Subject: [railML3.2]: Proposal for removal of xs:any elements
Posted by Milan Wölke on Wed, 16 Mar 2022 12:32:08 GMT
View Forum Message <> Reply to Message

Hi Community,

bei unserer letzten Konferenz in Göteborg, Schweden hatte unser Common-Koordinator das Thema aufgebracht, in railML3 die Art und Weise zu ändern, mit der Erweiterungen für railML erstellt werden können. Thomas hat in dort eine neue Vorgehensweise vorgestellt, die sich allerdings mit der gegenwärtigen Herangehensweise ausschließt. Er hat dazu einen Forumpost erstellt, der leider bisher nicht viel Beachtung gefunden hat.

https://www.railml.org/forum/index.php?t=msg&th=638& start=0&

Die railML.org-Koordinatoren halten es für eine gute Idee, diese neue Technik zu nutzen, da es unter anderem erlaubt, auch Dokumente mit eigenen Erweiterungen der jeweiligen railML-Schnittstellen zu validieren. Da eine solche Änderung aber eben dazu führt, dass bestehende Erweiterungen mit einer neuen railML Version nicht mehr funktionieren werden, brauchen wir euer Feedback um zu prüfen, ob die Vorteile, die wir auf dem neuen Weg sehen, auch euch überzeugen. Bitte lasst uns unter dem Thread oben wissen, wie ihr darüber denkt.

Zum Hintergrund:
Bisher war die gängige Praxis im offiziellen railML Schema Erweiterungspunkte vorzusehen. Damit wurde zugelassen, dass ein XML, das an einer solchen Stelle Tags enthält, die nicht zu railML gehören, trotzdem als valides railML angesehen wurde. Der Nachteil dieses Vorgehens ist, dass es damit nicht möglich ist, durch das Erweiterungsschema selbst festzulegen, wo diese railML-fremden Tags zulässig sind und wo nicht. Eine Erweiterung, die dafür gedacht ist, Öffnungszeiten für ein Stationsgebäude zu erfassen, könnte so auch dazu genutzt werden, Öffnungszeiten für einen Umlauf zu erfassen oder für eine Zugnummer. Dies ist fachlich nicht sinnvoll. Mit den Erweiterungspunkten ist es nicht möglich, dies einzuschränken. Mit der neu vorgeschlagenen Herangehensweise (siehe Forenpost) wäre das kein Problem mehr. Zusätzlich könnten auch Tools zur Codegenerierung eingesetzt werden, um effizienter Code zum Im- und Export von railML zu implementieren.

Wir schlagen daher vor die Erweiterungspunkte mit railML 3.2 durch die neue Vererbungsmethode abzulösen. Bitte lasst uns unter dem Link oben wissen, wenn aus eurer Sicht fachlich/technisch etwas dagegen spricht.

At our last conference in Gothenburg, Sweden, our Common Coordinator raised the issue of changing the way extensions for railML can be created in railML3. Thomas presented a new approach there, which is, however, mutually exclusive with the current approach. He has created a forum post on this, which unfortunately has not received much attention so far.

https://www.railml.org/forum/index.php?t=msg&th=638& start=0&

The railML.org coordinators think it is a good idea to use this new technique, as it allows, among

other things, to validate documents with custom extensions of the respective railML interfaces. However, since such a change will mean that existing extensions will no longer work with a new railML version, we need your feedback to check whether the advantages we see in the new way are convincing to you as well. Please let us know what you think under the thread above.

Background:
Previously, the common practice was to provide extension points in the official railML schema. This allowed an XML that contained non-railML tags at such a point to still be considered valid railML. The disadvantage of this approach is that it is not possible to specify through the extension schema itself where these non-railML tags are allowed and where they are not. An extension that is intended to record opening hours for a station building could thus also be used to record opening hours for a circulation or for a train number. From a technical point of view, this does not make sense. With the extension points it is not possible to restrict this. With the newly proposed approach (see forum post) this would no longer be a problem. In addition, code generation tools could also be used to implement code for importing and exporting railML more efficiently.

We therefore propose to replace the extension points with railML 3.2 with the new inheritance method. Please let us know at the link above whether there are any professional/technical arguments against this from your point of view.

Best regards, Milan
