Subject: the use of @dir in railML.
Posted by Torben Brand on Wed, 12 Sep 2018 12:57:50 GMT
View Forum Message <> Reply to Message

I would like to address the use of @dir in railML.
The direction of the objects on the track is always given in railML through the track direction (from the <trackBegin> to the <trackEnd> or seen in increasing relative position values)
@dir denotes the validity of the objects as seen from the direction of travel by the train. Or as it says similar in the gradientChange wiki: "dir: This defines the validity of gradientChange along the track."
Suggestion for a more clear term in railML3 that creates less confusion would be @validDir (or something similar).

Direction restrictions come in three pre-set levels in railML:
"lax" with the enumeration values "up","down","unknown","none","both"
"delimited" with the enumeration values "up","down","unknown"
"strict" with the enumeration values "up" or "down"

The values are defined to (taken from wiki for gradientChange):

Possible values are:
• none: gradientChange has no direction restriction.
• up: This denotes the direction from the <trackBegin> to the <trackEnd> (increasing relative position values).
• down: This goes opposite to up (decreasing relative position values).
• both: gradientChange is valid in both directions.
• unknown: gradientChange is restricted to a certain direction, but this direction is not known.

First having the value "none" and "both" make no sense. This as they both cover the same thing (glass is half full or half empty)

The wiki pages describing @dir for gradientChange, trackCircuitBorder and trainDetector (possible other pages that use @dir, I have not checked) are inconsistent.

The possible values listed are the 5 lax values.
Under constraint it is referenced to: "dir: xs:string, generic type for more constrained direction statements: enumeration up, down, unknown; derived from tLaxDirection; optional "
tLaxDirection are the 5 listed values, but the values listed in constraint are the tDelimitedDirection values.
In the XSD the tStrictDirection values are used for gradientChange and tDelimitedDirection values are used for trainDetector and trackCircuitBorder. So only the values "up" and "down" (and "unknown") are allowed under @dir.
As the XSD is the master I suggest to edit the wiki pages to the following:

Possible values are:
• up: This denotes the direction from the <trackBegin> to the <trackEnd> (increasing relative position values).

• down: This goes opposite to up (decreasing relative position values).

Setting no @dir attribute defines that the object has no direction restraint. This corresponds to the lax values "none" or "both". Modelling of "unknown" is currently not possible for @dir in railML2.

Constraint

dir: xs:string, generic type for more constrained direction statements: enumeration up, down; derived from tStrictDirection; optional

gradientChange: https://wiki.railml.org/index.php?title=IS:gradientChange
trackCircuitBorder: https://wiki.railml.org/index.php?title=IS:trackCircuitBorde r
trainDetector: https://wiki.railml.org/index.php?title=IS:trainDetector