Subject: Re: railML 3.x: Data Modelling Patterns
Posted by Joerg von Lingen on Mon, 19 Nov 2018 04:25:05 GMT
View Forum Message <> Reply to Message

Dear all,

Christian's proposal for the modelling pattern is quite useful to normalise the sub-schemas within railML. Basically I
agree with the set patterns. However, it shall be used as a guideline and not as a strict corset. The railway world is
wide spread and requires flexibility.

I see currently the following issues where the strict adherence to the patterns is not useful:
1) Hierarchy
RS: The view would be formation/vehicle which are at the same time containers. Seeing Engine/Wagon/Maintenance/Classification/... as objects their parts definitely have several levels. One could argue to
use the components as containers within the "super"-container vehicle. But this would bring a lot overhead in referencing.
IL: According the pattern we have again "super"-containers without real view-level - AssetsForIL/Controller/SignalBox/GenericIM. The objects in these containers sometimes need more than one part-level.

2) Layer
IL: Elements like Controller or SignalBox cannot live without the remaining AssetsForIL/GenericIM because they need them
as basis.

3) Extension points
RS: In the 2.4 version there are not much extension points requested. At least most of enumerations can be extended.
IL: A good portion of elements are derived from EntityIL which provides the possibility of any-elements. However,
extensions with any-elements were not requested. The majority of enumerations cannot extended as this is mostly not
sensible.

4) Booleans
RS/IL: I would prefer option 1 and make the attributes in question optional.

5) Naming
IL: The use of verbs in the names enhance the legibility as it points already to the related function of the element. In
addition it would cause conflicts or confusion, if a name like "overlap" appears at several locations due to the
referencing.

Thanks to Christian for his work.