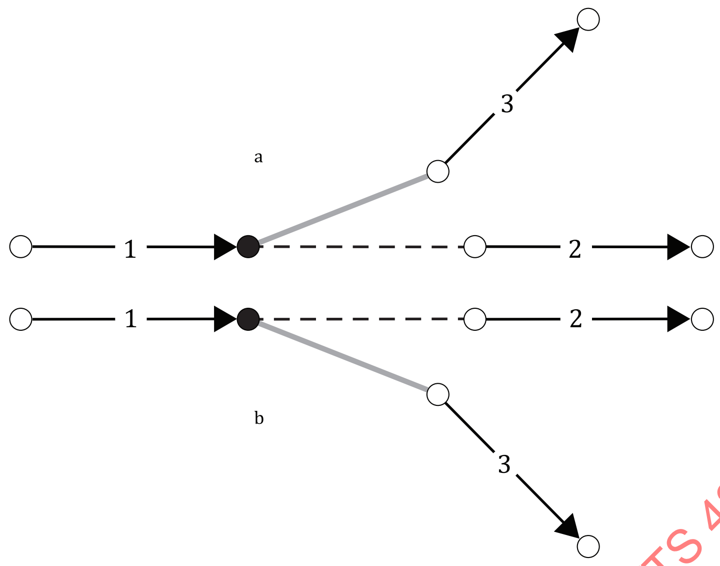
a Incoming left.

b Incoming right.

Figure 3 — Incoming branch

5.2.2.2.3 Outgoing branch

Figure 4 depicts a switch with an outgoing branch track: The switch element is situated in the end of track 1. From there, the main course of the switch continues with the begin of track 2. Track 3 is connected with the branching track, which is of orientation “outgoing” to the “left” or to the “right” related to the orientation of track 1.



a Outgoing left.
b Outgoing right.

Figure 4 — Outgoing branch

5.2.3 Temporal availability of infrastructure elements and speed profiles

The temporal aspect of the restriction is defined by four attributes:

1) `operatingPeriodRef`

`xs:IDREF`: Reference to an <`operatingPeriod`>. It defines the dates on which the restriction begins. If the <`operatingPeriod`> contains more than one day, it is a periodic restriction.

2) `startTime`

`xs:time`: Start time of the restriction. If no `startTime` is specified, 0.00 is assumed.

3) `endTime`

`xs:time`: End time of the restriction. If no `endTime` is specified, 24.00 is assumed.

4) `endDayOffset`

`xs:nonNegativeInteger`: Number of midnight transitions over which the restriction lasts. If no `endDayOffset` is specified, a value of 0 is assumed.

The duration of a restriction is calculated from the difference between `endTime` and `startTime`, and the number of midnight transitions multiplied by 24 hours:

```
duration = endTime – startTime + 24h * endDayOffset
```

If the end time is before the start time, `endDayOffset` needs to be greater than 0, since the restriction lasts at least once before midnight.

5.2.4 Defining track usage of a train in stations

In general, there are two ways of describing which station tracks are used by a train:

1) referring to infrastructure elements (station tracks, platform edges), which can describe the elements in more detail; or

2) using the attribute trackInfo of the element <ocpTT> for track related information

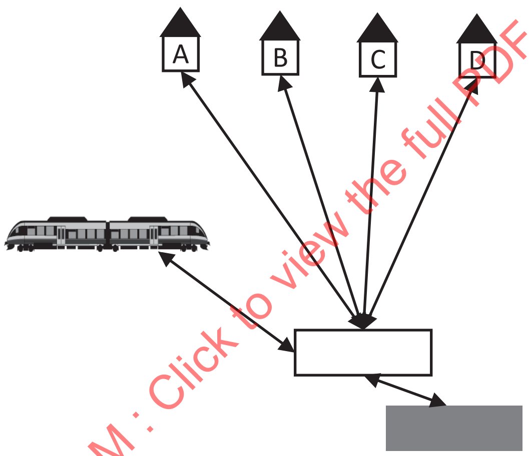
In case the infrastructure part of a RailDax file already includes information about station tracks, the first option should be used. The train schedule should refer to the infrastructure data. The two ways of describing station track usage should not be mixed within the same RailDax file.

5.3 Concepts timetable

5.3.1 Overview

The optimal infrastructure model for a RailDax timetable view is the macroscopic infrastructure model, which considers tracks as connections between OCPs. However, the RailDax infrastructure model described in the Infrastructure subschema is a microscopic one, where tracks are seen as connections between switches.

A <trainPart> is described as passing a sequence of several operation or control points <ocpTT>. The related <ocp> elements and the used parts of <track>s in between are part of the <infrastructure> subschema. The way from one <ocpTT> to the next is described in the sub-element <sectionTT>. Part of the description is the reference to the infrastructure <track> element by using <trackRef>, which allows the possibility of switching between the two different levels of detail.



Key

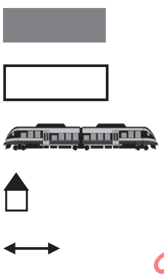


Figure 5 — Link between <ocpTT>s, <formations> and <trainPart>s

5.3.2 Train types, categories and passenger usage

5.3.2.1 Finding out whether a train or train part is for passengers or freight

Currently there is no attribute at <train> nor <trainPart> from which it can be directly deduced whether a train (part) is for passengers or freight. Rather, the attribute categoryRef of the element <trainPart> has to be "traced back".

The categories are summarized in an own list at <timetable>. There, the attributes trainUsage and deadRun can be used to determine whether a type of train is normally used for passengers or freight.

It is up to the reading software how it deals with such attributes missing. It can opt for a default case, or ask the user, or force the usage of these attributes, e.g. by providing an error message.

The attribute categoryRef, especially of <trainPart>, refers to train categories which are commonly known as products in practice. They are normally used for publishing. It has to be noted that a train can consist of more than one product at the same time because different train parts may reference different products.

In contrary to products there is the operational train category. This is something internal, not to be published. Also, a train can have only one operational category at the same time. For this kind of train category, there is the attribute categoryRef at the element <train> in RailDax. It is sub-placed below <trainPartSequence> because the operational category may change between different sections of the train's route. The many long-distance trains which run empty before and after their published route are examples in practice for such changes.

The attribute categoryRef at <train> for the operational category is normally used at operational trains only (elements <train> with occurrence type='operational').

Currently there is no explicit distinction between *product* and *operational category* in RailDax – an element <category> can represent an operational category (i.e. referenced by a <train>) as well as a product (i.e. referenced by a <trainPart>).

Here again it is up to the reading software how to deal with potential contradictions of this contextual redundancy in case a train (part) is shown, for example, as passenger hauling by its <trainPart>. categoryRef and at the same time as non-passenger hauling by its <train>.categoryRef. It is recommended to "consult" the product for traffic properties (such as passenger hauling) and to "consult" the operational category for operational properties (which may be for instance categoryPriority).

5.3.2.2 Overriding of places

To explicitly declare a train or train part as passenger hauling, it shall stringently reference a properly defined <category> by its categoryRef. But, in this relation, it has to be considered that sometimes the so-called overriding of places happens. Here, the element <passengerUsage> of <trainPart> is used to override (correct) the place capacities inherited from a formation (by the attribute formationRef). If the place capacities are corrected to 0 by this method, this implicitly includes that the train part cannot be passenger-hauling in a certain sense, even if its categoryRef says something else. This method is used, for example, to declare some carriages in a train to be closed. (Some carriages = part of the train that may normally be used by passengers, but is closed because it does not fit at a given platform.)

Though it can alternatively be expressed by assigning an "empty-run product" to this train part, it is not always common to do so. Sometimes it is up to the custom of the user and therefore cannot be forced by the writing software. Also, normally only some parts of the train should be set to 0, not all parts. In any case, when determining such properties as passenger, freight, or public, it is recommended for reading software to also check any possible overriding of places and to act accordingly.

5.3.3 How to reference infrastructure[12]

A train part is described as passing a sequence of several operation or control points <ocpTT>. The related <ocp> elements and the used parts of <track>s in between are part of the <infrastructure>. The way from one <ocpTT> to the next is described in the sub-element <sectionTT>. Part of the description is the reference to the infrastructure <track> element by using <trackRef>.

Different usages of the infrastructure schema are as follows:

— The macroscopic infrastructure model considers tracks as a connection between OCPs. This model is typical for a timetable view.

— The microscopic infrastructure model considers tracks as a connection between switches. This model is the preferred one for a simulation tool or a signal box view of the infrastructure.

To create unique references from a train part to the infrastructure with these two models, it is assumed that:

— in the macroscopic model in RailDax there are 'long' tracks, i.e. the tracks cross several switches, junctions, or OCPs;

— in the microscopic model in RailDax there are 'short' tracks, i.e. the tracks are only defined between two switches or junctions but do not cross any switch or junction.

5.3.4 Midnight overrun[13]

5.3.4.1 General

Midnight overruns generally have two aspects to be considered.

— The time of two consecutive arrivals/departures jumps back (e.g. from '23:57:53' to '00:00:19'). This has to be considered when calculating total run times of a train run.

— Arrivals/departures of a `<trainPart>` after midnight will take place on a different day than before midnight related to its `<operatingPeriod>`. Since there is only one `<operatingPeriod>` per `<trainPart>`, there has to be a means to determine the actual days, when a `<trainPart>` arrives or departs at a certain `<ocpTT>`.

It is not desirable to split the train's run into one `<trainPart>` before and another after midnight and to create separate `<operatingPeriod>`s for both. Instead, the different operating days of the `<trainPart>` before and after midnight are expressed using different values for the attributes arrivalDay and departureDay. These attributes represent the day offset of the arrival/departure of a `<trainPart>`'s `<ocpTT>` in respect of its `<operatingPeriod>`. For operational trains (`<train>` with type = 'operational') they are a counting of the number of midnight overruns relative to a reference place (e.g. departure). They are optional with default value 0, e.g. they do not have to be written as long as a train doesn't run over midnight. But, after the first run over midnight, they are written with values > 0.

5.3.4.2 Reference place for day indexes or midnight overruns[13]

— It is intended that the `arrivalDay`/`departureDay` counting starts with 0 at the first departure of a train.

— Therefore, the value -1 can occur in rare cases of an arrival before the first departure ("arrival from nowhere", from outside the scope of the Railway Data Exchange (railDax) file).

— In case a train did already run over midnight before the first departure in the railDax data, its first `<ocpTT>` may have an `arrivalDay`/`departureDay>0`.

— If a `<train>` consists of several `<trainPart>`s which are sequentially linked, the day counting normally refers to the first departure of the whole `<train>`. So, it may occur that a single `<trainPart>` already starts with day counting > 0.

— A "back-jump" of the day counting may occur especially from the view of a commercial train (`<train>` with **`type`**='commercial'): This means that the train first refers to `<trainPart>`s which did run over midnight and later refers to `<trainPart>`s which did not run over midnight.

5.3.5 Reversing trains and formations[14]

5.3.5.1 General

In the `<timetable>` scheme, there are two attributes concerning the turn of the running direction of a train, train part and/or formation: `<ocpTT>`.trainReverse and `<formationTT>`.orientationReversed. They are considered to be redundant and may be mixed. Their differences are explained in the following subclauses.

5.3.5.2 Concerning the meaning of <ocpTT>.trainReverse

<ocpTT>.trainReverse denotes that the train(part) changes direction, no matter whether the formation changes, reverses or neither.

— `trainReverse` with change of formation = e.g. running around with the engine, also known as headshunt;

— `trainReverse` with reversing of formation = train(part) of several MUs or push-pull train;

— `trainReverse` with neither of the above = train(part) consists of a single vehicle (MU or engine).

This information is mainly intended for passenger information (systems) which sometimes print a sign such as <-> to notify the passenger where the running direction of the train changes.

5.3.6 Train coupling and sharing[15]

This subclause explains basic concepts of coupling and sharing trains within the timetable subschema. To couple train parts (trains) and share them at another station is a typical phenomenon in timetable calculations for multiple reasons. The principle of "train coupling and sharing" requires the same operating days of two trains.

The term "train coupling and sharing" describes the situation where two or more parts of a train (nowadays mostly multiple units) run joined at one section and separated at another section of line.

NOTE This principle has been well-known throughout the world of railways for a long time.

One of the base philosophies of RailDax is to satisfy the requirements of everyday railway operation such as doubling of trains for raised capacity demands, direct through-coaches, and even train coupling and sharing. Since railML's Version 2, this is achieved by "deconstructing" trains in smallest, atomic fragments. These atomic fragments of trains are called train parts.

The actual train information such as times and vehicles are properties of the train parts. A <train> structure only joins train parts to trains, but besides this normally holds no additional information.

While "operating days" or "no. of vehicles" may change during a train's run, all such properties of a <trainPart> stay constant.

The element <train> can describe either an operational or a commercial train. This is defined by the attribute `type` which either is `operational` or `commercial`.

The characteristic attribute of operational trains is that at one moment there is only one train allowed at a section of line track. This train clearly has to be defined by one "primary key" (called 'head code' or 'train number'). These aspects come partly from reasons of security (as for instance communication between signal boxes).

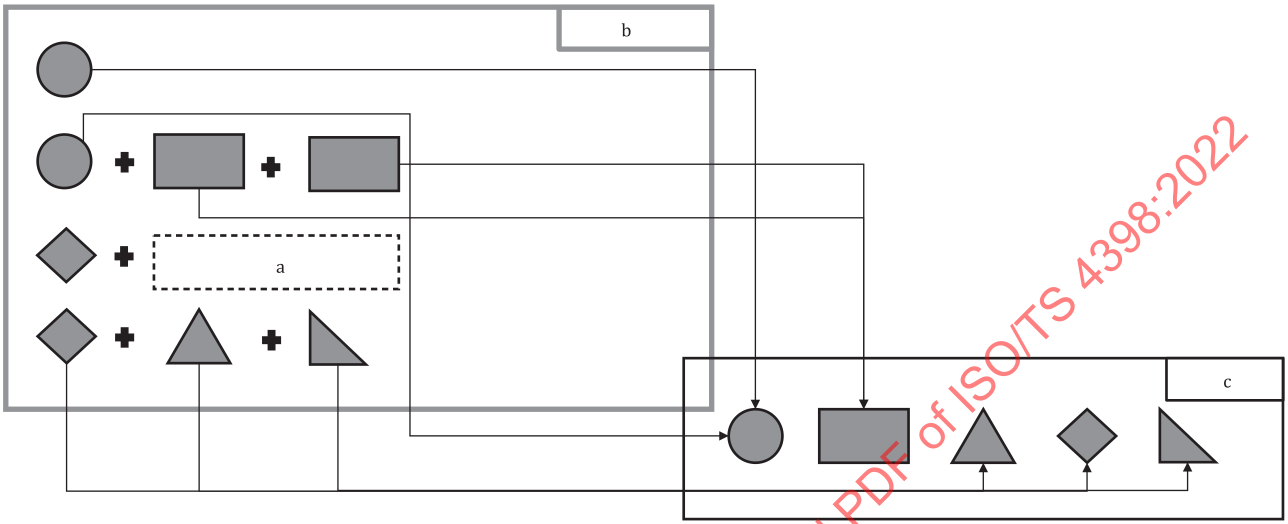
5.4 Concepts: rolling stock

5.4.1 Formations

The <formations> element can contain an infinite number of children elements <formation>, which form the data sets. The element <formation> contains all data related to a composition of single vehicles forming a complete train or a part of a train. It is not excluded to describe a train consisting of only one single vehicle.

