railML.org®

railML® Infrastructure v2.2 final

Summary of the modifications

Christian Rahmig

Updated 18.09.2013

Deutsches Zentrum für Luft- und Raumfahrt e.V.
in der Helmholtz-Gemeinschaft

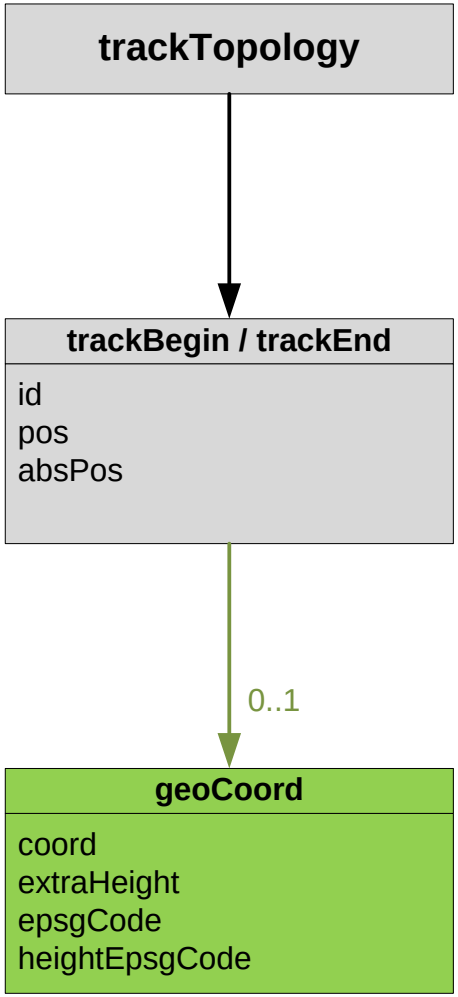
Overview

- Track Node Coordinates
- Mileage Changes
- Directions
- Speed Profiles and Speed Changes
- Platforms and Service
- Stop Posts
- Line Codes / Infrastructure Manager Codes
- Operation Control Points
- Balises and Train Protection
- Signals
- States of infrastructure

TRACK NODE COORDINATES

Track Node coordinates in railML® 2.2

- the **geoCoord** element has been added in type "tTrackNode“
- More details: Trac Ticket #135
  https://trac.assembla.com/railML/ticket/135

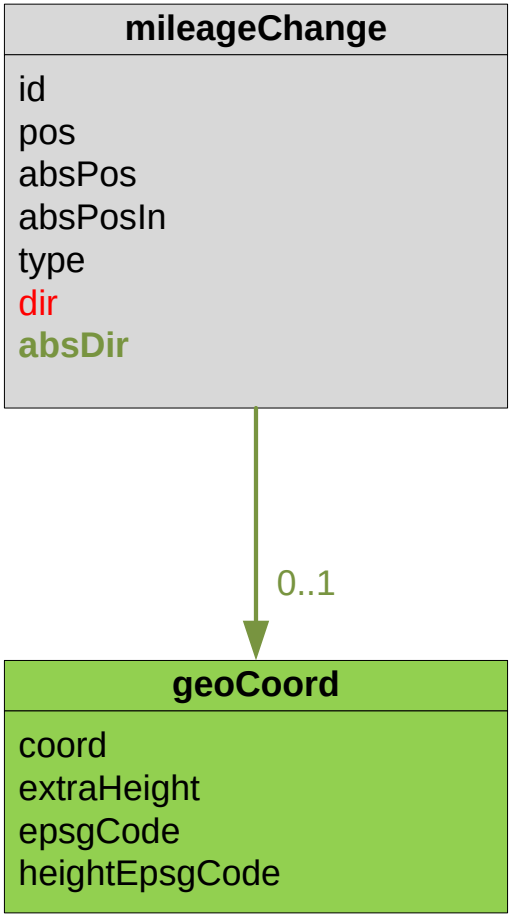


MILEAGE CHANGES

Mileage direction and coordinates in railML® 2.2

- the absolute (mileage) direction starting from this point on the track
- optional parameter ***absDir*** with values 'raising' and 'falling'
- marking parameter ***dir*** with values 'up' and 'down' as deprecated

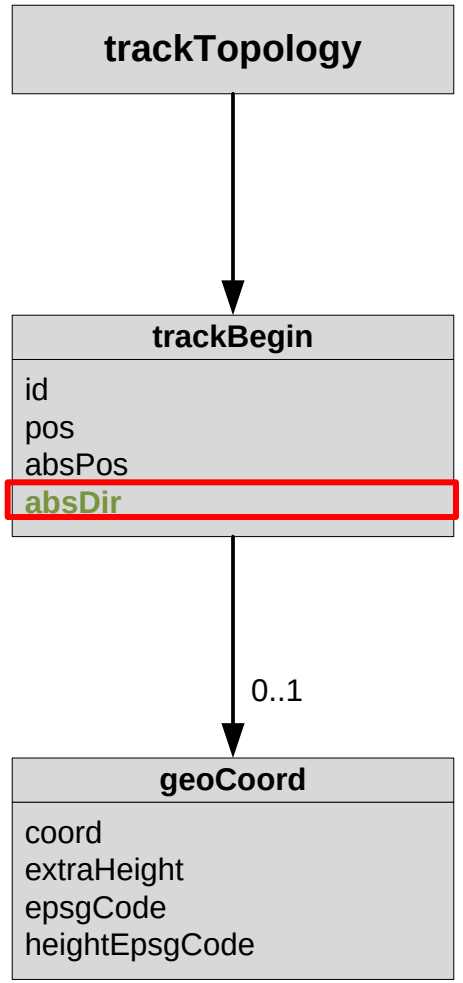
- the **geoCoord** element has been added in type "tMileageChange"
- More details: Trac Ticket #134
  https://trac.assembla.com/railML/ticket/134



Track Begin mileage direction in railML® 2.2

- describes an absolute (mileage) direction at the beginning of a track
- optional parameter ***absDir*** with values 'raising' and 'falling,

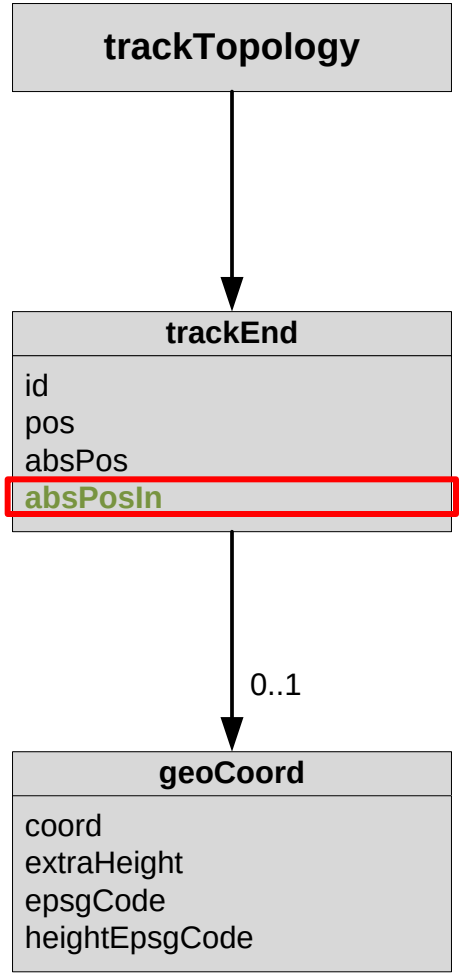
- More details: Trac Ticket #159
  https://trac.assembla.com/railML/ticket/159



Track End mileage position in railML® 2.2

- describes the absolute (mileage) position of the (incoming) line in the end of a track
- optional parameter ***absPosIn*** with values of type tLengthM

- More details: Trac Ticket #159
  https://trac.assembla.com/railML/ticket/159

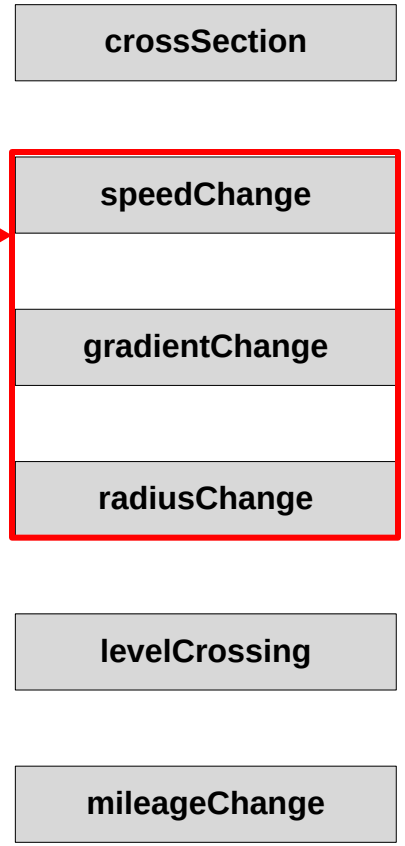


DIRECTIONS

Direction types in railML® 2.2

- **tStrictDirection** is the most constrained direction statement {up, down}
- **tDelimitedDirection** allows for an unknown direction {up, down, unknown}
- **tExtendedDirection** provides a non-directional value {up, down, unknown, none}
- **tLaxDirection** is the generic type for any direction statements {up, down, unknown, none, both}
- **tMileageDirection** defines the change of the mileage {raising, falling}

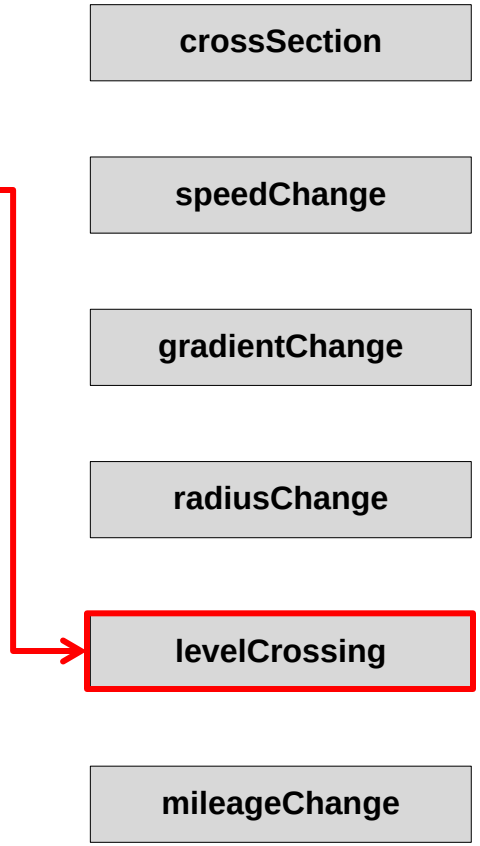
- More details: Trac Ticket #62
  https://trac.assembla.com/railML/ticket/62



Direction types in railML® 2.2

- **tStrictDirection** is the most constrained direction statement {up, down}
- **tDelimitedDirection** allows for an unknown direction {up, down, unknown}
- **tExtendedDirection** provides a non-directional value {up, down, unknown, none}
- **tLaxDirection** is the generic type for any direction statements {up, down, unknown, none, both}
- **tMileageDirection** defines the change of the mileage {raising, falling}

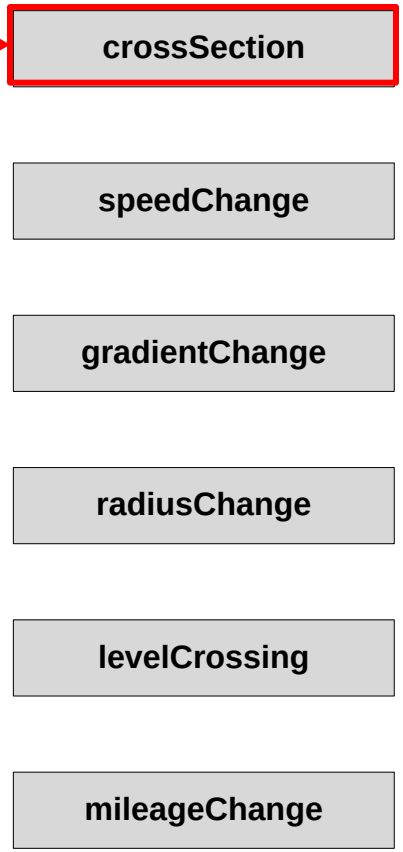
- More details: Trac Ticket #62
  https://trac.assembla.com/railML/ticket/62



Direction types in railML® 2.2

- **tStrictDirection** is the most constrained direction statement {up, down}
- **tDelimitedDirection** allows for an unknown direction {up, down, unknown}
- **tExtendedDirection** provides a non-directional value {up, down, unknown, none}
- **tLaxDirection** is the generic type for any direction statements {up, down, unknown, none, both}
- **tMileageDirection** defines the change of the mileage {raising, falling}

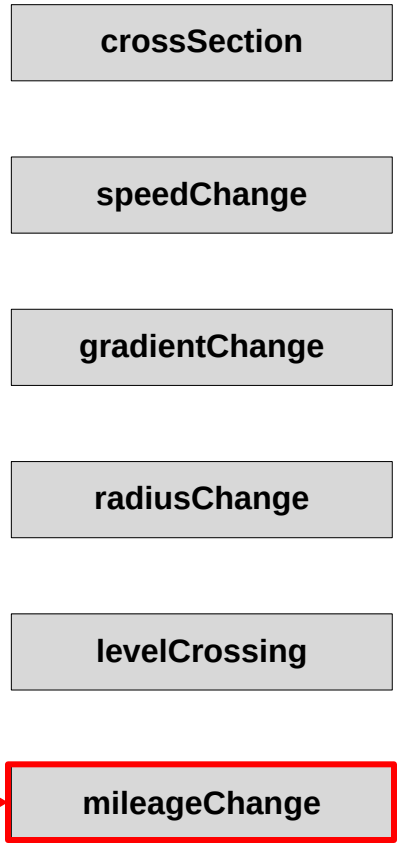
- More details: Trac Ticket #62
  https://trac.assembla.com/railML/ticket/62



Direction types in railML® 2.2

- **tStrictDirection** is the most constrained direction statement {up, down}
- **tDelimitedDirection** allows for an unknown direction {up, down, unknown}
- **tExtendedDirection** provides a non-directional value {up, down, unknown, none}
- **tLaxDirection** is the generic type for any direction statements {up, down, unknown, none, both}
- **tMileageDirection** defines the change of the mileage {raising, falling}

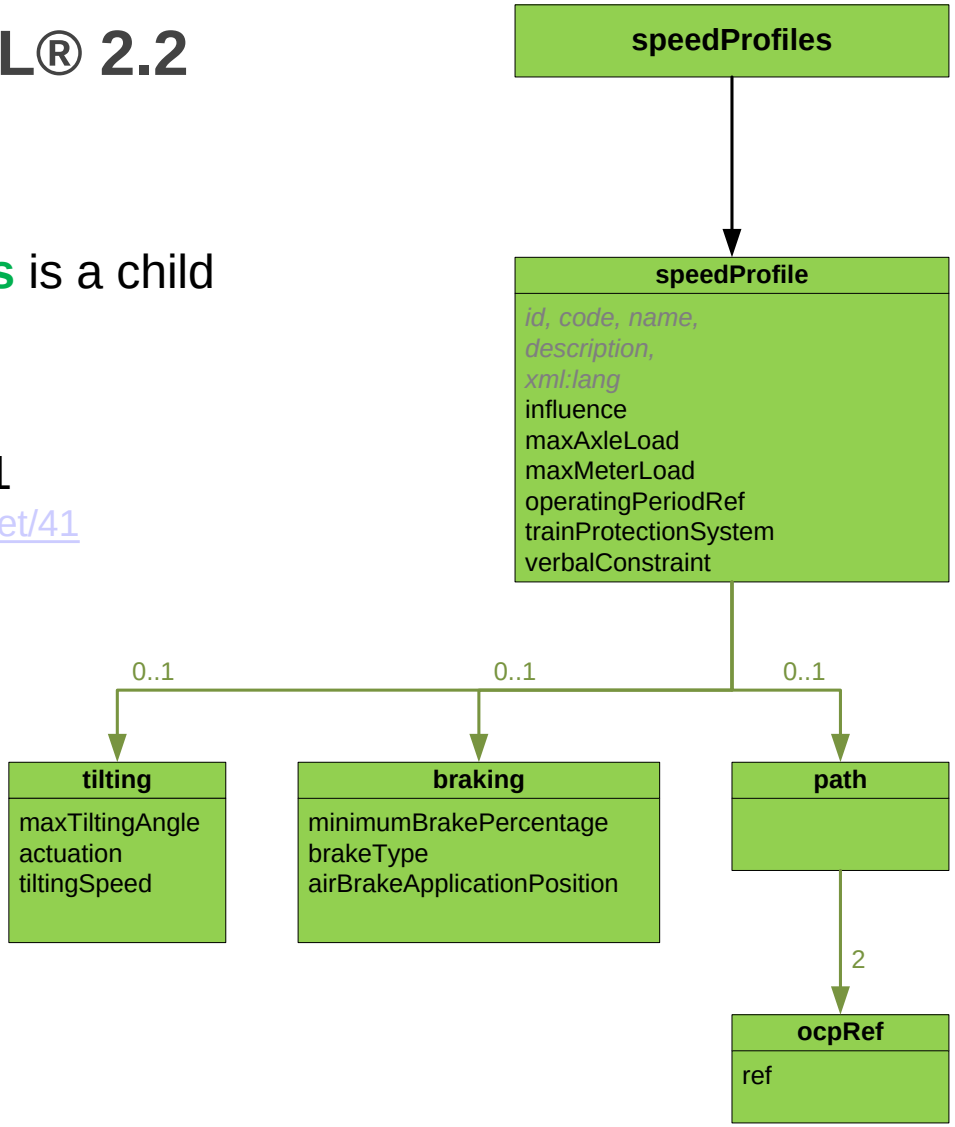
- More details: Trac Ticket #62
  https://trac.assembla.com/railML/ticket/62



