Subject: [railML3] Overview of how to deal with schema changes
Posted by on Mon, 22 Apr 2024 12:51:00 GMT
View Forum Message <> Reply to Message

Is there a general overview of how to deal with changes between minor versions in railML3.x? I am interested in both the schema development perspective as well as the implementation of the interface as import/export software. There was already a discussion in this forum and also a news article (https://www.railml.org/en/public-relations/news/reader/handl
ing-minor-changes-within-versions.html) that also contains a conclusion which approach was chosen for railML3.x, but I did not find a general description, especially not concerning the software implementation part.

According to the mentioned news article, the (planned) removal of attributes / elements must first be marked as "deprecated" in a minor version. Unfortunately, the article does not explicitly describe what to do when making changes to elements/attributes. I assume that a minor version with both representations (deprecated and current) must also be published in this case.

It is also not clear to me what this means for the implementation (and certification) of export/import interfaces. Does an export or import interface for a version 3.x have to support all regular elements of this version as well as all elements marked as "deprecated"?

I would find it very useful if this inforation could be summarised in a general document as a help for railML schema developers and software implementers.

Best Regards
Christian
--
iRFP e. K. · Institut für Regional- und Fernverkehrsplanung
Hochschulstr. 45, 01069 Dresden
Tel. +49 351 4706819 · Fax. +49 351 4768190 · www.irfp.de
Registergericht: Amtsgericht Dresden, HRA 9347

Subject: Re: [railML3] Overview of how to deal with schema changes
Posted by Thomas Nygreen on Fri, 26 Apr 2024 15:49:34 GMT
View Forum Message <> Reply to Message

Dear Christian,

Strictly speaking, the discussion and decided approach only covers how to remove something that has already been included in railML 3.x. In some cases this may happen without it being replaced, but as you imply, the normal case is that something new will take its place, normally in the same minor version where the old modelling is deprecated. We have already identified some types of changes where the old and new modelling will conflict, and I am working on a suggestion on how to extend the decided policy in the same spirit of balancing stability and progress: https://development.railml.org/railml/version3/-/issues/535

I will include your input in the process. Others in the community are also very welcome to

contribute.

Best regards
Thomas

Subject: Re: [railML3] Overview of how to deal with schema changes
Posted by on Thu, 02 May 2024 09:02:16 GMT
View Forum Message <> Reply to Message

Dear Thomas,

Thank you for your reply. I am very interested in your work on dealing with general changes to the schema between minor versions.
It is also not yet clear to me what the benefit of the current rule with the 'deprecated' elements in the context of the implementation of import/export software interfaces: It looks to me that at least one of the two sides of the data exchange (import or export interface) must support the regular as well as the 'deprecated' elements/attributes. However, as so far only deprecated elements/attributes are to be marked as 'deprecated', general upward/downward compatibility between minor versions is not guaranteed. So far, I don't really understand the idea behind the 'deprecated' markings and am therefore also interested in background information.

Best Regards
Christian
--
iRFP e. K. · Institut für Regional- und Fernverkehrsplanung
Hochschulstr. 45, 01069 Dresden
Tel. +49 351 4706819 · Fax. +49 351 4768190 · www.irfp.de
Registergericht: Amtsgericht Dresden, HRA 9347