5.4.2 Vehicles

The <vehicles> element can contain an infinite number of children elements <vehicle>, which form the data sets. The element <vehicle> contains all data related to a single vehicle or a vehicle family. The item is referenced via its attribute ID. In case of a vehicle family, all data common for all vehicles of this family can be stored. The data set of a single vehicle can refer to the common values and define more or deviating characteristics valid only for this particular vehicle.



Key

vehicleRef

locomotive type 1

passenger wagon

locomotive type 2

freight wagon type 1

freight wagon type 2

a Unknown vehicles.

b Formations.

Figure 6 — Link between <vehicle>s and <formation>s

6 RailDax high level description

6.1 General overview

Railway data for exchange between railway applications by RailDax are described in three subschemas for productive use: infrastructure (IS), rolling stock (RS) and timetable and vehicle rostering (TT).

The infrastructure subschema is focused on the description of the railway network infrastructure including all its various facets that are needed by the data exchange applications. In particular, the infrastructure schema contains information about topology, coordinates and geometry.

Railway infrastructure elements enclose a variety of railway-relevant assets that can be found on, under, over or next to the railway track, e.g. balises, platform edges and level crossings. This schema also encompasses elements that are closely linked with the railway infrastructure, but that "cannot be touched", e.g. speed profiles and track conditions.

The rolling stock schema provides a container for all data about any kind of railway vehicle including locomotives, multiple units, passenger and freight wagons. The second part of the schema enables the combination of single vehicles to formations as a fixed composition within a train or an entire train. It is intended to use this data schema for vehicle management as well as for detailed run-time calculations.

The timetable subschema is used for all data necessary to exchange any kind of timetable for operational or conceptional purposes including information about operating periods, trains, connections and rostering.

An index of all element names is included in Annex B.

6.2 XML methodology

RailDax is implemented in a model in a specific language that supports declarative modelling constructs. RailDax uses W3C XML schemas for this purpose. An XML document contains XML elements organized in a hierarchical tree structure from a common root element. The name of an XML element is commonly put in angled brackets (e.g. `<elementName>`), reflecting the code structure in an XML document.

Apart from the root element, all other elements belong to exactly one other element, its parent element. All elements belonging to the same parent element are the child elements of that parent, sometimes also referred to as sub-elements. In RailDax, each element either describes an object relevant for the use cases or forms a container for multiple child elements sharing the same type. One example of a container is the `<trainParts>` element (multiplicity 0..1) that contains multiple (1..*) `<trainPart>` elements.

There is no inheritance in the parent-child relationship. Child elements describe a part of the parent element in more detail relevant for its purpose. The properties of an object are described through XML attributes either belonging to the element representing the object, or to a child element representing a part of the object. XML attributes provide either a value or a reference to another element, and have a maximum multiplicity of 1, so repeatable properties are always specified through child elements. Child elements are also used by RailDax to group multiple attributes that describe the same topic. An example is the frequent repeatable child element `<additionalName>` that contains the attributes name, description and xml:lang.

6.3 RailDax high level UML

6.3.1 Overview

The RailDax unified modelling language (UML) schema contains the following sub-schemas, as illustrated in Figure 7.

- Infrastructure (IS)
  - This sub-schema is focused on the description of the railway network functional infrastructure.
- Timetable and vehicle rostering (TT)
  - This sub-schema is used for all data necessary to exchange any kind of timetable for operational or conceptional purposes.
- Rolling stock (RS)
  - This sub-schema is intended to be used for vehicle management as well as for detailed run-time calculations.

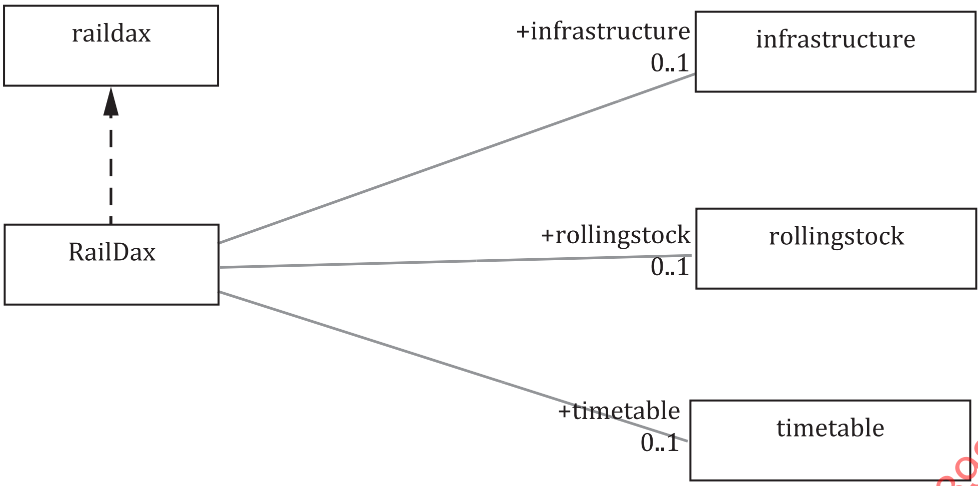


Figure 7 — RailDax high level UML

6.3.2 Infrastructure

6.3.2.1 Overview

The element `<infrastructure>` is part of the RailDax schema and contains the following child elements, as illustrated in Figure 8.

— InfraAttrGroups

  — The element `<infraAttrGroups>` is a container for pre-defined infrastructure attributes, which are referenced at the appropriate tracks.

— Tracks

  — The element `<tracks>` is a container for <`track`> elements.

— TrackGroups

  — The element `<trackGroups>` is a container that allows the definition of logically grouped tracks under various criteria. Examples for a track group can be:

    — all tracks belonging to a line;

    — all tracks belonging to a locallyControlledArea;

    — all tracks belonging to a station;

    — all tracks assigned to an interlocking;

    — etc.

— OperationControlPoints

  — The element `<operationControlPoints>` is a container for operation or control point elements.

— Controllers

  — The element `<controllers>` is a container for elements of the command and control systems domain which "cannot be touched in real life".

— SpeedProfiles

  — The element `<speedProfiles>` is a container for `<speedProfile>` elements.

— Routes

  — The element `<routes>` is a container for `<route>` elements.

— GenericAreas

  — The element `<genericAreas>` is a container for `<genericArea>` elements.

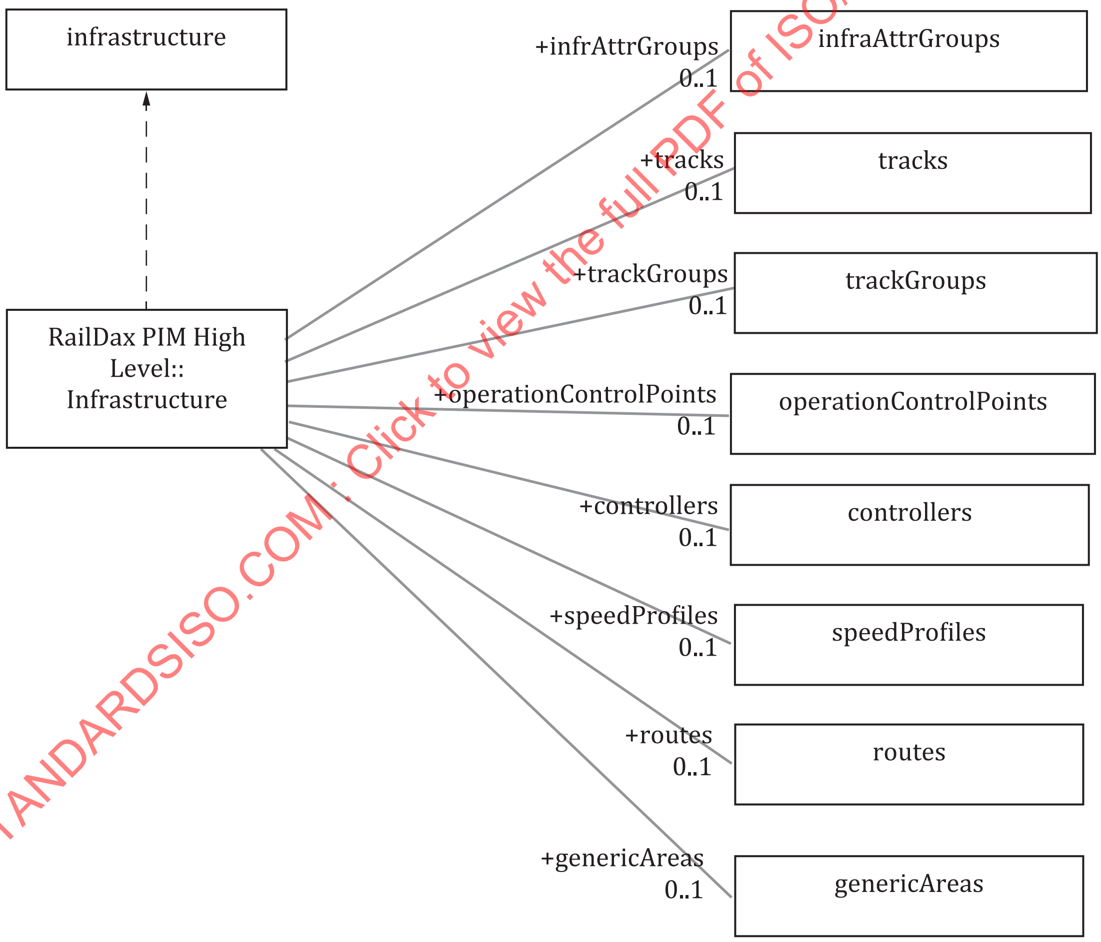


Figure 8 — Infrastructure

6.3.2.2 Controllers

The element <controllers> is a child element of <infrastructure> and contains the following child element, as illustrated in Figure 9.

- Controller
  - The element <controller> defines a facility that controls some track-side facilities. It may be something like an interlocking.

The element <controller> contains the following child elements:

- ocpRef
  - The element <ocpRef> is a reference from a controller to one or more OCP(s).
- trackVacancyDetectionArea
  - The element <trackVacancyDetectionArea> defines the track vacancy detection areas (TVD) controlled by this controller. The track vacancy detection area (also called track section) is an area which was created to serve the schematic track plan for signalling and a more detailed description of the route settings (<releaseGroup>).
- localOperationArea
  - The element <localOperationArea> defines the local operation areas controlled by this controller. The local operation area brings the assets in a special mode where they can be operated freely from on-site devices, e.g. button panel near a point. These assets are not available for any normal operation by the interlocking operator. The activation is done from the interlocking operator giving authorization for the mode. The return of operational control can be done by commands from the interlocking operator or a device on-site (special deactivation button). Local operation areas are mainly used for shunting purpose without the use of any route. The traffic safety within this area is solely dependent on the on-site staff. The active status of a local operation area is indicated to the railway staff by special signals or special signal aspects. The limits are defined by the end of TVD sections, i.e. axle detection points or insulated joints.
- workZone
  - The element <workZone> defines work zones that are controlled by the controller. A work zone is an area that can be separated from the network for special purposes. When this happens, the assets of the area are not available for normal operation or train traffic. The activation and deactivation are controlled by special routines to ensure the safety of any workers in that zone. A work zone cannot be reopened for traffic without proper action and consent from the outside staff. A work zone is mainly activated for the protection of staff from train traffic. There is normally no intention for any train movements inside it. The limits of the zone are defined by the end of TVD sections, i.e. axle detection points or insulated joints.

Figure 9 — Controllers

6.3.2.3 InfraAttrGroups

The element <infraAttrGroups> is a child element of <infrastructure> and contains the following child element, as illustrated in Figure 10.

— InfraAttributes

 — The element <infraAttributes> describes the properties of an infrastructure group, such as owner, operation mode, train protection type, electrification, power transmission, axle weight, gauge, speeds, EPSG codes and general attributes. Thus, standard infrastructure types can be defined, e.g. according to established standards.

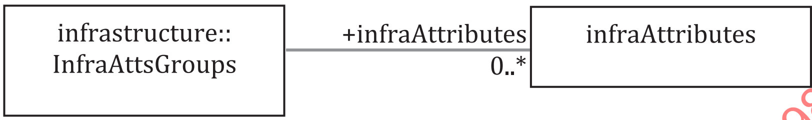


Figure 10 — InfraAttrGroups

6.3.2.4 OperationControlPoints

The element <operationControlPoint> is a child element of <infrastructure> and contains the following child element, as illustrated in Figure 11.

— OCP

 — The element <ocp> is a container for operation or control points (places). Each <ocp> illustrates one operation or control point in the infrastructure.

The element <ocp> contains the following child elements:

— PropOperational

 — The element <propOperational> contains attributes which further refine the operational properties of an <ocp>.

— PropService

 — The element <propService> contains attributes which further refine the service properties of an <ocp>.

— PropEquipment

 — The element <propEquipment> is a container which provides two variants to define the (technical) equipment of an <ocp>. Either (1) a set of flags (attributes) contains a summary of the OCP's technical properties, or (2) a list of references to track-elements which belong to the OCP. Equipment defined for these tracks implicitly belongs to the <ocp>.

— PropOther

 — The element <propOther> contains attributes which define additional properties of an <ocp>.

— Designator

 — The element <designator> allows external primary keys for the <ocp>.

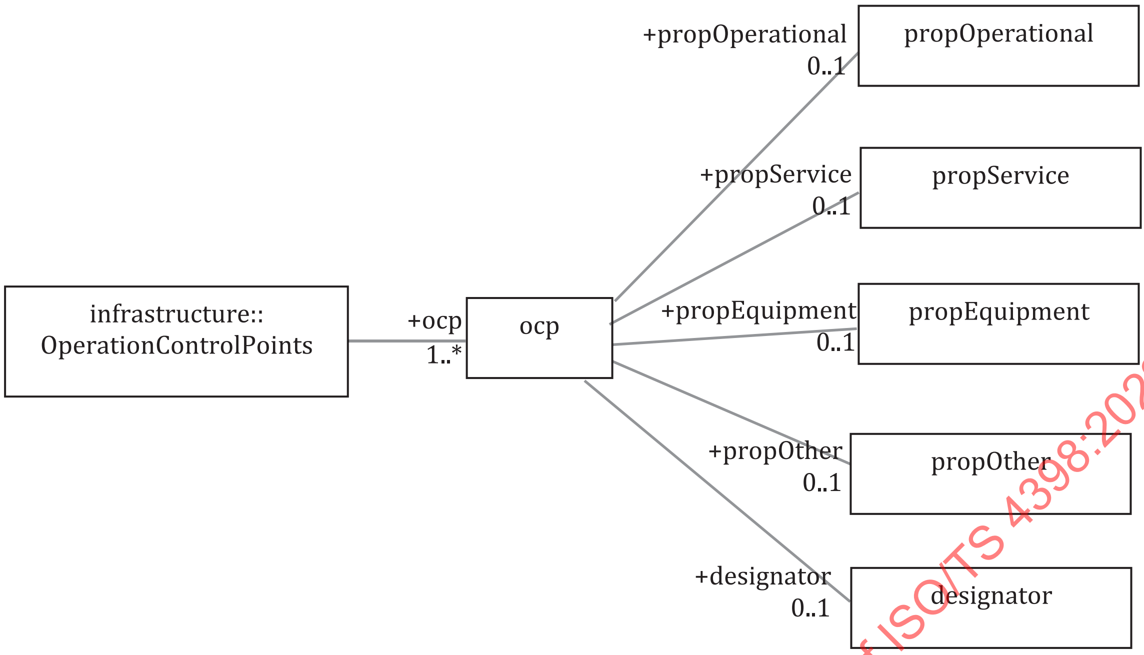


Figure 11 — OperationControlPoints

6.3.2.5 SpeedProfiles

The element <speedProfiles> is a child element of <infrastructure> and contains the following child element, as illustrated in Figure 12.