SPEED

Speed profiles in railML® 2.2

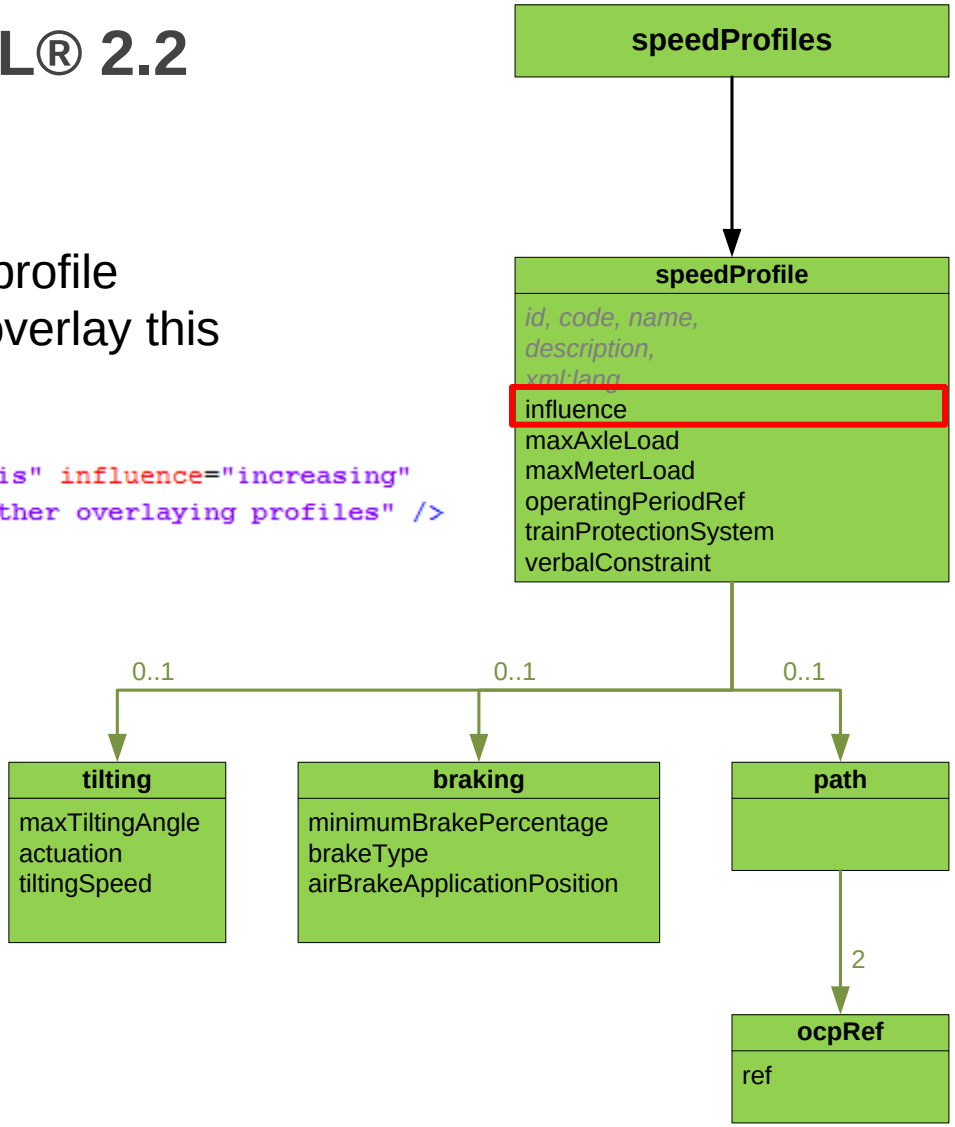
- The container **speedProfiles** is a child element of **infrastructure**
- More details: Trac Ticket #41
  https://trac.assembla.com/railML/ticket/41



Speed profiles in railML® 2.2

- Example 1: the base speed profile (further speed profiles may overlay this speed profile)

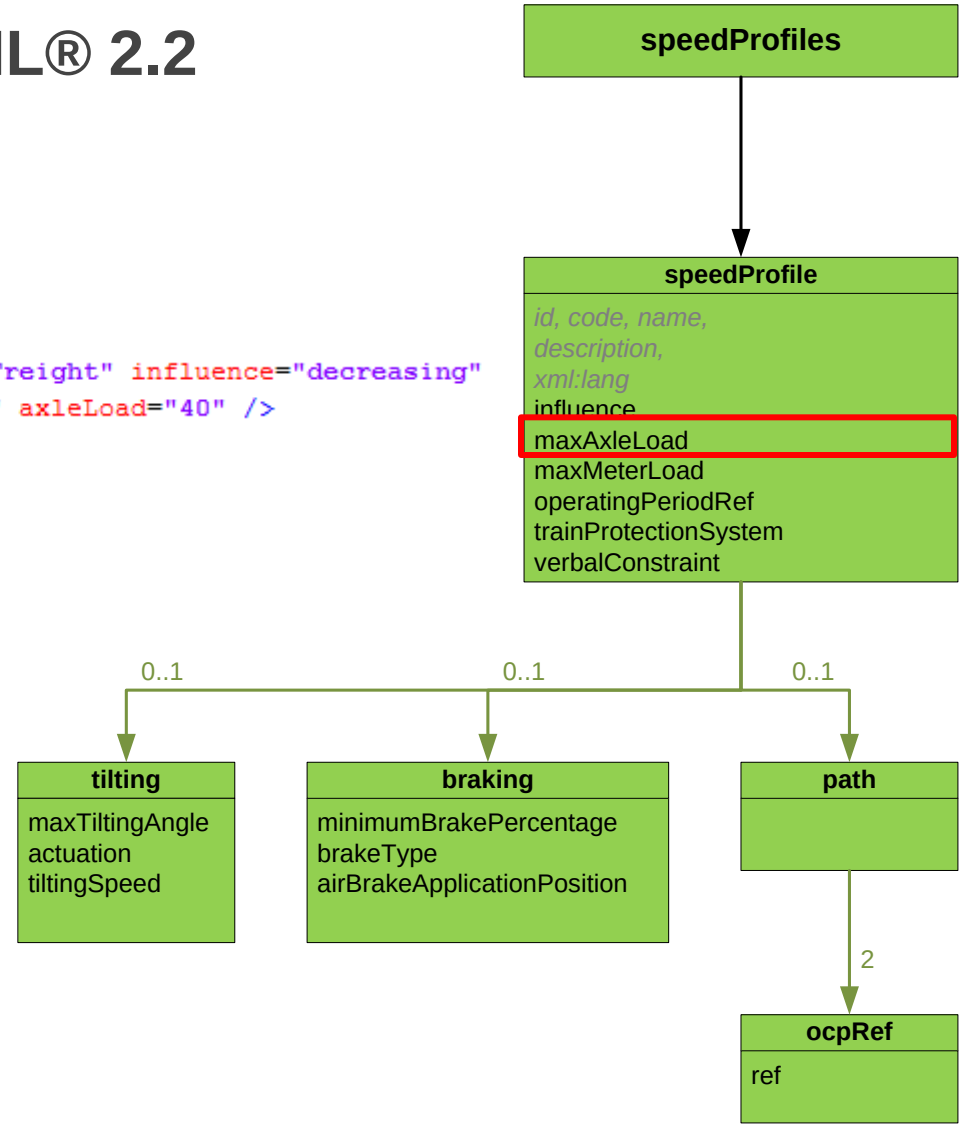
```
<rail:speedProfile id="sp01" code="A12" name="Basis" influence="increasing"
description="Speed profile as general basis for other overlaying profiles" />
```



Speed profiles in railML® 2.2

- Example 2:

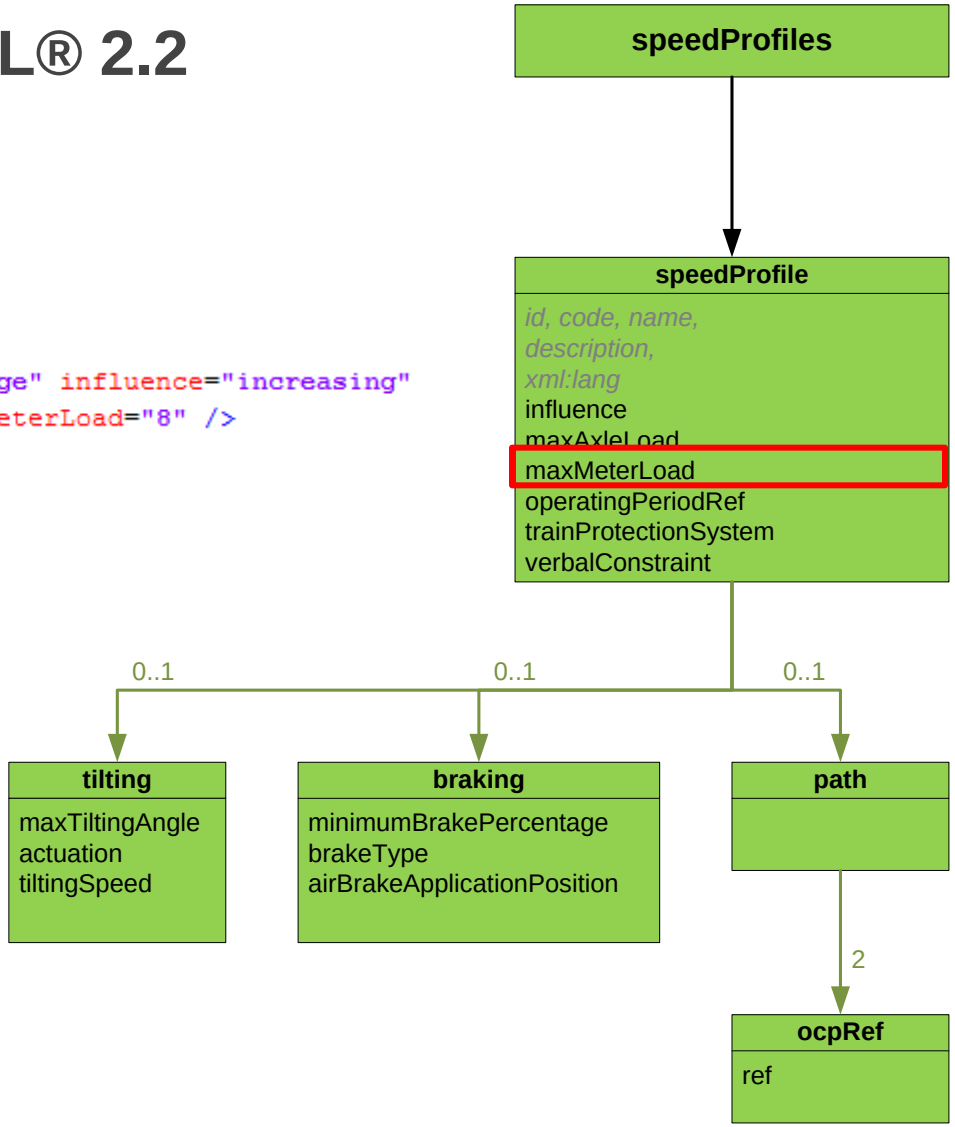
```
<rail:speedProfile id="sp02" code="B3" name="Heavy Freight" influence="decreasing"
description="Speed profile for heavy freight trains" axleLoad="40" />
```



Speed profiles in railML® 2.2

- Example 3:

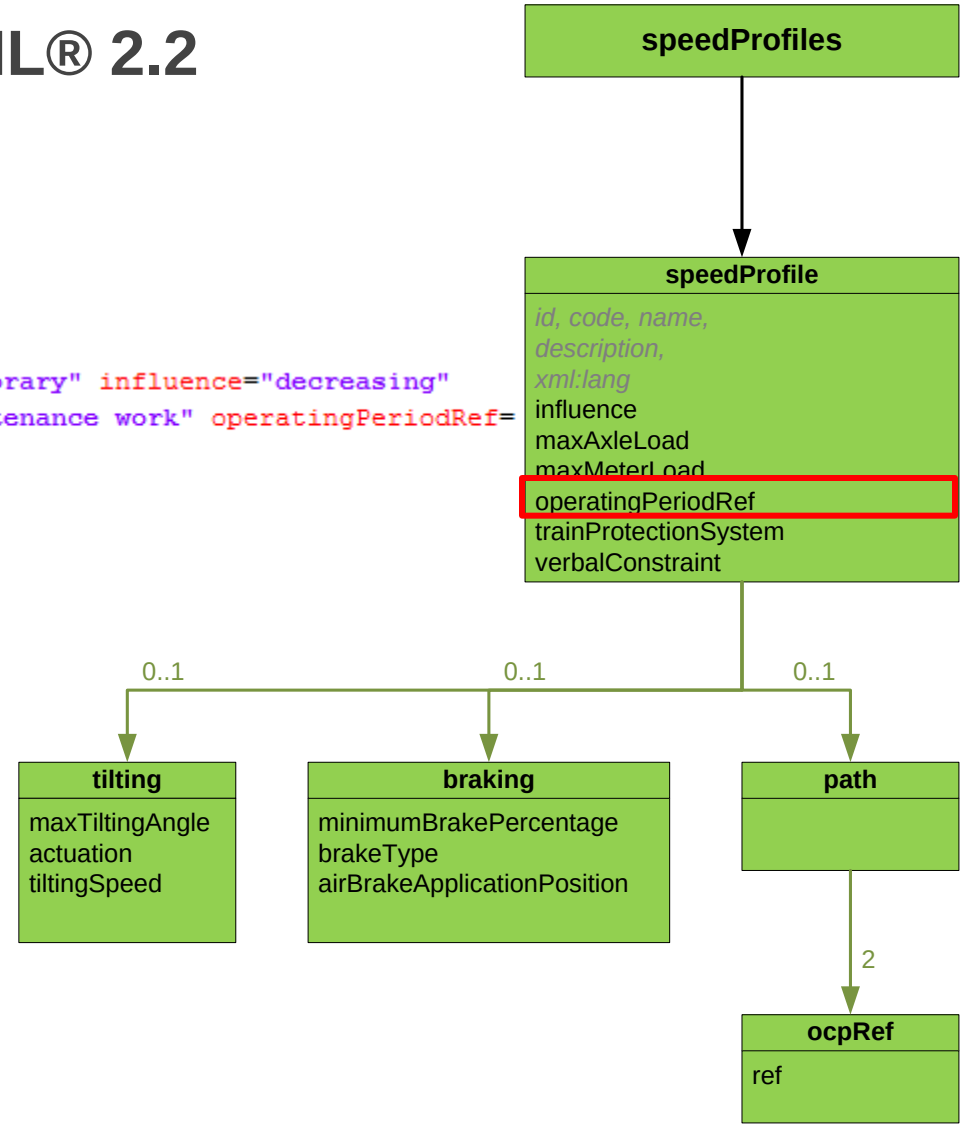
```
<rail:speedProfile id="sp03" code="C1" name="Bridge" influence="increasing"
description="Speed profile for typical bridges" meterLoad="8" />
```



Speed profiles in railML® 2.2

- Example 4:

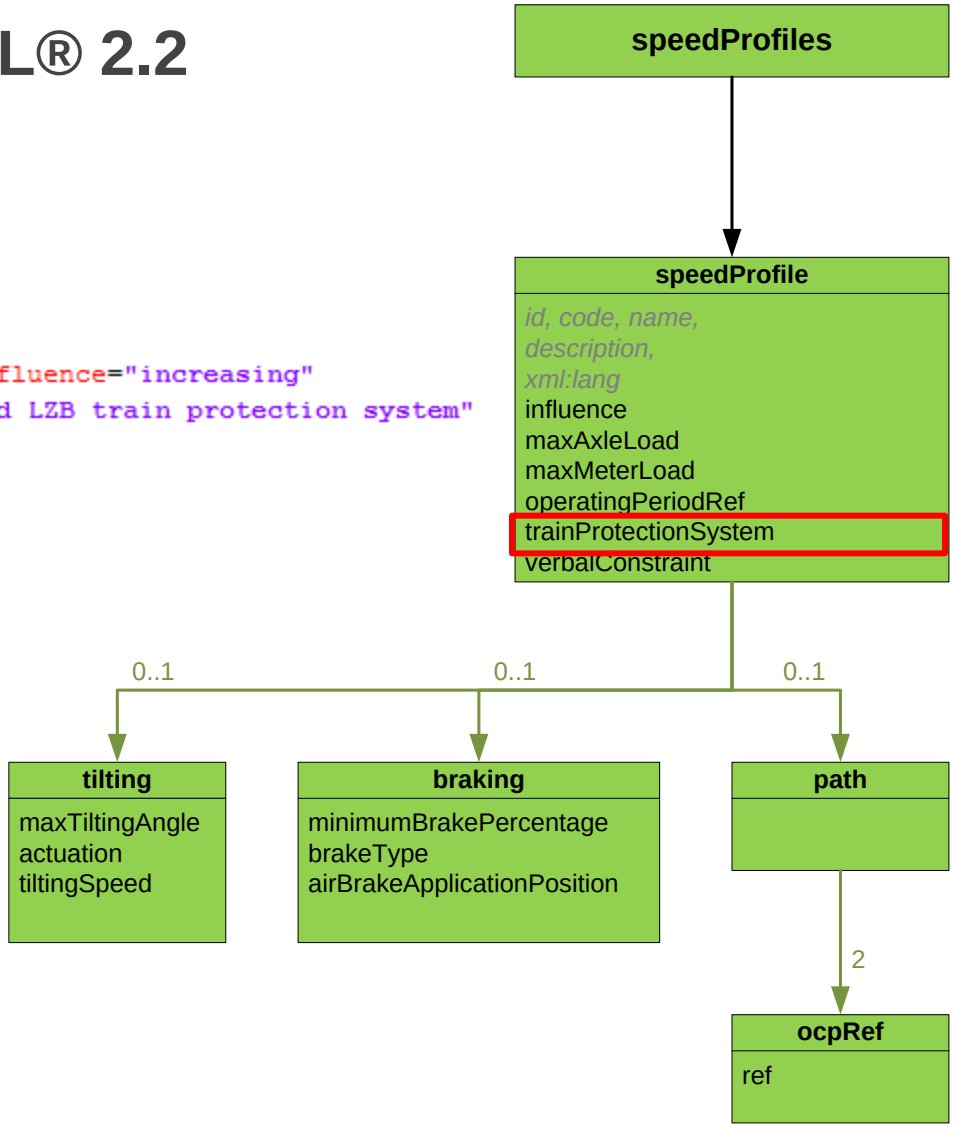
```
<rail:speedProfile id="sp04" code="T" name="Temporary" influence="decreasing"
description="Temporary speed profile due to maintenance work" operatingPeriodRef=
"op_1" />
```



