Subject: [railML3] Refactoring of states (e.g. infrastructure states)
Posted by Vasco Paul Kolmorgen on Thu, 17 Oct 2024 15:12:55 GMT
View Forum Message <> Reply to Message

Dear all,

In response to requests of the community to also allow modelling of status information regarding interlocking we decided to refactor how of how states, such as infrastructure states information, are encoded is encoded in railML. In order to provide a solution that is more general and also allows for example rolling stock information to carry status information we came up with a more general approach. As of version 3.3 what is known in railML 3.2 as infrastructure states will be defined in common. We introduced a new top-level element for this names <states>.

With it, it is possible to specify states that apply to the whole railML document, similar to what before could be done with
<infrastructureStates> although not limited to just infrastructure.
Elements in infrastructure, interlocking and rollingstock have been extended each by an optional repeatable element called <elementState>. This element state allows specifying an overriding status for the enclosing element. The states in common as well as the element state each provide child elements to specify their validities.

The general idea is to define the general status in the common section, while making sure that the states defined there do no overlap in time. Wherever an element, such as an overcrossing deviates from the state defined for the document under common, an element state can be defined to express that. These element states by nature can and should overlap in time with the states defined under common, however they should not overlap with each other (within the same element).

With this extension it will be possible from railML 3.3 onward to also encode the planning for interlocking and rollingstock between applications.

We just published this for review with the beta2 of railML 3.3 this week. All the railML v3.3 betas can be downloaded free of charge from our website after login:
https://www.railml.org/app.php/en/download/schemas.html

We invite you to download the beta version and give us your feedback until October 28th via the railML forum, by e-mail or in the working group meetings.

The ticket for this change is available at
https://development.railml.org/railml/version3/-/issues/552.

Best regards,
--
Vasco Paul Kolmorgen - Governance Coordinator
railML.org (Registry of Associations: VR 5750)
Phone railML.org: +49 351 47582911
Altplauen 19h; 01187 Dresden; Germany

Subject: Re: [railML3] Refactoring of states (e.g. infrastructure states)
Posted by Dominik Looser on Wed, 23 Oct 2024 13:27:29 GMT
View Forum Message <> Reply to Message

Dear coordinators,

Thank you for giving us the opportunity to give feedback to the railML 3.3 beta2 release.
We noticed that some important objects do not support a <elementState>-subelement:
- gradientCurve
- horizontalCurve
- netElement
- organizationalUnit

For the organizational unit, we believe that Thomas Nygreen's suggestion (per mail) to add start and end validity would suffice.

Regarding the infrastructure elements, we believe the following use case is both realistic and also currently modelled in railOscope:
When a new track edge is created in railOscope (such as a second track added alongside an existing one) it can be designated with the status "planned." We would like to export such track edges as a <netElement> with an <elementState value="planned" />. This is also supported in a similar way in railML 2.5 and 3.2.
Additionally, all new elements located on this track, such as signals, gradient, or radius elements, would similarly get the "planned" status within railOscope. These elements would be exported using the <elementState /> subelement for both <gradientCurve> and <horizontalCurve>.
The <elementState> subelement also provides a clear method for defining other states, such as "disabled" or "dismantled" (including their timeframes), which is important for managing infrastructure projects.
This is mostly relevant in the SCTP-use case, where a schematic track plan can graphically differentiate between different infrastructure states.

We believe that railML 3.3 should support the description of these changes and the netElements and these infrastructure elements are part of it.

Thank you and best regards,
Dominik Looser

Subject: Re: [railML3] Refactoring of states (e.g. infrastructure states)
Posted by Thomas Nygreen on Thu, 24 Oct 2024 16:39:30 GMT
View Forum Message <> Reply to Message

Dear Dominik,

The conclusion of the coordinators is that netElement, gradientCurve and horizontalCurve are abstract entities that do not take on different states. We will therefore not include elementState on these in railML 3.3. A netElement is not a track, it is an abstract part of the topology, on which one can place tracks and other infrastructure entities. The netElement (or gradientCurve or

horizontalCurve) itself is not "operational", "closed" or "dismantled", but any track or other entity positioned along the netElement can have a state, allowing you to model the use case you describe with railML 3.3.

netElement, gradientCurve and horizontalCurve support basic RTM validity, so one is able to specify a start and/or end of availability/applicability.

Please let us know if there is anything we have overlooked in your use case.

We will add start and end of validity for organizationalUnit.

Best regards,
Thomas

Subject: Re: [railML3] Refactoring of states (e.g. infrastructure states)
Posted by Torben Brand on Fri, 25 Oct 2024 12:04:59 GMT
View Forum Message <> Reply to Message

Dear Thomas,
Jernbanedirektoratet agree with setting elementState on track instead of netElements (as they are abstract). The solution with setting start and end validity for organizationalUnit is fine for us.

Jernbanedirektoratet disagrees with not giving geometry elements like gradientCurve an elementState. This as we often have plans that have objects planned on the track, but the gradients have not been planned yet, but we need them for RTC. The gradients are thus estimated and get a state different from the track with the value "conceptual".

Furthermore the gradients can be individually defined in RailOscope and we need to exchange the information with Bane NORs new DIM tool.

Thus, we hope to have the possibility to set elementState on gradient (and horizontal) curves as an optional item in railML3.3.

Subject: Re: [railML3] Refactoring of states (e.g. infrastructure states)
Posted by Georg Boasson on Fri, 25 Oct 2024 12:22:26 GMT
View Forum Message <> Reply to Message

Dear Thomas

In my opinion the netElement is the basic building block in the description of the topology. Since the topology might change during the lifetime of the railway it also make sense to have the possibility to give the netElement a state similar to other building blocks. I don't see netElements as abstract entities.

May be I am wrong, but I hope you can consider my thoughts here.

Best regards
Georg