— SpeedProfile

  — The element <speedProfile> contains information about the required train speed profile characteristics.

The element <speedProfile> contains the following child elements:

— Tilting

  — The element <tilting> contains properties describing the required train tilting characteristics.

— Braking

  — The element <braking> contains properties describing the required train braking configuration.

— Path

  — The element <path> indicates a train run between two neighbouring OCPs.

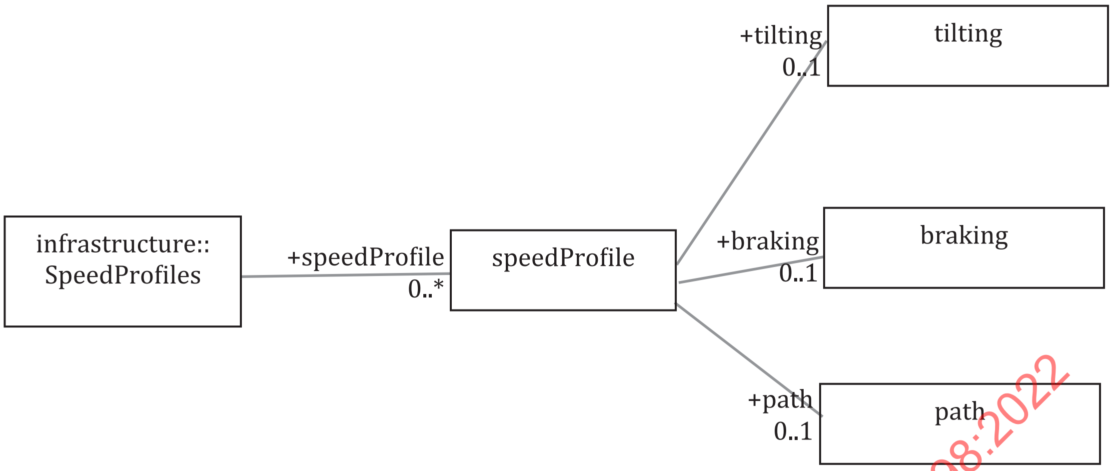


Figure 12 — SpeedProfiles

6.3.2.6 TrackGroups

The element <trackGroups> is a child element of <infrastructure> and contains the following child elements, as illustrated in Figure 13.

— Line

  — The element <line> groups all tracks belonging to a line. The line typically has its own mileage and allows for the regular operation of trains.

Figure 13 — TrackGroups

6.3.2.7 Tracks

6.3.2.7.1 Overview

The element <tracks> is a child element of <infrastructure> and contains the following child element, as illustrated in Figure 14.

— Track

  — The element <track> represents one of possibly multiple tracks (= "pair of rails") that make up a line. The <track> and its child elements contain all information about the track's topology and the trackside elements associated with that track.

The element <`track`> contains the following child elements:

— TrackTopology

  — The element <`trackTopology`> is a container for several topology-related elements that “cannot be touched in real life”.

— TrackElements

  — The element <`trackElements`> is a container for elements which can be (more or less) “touched in real life”.

— OcsElements

  — The element <`ocsElements`> is a container for operation and control system elements.

— impairmentSections

  — The element <`impairmentSections`> is a container element for track sections with impairments for railway operation

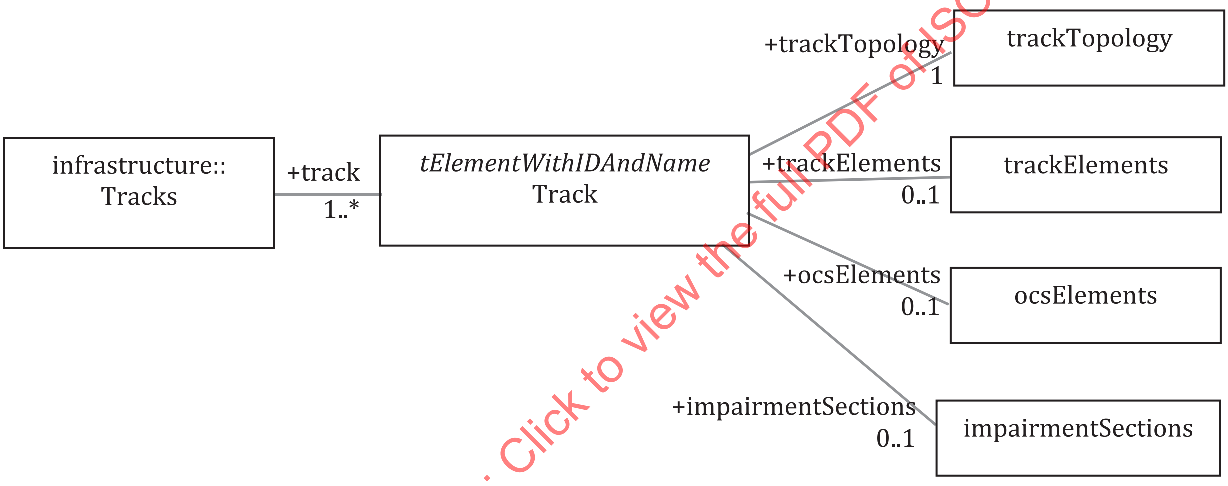


Figure 14 — Tracks

6.3.2.7.2 OcsElements

The element <ocsElements> is a child element of <tracks> and contains the following child elements, as illustrated in Figure 15.

— Signals

  — The element <signals> is a container for signalling elements.

— TrainDetectionElements

  — The element <trainDetectionElements> is a container for track-side sensor elements.

— Balises

  — The element <balises> represents a container for all balise elements which can either be a single balise or a balise group.

— TrainProtectionElements

  — The element <trainProtectionElements> is a container for track-side equipment of train protection systems.

— StopPosts

  — The element <stopPosts> is a container for <stopPost> elements.

— Derailers

  — The element <derailers> is a container for all elements of derailers. Derailers are devices used to prevent fouling of a rail track by unauthorized movements of trains or unattended rolling stock.

— TrainRadioChanges

  — The element <trainRadioChanges> is a container for track-side equipment of train radio/communication systems.

— Locks

  — The element <locks> is a container element for instances of <lock>. A <lock> is a physical railway infrastructure element that is used for logical locking connections between switches, derailers, level crossings, or other signalling components.

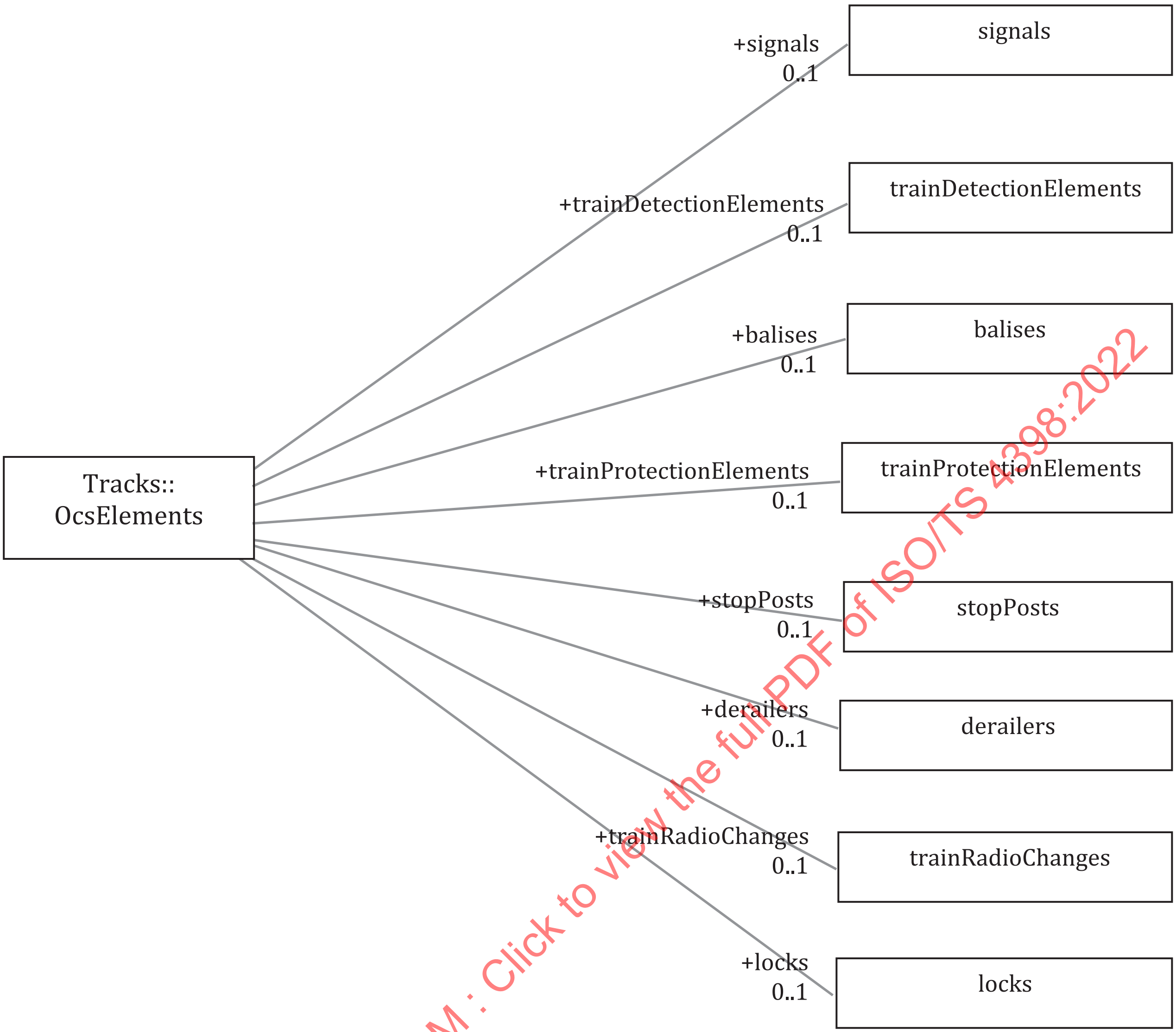


Figure 15 — OcsElements

6.3.2.7.3 TrackElements

The element <trackElements> is a child element of <tracks> and contains the following child elements, as illustrated in Figure 16.

- SpeedChanges
  - The element <speedChanges> is a container for all elements of <speedChange>. They are used to determine the position of a track where the maximum allowed speed changes.
- GradientChanges
  - The element <gradientChanges> is a container for all elements of <gradientChange>. They are used to determine the position of a track where the track's gradient changes.
- RadiusChanges
  - The element <radiusChanges> is a container for all elements of <radiusChange>. An unlimited number of changes may be defined.
- Tunnels
  - The element <tunnels> represents a container for an unlimited amount of several <tunnel> elements.
- Bridges
  - The element <bridges> represents a container for an unlimited amount of several <bridge> elements.
- LevelCrossings
  - The element <levelCrossings> is a container for all elements of <levelCrossing>. They are used to determine the crossings of a track with roads, paths, etc.
- OwnerChanges
  - The element <ownerChanges> is a container for all elements of <ownerChange>. An unlimited number of changes may be defined.
- OperationModeChanges
  - The element <operationModeChanges> is a container for all elements of <operationModeChange>. An unlimited number of changes may be defined.
- TrainProtectionChanges
  - The element <trainProtectionChanges> is a container for all elements of <trainProtectionChange>. An unlimited number of changes may be defined.
- ElectrificationChanges
  - The element <electrificationChanges> is a container for all elements of <electrificationChange>. They are used to determine the position of a track where the electric supply changes.
- PowerTransmissionChanges
  - The element <powerTransmissionChanges> is a container for all elements of <powerTransmissionChange>. An unlimited number of changes may be defined.
- AxleWeightChanges
  - The element <axleWeightChanges> is a container for all elements of <axleWeightChange>. They

are used to determine the position of a track where the maximum allowed axle weight changes.

— GaugeChanges
  — The element `<gaugeChanges>` is a container for all elements of `<gaugeChange>`. They are used to determine the position of a track where the gauge (the track's width) changes.
— ClearanceGaugeChanges
  — The element `<clearanceGaugeChanges>` is a container for all elements of `<clearanceGaugeChange>`. They are used to determine the position of a track where the clearance gauge (also called loading gauge) changes.
— GeoMappings
  — The element `<geoMappings>` is a container for `<geoMapping>` elements.
— TrackConditions
  — The element `<trackConditions>` is a container for `<trackCondition>` elements.
— PlatformEdges
  — The element `<platformEdges>` is a container for all elements of `<platformEdge>`. They are used to describe the border line between the platform and the railway track in a station.
— ServiceSections
  — The element `<serviceSections>` is a container for all elements of `<serviceSection>`. They are used to describe special areas of railway tracks in stations where specific service infrastructure is located.

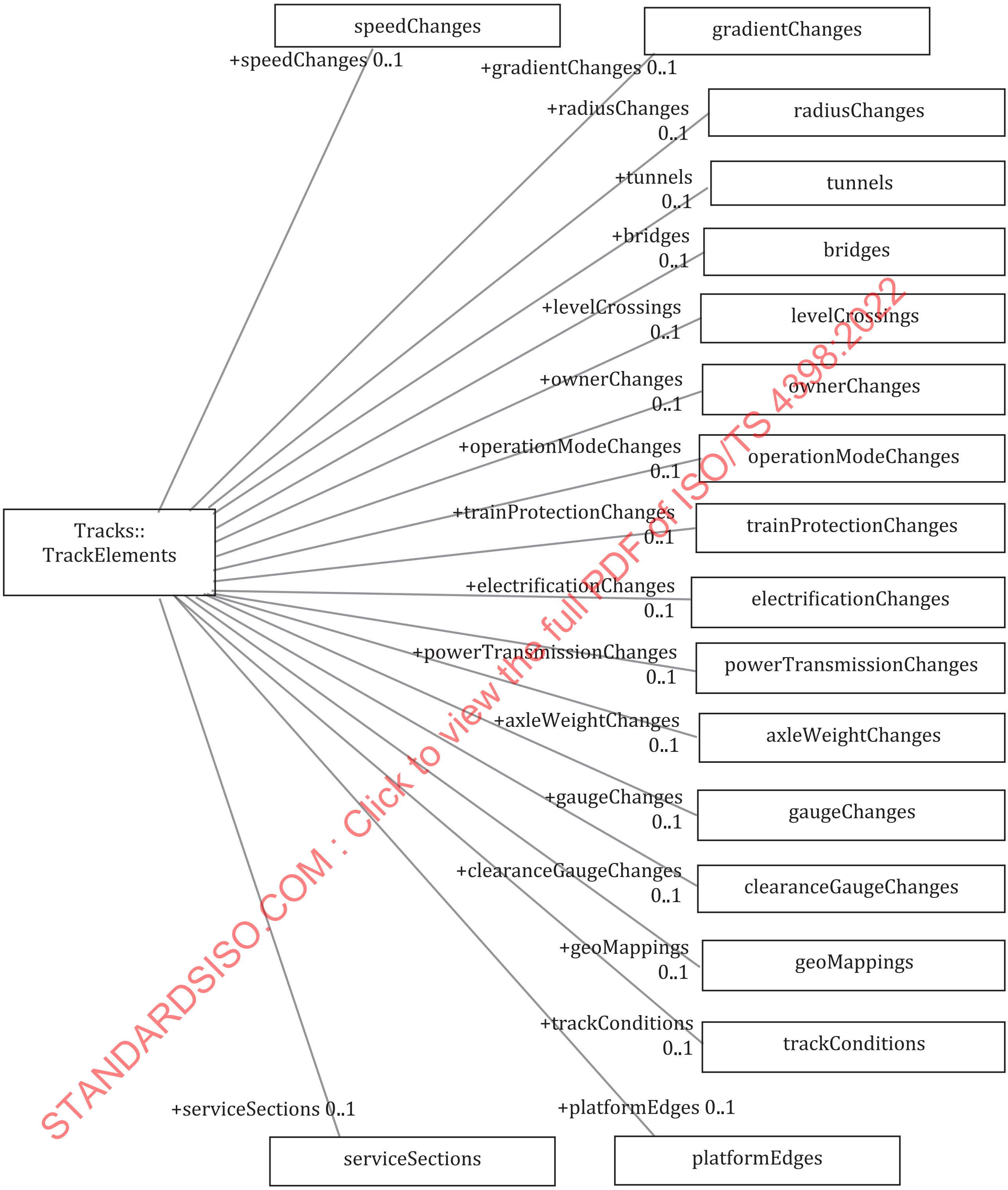


Figure 16 — TrackElements

6.3.2.7.4 TrackTopology

The element <trackTopology> is a child element of <tracks> and contains the following child elements, as illustrated in Figure 17.