Speed profiles in railML® 2.2

- Example 5:

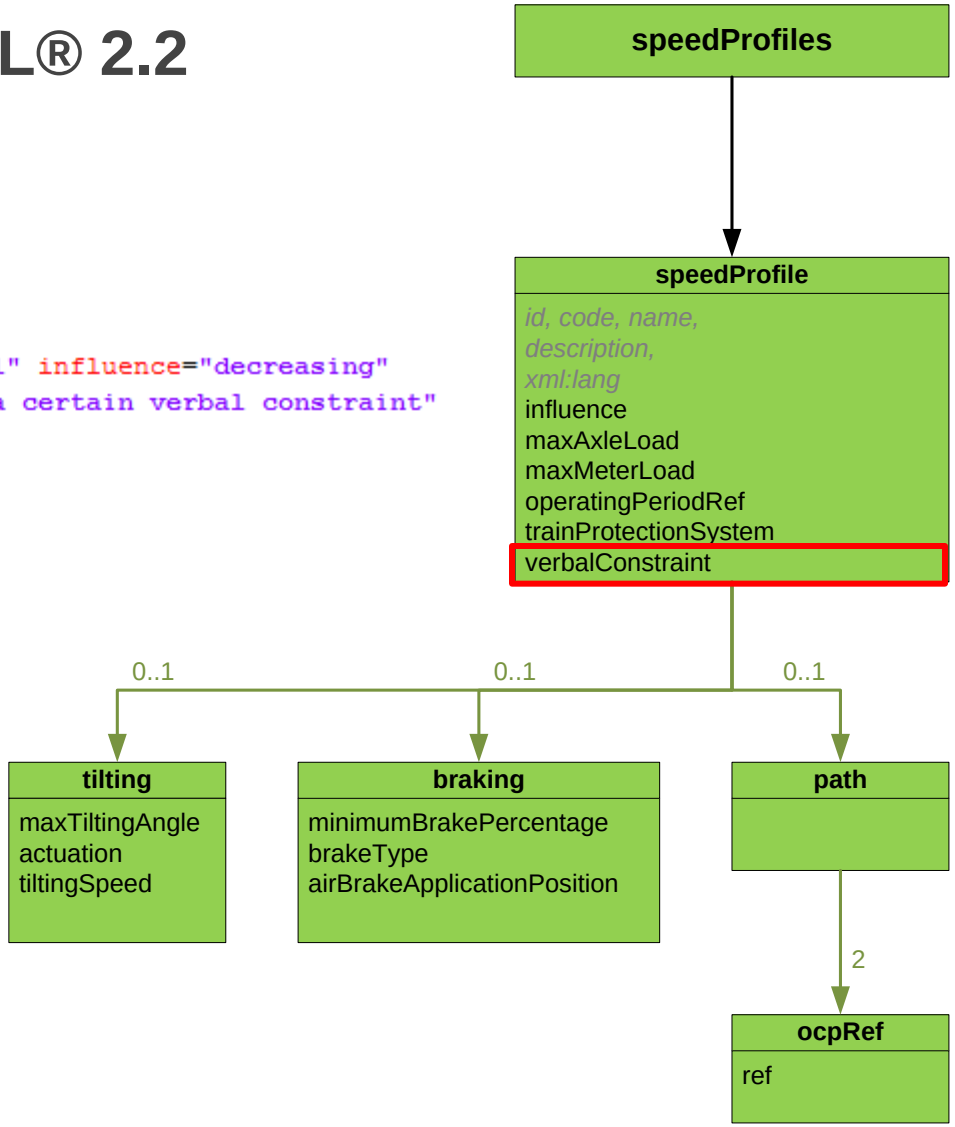
```
<rail:speedProfile id="sp6" code="HS" name="LZB" influence="increasing"
description="Speed profile for trains with activated LZB train protection system"
trainProtectionSystem="LZB"/>
```



Speed profiles in railML® 2.2

- Example 6:

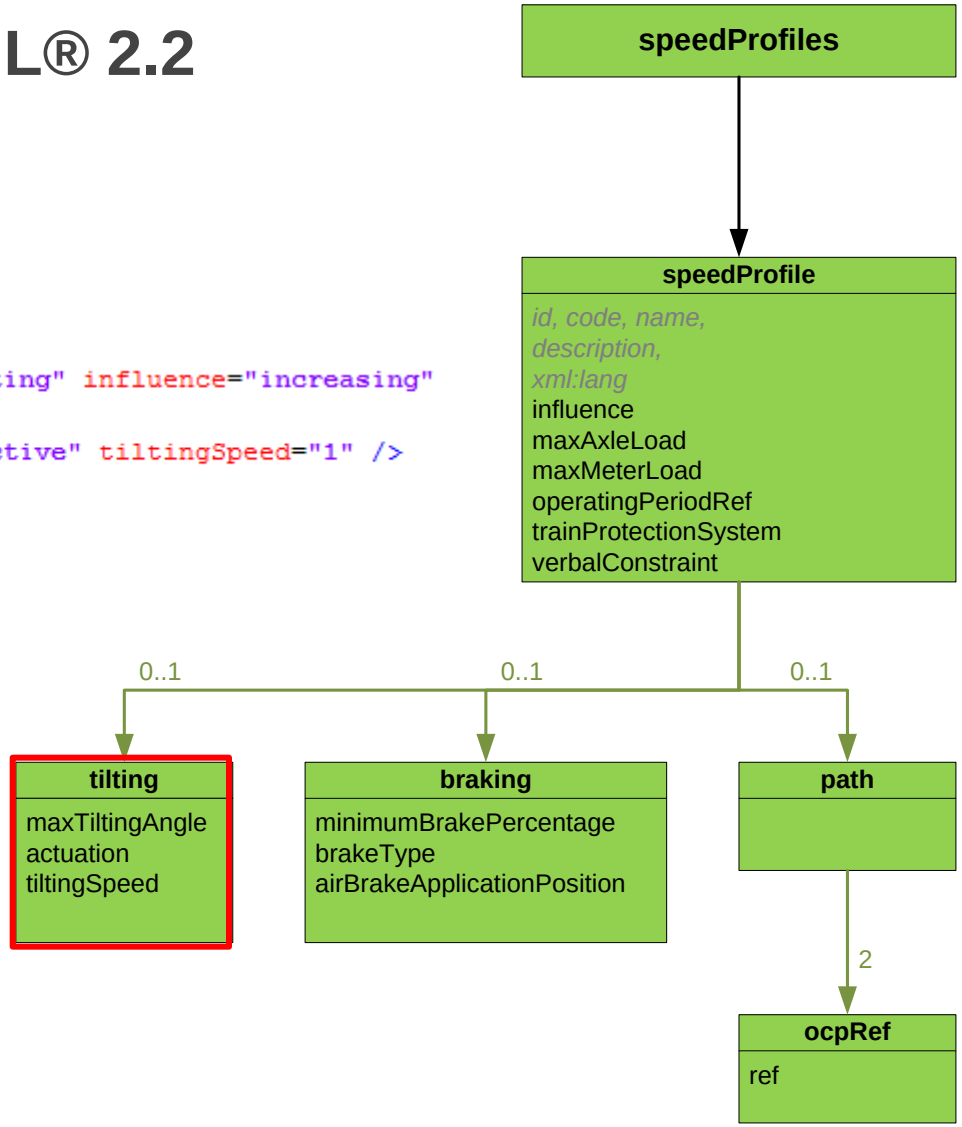
```
<rail:speedProfile id="sp06" code="V" name="Verbal" influence="decreasing"
description="Speed profile for trains fulfilling a certain verbal constraint"
 verbalConstraint="Desiro"/>
```



Speed profiles in railML® 2.2

- Example 7:

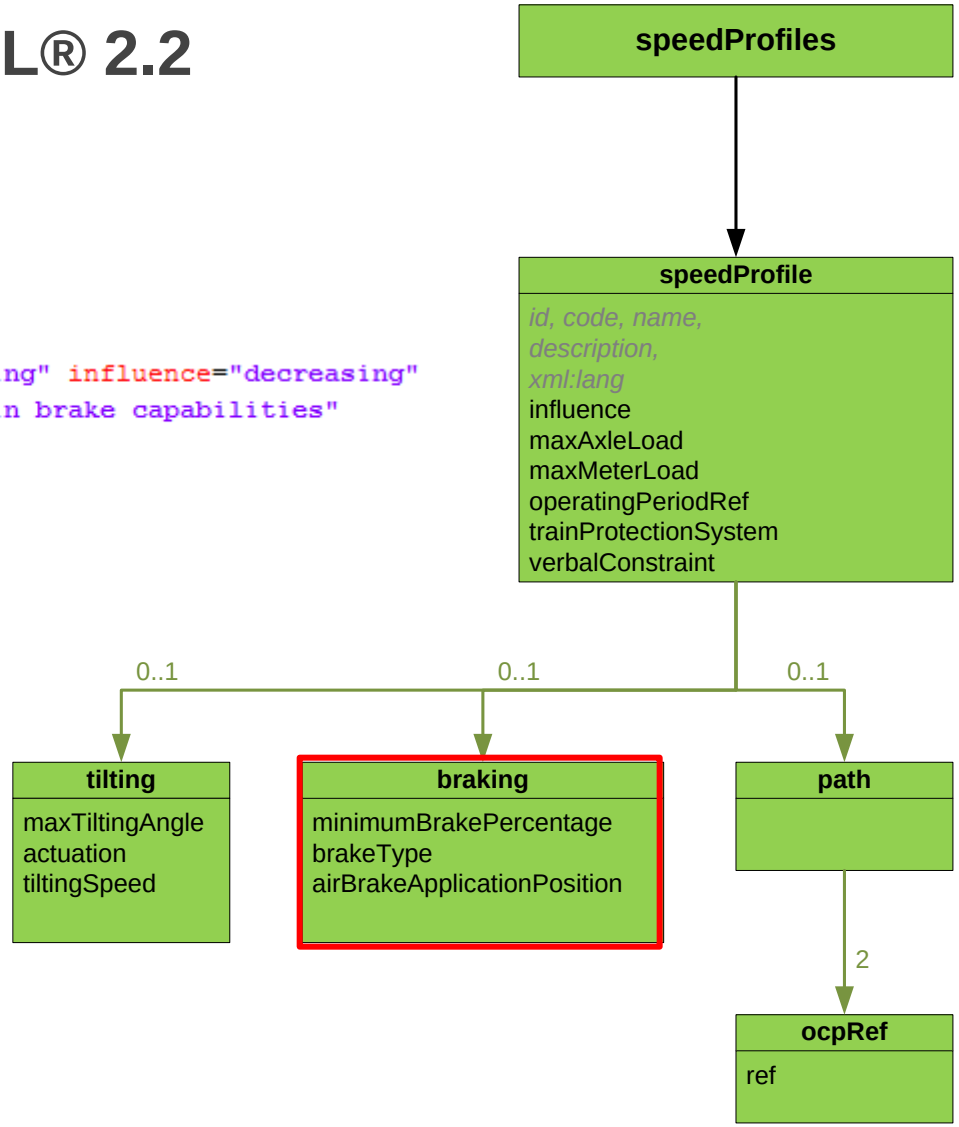
```
<rail:speedProfile id="sp05" code="T5" name="Tilting" influence="increasing"
description="Speed profile for tilting trains">
  <rail:tilting maxTiltingAngle="8" actuation="active" tiltingSpeed="1" />
</rail:speedProfile>
```



Speed profiles in railML® 2.2

- Example 8:

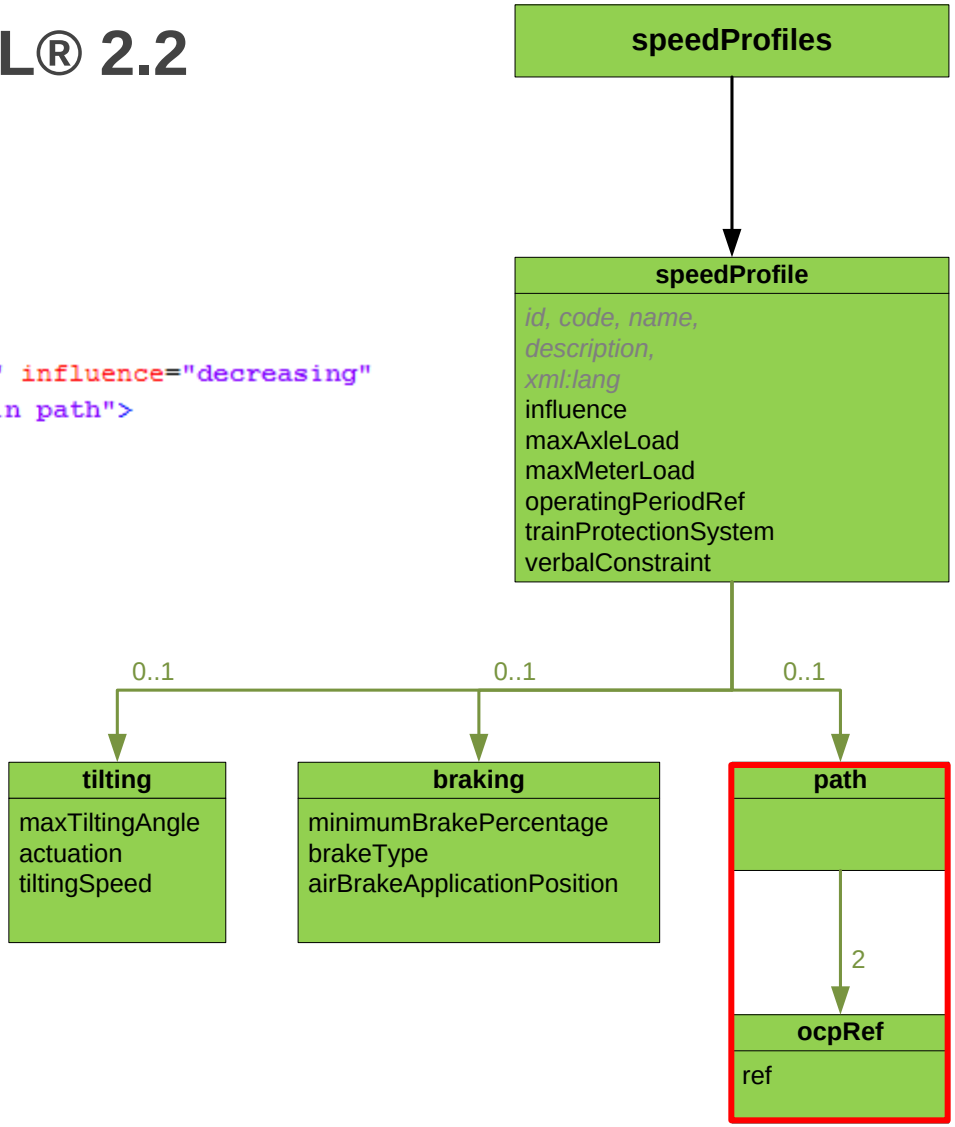
```
<rail:speedProfile id="sp08" code="B" name="Braking" influence="decreasing"
description="Speed profile for trains with certain brake capabilities"
minimumBrakePercentage="123"/>
```



Speed profiles in railML® 2.2

- Example 9:

```
<rail:speedProfile id="sp09" code="P" name="Path" influence="decreasing"
description="Speed profile for trains on a certain path">
  <rail:path>
    <rail:ocpRef ref="ocp01" />
    <rail:ocpRef ref="ocp02" />
  </rail:path>
</rail:speedProfile>
```



Speed changes in railML® 2.2

- A **speedProfile** is referenced from a **speedChange** using the new parameter ***profileRef***
- ***trainCategory*** is marked deprecated and replaced by ***etcsTrainCategory*** {integer}

page 38 in ERA_ERTMS_040001 "Assignment of values to ETCS variables" v1.9 from 10/07/12

- ***trainRelation*** references the part of train from where on the speed change is valid {headOfTrain, midOfTrain, endOfTrain}

Speed changes in railML® 2.2

- Specification boolean for a ***mandatoryStop,*** e.g. at a level crossing
- Specification boolean for a ***signalised*** speed profile, e.g. with a panel

- More details: Trac Tickets #41, #100, #190, #193
  https://trac.assembla.com/railML/ticket/41
  https://trac.assembla.com/railML/ticket/100
  http://trac.assembla.com/railML/ticket/190
  http://trac.assembla.com/railML/ticket/193

