Subject: Re: V1.00 RC1: switchRef/crossingRef
Posted by Matthias Hengartner on Mon, 04 Oct 2004 12:57:30 GMT
View Forum Message <> Reply to Message

> Jepp, that's the way I would prefer it, too. But as usual, things are
> not as easy as in your example, although it was perfect to understand
> your intention. So I will do my very best to make things complicated ;-)

Yes, you're right, my example was quite a "model example".

>
> If we start or end a track with a switch, we can distinguish between 2
> cases:

BTW: These 2 cases can have 2 sub-cases: The switch can be placed on
trackEnd (a) or on trackBegin (b), and the "orientation"-attribute refers to
the direction of the track (which is defined by trackBegin and trackEnd).
See below in the ASCII-drawing.

>
>
> (1) the switch element belongs to the straight track
>
>
>   / first connected track
>   /
>   o
>
>   o
>   /
>   /
> -------o---------o   o------ second connected track
(1a) ---> (trackEnd)
(1b) <--- (trackBegin)

>
>
>
> (2) the switch element belongs to the branch track
>
>
>   /
>   /   /   ^
>   o   /   /
>   /   V (2a)   / (2b)
>   / (trackEnd) (trackBegin)

> first -----o   o---------o   o------ second connected track
>

> The crucial thing is the required "orientation"-attribute in the
> <connection>-element of a switch. "orientation" can be either
> "incoming", "outgoing", "right angled" (???) or "unknown". Which value
> is to be chosen for the second track in case (1) and for both tracks in
> case (2)?
>
> I suggest an additional value "straight" (which perfectly coincides with
> the possible values for "trackContinueCourse")

In case (1a), I'd take "outgoing" for _both_ connected tracks (1b: "incoming"). The attribute "course" would have the value "straight" for the second connected track (and "left"/"right" for the first [1a/1b]).
So in my opinion, we _could_ introduce the value "straight" in the "orientation"-attribute, but there's no need for it.

> and the __convention__ to
> let the <switch>-element be part of track at the switch's tip. Thus,
> role of every track is unambiguous.

I agree fully with you!
If I _had to_ realize case (2) in railML, I probably would say that the first connected track is "outgoing" (2a) / "incoming" (2b), and the second connected track is "incoming" (2a) / "outgoing" (2b).
But this is of course a very "dirty" implementation. And I don't think that the possibility to implement case (2) is really needed (It can easily be avoided).

*****
Another possibility... We could abandon the special treatment of switches/crossings which are placed on trackBegin/trackEnd (I feel a little uncomfortable about saying this, because this idea is penned by me...).
However, then we'd have a <simpleConnection> and a <switch> which have the same position (on trackBegin/trackEnd). So, in case (1) we'd have a <simpleConnection> to one of the connected track and a <switch>/<connection> to the other. In case (2), we'd have the <simpleConnection> to the first and a <switch>/<connection> to the second connected track. It would be clear, which "orientation" a track has, as a simpleConnection is always "straight".
Disadvantages of this solutions are:
- we have data redundancy (but not very much)
- we have to compare the position of the switches/crossings/simpleConnections to get the information, that a

switch/crossing is placed on a trackEnd/trackBegin

Or, final idea (for the moment ;-)): We could combine these 2 approaches:
We could have a <simpleConnection> with a reference to a
<switch>/<crossing>.

What do you think?

Best regards from sunny Zurich
Matthias Hengartner

Matthias Hengartner

++ 41 1 633 31 09
hengartner@ivt.baug.ethz.ch