— TrackBegin

  — The element <trackBegin> defines the start of a track in the RailDax file. It could be seen as a "vertex" in a "graph" where the track is an "edge".

— TrackEnd

  — The element <trackEnd> defines the end of a track in the RailDax file. It could be seen as a "vertex" in a "graph" where the track is an "edge".

— MileageChanges

  — The element <mileageChanges> is a container for all elements of <mileageChange>. It is used to determine the position of a track where the track's mileage system changes.

— Connections

  — The element <connections> is a container keeping all connecting elements in RailDax which can either be a <switch> or a <crossing>.

— CrossSections

  — The element <crossSections> represents a container for several <crossSection> elements.

— Borders

  — The element <borders> represents a container for several <border> elements.

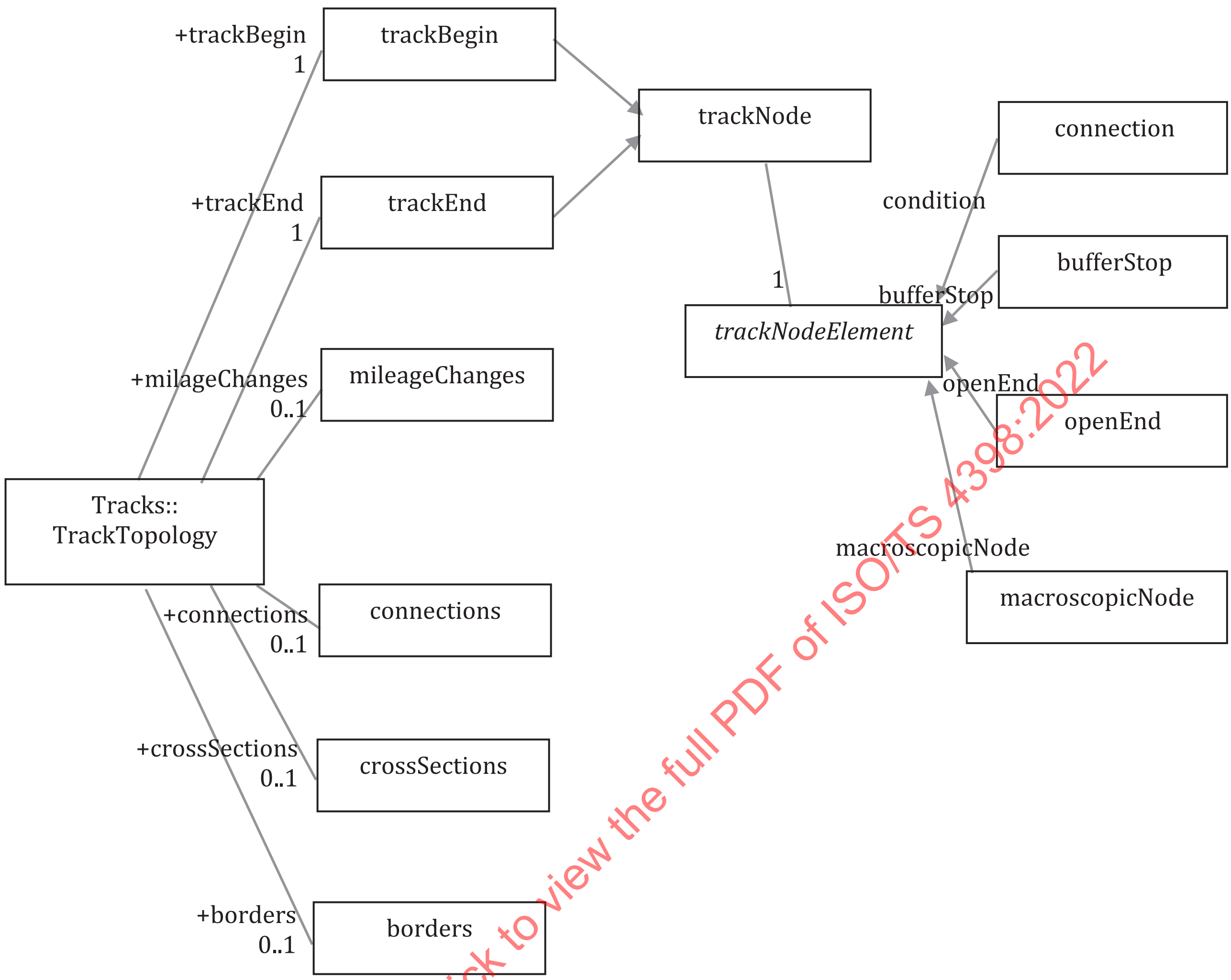


Figure 17 — TrackTopology

6.3.2.7.5 Connections

The element <`connections`> is a child element of <`tracks`> and contains the following child elements, as illustrated in Figure 18.

— Crossing

  — The element <`crossing`> defines properties of diamond crossings. A diamond crossing is defined as a level junction of two tracks without the possibility of changing between these tracks.

— Switch

  — The element <`switch`> defines a standard railroad 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.

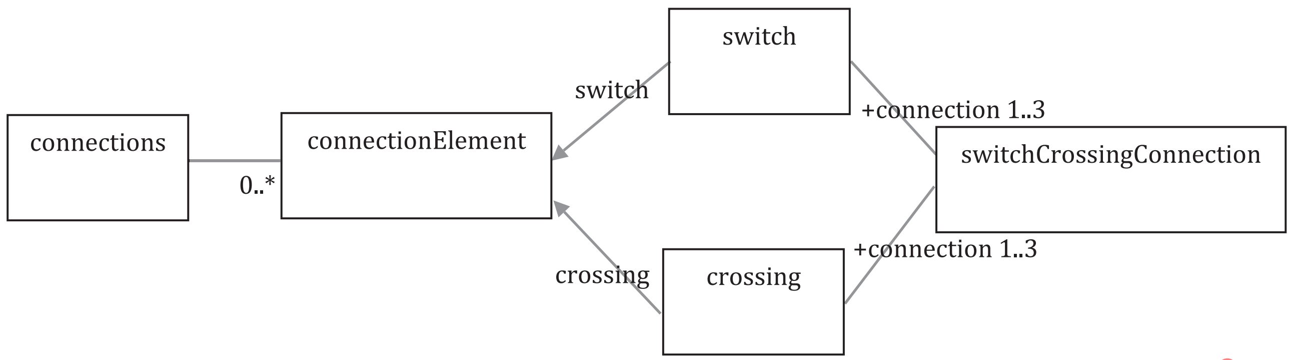


Figure 18 — Connections

6.3.2.8 Routes

The element <routes> is a child element of <infrastructure> and contains the following child element, as illustrated in Figure 19.

— route

  — The element <route> is a container for a signal route. A route is defined as a configured piece of infrastructure, which allows a train to pass a set railway path safely and legally. The use cases listed under "operational timetable simulation" have a need to model routes without interlocking. This means that there is a list of routes. The routes are described with their properties independently from one another.

The element <route> contains the following child elements:

— switchAndPosition

  — The element <switchAndPosition> contains attributes which further refine the path of the route by defining all switches (@switchRef) with their set course (@switchPosition) along the route.

— overlapSwitchAndPosition

  — The element <overlapSwitchAndPosition> contains attributes which further refine the path of the overlap defining all switches, @switchRef, with their set course, @switchPosition, along the path of the overlap.

— releaseGroup

  — The element <releaseGroup> is a container which refers to an extended feature of a route, which gives the optional possibility to group track sections into a release group. This is used to model partial route releases. The release group is released when all track sections of a group are cleared.
  All the track sections listed in <nor:releaseGroup> also form the track sections that have to be controlled for the route to be set. As some track sections are included to form flank protection (and are thus not part of the routes path), they have to be declared as such.

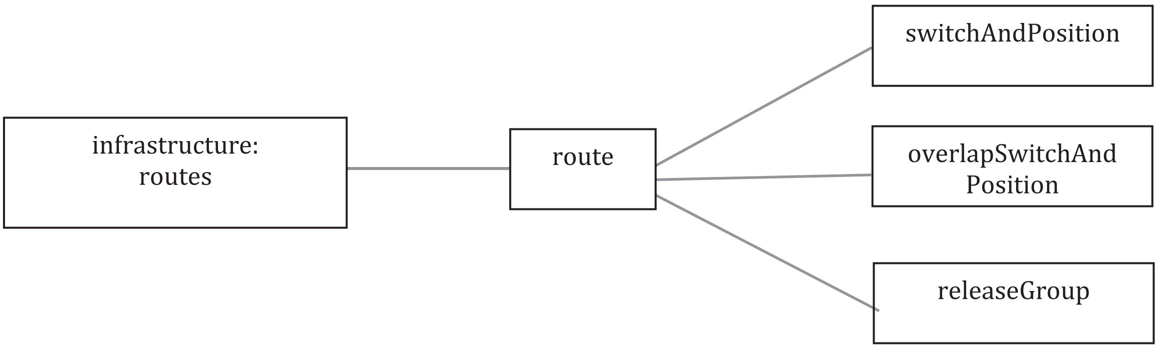


Figure 19 — Routes

6.3.2.9 GenericAreas

The element <genericAreas> is a child element of <infrastructure> and contains the following child element, as illustrated in Figure 20.

— genericArea

  — The element <genericArea> is a container to describe several different types of areas, e.g. specification of specially controlled areas within the network.

The element <genericArea> contains the following child elements:

— location

  — The element <location> is the location of the area within the track network or with reference to a coordinate reference system.

— IsLimitedBy

  — The element <isLimitedBy> references the areas borders.

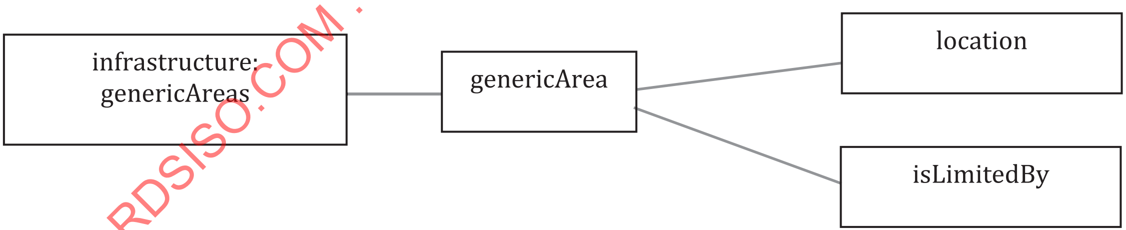


Figure 20 — GenericAreas

6.3.3 Timetable

6.3.3.1 Overview

The element <timetable> is part of the RailDax schema and contains the following child elements, as illustrated in Figure 21.

— TrainParts

  — The element <trainParts> groups <trainPart> elements.

— Trains

  — The element <trains> groups <train> elements.

— TrainGroups

  — The element <trainGroups> groups <trainGroup> elements.

— Rosterings

  — The element <rosterings> groups <rostering> elements.

— Annotations

  — The element <annotations> groups <annotation> elements.

— Categories

  — The element <categories> groups <category> elements.

— TimetablePeriods

  — The element <timetablePeriods> groups <timetablePeriod> elements, lasting typically a year.

— OperatingPeriods

  — The element <operatingPeriods> groups <operatingPeriod> elements within a timetable period.

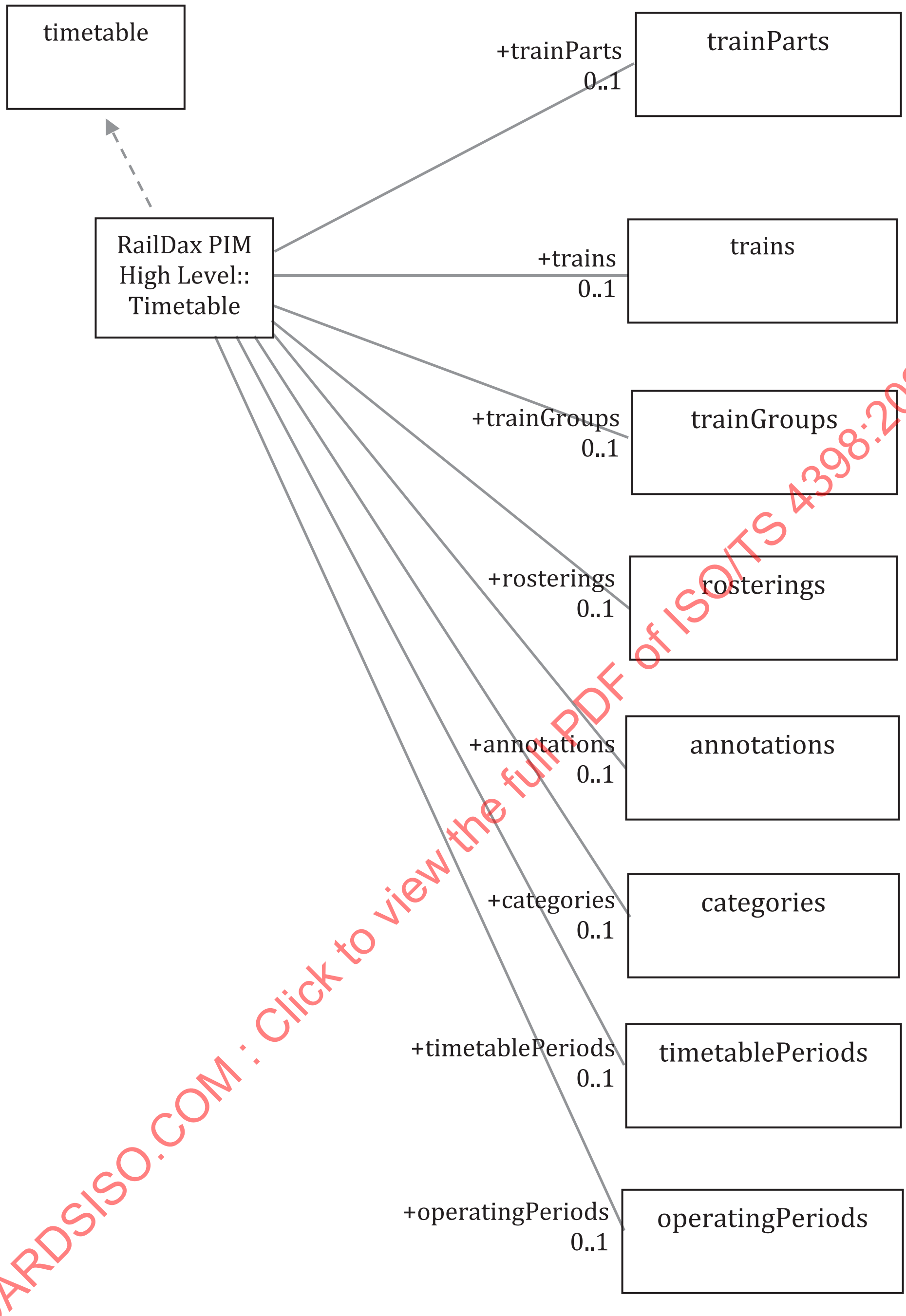


Figure 21 — Timetable

6.3.3.2 TrainGroups

The element <trainGroups> is a child element of <timetable> and contains the following child element, as illustrated in Figure 22.