Speed changes in railML® 2.2

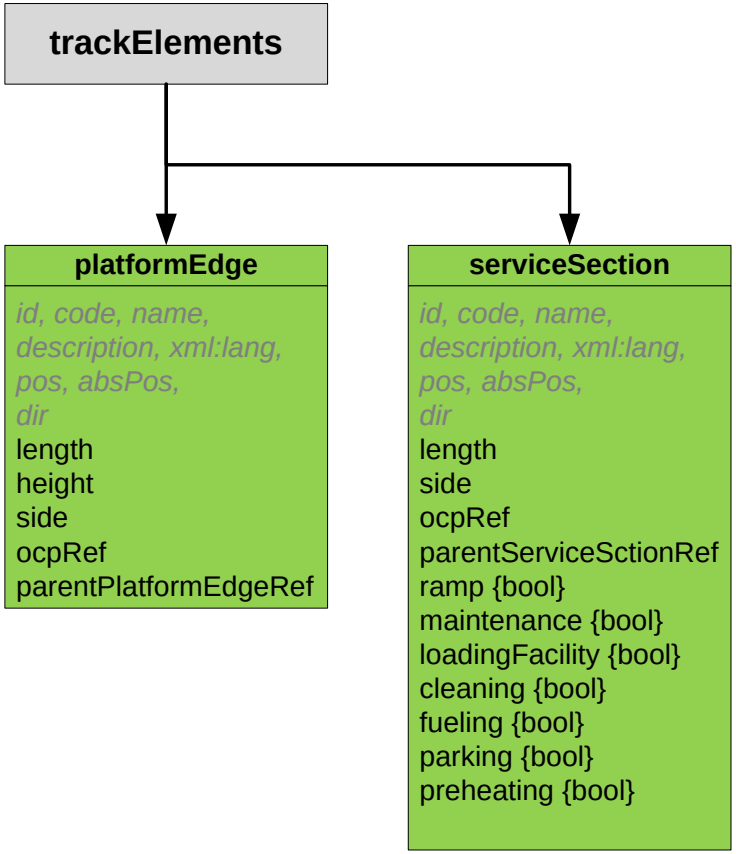
- Example for a speedChange:
  - ***General attributes (id, name, code, description, xml:lang)***
  - ***Positional data (pos, absPos, dir)***
  - ***vMax*** = 80
  - ***profileRef*** = …
  - ***trainRelation*** = headOfTrain
  - ***stop*** = false
  - ***signalised*** = true

http://www.tf-ausbildung.de

PLATFORMS & SERVICE

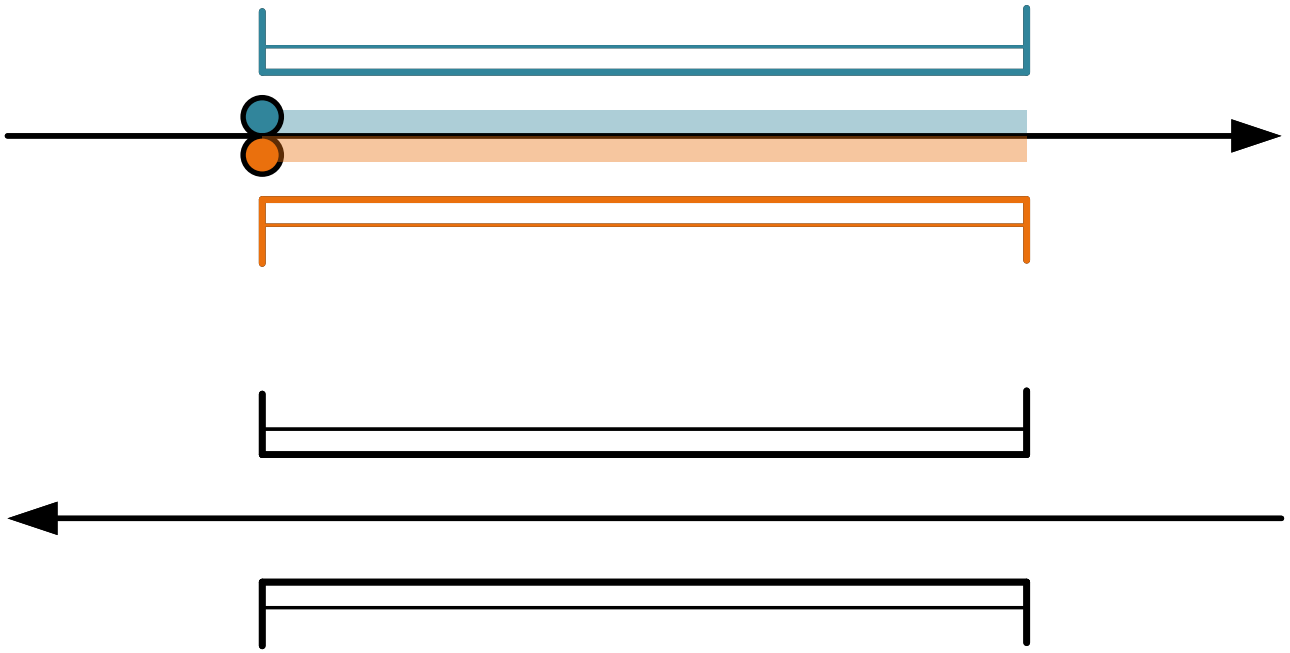
Modelling of platforms in railML® 2.2

- **platformEdge** and **serviceSection** are new track elements
- Parameters:
  - ***Positional data (pos, absPos, dir)***
  - ***Dimensions (length, height)***
  - ***side*** {left, right}
  - ***References*** (to ocp and to parent platform edge / service section)
  - ***type booleans*** {ramp, maintenance, loadingFacility, cleaning, fueling, parking, preheating, other}
- More details: Trac Ticket #122
  https://trac.assembla.com/railML/ticket/122



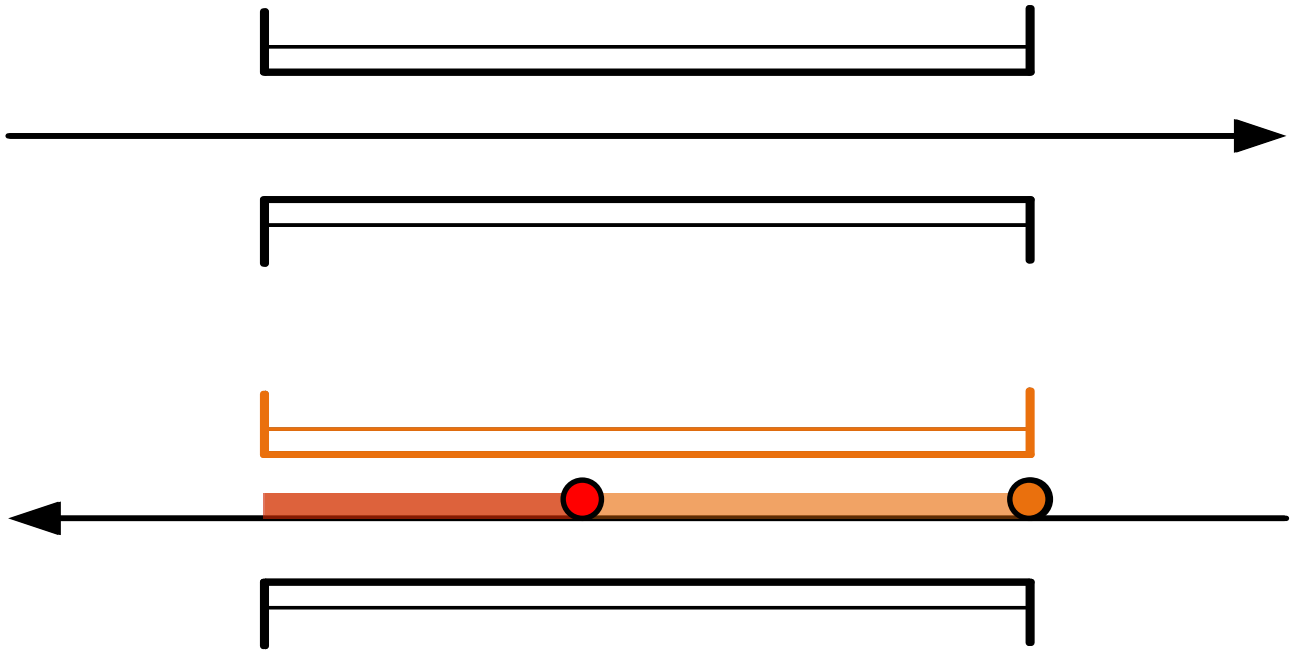
Modelling of platforms in railML® 2.2

- Example 1: platform edges on both sides of the track



Modelling of platforms in railML® 2.2

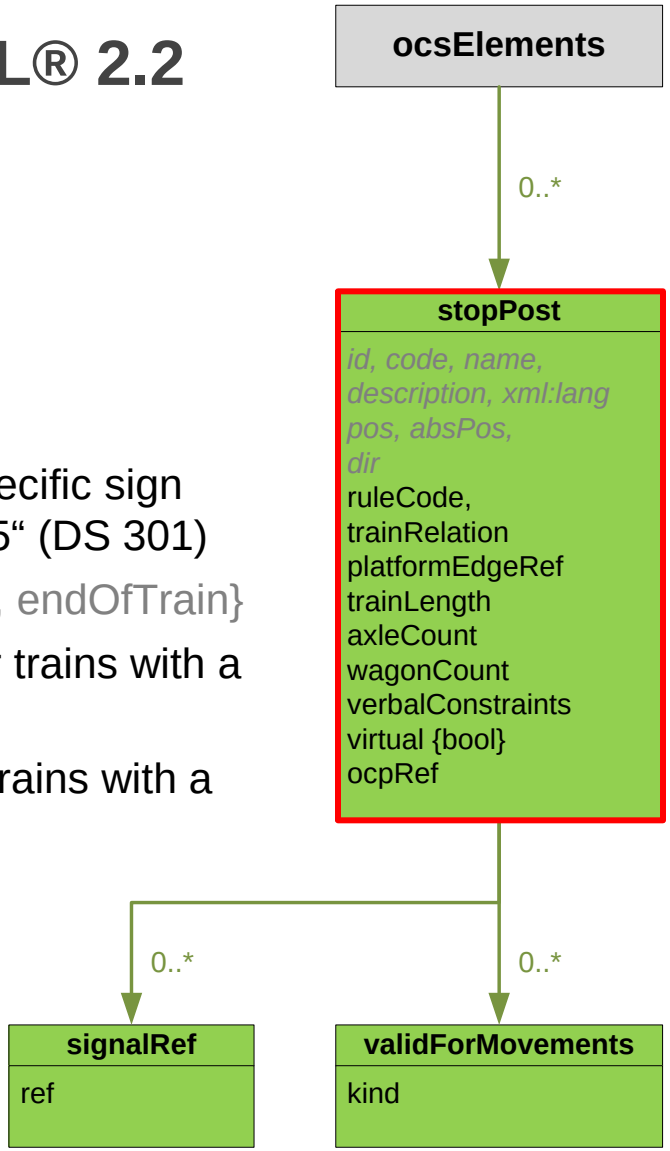
- Example 2: parent platform edges



STOP POSTS

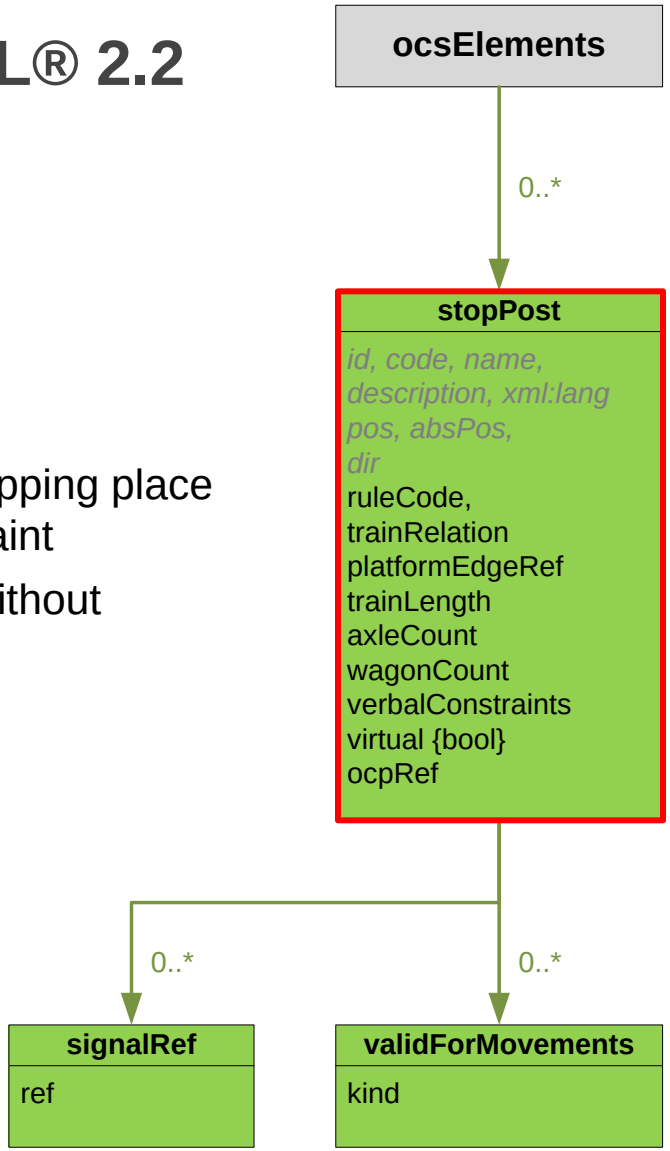
Modelling of stop posts in railML® 2.2

- **stopPost** is a new ocs element
- Parameters:
  - ***Identification and positional data***
  - ***ruleCode*** {string} defines the country-specific sign code for the signalling element, e.g. „Ne5“ (DS 301)
  - ***trainRelation*** {headOfTrain, midOfTrain, endOfTrain}
  - ***trainLength*** defines a stopping place for trains with a certain length
  - ***axleCount*** defines a stopping place for trains with a certain number of axles
  - ***wagonCount*** …
  - […]



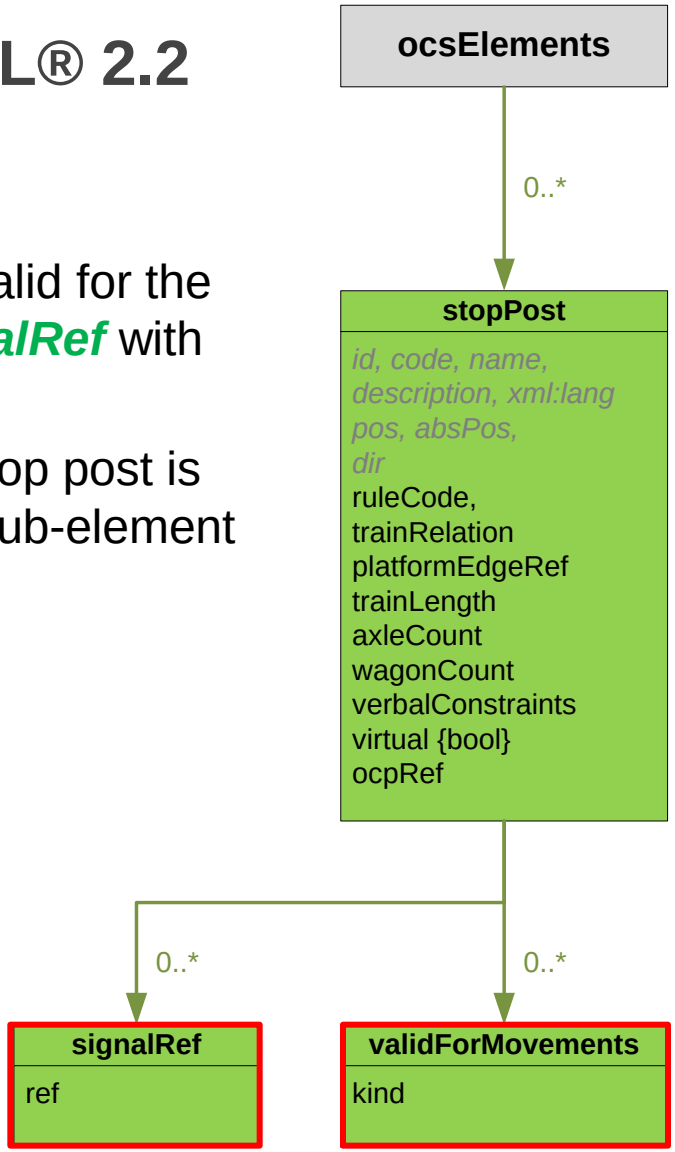
Modelling of stop posts in railML® 2.2

- **stopPost** is a new ocs element
- Parameters:
  - […]
  - ***verbalConstraints*** {string} defines a stopping place for trains fulfilling a certain verbal constraint
  - ***virtual*** {bool} defines a stopping place without physical stop post
  - ***References (platformEdge, ocp)***



