Subject: Re: [railML3.1] applicationDirection and placing of elements
Posted by christian.rahmig on Fri, 24 Jul 2020 12:23:12 GMT
View Forum Message <> Reply to Message

Dear Peter,

Peter Vancsa wrote on Mon, 13 July 2020 18:40
...
Since the NetElement "ne_b05" is not liked to the linearPositioningSystem "lps01", I should use intrinsic coordinates. However, intrinsic coordinates are linked to positions, so an intrinsic coordinate of 0.5 would directly correlate to a position of half of the length of the net element. With this knowledge, 'pos' always correlates to an exact 'intrinsicCoord', right?

Also, since i know the linearPositioningSystem through one of the netRelations ("nr_b02b05" or "nr_b04b05") of this netElement at intrinsic coordinate 1, i can compute the other one (intrinsic coordinate 0) with the edge direction and the length.
I am not sure what advantages 'intrinsicCoord' have over 'pos' since they always have to correlate to the edge length.

Is there a rule to use 'pos' when the net element is linked to a linearPositioningSystem and 'intrinsicCoord' when it's not?

From railML schema perspective there is no rule when to use @pos and when to use @intrinsicCoord. The only rule is that the location of the element should be unambiguously defined. You are right: the intrinsic coordinate (0..1) can be calculated from a pos value knowing the total length of the NetElement. It is also possible to calculate the pos value from an intrinsic coordinate. There may be different use cases requiring different implementations of locations. So, the idea is: we define the "leading information" (pos value or intrinsic coordinate) in the use case and since the other information can be always calculated from the leading one, there is neither a loss of data nor conflicting values.

However, it is a good questions for the community: do you use (relative) position values or intrinsic coordinates as leading information in your systems and export interfaces?

Peter Vancsa wrote on Mon, 13 July 2020 18:40
Considering that the switchIS id="swi03" ("69W04") should have the spotLocation with applicationDirection="reverse", should the continueCourse for this switch not also have be defined as "left"? In the visualization (pdf) of this sample, it looks like the continueCourse is the branchCourse. I am not sure if the visualization is not correct here, or if the definition in the railml file is not precise.
See the attachment "continueCourse.jpg"

You are right considering a "geometric" perspective: The switch's "straight branch" ends in track 21, the "branching track" leads to switch 69W03. If we define "trackContinueCourse" on an operational basis (the branch with the highest traffic / operational main track), selecting the right

branch is correct. Following this approach, the "branchCourse" is the straight track leading to the track 21.

This is again a question to the active community: How shall be @trackContinueCourse and @branchCourse be interpreted?
a) operational view: trackContinueCourse = track which is mainly used by railway traffic
b) geometric view: trackContinueCourse = track with the higher radius or which is straight

How did and do you use these attributes in railML 2.x in your exports? What is best practice? We need your feedback to improve our documentation in wiki (https://wiki2.railml.org/index.php?title=IS:switch).

Peter Vancsa wrote on Mon, 13 July 2020 18:40
And lastly,
The trainDetectionElement id="tde13" has a spotLocation withouh a 'pos' or 'intrinsicCoord' defined, it has however a linearCoordinate with a 'measure'. Is that a valid definition for a spotLocation? Or should I in such a case compute the 'pos' or 'intrinsicCoord' from the linearCoordinate's 'measure'?
...

Yes, the location of "tde13" is correctly defined. The attributes @pos and @intrinsicCoordinate can be directly calculated from the <linearCoordinate>@measure. As mentioned before, there is no rule when to use which notation, but it must be guaranteed that the other values can be calculated from the given one.

If you (and the rest of the community) have different opinions or further questions regarding this topic, please let's continue the discussion. We need solutions that you can practically work with...

Best regards
Christian