— TrainGroup

  — The element <trainGroup> is linking different individual trains to a group. This could for instance be used for an interval group of trains belonging together in a train path allocation process.

Figure 22 — TrainGroups

6.3.3.3 TrainParts

6.3.3.3.1 Overview

The element <trainParts> is a child element of <timetable> and contains the following child element, as illustrated in Figure 23.

— TrainPart

  — The element <trainPart> describes the most basic part of a train. Hence there is no change of the formation or operating period allowed during a train parts route.

The element <trainPart> contains the following child elements:

— FormationTT

  — The element <formationTT> describes how a formation is used for a <trainPart>.

— OperatingPeriodRef

  — The element <operatingPeriodRef> is used for referencing the <operatingPeriod> for the <trainPart>.

— OcpsTT

  — The element <ocpsTT> groups <ocpTT> elements applying for the respective <trainPart>. As the <ocpTT> elements represent <ocps> which the <trainPart> passes, the <ocpsTT> contains the route.

— OrganizationalUnitBinding

  — The element <organizationalUnitBinding> describes what organizational units are responsible for this <trainPart>.

— AnnotationRef

  — The element <annotationRef> allows for referencing an <annotation>.

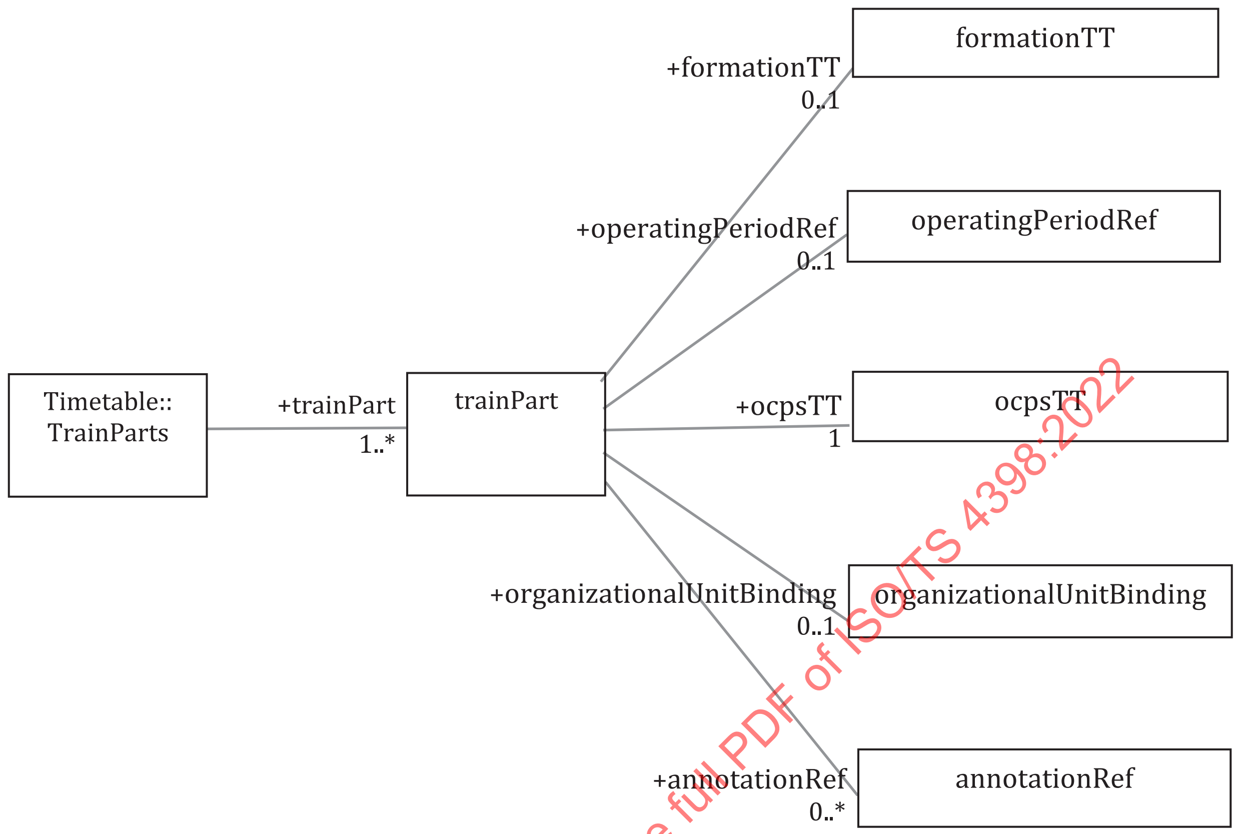


Figure 23 — TrainParts

6.3.3.3.2 OcpsTT

The element <ocpsTT> is a child element of <trainParts> and contains the following child element, as illustrated in Figure 24.

— OcpTT

 — The element <ocpTT> describes a single location along the train path. Normally, an <ocpTT> belongs to a station with an arrival and departure time. But an <ocpTT> could also be a blocking signal with or without passing time, or some control point on the train path. It is always referencing a certain <ocp> defined in the infrastructure part, providing some timetable related information for it.

The element <ocpTT> contains the following child elements:

- Times
  - The element <times> describes arrival and departure times of a train with their scope.
- Connections
  - The element <connections> groups <connection> elements.
- Statistics
  - The element <statistics> groups <statistic> elements.
- SectionTT
  - The element <sectionTT> describes data concerning the way from one <ocpTT> to the next one in the order of the sequence. This includes references to the used infrastructure as well as <runTimes>.
- StopDescription
  - The element <stopDescription> describes data concerning the stop (or pass) at the parent <ocpTT>.

The element <connection> contains the following child elements:

- ExternalReference
  - The element <externalReference> contains information about the train identification number.
- AnnotationRef
  - The element <annotationRef> contains texts and messages dedicated for a passenger information system.

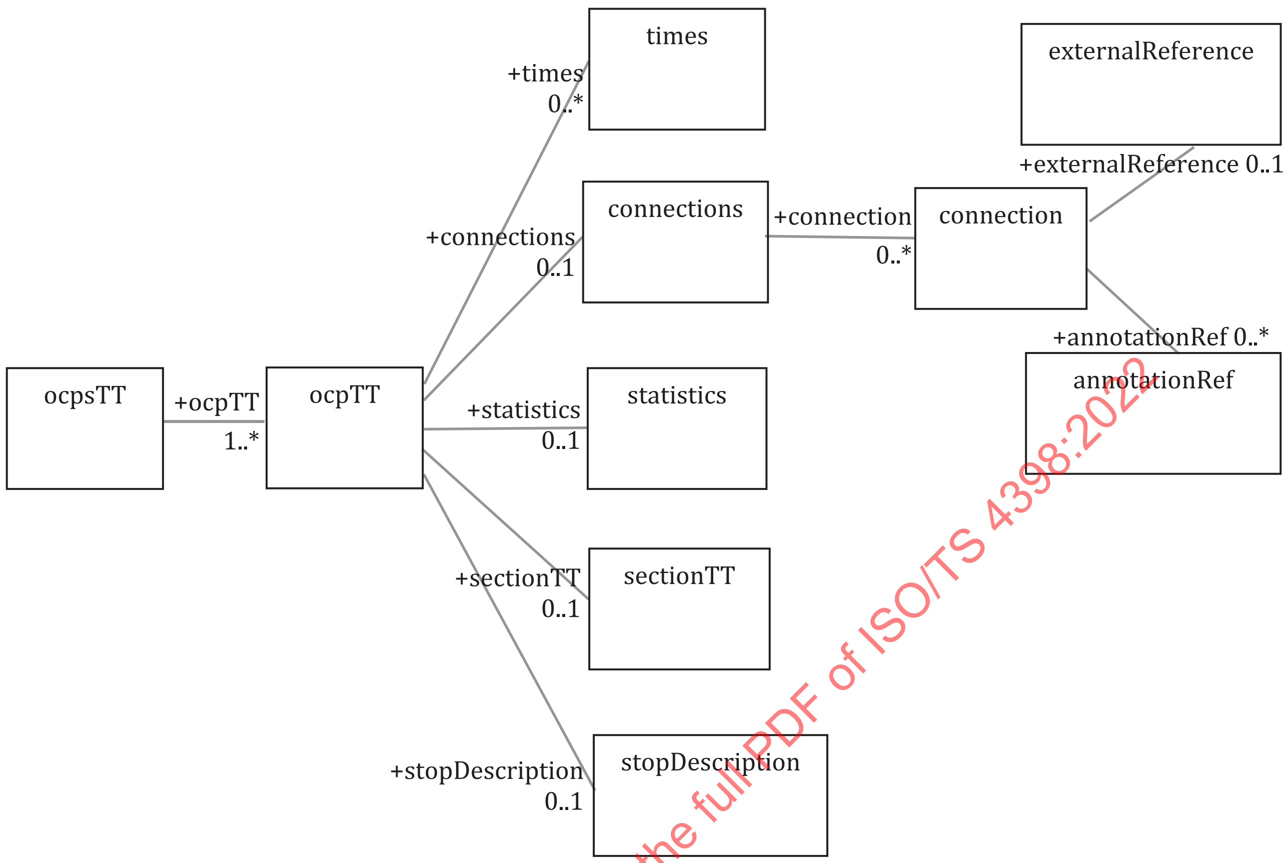


Figure 24 — OcpsTT

6.3.3.4 Trains

6.3.3.4.1 Overview

The element <trains> is a child element of <timetable> and contains the following child element, as illustrated in Figure 25.

— Train

— The element <train> describes a train for different perceptions. A train could either be an “operational train” in the view of a signal box, or it can be “commercial train” from the view of a passenger.

The element <train> contains the following child elements:

— TrainPartSequence

— The element <trainPartSequence> groups all elements <trainPart> belonging to a train. The sequence herein is supposed to be used for successive parts along the train route. This is not to be confused with the sequence of rolling stock inside a train formation or a chronological sequence during an operating period or during a day. Therefore, all referenced elements <trainPart> of a <trainPartSequence> should have the same starting point and end point. Otherwise, smaller (shorter) <trainParts> should be defined.

— TafTapTsiTrainID

— The element <tafTapTsiTrainID> describes the so-called unique <transportID>, used for the explicit identification of a train, especially for international train runs. The number will be implemented during the implementation of the TAF/TAP TSI (Telematics Applications for

Freight services, Telematics Applications for Passenger services, Technical Specifications for Interoperability).

— Replacements

— The element `<replacements>` is a container element for instances of `<replacementOf>`. As such, `<replacements>` contains the mapping of train replacements.

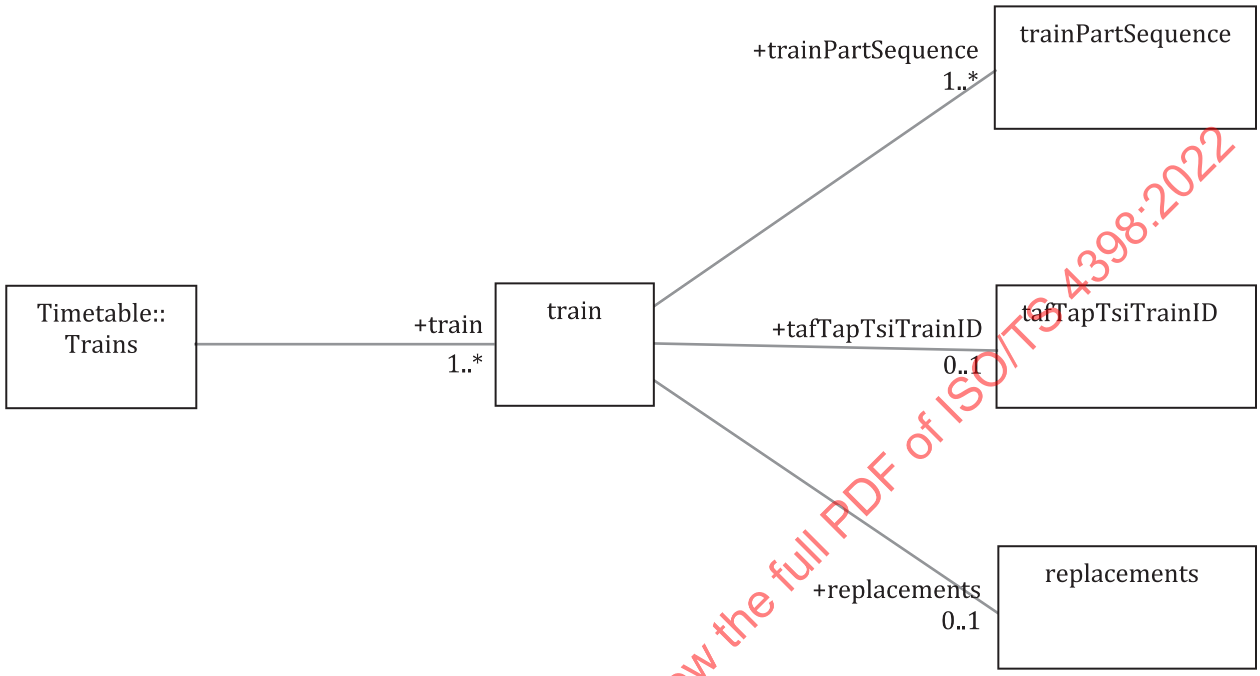


Figure 25 — Trains

6.3.3.4.2 TrainPartSequence

The element <trainPartSequence> is a child element of <trains> and contains the following child elements, as illustrated in Figure 26.

— TrainPartRef

  — The element <trainPartRef> is used for referencing one <trainPart> belonging with others to the same <trainPartSequence>.

— SpeedProfileRef

  — The element <speedProfileRef> is referencing the <speedProfile> element used by the <trainPartSequence> and described in detail in an infrastructure branch of the RailDax file. In special cases, there could occur more than one speed profile.

— EquipmentUsage

  — The element <equipmentUsage> and its sub elements describe how the equipment of the rolling stock is used or required by a certain <train>. This is not to be mixed with the physical-timetable-independent <rollingstock>.

— BrakeUsage

  — The element <brakeUsage> defines the brake system for the whole train as a "fixed formation".

— TopologyReference

  — The element <topologyReference> can be used to refer to the underlying topology, which was used to determine the train running path and calculate the runtimes. Specifying the topology register and an optional time period, the attribute entry is used to refer to the external version/variant within the topology register. This assumes that the reading system also knows the topology register and can interpret the version/variant definition. When specifying multiple topology references, these are required to have a disjoint time period and, combined together, cover at least the operating days of the train.