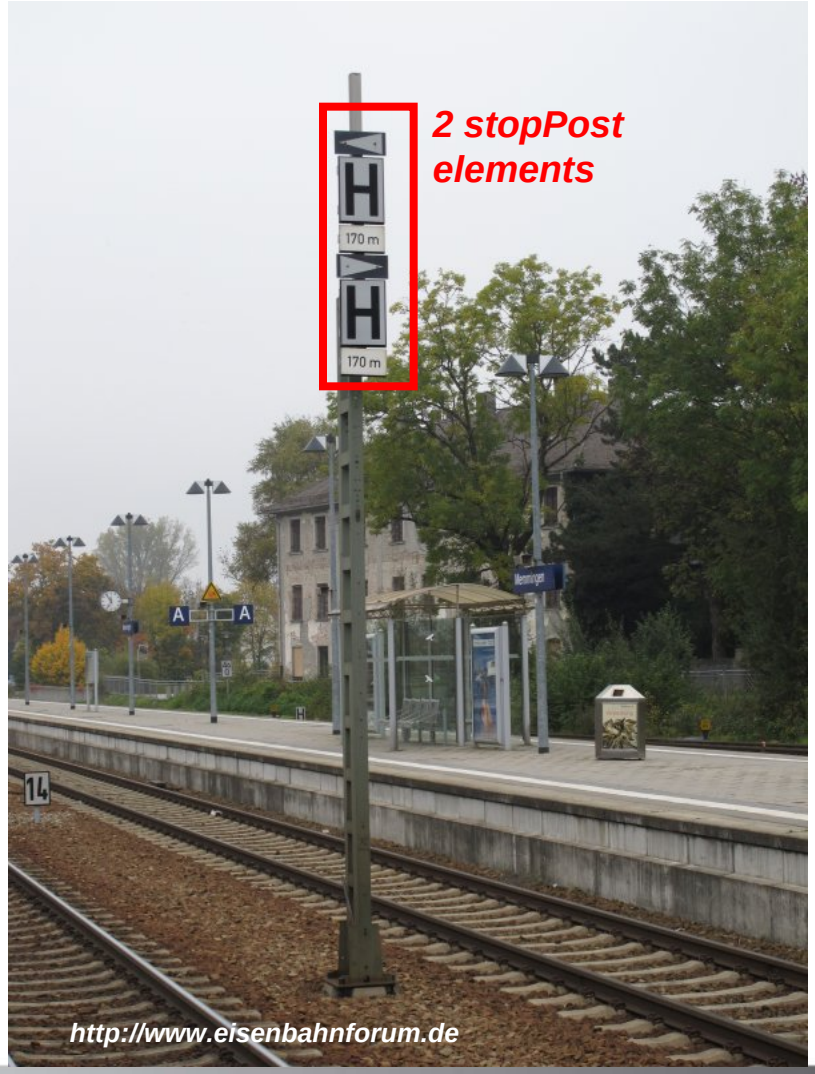
Modelling of stop posts in railML® 2.2

- For referencing various signals that are valid for the stop post, use the new sub-element ***signalRef*** with its parameter ***ref***
- For specifying the train movements the stop post is valid for, the parameter ***kind*** of the new sub-element **validForMovement** provides the values:
  - freightTrains
  - passengerTrains
  - allTrains
  - shunting
- More details: Trac Tickets #167, #198
  https://trac.assembla.com/railML/ticket/167
  https://trac.assembla.com/railML/ticket/198



Modelling of stop posts in railML® 2.2

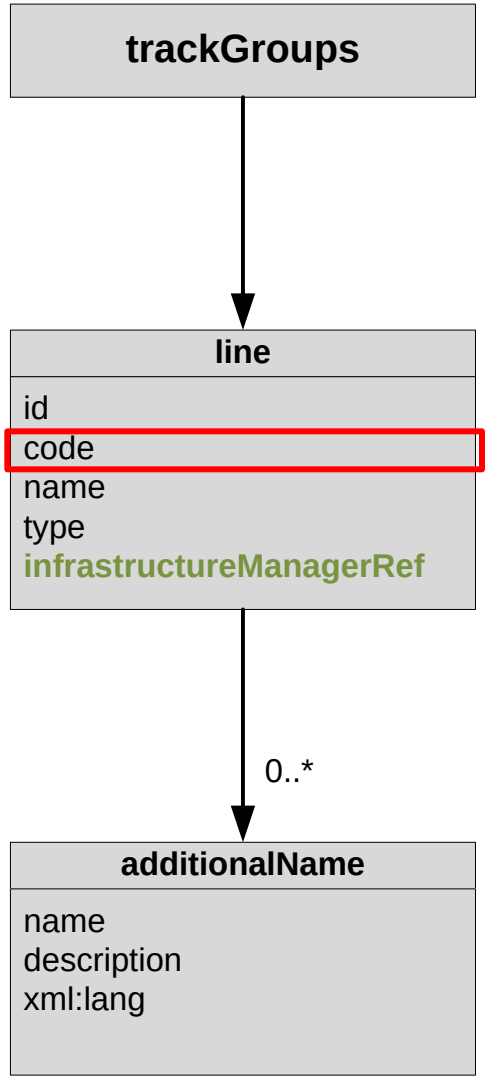
- example:



LINE CODES

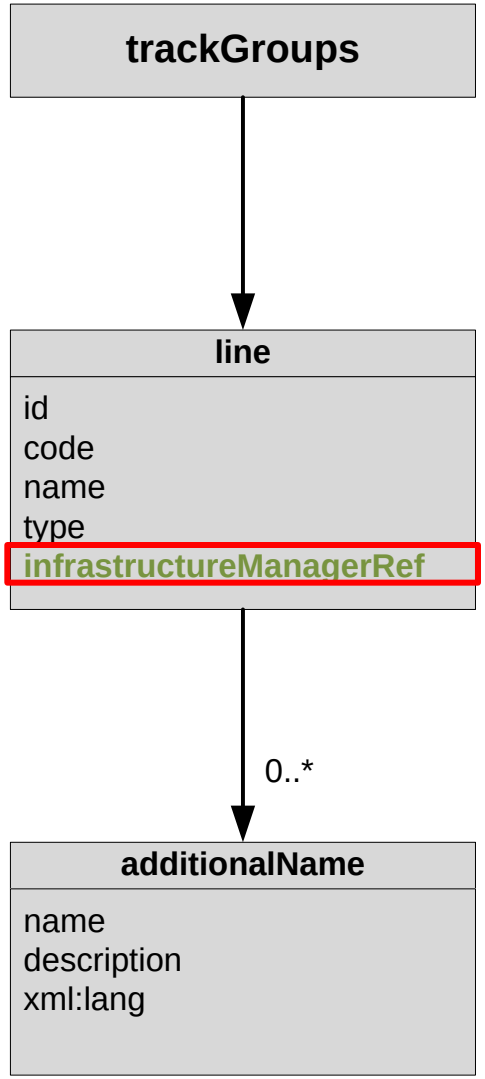
Different Line codes in railML® 2.2

- ***code***: defining the number of the line (without the UIC Country Code)



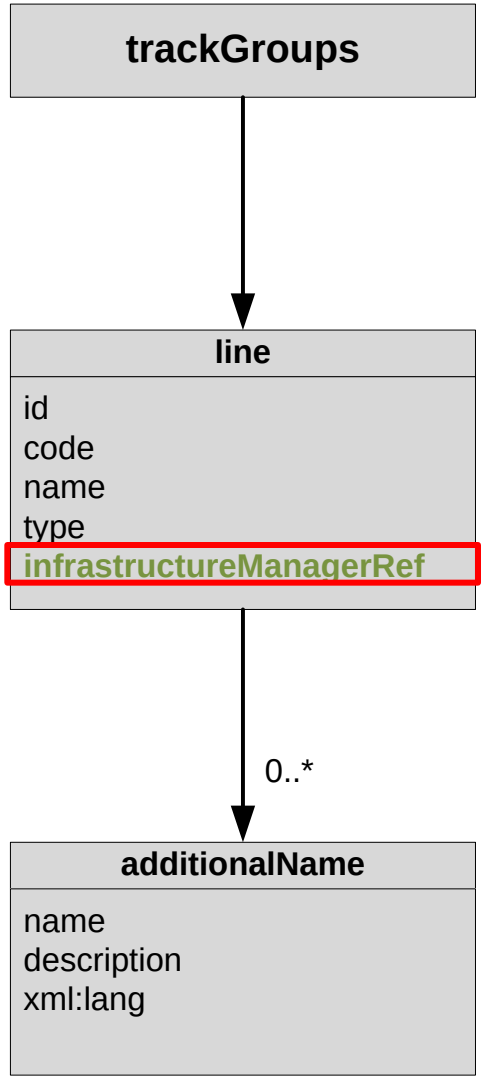
Different Line codes in railML® 2.2

- ***infrastructureManagerRef***: a string attribute referencing the code of an infrastructure manager
- The various infrastructure managers are listed in a separate XML file, which is a loosely coupled part of the schema: ***InfrastructureManagerCodes.xml***



Different Line codes in railML® 2.2

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!-- $Id: InfrastructureManagerCodes.xml 550 2013-02-18 15:47:52Z
susanne.wunsch $ -->
<infrastructureManagerCodes xmlns="http://www.railml.org/lists">
  <infrastructureManager code="ADF">
  <infrastructureManager code="AVG">
    <name>Albtal-Verkehrs-Gesellschaft mbH</name>
    <isoCountryCode>DE</isoCountryCode>
  </infrastructureManager>
  <infrastructureManager code="BC">
  <infrastructureManager code="BDK">
  <infrastructureManager code="BLS">
  <infrastructureManager code="BRE">
  <infrastructureManager code="CFL">
  <infrastructureManager code="CFM">
  <infrastructureManager code="CFR">
  <infrastructureManager code="CIE">
  <infrastructureManager code="DBN">
    <name>DB Netz</name>
    <isoCountryCode>DE</isoCountryCode>
  </infrastructureManager>
  <infrastructureManager code="DRE">
  <infrastructureManager code="EDS">
  <infrastructureManager code="evb">
  <infrastructureManager code="EVR">
  <infrastructureManager code="GKB">
  <infrastructureManager code="GyS">
  <infrastructureManager code="HSB">
```



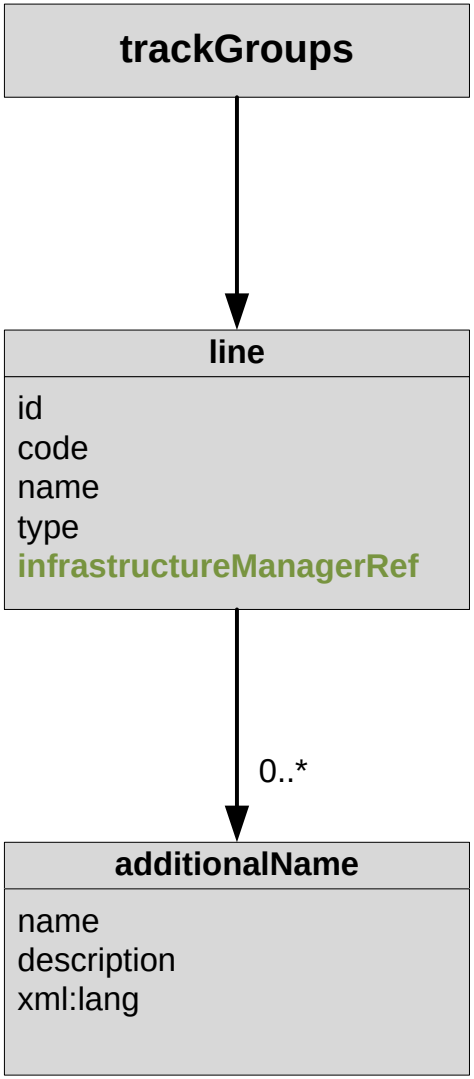
Different Line codes in railML® 2.2

- ***isoCountryCode***: a two-digit code defined as ISO 3166-1 alpha-2 in ISO 3166-1

Decoding table of ISO 3166-1 alpha-2 codes

AA	AB	AC	AD	AE	AF	AG	AH	AI	AJ	AK	AL	AM	AN	AO	AP	AQ	AR	AS	AT	AU	AV	AW	AX	AY	AZ
BA	BB	BC	BD	BE	BF	BG	BH	BI	BJ	BK	BL	BM	BN	BO	BP	BQ	BR	BS	BT	BU	BV	BW	BX	BY	BZ
CA	CB	CC	CD	CE	CF	CG	CH	CI	CJ	CK	CL	CM	CN	CO	CP	CQ	CR	CS	CT	CU	CV	CW	CX	CY	CZ
DA	DB	DC	DD	DE	DF	DG	DH	DI	DJ	DK	DL	DM	DN	DO	DP	DQ	DR	DS	DT	DU	DV	DW	DX	DY	DZ
EA	EB	EC	ED	EE	EF	EG	EH	EI	EJ	EK	EL	EM	EN	EO	EP	EQ	ER	ES	ET	EU	EV	EW	EX	EY	EZ
FA	FB	FC	FD	FE	FF	FG	FH	FI	FJ	FK	FL	FM	FN	FO	FP	FQ	FR	FS	FT	FU	FV	FW	FX	FY	FZ
GA	GB	GC	GD	GE	GF	GG	GH	GI	GJ	GK	GL	GM	GN	GO	GP	GQ	GR	GS	GT	GU	GV	GW	GX	GY	GZ
HA	HB	HC	HD	HE	HF	HG	HH	HI	HJ	HK	HL	HM	HN	HO	HP	HQ	HR	HS	HT	HU	HV	HW	HX	HY	HZ
IA	IB	IC	ID	IE	IF	IG	IH	II	IJ	IK	IL	IM	IN	IO	IP	IQ	IR	IS	IT	IU	IV	IW	IX	IY	IZ
JA	JB	JC	JD	JE	JF	JG	JH	JI	JJ	JK	JL	JM	JN	JO	JP	JQ	JR	JS	JT	JU	JV	JW	JX	JY	JZ
KA	KB	KC	KD	KE	KF	KG	KH	KI	KJ	KK	KL	KM	KN	KO	KP	KQ	KR	KS	KT	KU	KV	KW	KX	KY	KZ
LA	LB	LC	LD	LE	LF	LG	LH	LI	LJ	LK	LL	LM	LN	LO	LP	LQ	LR	LS	LT	LU	LV	LW	LX	LY	LZ
MA	MB	MC	MD	ME	MF	MG	MH	MI	MJ	MK	ML	MM	MN	MO	MP	MQ	MR	MS	MT	MU	MV	MW	MX	MY	MZ
NA	NB	NC	ND	NE	NF	NG	NH	NI	NJ	NK	NL	NM	NN	NO	NP	NQ	NR	NS	NT	NU	NV	NW	NX	NY	NZ
OA	OB	OC	OD	OE	OF	OG	OH	OI	OJ	OK	OL	OM	ON	OO	OP	OQ	OR	OS	OT	OU	OV	OW	OX	OY	OZ

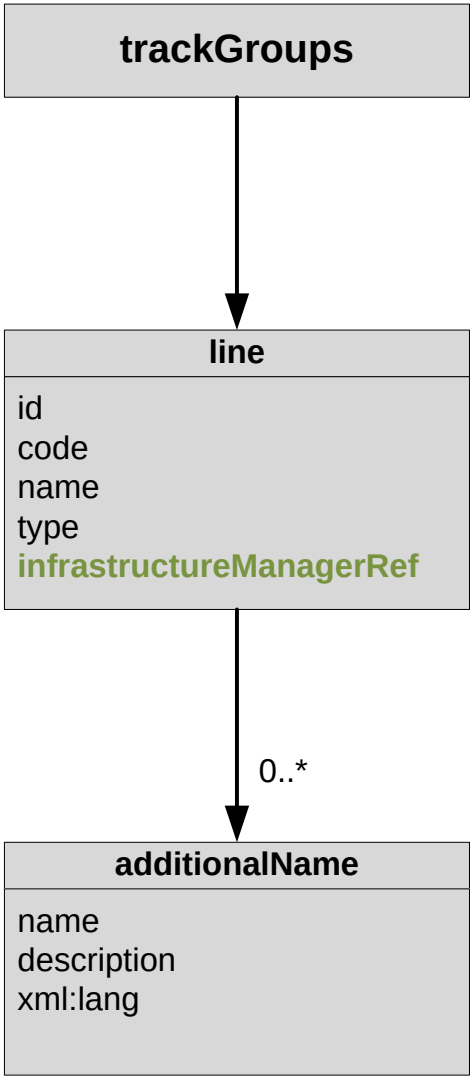
http://en.wikipedia.org/wiki/ISO_3166-1_alpha-2



Different Line codes in railML® 2.2

- ***code***: defining the number of the line (without the UIC Country Code)
- ***infrastructureManagerRef***: a string attribute referencing the code of an infrastructure manager

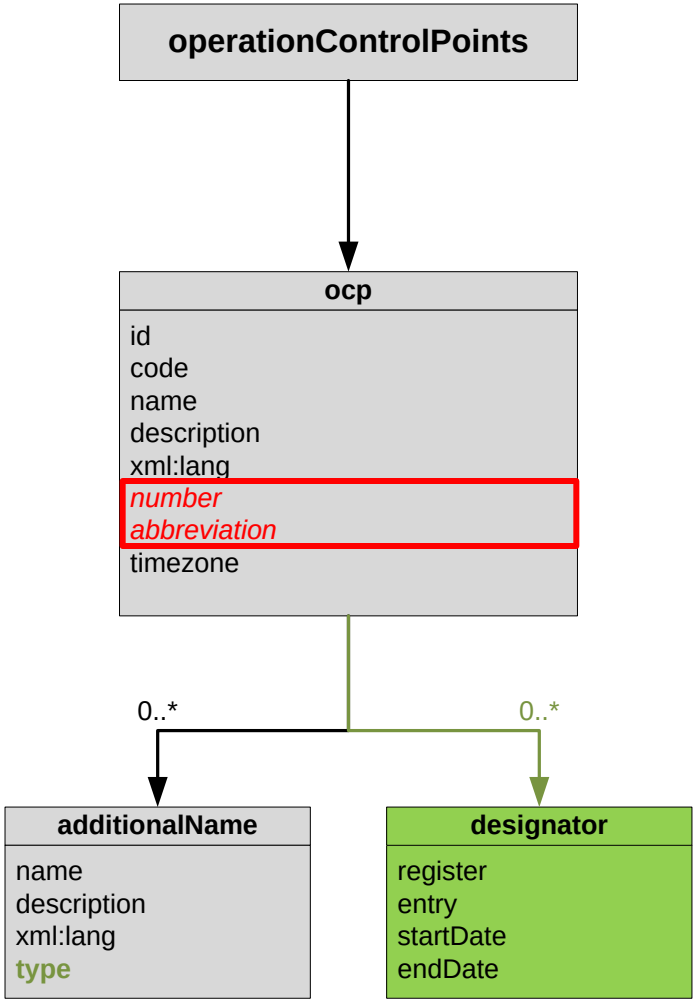
- More details: Trac Ticket #152
  https://trac.assembla.com/railML/ticket/152



OPERATION CONTROL POINTS

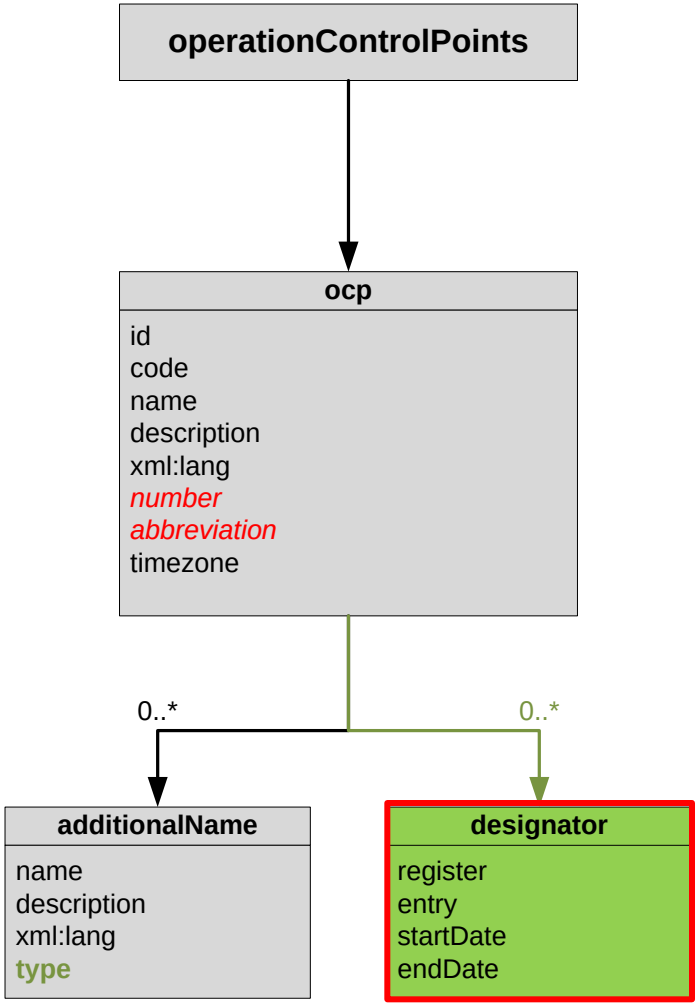
Different OCP codes in railML® 2.2

- The parameters ***number*** and ***abbreviation*** are marked as deprecated since railML 2.1



Different OCP codes in railML® 2.2

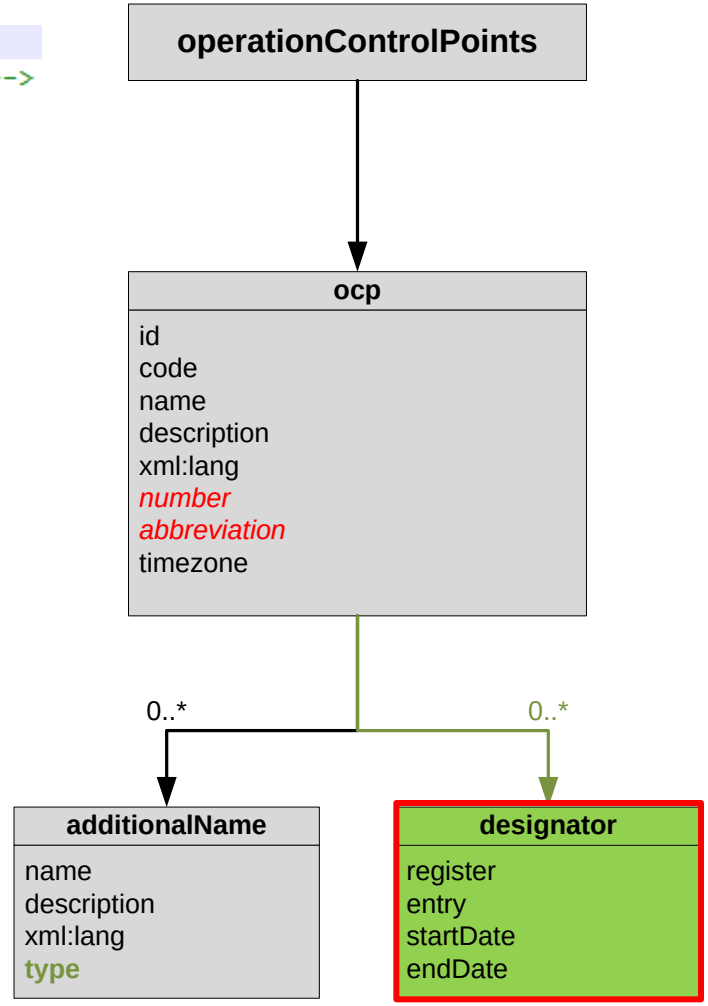
- using the new element **designator** it will be possible to address "local" register codes for the same OCP
- Parameters:
  - ***Register*** defines a (local) register for an OCP's name/code
  - ***Entry*** contains the OCP's name/code in the certain register
  - ***startDate*** defines the first day of validity for the register
  - ***endDate*** defines the last day of validity for the register
- Example: `register='IBNR' entry='8509404'`



Different OCP codes in railML® 2.2

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!-- $Id: Registers.xml 550 2013-02-18 15:47:52Z susanne.wunsch $ -->
<registers xmlns="http://www.railml.org/lists">
  <register code="DB640">
    <name>Dienstbehelf Nr. 640 (ÖBB)</name>
  </register>
  <register code="DIDOK">
    <name>Dienststellendokumentation (Schweiz)</name>
  </register>
  <register code="ENEE">
    <name>ENEE Database Station Code (UIC)</name>
  </register>
  <register code="IBNR">
    <name>IBNR (Europäisches Fahrplanzentrum)</name>
  </register>
  <register code="RL100">
    <name>Betriebsstellenkürzel (DB)</name>
    <remarks>was DS100 or DV100 or Ril100</remarks>
  </register>
</registers>
```

