Subject: Re: [railML3|alpha] Suggestion for an enhanced topology model
Posted by Alain Jeanmaire on Thu, 27 Oct 2016 15:06:38 GMT
View Forum Message <> Reply to Message

Dear Martin,
Thank you for your interest to this major evolution, and for those relevant questions. I will try below to give you a clear explanation; do not hesitate to come back if any need for further details.

The idea of modelling a railway network topology as a mathematical graph comes very naturally. After all, the network layout even looks like a graph, with the tracks as edges and the switches and crossings as nodes.

Deceived by this intuitive picture, it actually took me a long time to realise that the RailTopoModel is modeled the other way around.
--> if you refer to IRS 30100, this major (and disturbing) feature is clearly mentioned for attention (chapter 1.5 Modeling principles)
The tracks are nodes in the graph, not edges! OK The edges are instead the passages through switches and crossings, which connect the tracks.
--> at this level you should stick to pure TOPOLOGY, that is nodes and edges, and not speak about pieces of equipment as iron tracks, switches or crossings, which will appear in the functional layer.
--> up to now, at toplogy level, the network is described independatly of the mean of transportation; i.e. can be road, rail or pedestrian path.
--> so, in a conexity graph, all topology objects are nodes, and edges only relate relations between topology objects.

I guess there were good reasons for choosing this model, even though I don't think I'm the only one to make this mistake. But the comprehension of the model is a secondary issue. What bothers me more is the fact that the switches and crossings are not at all included in the model.
--> at this toplogy level, we definitely do not YET include functional objects as switches and crossings
--> those functional objects will be modeled furher as "NetEntity"

This is a problem, because the switch or crossing as such implies a restriction on the use of the relations through it. Take the example of a diamond crossing. We will have four tracks, connected in pairs by navigable relations. The model does not tell us that there is a restriction against using both these relations for traffic at the same time. But many applications are dependant on that information. Interlockings obviously, but also CTC pretest function, TMS traffic optimisation, time table simulation tools etc. Some of these may not be railML use cases today, but they will for sure be in the future.
--> those capacities or restrictions are perfectly modeled in the topology layer, using the notion of "navigability", within the "PositionedRelation", independantly of the function/equipement which will ensure this capacity or restriction.

I believe railML needs to cover the gaps that RTM leaves open in this respect. My suggestion is to introduce topology relations on asset level.

Please look at the UML diagram below. In addition to the well known RTM classes, I have

introduced the classes Track and NetworkNode. NetworkNode is an abstract base class with a number of possible descendants, shown here is Switch. The Track has a one-to-one relation with a LinearNetElement, and two relations to the nodes at each end. Depending on the subclass, the node will have relations to relevant PositionedRelations. In the case of the switch, the relations indicate which course of the switch each PositionedRelation corresponds to (this eliminates the need for the SwitchBegin and SwitchEnd assets).
--> as mentioned above, a main structuring principle of RTM is not to mix topology and functional objects, then split the system into layers: topology, functional objects, physical assets,...
--> the functional objects (switch, track,... are covered by "NetEntity".

--> Your proposal goes with different principles... as mentioned in IRS and in the earlier feasibility study, this choice is related to the limitation encountered in the multiple previous experiences, with the clear will to be agnostic of any usages and therefore open to any further development, with minimum risk of refactoring.

I hope those explantations will clarify the choice we have done for long term with RTM and railML3.

Alain jeanmaire / Gilles Dessagne
SNCF Réseau