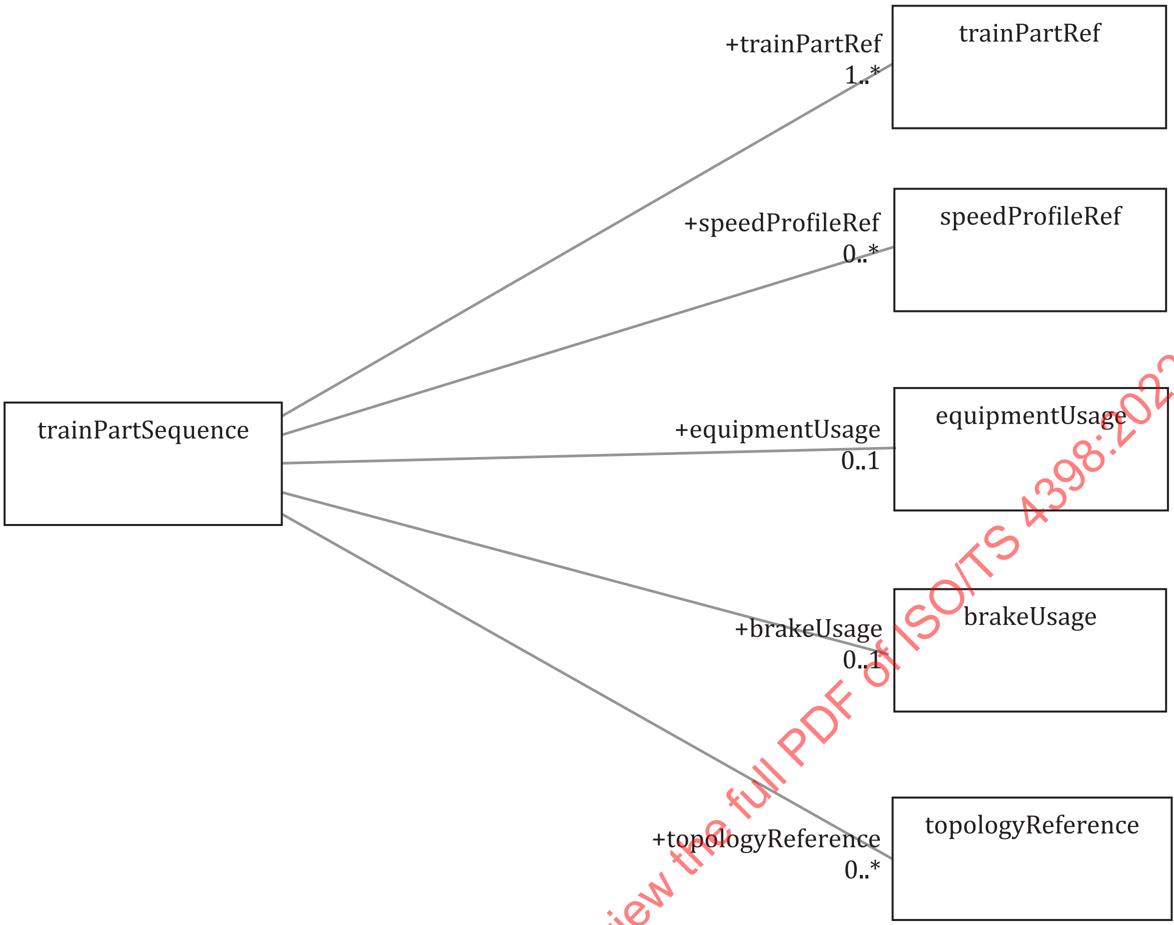


Figure 26 — TrainPartSequence

6.3.3.5 Rostering

The element <rostering> is a child element of <timetable> and contains the following child elements, as illustrated in Figure 27.

- BlockParts
  - The element <blockParts> is used for grouping one or more <blockPart>.
  - The sub-element <blockPart> contains all data which are related to an atomic particle of a block.
- Blocks
  - The element <blocks> is used for grouping one or more <block>.
  - The sub-element <block> contains all data which are related to a "vehicle duty" containing a sequence of atomic tasks (blockParts). The item is referenced via its ID attribute. Every blockPart shall be part of a block.
- Circulations
  - The element <circulations> is used for grouping one or more <circulation>.
  - The sub-element <circulation> is for chaining the blocks into a complete rostering.

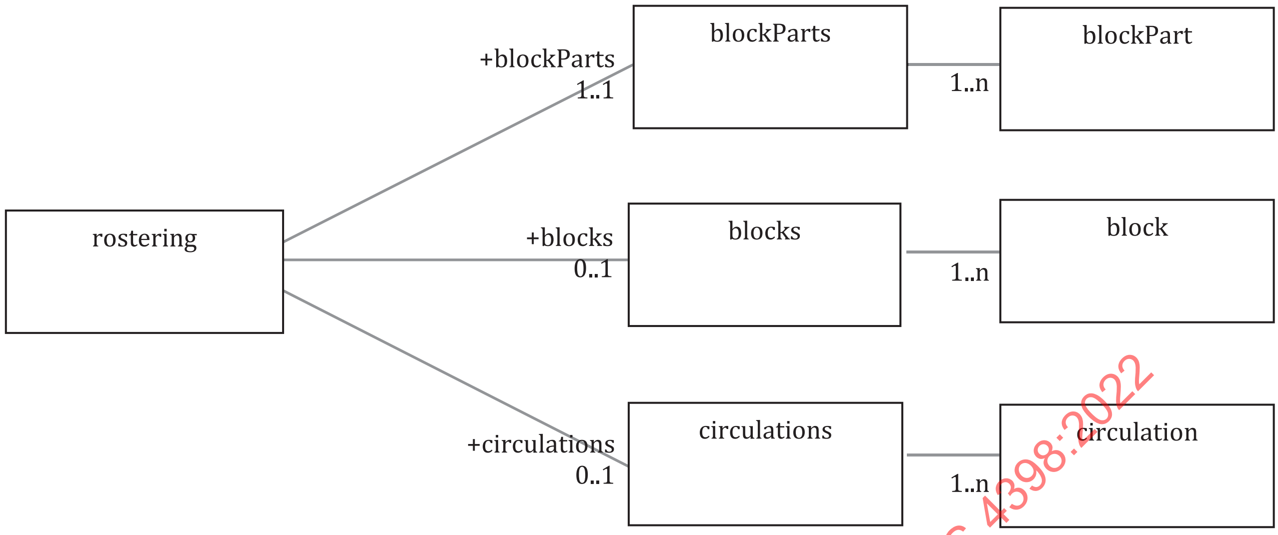


Figure 27 — Rostering

6.3.4 Rolling stock

6.3.4.1 Overview

The element <rollingstock> is part of the RailDax schema and contains the following child elements, as illustrated in Figure 28.

— Vehicles

  — The element <vehicles> is a container for the data of a single vehicle or a vehicle family. It can contain an infinite number of child elements <vehicle>, which form the data sets.

— Formations

  — The element <formations> is a container for the data of complete trains or parts of a train composed of several vehicles. It can contain an infinite number of child elements <formation>, which form the data sets.

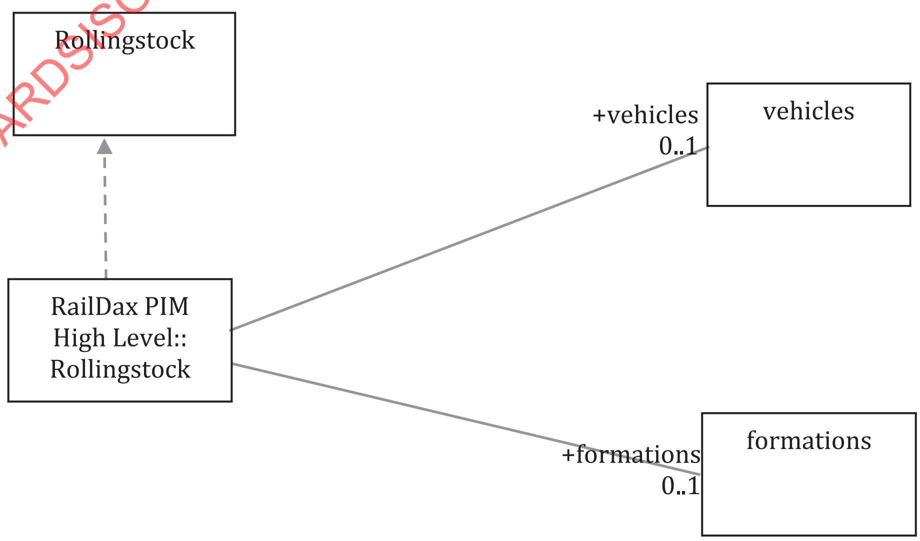


Figure 28 — Rolling stock

6.3.4.2 Formations

6.3.4.2.1 Overview

The element <formations> is a child element of <rollingstock> and contains the following child element, as illustrated in Figure 29.

— Formation

  — The element <formation> contains all data related to a composition of single vehicles forming a complete train or a part of a train. It is not excluded to describe a train consisting of only one single vehicle.

The element <formation> contains the following child elements:

— TrainOrder

  — The element <trainOrder> is a container for the data about the positions and the number of particular vehicles within that formation.

— CategoryRef

  — The optional element <categoryRef> forms a list of references to train categories in the TimeTable part at <category>, e.g. limited express, the formation can run as in service. The possible number of different types referenced here is infinite.

— TrainEngine

  — The element <trainEngine> contains the acceleration values for the entire train or train part. These values may be used for a rough estimation of runtime. The user has to ensure that the values are consistent with any acceleration values given for single vehicles forming this formation. The values shall be based on the typical load situation of the formation and considering no gradient or curve as additional resistance.

— TrainBrakes

  — The element <trainBrakes> describes the type of vehicle brakes and their application as brake effort or deceleration. There is a distinction between normal brake operation and emergency brake application.

— TrainBrakeOperation

  — The element <trainBrakeOperation> contains all data for description of brake operation.

— TrainResistance

  — The element <trainRestistance> contains the resistance to the train's movement as a sum of rolling resistance on tangent level track, grade resistance and curve resistance. RailDax offers two basic possibilities of train resistance value representation: valueTable or mathml.

— SpeedProfileRef

  — The element <speedProfileRef> contains the permissible speed profile for the entire formation as reference to the defined speed profiles in infrastructure.

— TechnicalStopActivity

  — The element <technicalStopActivity> contains a list of stop activities with their related technical times.

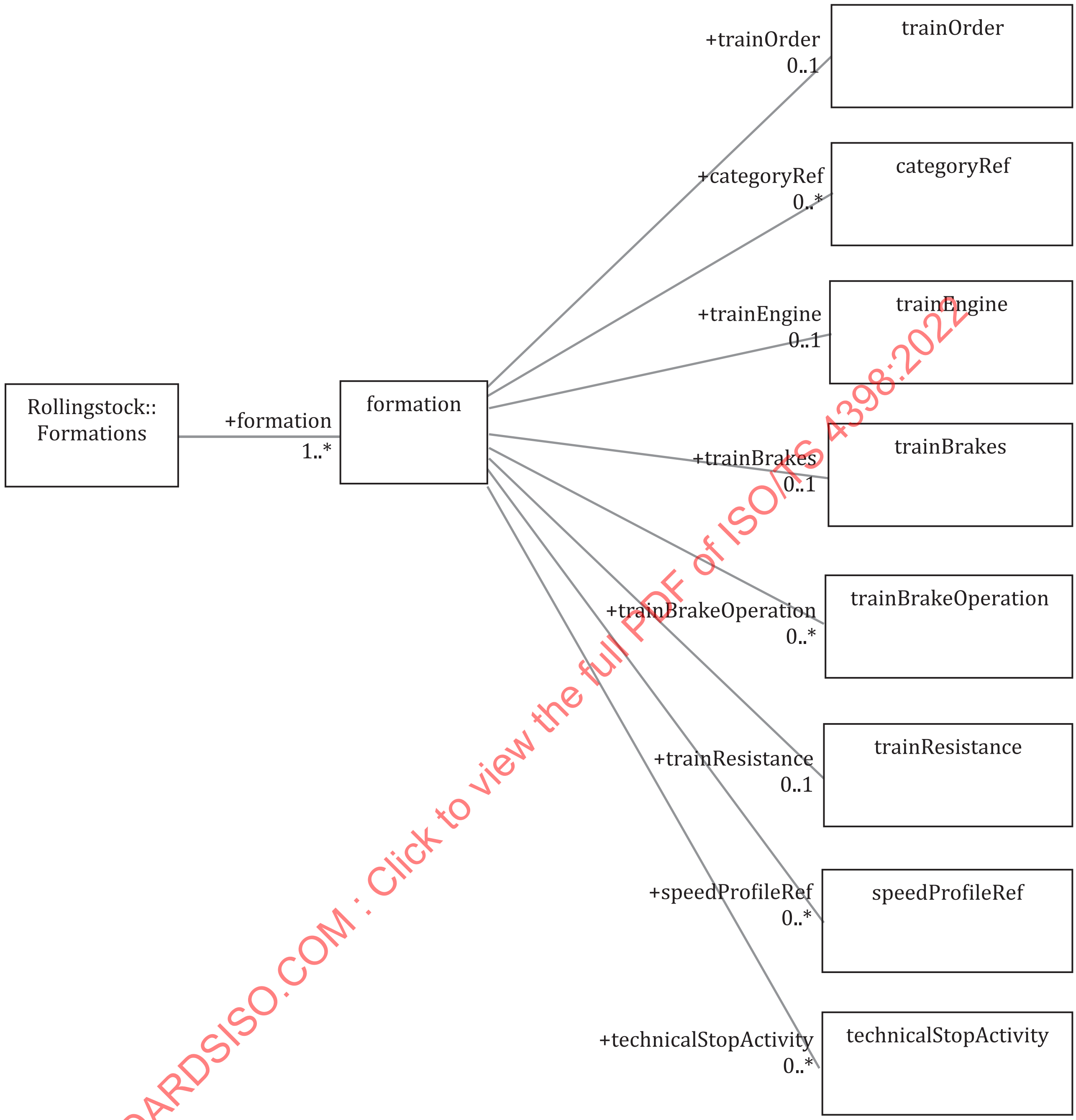


Figure 29 — Formations

6.3.4.2.2 TrainOrder

The element <trainOrder> is a child element of <formations> and contains the following child element, as illustrated in Figure 30.

— VehicleRef

  — The element <vehicleRef> contains the data about the positions and the number of particular vehicles within that formation. One occurrence of the element stands for at least one vehicle. The number of vehicles composed to a formation is infinite.

Figure 30 — TrainOrder

6.3.4.3 Vehicles

6.3.4.3.1 Overview

The element <vehicles> is a child element of <rollingstock> and contains the following child element, as illustrated in Figure 31.

— Vehicle

  — The element <vehicle> contains all data, which are related to a single vehicle or a vehicle family. The item is referenced via its attribute ID. In case of a vehicle family all data can be stored, which are common for all vehicles of this family. The data set of a single vehicle can refer to the common values and define more or deviating characteristics valid only for this particular vehicle.

The element <vehicle> contains the following child elements:

— Classification

  — The element <classification> provides data of informative or organizational character about the vehicle.

— Engine

  — The element <engine> is the container for all data of the propulsion and control system. It is used for all motored vehicles and driver's cab vehicles (only <monitoring> portion).

— Wagon

  — The element <wagon> contains all data with regard to possible payload of the vehicle.

— VehicleBrakes

  — The element <vehicleBrakes> is a container for data about the various vehicle brake systems except the propulsion. The brake effort produced by the propulsion is described in <propulsion> and the related child elements with respect to the different supply variants.

— LoadLimitMatrix

  — The element <loadLimitMatrix> contains the several possible load limits for the vehicle.

— CurvingLimitation

  — The element <curvingLimitation> contains all data related to the physical limitations of curve radiuses the vehicle can negotiate in horizontal and vertical direction.

— MaintenanceIntervals

  — The element <maintenanceIntervals> is a container element for instances of <maintenanceInterval>.