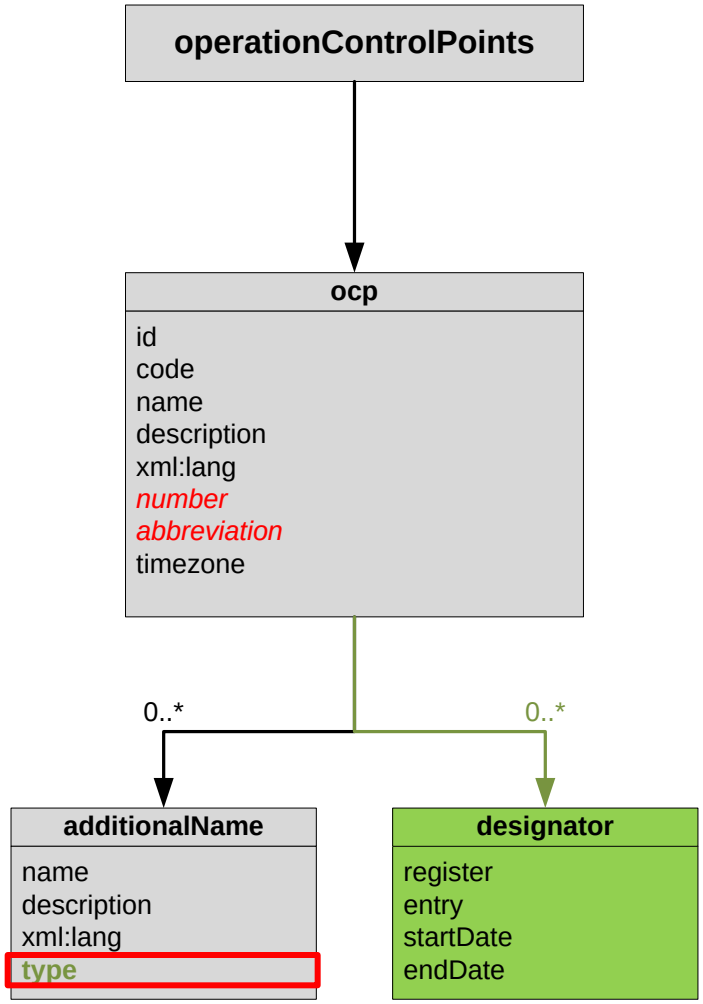
- The register entries are stored in a separate XML file **Registers.xml** that is loosely coupled with the schema



Different OCP codes in railML® 2.2

- For further specifying an OCP's **additionalName** the new parameter ***type*** provides the values
  - operationalName
  - trafficName
  - localName

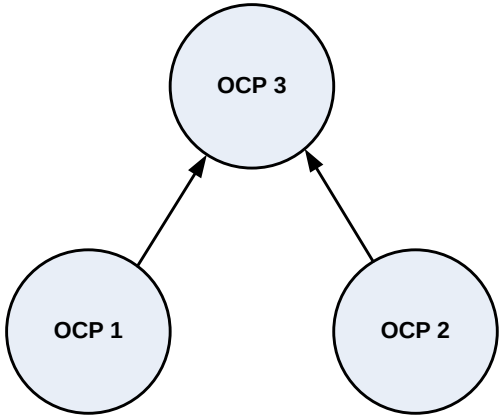
- More details: Trac Ticket #112
  https://trac.assembla.com/railML/ticket/112

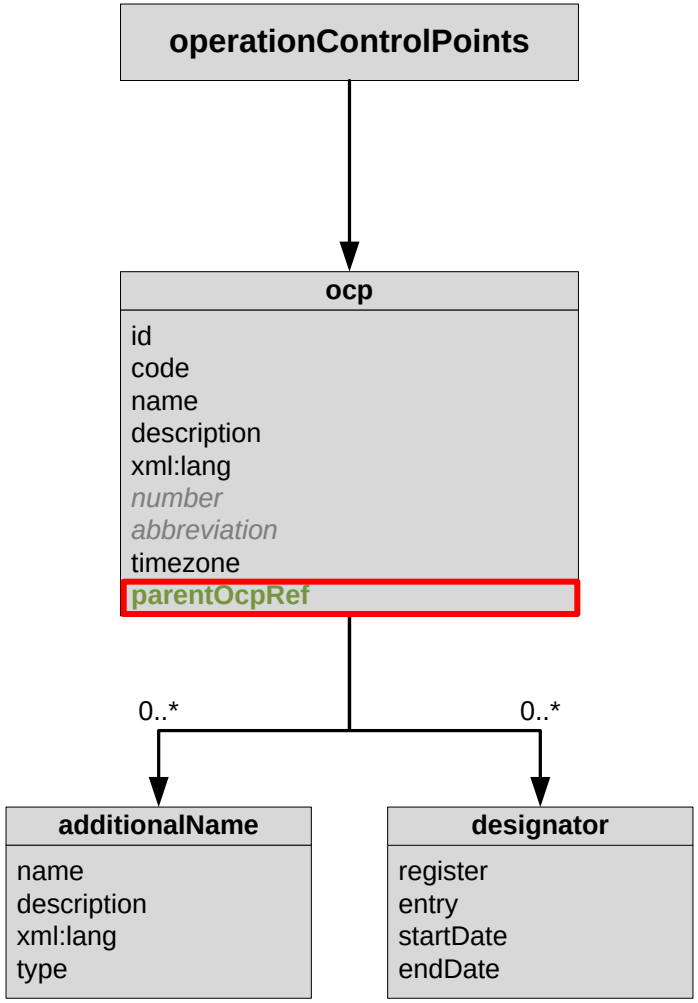


Grouping of OCPs in railML® 2.2

- The new optional parameter ***parentOcpRef*** references the one and only parent ocp of this ocp

- More details: Trac Ticket #153
  https://trac.assembla.com/railML/ticket/153

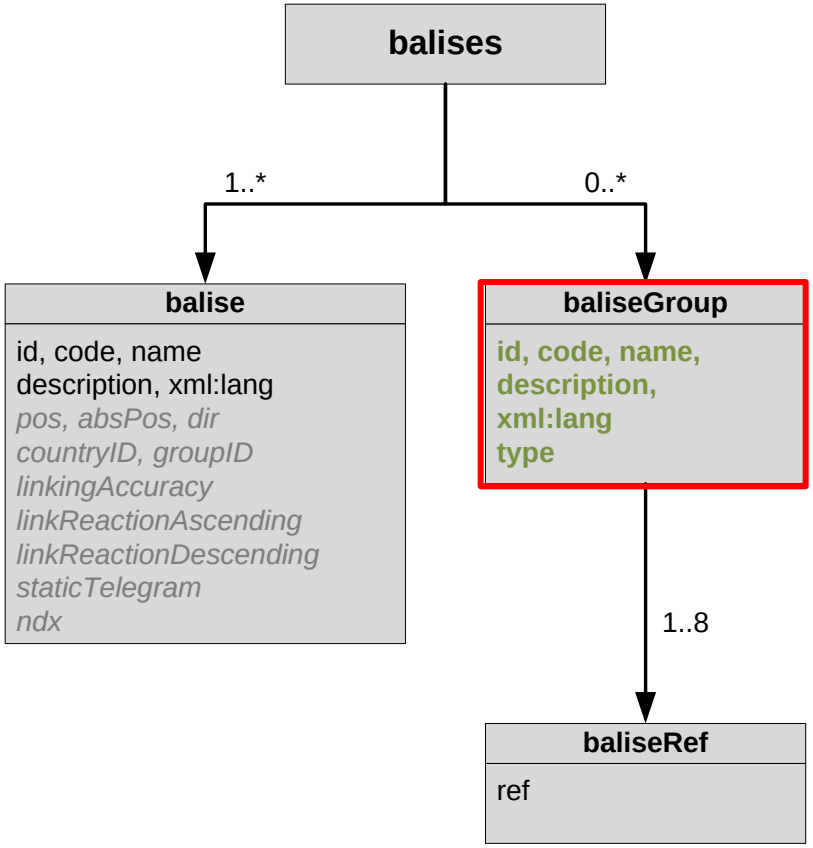




BALISES & TRAINPROTECTION

BaliseGroups in railML® 2.2

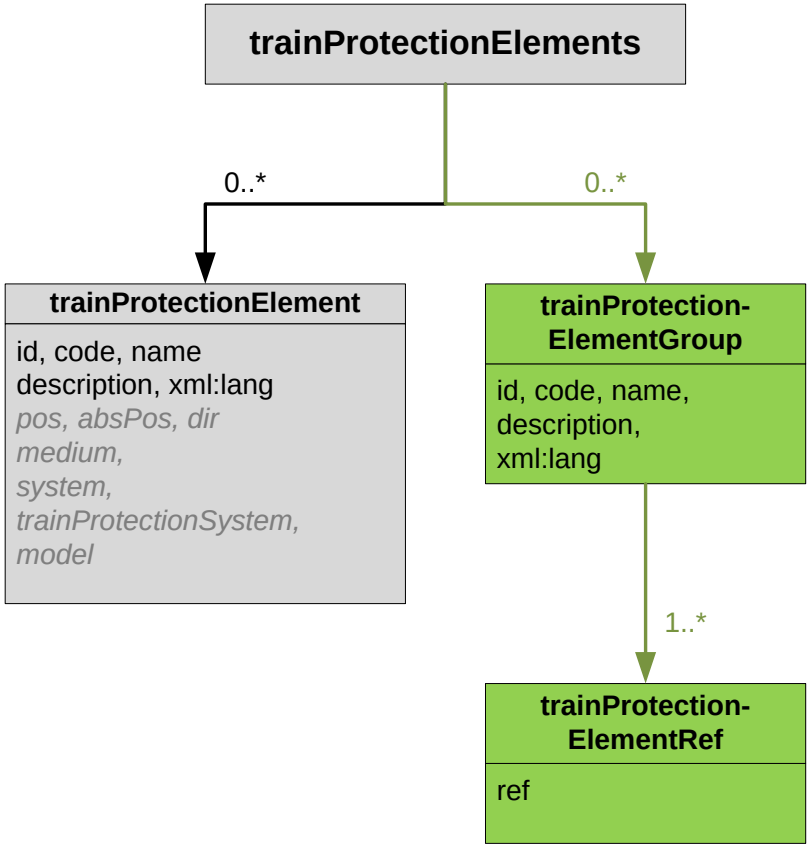
- **baliseGroup** becomes an identifiable object
- Parameters:
  - ***Identification (id, name, code, description, xml:lang)***
  - ***type*** {infill, signal, fixed}
- The balise and the balise group can be referenced from a signal
- The direct grouping of balises within **baliseGroup** is suggested for railML® 3.0



- More details: Trac Ticket #174
  https://trac.assembla.com/railML/ticket/174

TrainProtectionElementGroups in railML® 2.2

- **trainProtectionElementGroup** is a new identifiable object
- Parameters:
  - ***Identification (id, name, code, description, xml:lang)***
- The trainProtectionElement and the trainProtectionElementGroup can be referenced from a signal

