Subject: Re: the use of @dir in railML.
Posted by Thomas Nygreen JBD on Mon, 08 Oct 2018 13:14:03 GMT
View Forum Message <> Reply to Message

I would like to look a bit broader at the use and interpretation of the @dir attribute. As Torben points out, the documentation is lacking, and in some cases incorrect. Currently this attribute is not documented in the wiki or XSD at all, apart from the quite uninformative "This defines the validity of <element> along the track" and the restrictions described in more technical terms. The description that Torben mentions says "derived from tLaxDirection" because it is a description of the (badly named) tDelimitedDirection, and this type is indeed derived from tLaxDirection.

I agree that "none" and "both" are two sides to the same coin, and one of them can be removed. Regarding "unknown", there is according to the documentation a subtle difference between @dir="unknown" and a missing @dir: with "unknown" the element "is restricted to a certain direction, but this direction is not known", while with a missing @dir we do not know if there is any restriction at all (or in most cases we assume that there is not). However, I have yet not seen a real use case, and unless anyone else has, it can maybe also be removed.

Suggested change:
* Remove "unknown" and "none" from enumeration

There are 26 elements in the railML 2.3/2.4 infrastructure schema that support the dir attribute. Additionally there are two elements where this attribute is deprecated (crossSection and mileageChange), which I will disregard below. The elements can be divided into three categories: elements without extent, elements with extent and elements that describe a change that occurs at a given point.

The first group, elements without extent (i.e. without a @length) include: balise, border, derailer, signal, stopPost, trackCircuitBorder, trainDetector and trainProtectionElement. The interpretation of @dir for most of the physical elements in this group is quite simple: they apply only to trains in that direction. For trackCircuitBorder, however, a rail joint cannot be isolated in only one direction. For the more abstract border element, I am not quite sure about the purpose of @dir. The only one I can think of is that there could be borders that are placed differently in the two directions.

Suggested changes:
* DEPRECATE @dir for border and trackCircuitBorder
* Remove "both" from enumeration for derailer, signal and stopPost unless anyone has an example

The second group, the elements with an extent (i.e. with @length) include: brigde, levelCrossing, platformEdge, serviceSection, trackCondition and tunnel. This group is less simple. Using the same logic and definition as above would imply that a tunnel, bridge etc. with @dir="up" is "not there" if you drive in the "down" direction, and can be ignored by those trains. If tunnels that are invisible and without air resistance in one direction actually exist, please inform me! The only element in this group that can be interpreted in this fashion is trackCondition. For the remaining elements the intention may have been to specify that it is only allowed to drive through/along the element in one direction, but this would mean that @dir must be interpreted very differently for these elements. Furthermore, such a restriction would not be a property of this element, but of the track it is placed on or the signalling and interlocking system. Consequently I suggest to deprecate @dir for all these elements, except trackCondition.

Suggested changes:
* DEPRECATE @dir for brigde, levelCrossing, platformEdge, serviceSection (?) and tunnel

The last group, elements that describe a change, include: axleWeightChange, clearanceGaugeChange, electrificationChange, gaugeChange, gradientChange, operationModeChange, ownerChange, powerTransmissionChange, radiusChange, speedChange, trainProtectionChange and trainRadioChange. In these cases the general interpretation of @dir is quite simple: the specified change only applies when traversing the track in the given direction. The best known example is probably speedChange, where it is common knowledge that speed profiles my differ in the two directions. The detailed interpretation can however be tricky, as I will soon describe. Also, for some of these characteristics, it does not make much sense allowing different values in different directions. I suggest to deprecate the @dir attribute for most of these elements, as I do not know of applications where their properties can differ by direction. I welcome all examples to the contrary. For speed, train protection and train radio, there can obviously be some differences between directions, at least in exactly where the change happens. For gradients, there are use cases where the exact gradient at every point is unknown but the average or characteristic gradient between consecutive signals is known, and since signal placement may vary between directions, so will these averaged gradients (even if the track itself must have the same gradient in both directions).

Suggested changes:
* DEPRECATE @dir for axleWeightChange, clearanceGaugeChange, electrificationChange, gaugeChange, operationModeChange, ownerChange, powerTransmissionChange and radiusChange unless anyone has an example
* Only allow up/down for trainRadioChange

The remaining question for all change elements when @dir is used, is what part of the track the given values apply to. Let's start with a speedChange example without @dir:

```
<speedChange pos="0" vMax="40"/>
<speedChange pos="100" vMax="80"/>
```

This is simple to interpret as the following. Generally, the new values apply to the segment of the track between this element and the next in the direction of the track.

```
Pos     0                  100                  200
Track   |-------------------------------------------->
vMax    |       40        |         80
```

Now, if we alter one direction, what should the @pos value of the element with @dir="down" be?

```
Pos     0                 100                 200
Track   |-------------------------------------------->
vMax -> |       60        |         80
vMax <- |       40        |         80
```

I cannot find any documentation answering that question. According to a very old thread on this forum, the intended use was like this:

```
<speedChange pos="0" dir="up" vMax="60"/>
<speedChange pos="0" dir="down" vMax="40"/>
<speedChange pos="100" vMax="80"/>
```

While, in my experience, and also mentioned in a later post in the same old forum thread, the common practice is to use @pos="100" on the @dir="down" element. I find the intended use to be more consistent: the given new values always apply to the segment of track between this change and the next change of the same type found in the direction of the track. The @dir attribute, if given, only affects which direction (of travel) the values apply to.

BUT, it is important what the common practice is. So unless I am mistaken about the current common practice, the definition would be: unless @dir="down" the given new values apply to the segment of track between this change and the next change of the same type found in the direction of the track. When @dir="down", the given new values apply to the segment of track between this change and the previous change of the same type.

Regardless of choice of definition, it should be the same for all change elements.

Suggested change:
* Document what part of the track the new values apply to when using the @dir attribute