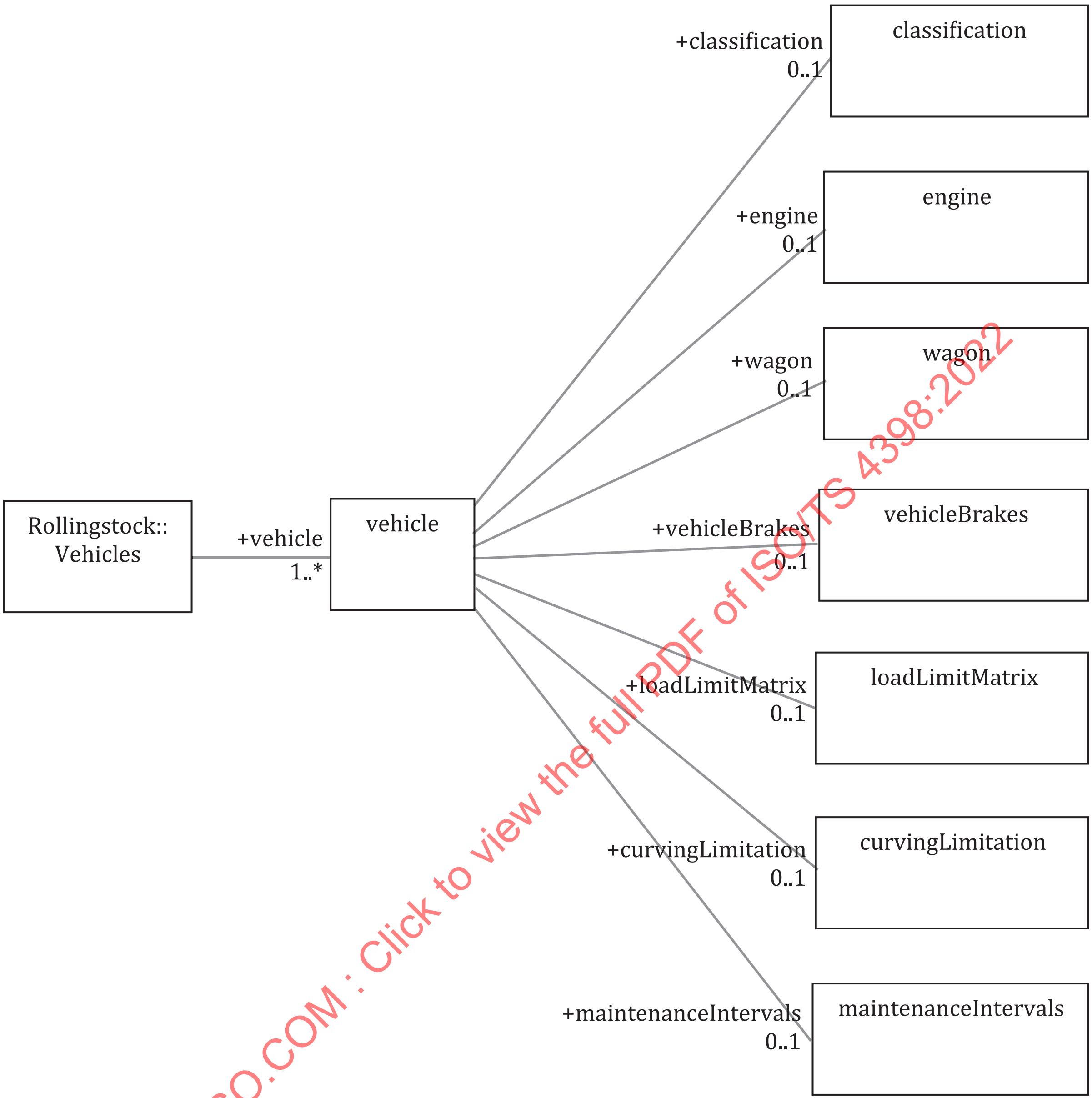


Figure 31 — Vehicles

6.3.4.3.2 Engine

6.3.4.3.2.1 Overview

The element <engine> is a child element of <vehicles> and contains the following child elements, as illustrated in Figure 32.

— Propulsion

  — The element <propulsion> contains all data for the propulsion system of motive power related to a particular supply system, i.e. for each supply system including diesel a separate instance of

<propulsion> is needed. The structure of its sub-elements conforms to the structure found on locos.

— Pantograph

  — The element <pantograph> contains all data of a vehicle pantograph.

— EnergyStorage

  — The element <energyStorage> is the container for all data of energy storage device on the vehicle. The main focus is on the electrical characteristics.

— Monitoring

  — The element <monitoring> contains all data about the installed train control and protection systems and any other supervision equipment of that vehicle. The element is used for any powered vehicle or any vehicle with a driver's cab.

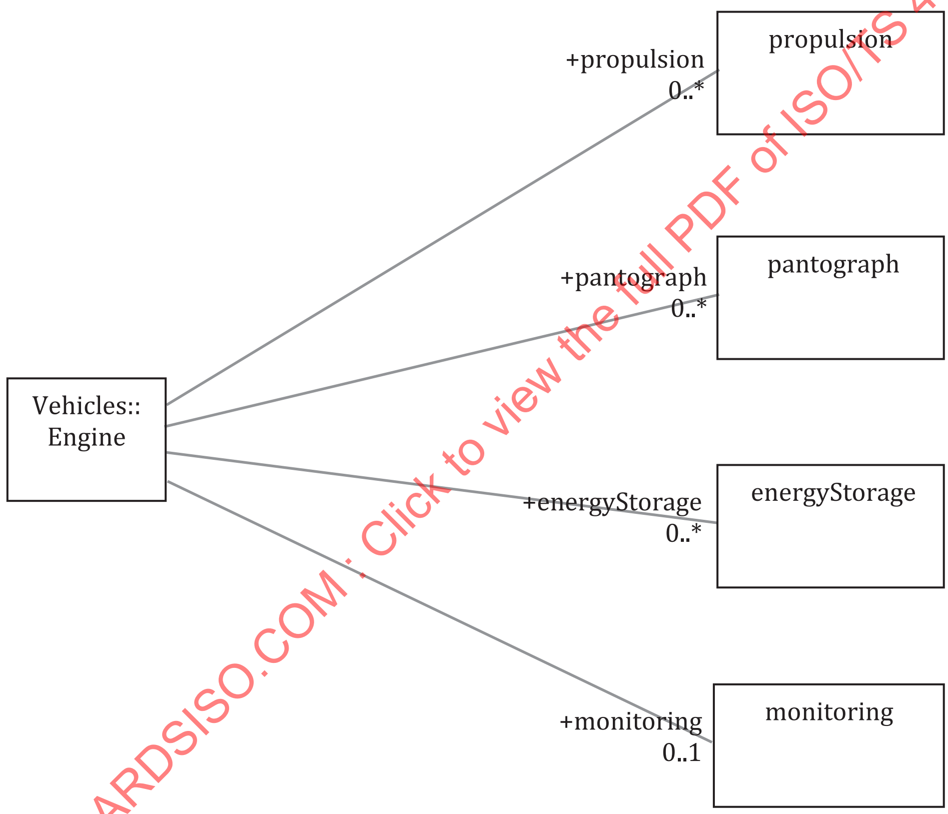


Figure 32 — Engine

6.3.4.3.2.2 Propulsion

The element <propulsion> is a child element of <engine> and contains the following child elements, as illustrated in Figure 33.

— Transformer

  — The element <transformer> contains all data of the main transformer for the propulsion system. The electrical values are related to primary side of transformer.

— FourQuadrantChopper

  — The element <fourQuadrantChopper> contains all data about the four-quadrant chopper conditioning the supply energy for the traction purpose.

— Link

  — The element <link> contains the data about the link circuit of the traction system.

— TractionInverter

  — The element <tractionInverter> contains all data about the inverter adopting the link circuit to the traction motor(s). It can adjust the voltage and frequency for traction motor according to the needs.

— TractionMotor

  — The element <tractionMotor> contains all data about the traction motor(s). It can be detailed to the equivalent circuit with the related resistances and inductances.

— Diesel

  — The element <diesel> contains all data of a diesel engine used for traction. At the moment this list is far from being exhaustive.

— Gear

  — The element <gear> contains all data for the power transmission from the traction motor to the wheels. This normally comprises a mechanical gear and a kind of coupling.

— AuxilliarySupply

  — The element <auxiliarySupply> contains the data of the vehicle's auxiliary power supply as needed for consideration of power consumption calculations. This can potentially comprise components like auxiliary converters or auxiliary windings of main transformer.

— TractiveEffort

  — The element <tractiveEffort> contains the description of available effort at wheels versus vehicle speed for the propulsion system in traction mode.

— BrakeEffort

  — The element <brakeEffort> contains the description of mechanical brake effort at the locomotive's wheels versus vehicle speed as produced by the electrical brake system.

— TractiveCurrent

  — The element <tractiveCurrent> contains the description of net current versus vehicle speed for the propulsion system in traction mode.

— BrakeCurrent

  — The element <brakeCurrent> contains the description of line current versus vehicle speed as produced by regenerative brakes of the vehicle, which may be fed back into the contact system. It does not consider any current, which is generally fed into the rheostatic resistors of the vehicle.

— TractiveCurrentLimitation

  — The element `<tractiveCurrentLimitation>` contains the description of net current limit versus net voltage for the propulsion system in traction mode.

— BrakeCurrentLimitation

  — The element `<brakeCurrentLimitation>` contains the description of line current limitation in braking mode versus line voltage at the vehicle's pantograph.

— TractiveVehicleEfficiency

  — The element `<tractiveVehicleEfficiency>` contains the description of efficiency versus vehicle speed for the entire propulsion system in traction mode.

— BrakeVehicleEfficiency

  — The element `<brakeVehicleEfficiency>` contains the description of the propulsion system efficiency versus vehicle speed in braking mode.

— RackTraction

  — The element `<rackTraction>` contains the data of the cog wheels used for rack traction and braking.

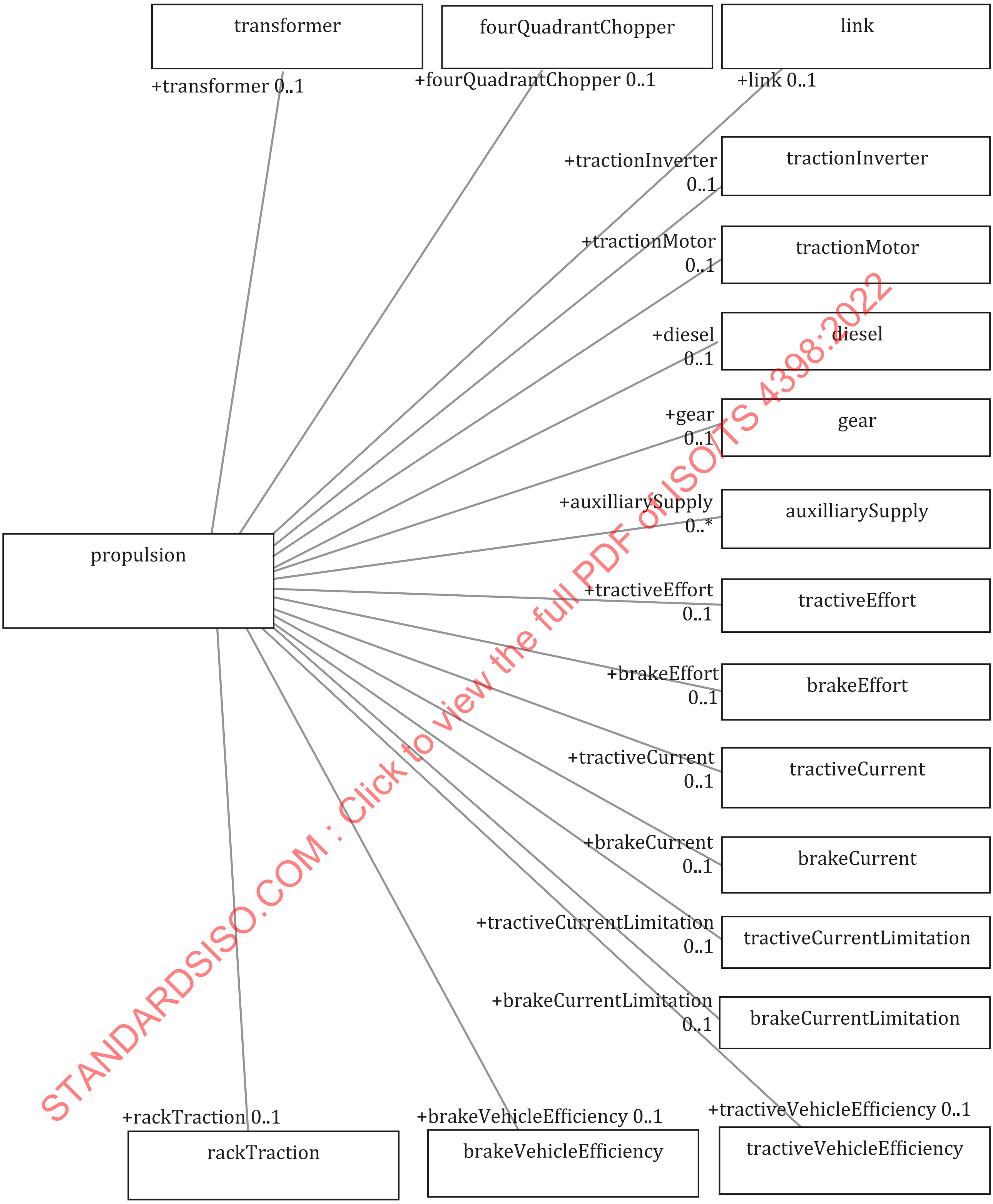


Figure 33 — Propulsion

6.3.4.3.3 Wagon

6.3.4.3.3.1 Overview

The element <wagon> is a child element of <vehicles> and contains the following child elements, as illustrated in Figure 34.

— Passenger

  — The element <passenger> contains all data related to passenger transport with the vehicle.

— Goods

  — The element <goods> contains the data about freight in the entire vehicle or special compartments.

— AuxilliarySupplySystem

  — The element <auxiliarySupplySystem> is used to name the supply systems with which the auxiliaries of the vehicle can be fed.

— DriversCab

  — The element <driversCab> contains the data per driver's cab of the vehicle.

— RackTraction

  — The element <rackTraction> contains the data of the cog wheels used for rack traction and braking.

— Couplers

  — The element <couplers> contains all data related to the coupling devices used between the vehicles.

— TrainClearanceGauge

  — The element <trainClearanceGauge> contains all data with regard to the clearance gauge of the vehicle, also called the minimum clearance outline. This rather complex information about minimum distances in terms of height and width from track centreline and top of rail surface is normally coded in related standards and national railway regulations. The information about the clearance gauges to which the vehicle is compliant allows for a decision as to whether it is suitable for a particular line.