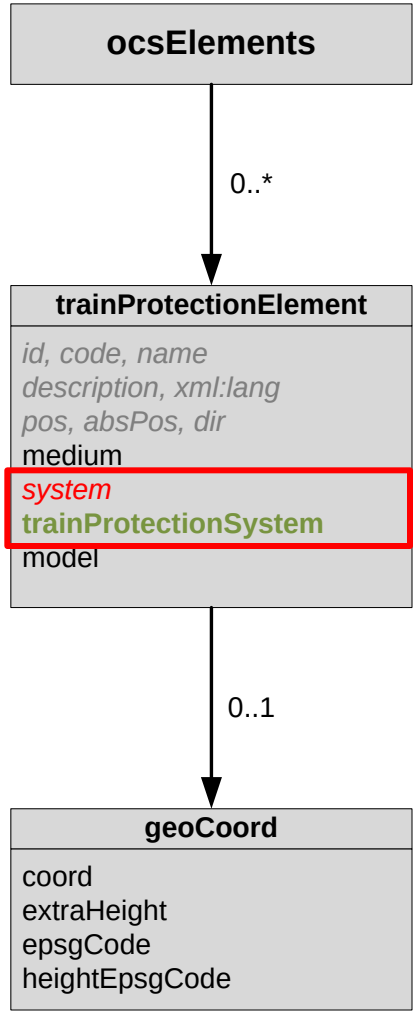


- More details: Trac Ticket #199
  https://trac.assembla.com/railML/ticket/199

TrainProtectionElements in railML® 2.2

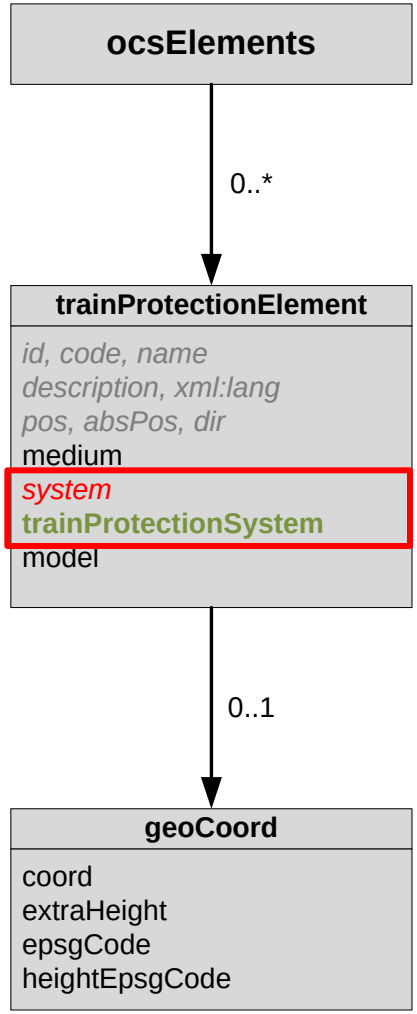
- The string parameter ***system*** is marked as deprecated
- Use the new string parameter ***trainProtectionSystem*** instead
- The possible entries are stored in the list trainProtectionSystemsAtTrack in a separate XML file **TrainProtectionSystems.xml** that is loosely coupled with the schema