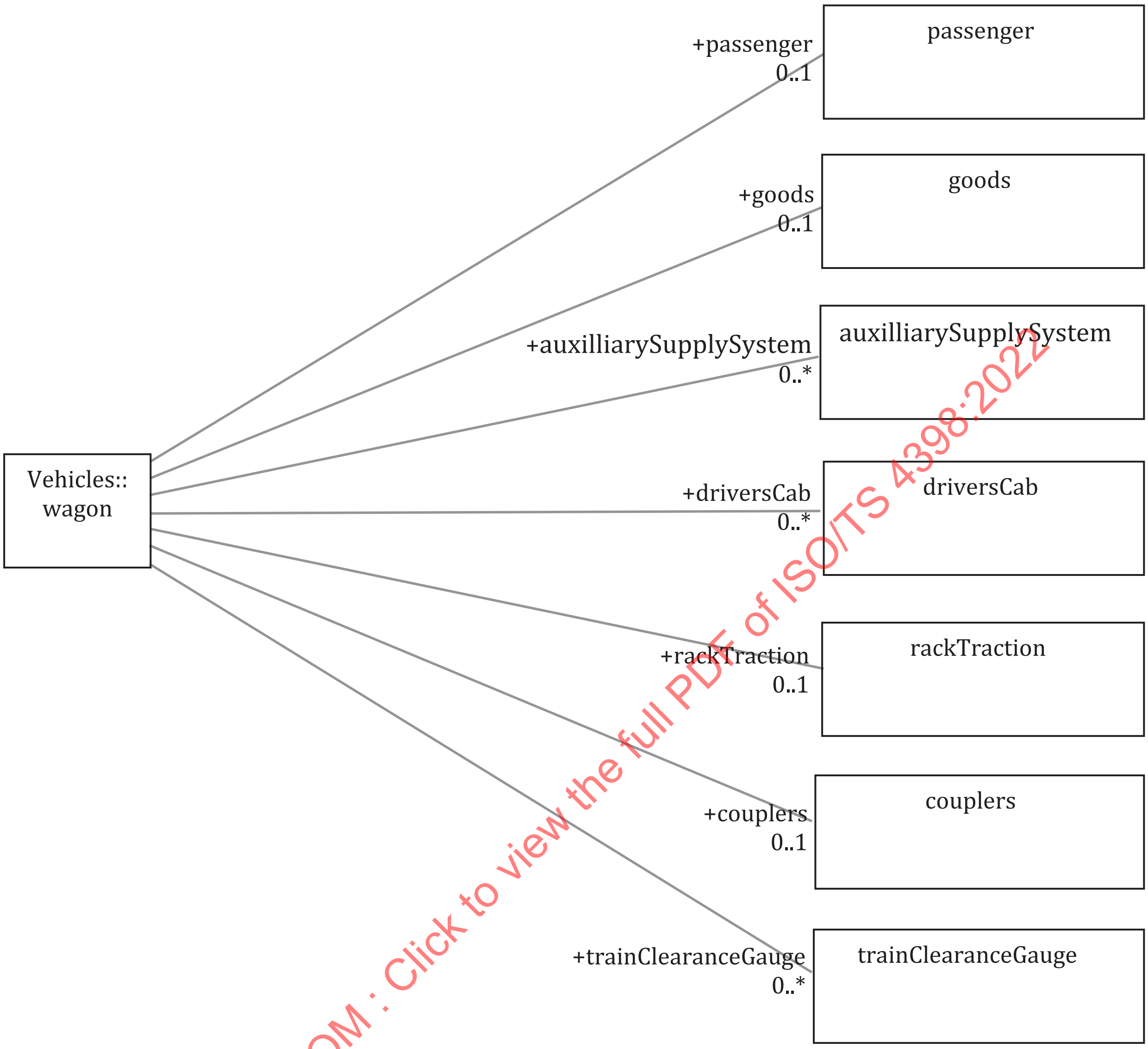


Figure 34 — Wagon

6.3.4.3.3.2 Goods

The element <goods> is a child element of <wagon> and contains the following child element, as illustrated in Figure 35.

— Service

— The element <service> can be used for a list of descriptions of any additional service provided per vehicle. This can be catering, baggage, toilets, low floor portion, conference compartment, internet access, etc. The element can also be used to specify particular provisions of a goods compartment.

Figure 35 — Goods

6.3.4.3.3.3 Passenger

The element <passenger> is a child element of <wagon> and contains the following child elements, as illustrated in Figure 36.

- Places
  - The element <places> contains the data related to particular passenger capacity with respect to the various categories of space provision for them.
- Service
  - The element <service> can be used for a list of descriptions of any additional service provided per vehicle. This can be catering, baggage, toilets, low floor portion, conference compartment, internet access, etc. The element can also be used to specify particular provisions of a goods compartment.
- Gangway
  - The element <gangway> contains all data for gangways allowing the interchange to neighbouring vehicles.
- Doors
  - The element <doors> contains all data related to the passenger doors of the vehicle. This is normally valid for one side of the vehicle.
- Tilting
  - The element <tilting> contains the data of the tilting mechanism if the passenger vehicle is fitted with one.

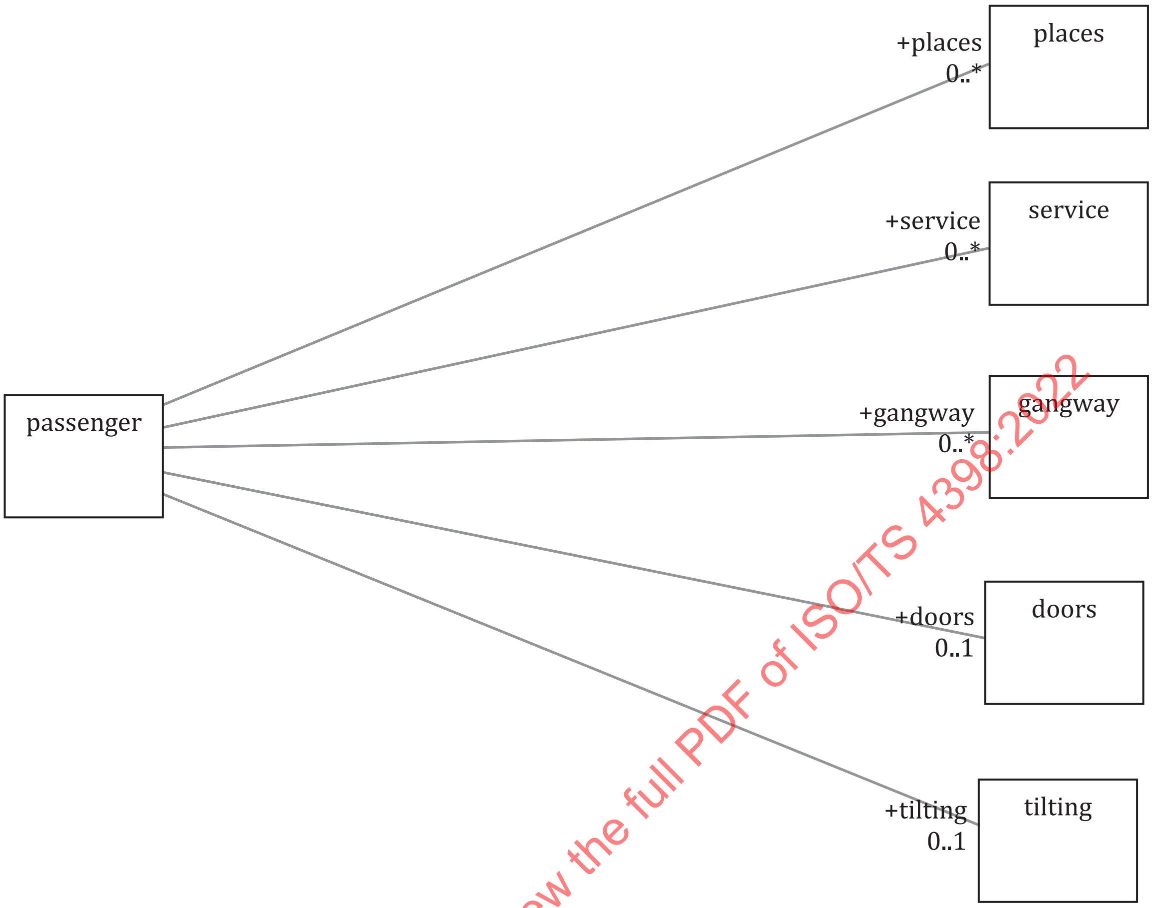


Figure 36 — Passenger

6.3.5 Metadata

6.3.5.1 Overview

The <`metadata`> main element describes the dataset itself. The focus is on administrative metadata to describe the source of the RailDax file. Examples are the user creating the file, the date of creation, tools that created the RailDax dataset and their databases.

7 Use cases

7.1 General

The purpose of the high-level use case (UC) is to plan service operations. This use case can be deconstructed into six specific use cases that build upon each other to achieve the high-level use case of service operations planning. These six use cases cover the fields:

— infrastructure,

— rolling stock,

— timetable,

— signalling.

All relevant objects within these fields are on a functional/operational level.

The six use cases are:

1) schematic track plans for infrastructure planning (infrastructure);

2) runtime calculations (rolling stock, builds on UC1);

3) network statement of an infrastructure manager annex asset descriptions (infrastructure);

4) passenger assets of rolling stock for operational planning (rolling stock);

5) operational timetable planning (timetable, builds on UC1, 2, 3 and 4);

6) operational timetable simulation (signalling, builds on UC1, 2, and 5).

7.2 Operational timetable planning for tenders, long- and short-term planning

7.2.1 Description

This use case describes a timetable which is transferred with the aim of describing which timetable is asked for in the competition, i.e. the number of trains, their routes and (regular) operating days and any other properties of the trains such as services or minimum seating capacities.

The purpose of such a data exchange with RailDax files, particularly in the context of competitions of a large production amount, is to facilitate the reproduction of the timetable in the bidders' own software systems. This allows a quicker calculation of the number of vehicles necessary or the price. Time consuming and potential faulty manual transcription of the timetable is avoided. The usage of RailDax for such data exchange (instead of more special file formats) keeps this procedure transparent and free from discrimination.

7.2.2 Data flows and interfaces

Based on experience, it is typical for RailDax files of this use case to contain a minimum infrastructure only. The main reason for this is perhaps that the issuing authority is not normally the infrastructure manager. Nevertheless, general infrastructure data are published in a competition as a kind of “passing through” issue from the infrastructure manager(s) through the issuing authority. The `<operatingPeriod>`s of the trains have regular weekdays only, with no special operating days and no certain date references. The timetables are typically not planned with regard to public holidays, etc. The amount of train kilometres is only calculated on a statistical basis (flat number of operating days, statistic year). However, there may be a `<timetablePeriod>` defined for one year as a placeholder for all the years. This is common practice for seasonal train operation, e.g. school trains or summer-only trains. Theoretically there may be a very long `<timetablePeriod>` covering all the years of the timetable's validity. Since this leads to very long bitmasks of `<operatingPeriod>`s, normally with no additional benefit, it is not common and not recommended.

7.2.3 Characteristics of data

Table 1 shows the characteristics of data for operational timetable planning.

Table 1 — Characteristics of data for operational timetable planning

Topic	Parameters	
How often do the data change (update)?	static (not changing)	
	yearly	x
	regular changes	x
	monthly	
	weekly	x
	daily	x

Table 1 *(continued)*

Topic	Parameters	
	realtime (seconds)	x
How big are the data fragments to be exchanged (complexity)?	tiny (attribute)	
	very small (element)	x
	small (operational point)	x
	big (station/yard)	
	huge (region)	x
	whole data set (network)	x
Which views are represented by the data (focus)?	Statistics	
	Signalling	
	Construction	
	Geometry	
	Geodesy	
	Energy	
	Timetable	x

7.3 Runtime calculations

7.3.1 Description

For various purposes it is necessary to simulate the process of railway operation for a fragment or an entire network in order to investigate questions concerning:

— the construction/fine tuning of a timetable;

— the feasibility of a timetable;

— the stability of a timetable;

— the driving capability of rolling stock;

— the capacity of infrastructure;

— feasibility studies for new infrastructure

7.3.2 Data flows and interfaces

Provision of rolling stock data by railway operator and/or manufacturer.

Import of rolling stock data into the simulation system (currently manual).

7.3.3 Characteristics of data

Table 2 shows the characteristics of data for runtime calculations.

Table 2 — Characteristics of data for runtime calculations

Topic	Parameters	
How often do the data change (update)?	static (not changing)	x
	yearly	
	regular changes	
	monthly	

Table 2 *(continued)*

Topic	Parameters	
	weekly	
	daily	
	realtime (seconds)	
How big are the data fragments to be exchanged (complexity)?	tiny (attribute)	x
	very small (element)	
	small (operational point)	
	big (station/yard)	
	huge (region)	
	whole data set (network)	
Which views are represented by the data (focus)?	Statistics	x
	Signalling	
	Construction	
	Geometry	
	Geodesy	
	Energy	
	...	

7.4 Passenger assets of rolling stock for operational planning

7.4.1 Description

The primary purpose of this use case is to determine a train's passenger capacity. A secondary purpose is to determine the function and quality of this capacity. At a later stage the use case will be extended to also include a train's freight capacity.

7.4.2 Data flow and interfaces

From rolling stock asset manager or rolling stock producer to capacity planner.

7.4.3 Characteristics of data

Table 3 shows the characteristics of data for passenger assets of rolling stock.

Table 3 — Characteristics of data for passenger assets of rolling stock

Topic	Parameters	
How often do the data change (update)?	static (not changing)	x
	yearly	
	regular changes	
	monthly	
	weekly	
	daily	
	realtime (seconds)	
How big are the data fragments to be exchanged (complexity)?	tiny (attribute)	x
	very small (element)	
	small (operational point)	
	big (station/yard)	

Table 3 *(continued)*

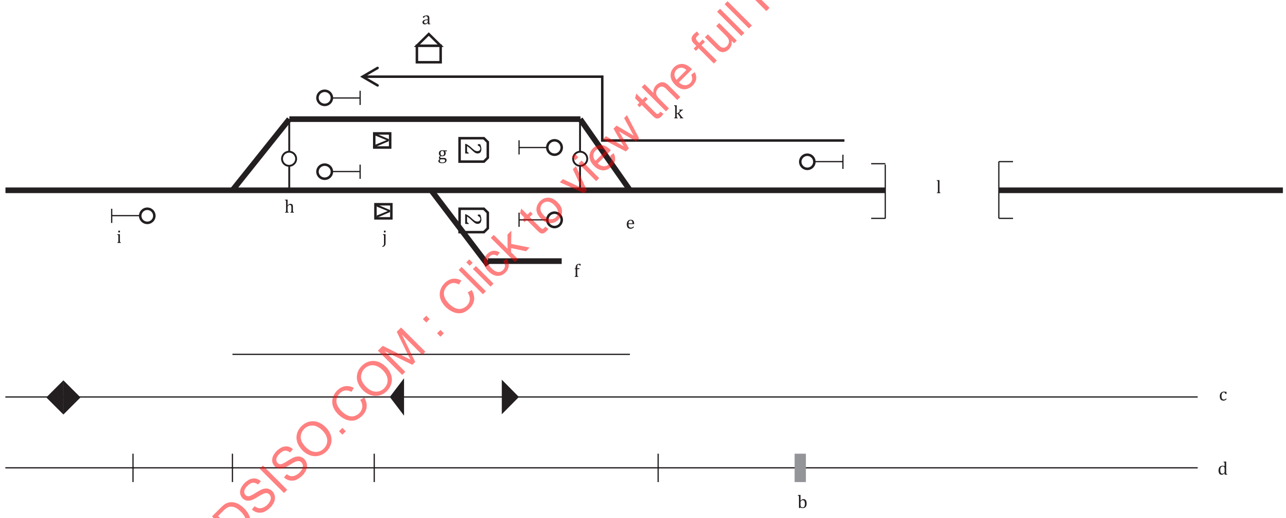
Topic	Parameters	
	huge (region)	
	whole data set (network)	
Which views are represented by the data (focus)?	Statistics	x
	Signalling	
	Construction	
	Geometry	
	Geodesy	
	Energy	
	...	

7.5 Schematic track plans for infrastructure planning

7.5.1 Description

Schematic track plans visualize the track's topology (all elements have screen coordinates resulting from their complete schematic layout), with functional elements placed on the tracks.

Figure 37 depicts an example for a schematic drawing.



Key

a Timing point.

b Discontinuation.

c Speed.

d Gradients.

e Switches.

f End of track.

g Stop marker.

h Signals.

i Clearance.

j Rep. balises.

k Route.

l Tunnel.

Figure 37 — Example schematic track plan