- More details: Trac Ticket #175
  https://trac.assembla.com/railML/ticket/175



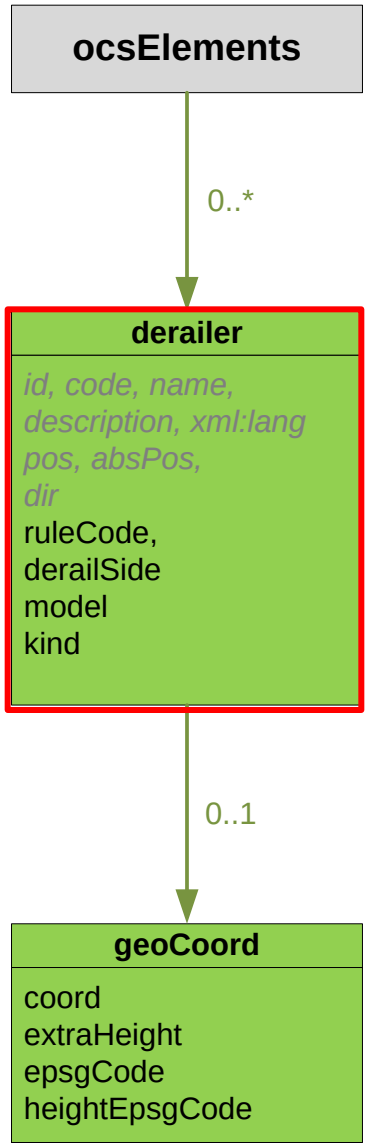
TrainProtectionElements in railML® 2.2

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!-- $Id: TrainProtectionSystems.xml 550 2013-02-18 15:47:52Z susanne.wunsch $ -->
<trainProtectionSystems xmlns="http://www.railml.org/lists">
  <trainProtectionSystemsAtTrack>
    <trainProtectionSystem code="ALSN">
    <trainProtectionSystem code="ASFA">
    <trainProtectionSystem code="ATB">
    <trainProtectionSystem code="ATBNG">
    <!-- This name is used for various systems with heterogenous devices
    <trainProtectionSystem code="ATSP">
    <trainProtectionSystem code="ATSS">
    <trainProtectionSystem code="AWS">
    <trainProtectionSystem code="BACC">
    <trainProtectionSystem code="Crocodile">
    <trainProtectionSystem code="CIR-ELKE">
      <name />
      <validFor />
    </trainProtectionSystem>
    <trainProtectionSystem code="CIR-ELKE2">
    <trainProtectionSystem code="CSS">
    <trainProtectionSystem code="DATC">
    <trainProtectionSystem code="EBICAB">
    <trainProtectionSystem code="ETCS">
    <trainProtectionSystem code="EVM">
    <trainProtectionSystem code="Fahrsp">
    <trainProtectionSystem code="GWATP">
```



Derailers in railML® 2.2

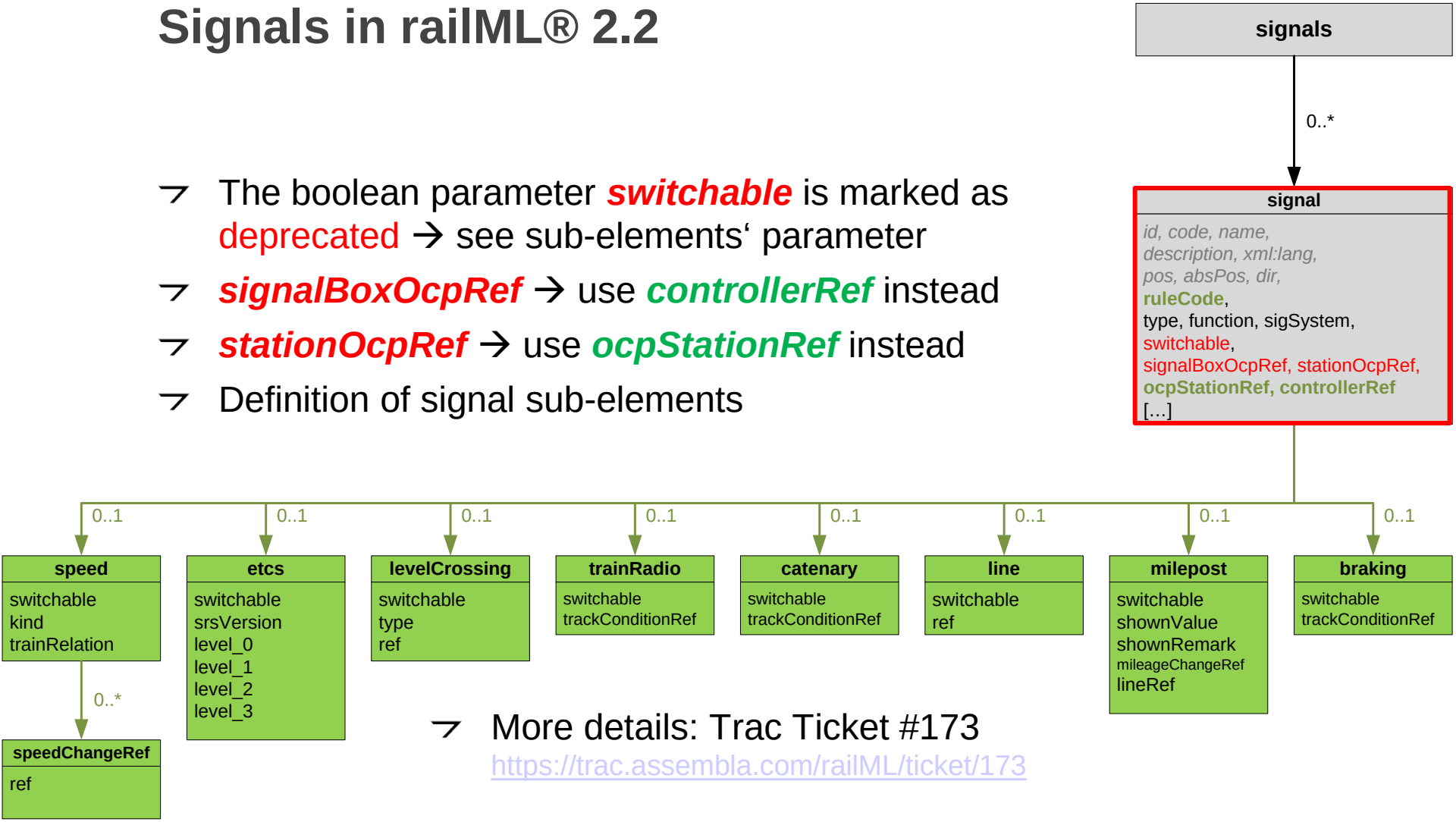
- **derailer** is a new ocs element
- Parameters:
  - ***Identification and positional data***
  - ***ruleCode*** {string} defines the country-specific sign code for the signalling element, e.g. ???
  - ***derailSide*** {left, right} specifies the side, to which the vehicle will be derailed
  - ***model*** {string} can be used for further specification of the derail device (e.g. producer)
  - ***kind*** {blockDerail, singleCatchPoints, doubleCatchPoints} specifies the derail device
- More details: Trac Ticket #232
  https://trac.assembla.com/railML/ticket/232



SIGNALS

Signals in railML® 2.2

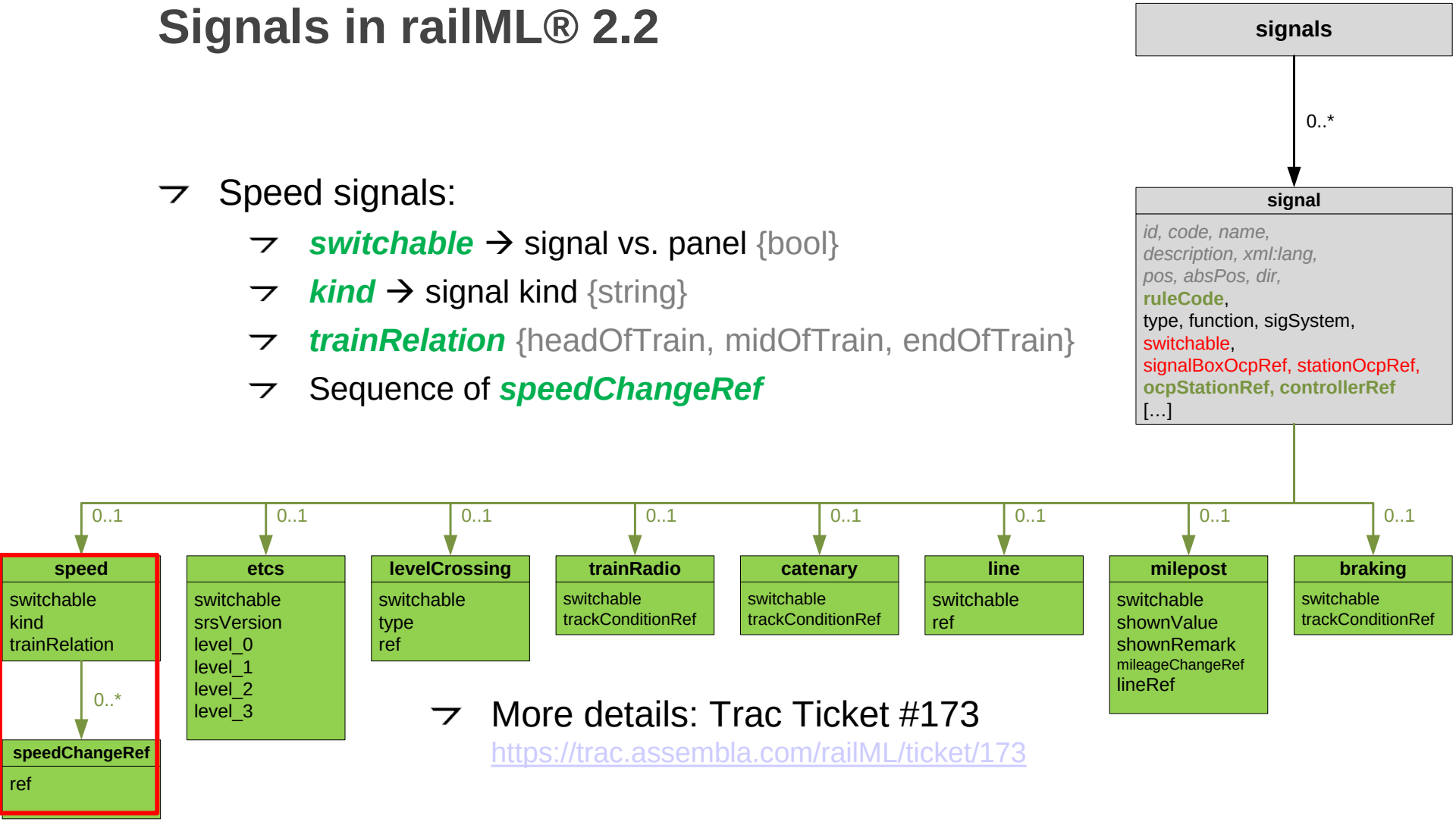
- The boolean parameter ***switchable*** is marked as deprecated → see sub-elements‘ parameter
- ***signalBoxOcpRef*** → use ***controllerRef*** instead
- ***stationOcpRef*** → use ***ocpStationRef*** instead
- Definition of signal sub-elements



- More details: Trac Ticket #173
  https://trac.assembla.com/railML/ticket/173

Signals in railML® 2.2

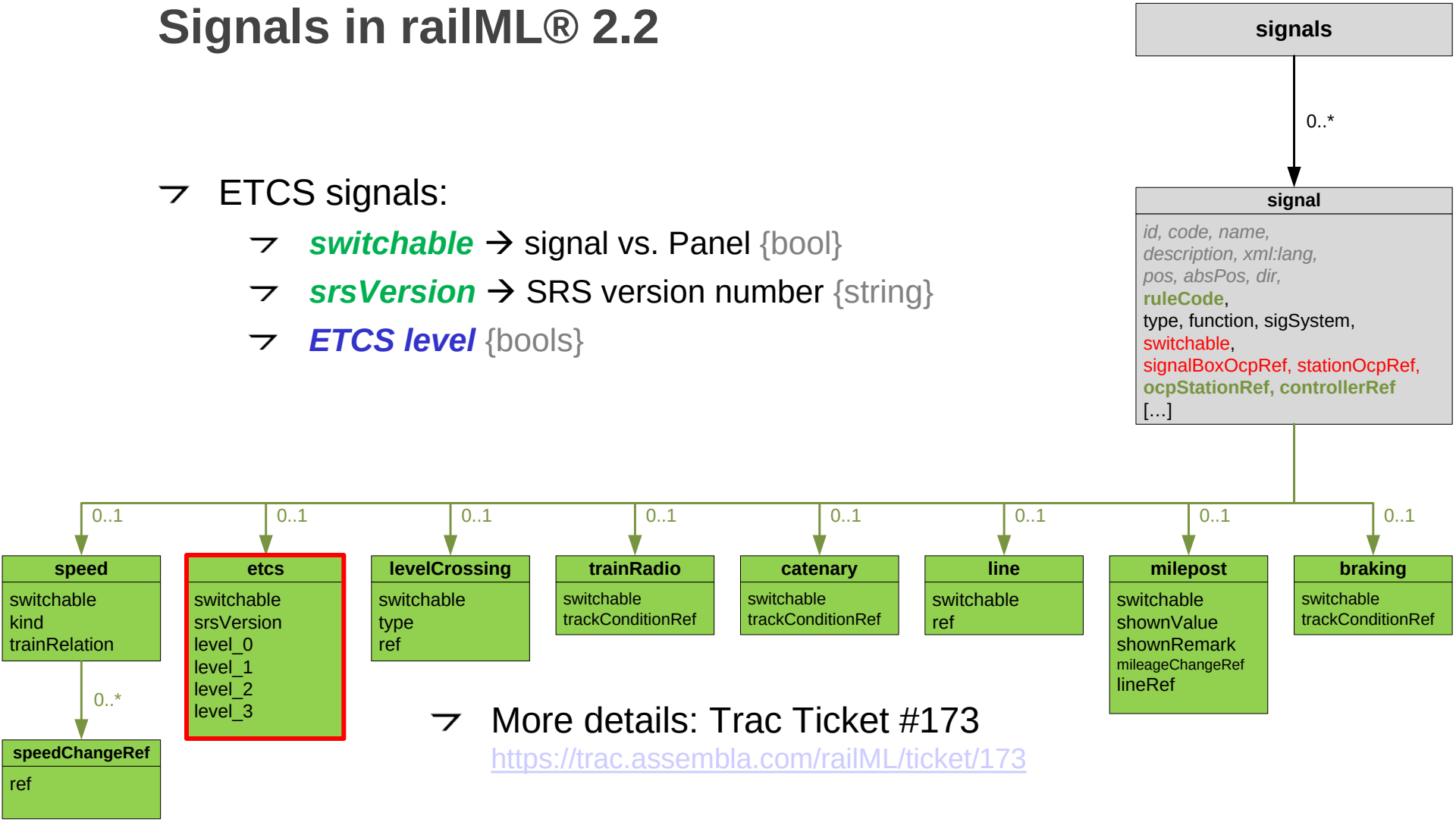
- Speed signals:
  - ***switchable*** → signal vs. panel {bool}
  - ***kind*** → signal kind {string}
  - ***trainRelation*** {headOfTrain, midOfTrain, endOfTrain}
  - Sequence of ***speedChangeRef***



- More details: Trac Ticket #173
  https://trac.assembla.com/railML/ticket/173

Signals in railML® 2.2

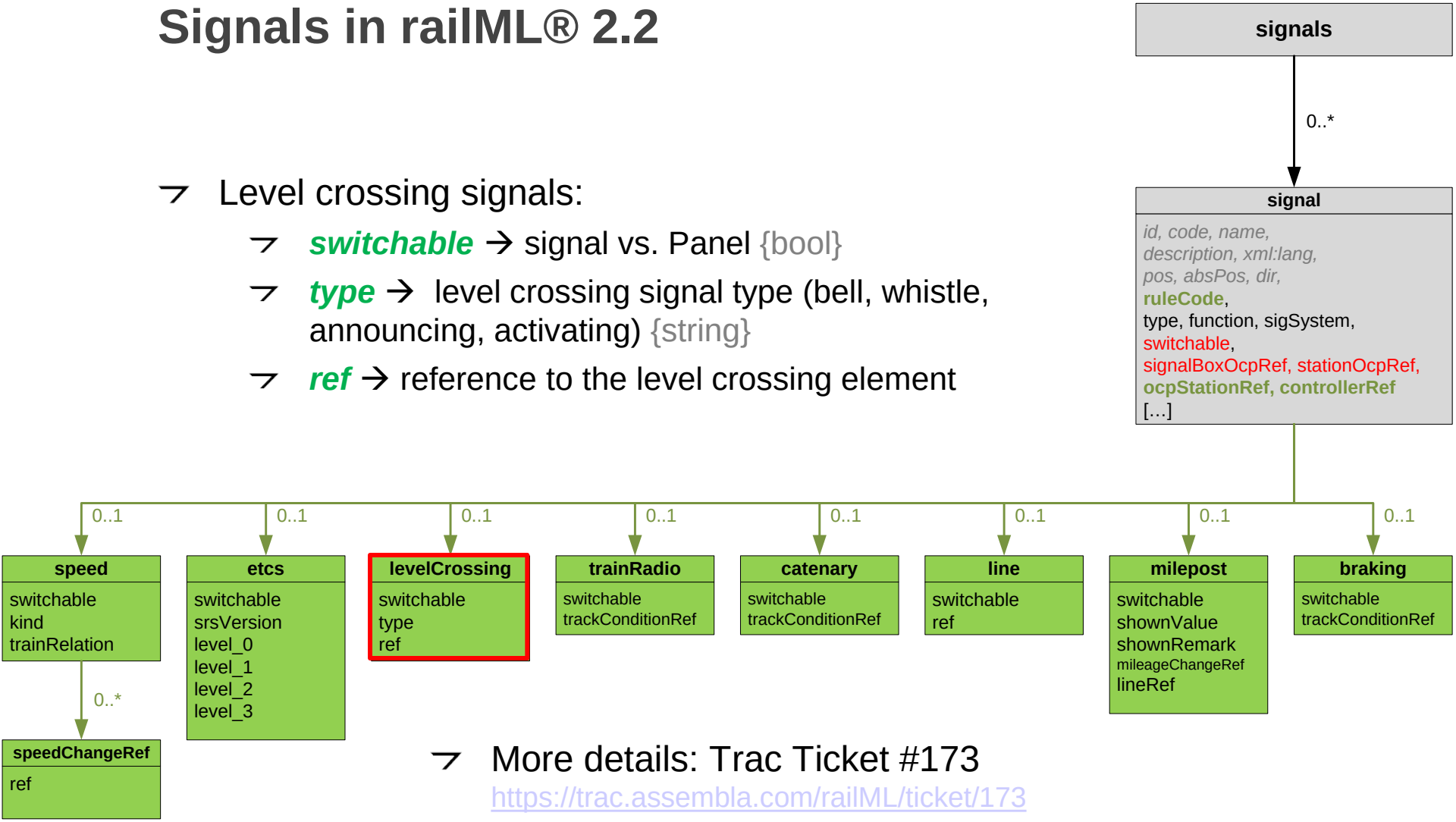
- ETCS signals:
  - ***switchable*** → signal vs. Panel {bool}
  - ***srsVersion*** → SRS version number {string}
  - ***ETCS level*** {bools}



- More details: Trac Ticket #173
  https://trac.assembla.com/railML/ticket/173

Signals in railML® 2.2

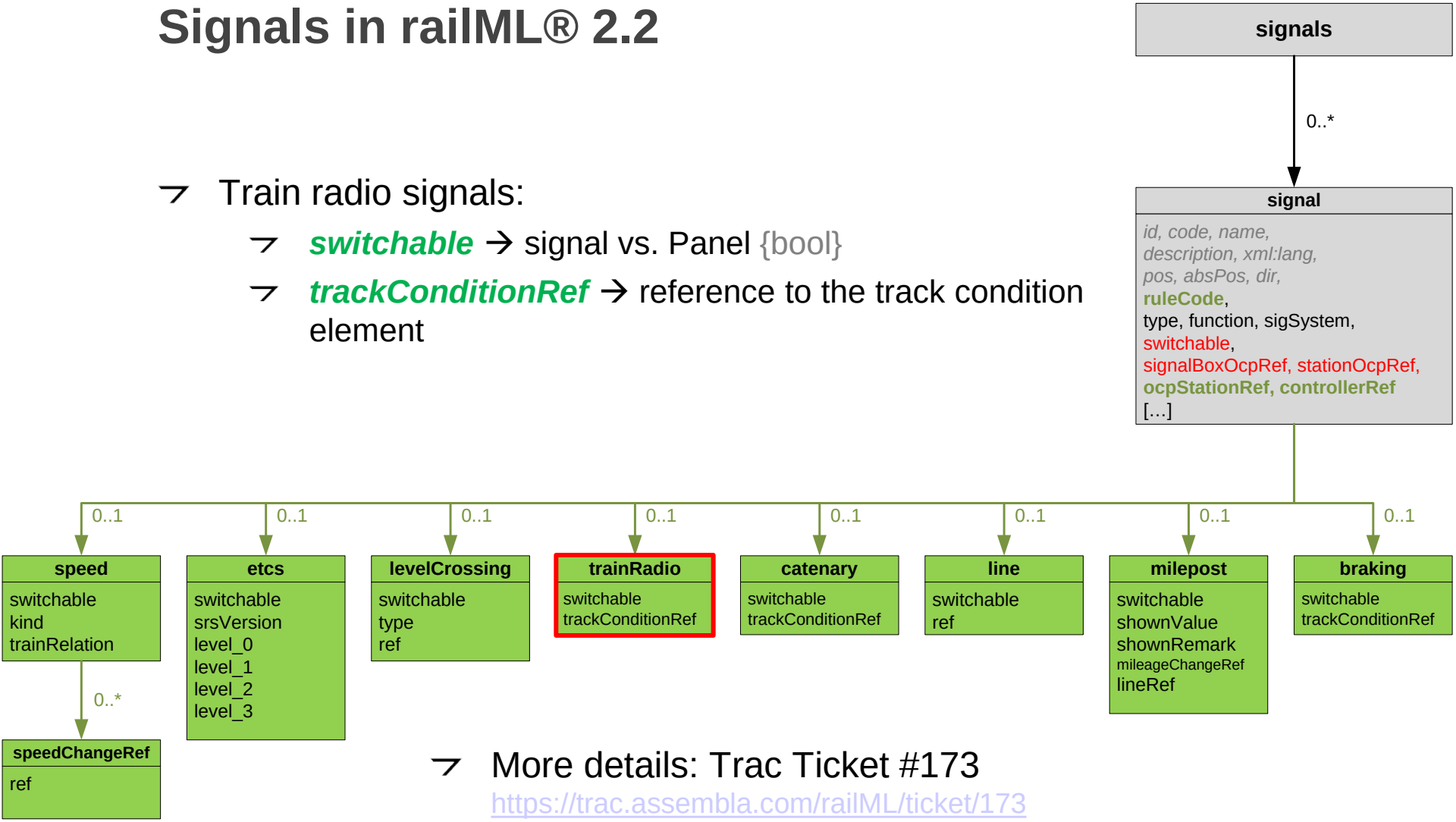
- Level crossing signals:
  - ***switchable*** → signal vs. Panel {bool}
  - ***type*** → level crossing signal type (bell, whistle, announcing, activating) {string}
  - ***ref*** → reference to the level crossing element



- More details: Trac Ticket #173
  https://trac.assembla.com/railML/ticket/173

Signals in railML® 2.2

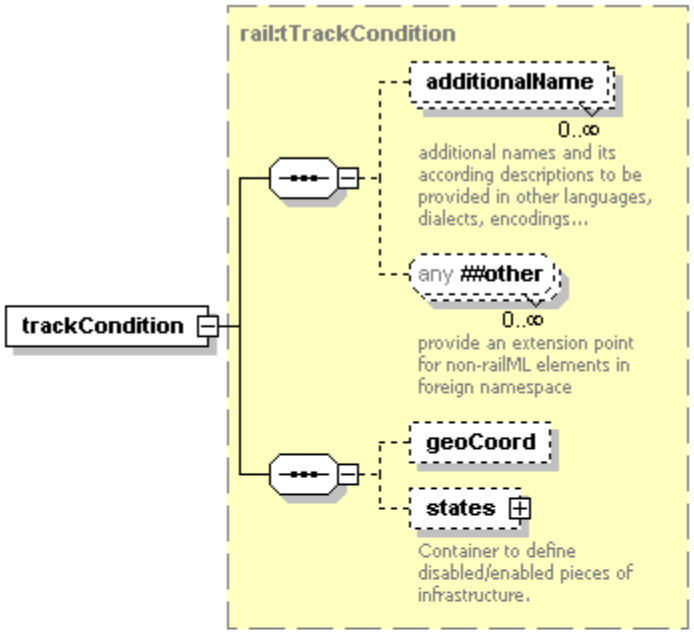
- Train radio signals:
  - ***switchable*** → signal vs. Panel {bool}
  - ***trackConditionRef*** → reference to the track condition element



- More details: Trac Ticket #173
  https://trac.assembla.com/railML/ticket/173

element **rail:eTrackConditions/trackCondition**

diagram



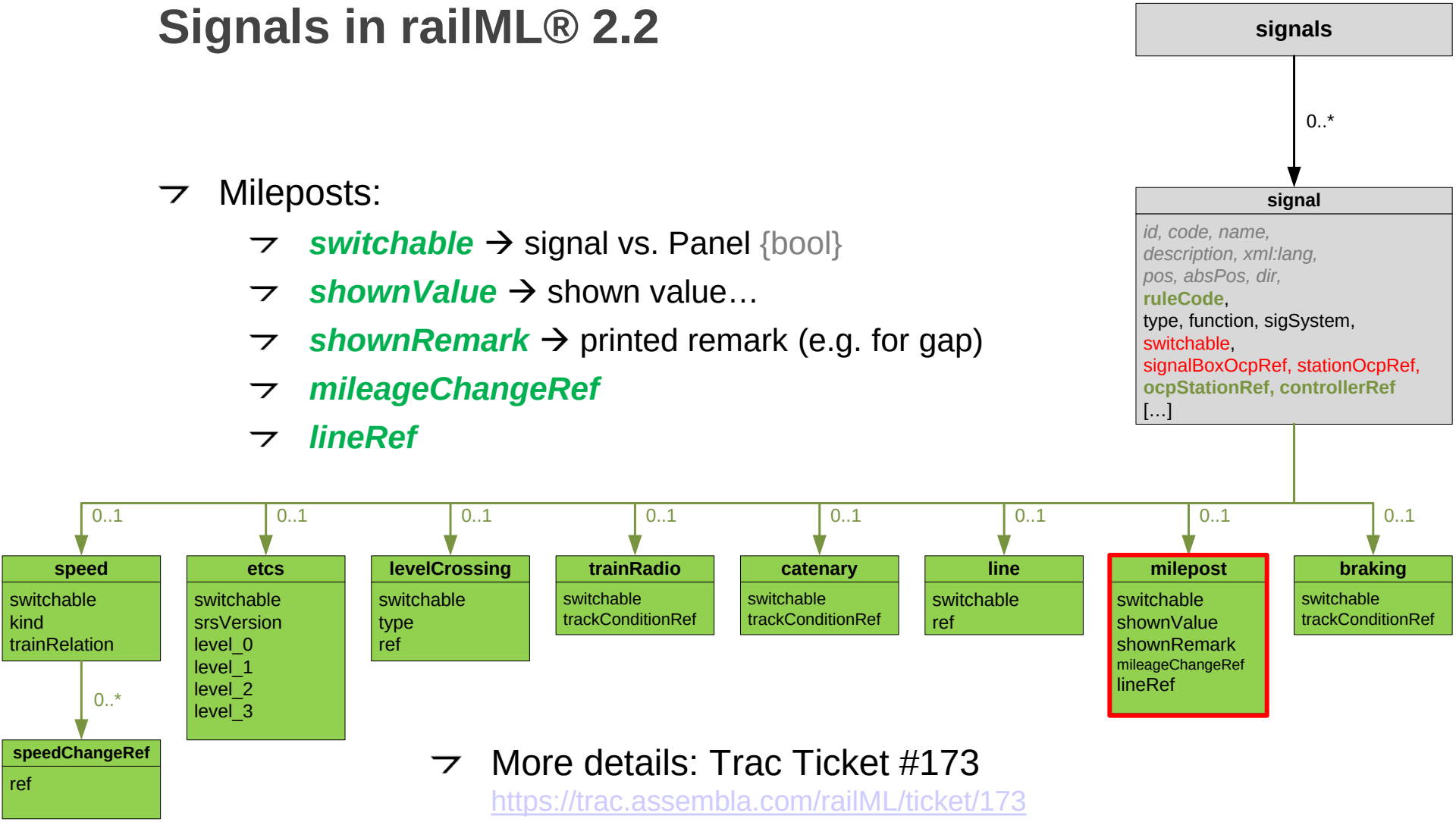
namespace	http://www.railml.org/schemas/2013
type	rail:tTrackCondition
children	additionalName geoCoord states

attributes

Name	Type	Use	Default	Fixed	Annotation
id	rail:tGenericID	required			documentation
code	rail:tGenericName				documentation
name	rail:tGenericName				documentation
description	rail:tElementDescription				documentation
xml:lang	xs:language				documentation
pos	rail:tLengthM	required			
absPos	rail:tLengthM				
absPosOffset	rail:tLengthM	optional			documentation
dir	rail:tLaxDirection				
length	rail:tLengthM				
type	rail:tTrackConditionType				

Signals in railML® 2.2

- Mileposts:
  - ***switchable*** → signal vs. Panel {bool}
  - ***shownValue*** → shown value…
  - ***shownRemark*** → printed remark (e.g. for gap)
  - ***mileageChangeRef***
  - ***lineRef***



- More details: Trac Ticket #173
  https://trac.assembla.com/railML/ticket/173

STATES

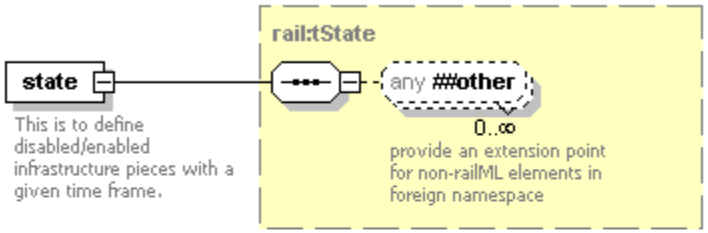
States of infrastructure in railML® 2.2

- States can be used to model blocked and disabled infrastructure
- Blocking: **tracks**, **lines**, **locallyControlledAreas**
- Disabled: **controller, switch, crossing, signal, balise, tunnel, bridge, levelCrossing, platformEdge, serviceSection, trainProtectionElement, trainDetector, trackCircuitBorder, trackCondition, derailer**

- More details: Trac Ticket #156 https://trac.assembla.com/railML/ticket/156

States in railML® 2.2

element **rail:tStates/state**

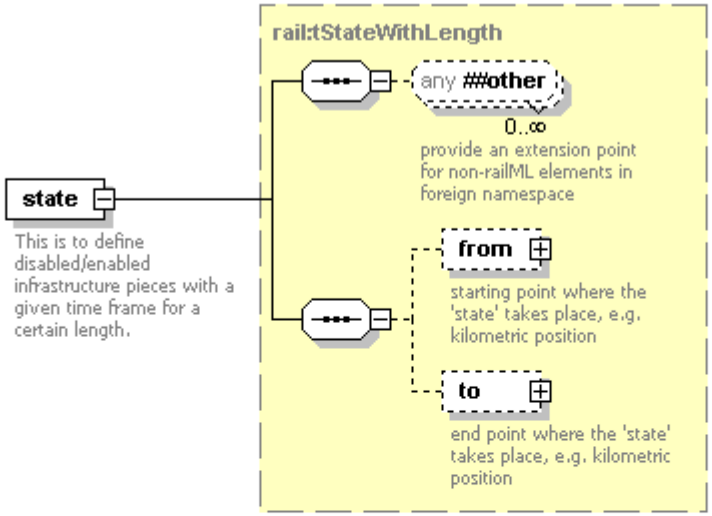
diagram	
namespace	http://www.railml.org/schemas/2013
type	rail:tState
attributes	Name / Type / Use / Default / Fixed / Annotation disabled / xs:boolean / required / / / documentation operatingPeriodRef / rail:tGenericRef / / / / documentation remarks / rail:tElementDescription / / / / documentation
annotation	documentation This is to define disabled/enabled infrastructure pieces with a given time frame.

- More details: Trac Ticket #156 https://trac.assembla.com/railML/ticket/156

States in railML® 2.2

element **rail:tStatesWithLength/state**

diagram



→ **track**

namespace	http://www.railml.org/schemas/2013
type	rail:tStateWithLength
children	from to

attributes

Name	Type	Use	Default	Fixed	Annotation
disabled	xs:boolean	required			documentation
operatingPeriodRef	rail:tGenericRef				documentation
remarks	rail:tElementDescription				documentation

annotation: documentation This is to define disabled/enabled infrastructure pieces with a given time frame for a certain length.

railML timetable common
rollingstock
infrastructure

Thank you for your attention!

Christian Rahmig
coord@infrastructure.railML.org