Subject: planned passenger capacities
Posted by Andreas Tanner on Wed, 16 Oct 2013 11:49:14 GMT
View Forum Message <> Reply to Message

Dear all,
we want to transmit the planned passenger capacity of a train section as opposed to the formation that the train is "implemented with" in a subsequent planning step. The natural place to put this property would be the trainpartSequence of the commercial train. The train parts (referenced both by the commercial and operational train) then have the formation with the number of seats etc.

In a more general sense, most properties of <formation> are possibly needed here. Also, they may vary across operating periods. So we are more or less in the position to need a reference to a trainPart - but not a <trainPartRef> since it's not one among the other train parts with their positions.

So I suggest to add a child <commercialTrainPartRef> for this purpose.

On the long run (railML3), it should be possible to model commercial and operational views on a train with different trainParts and having a "implements" relationship from the operational to the commercial view.

What do you think about this?

Best, Andreas.

Subject: Re: planned passenger capacities
Posted by on Tue, 19 Nov 2013 12:28:19 GMT
View Forum Message <> Reply to Message

Dear Andreas,

I guess your suggestion very much relates to the topic "how to implement different planning stages". Also, we have to distinguish between the "minimum necessary passenger capacity" (may be existing throughout all planning stages) and a kind of "pre-planned" minimum passenger capacity, which may be overwritten by an actual formation in a later planning stage.

So in my opinion we should first clarify how to deal with different planning stages and then come back to the passenger capacity.

Since this concerns all kinds of capacity (1st/2nd class, WC, beds, Restaurant a. s. o.) we should use a generic structure for trainParts / formations / vehicles.

We should also be (keep!) aware that originally the <trainPart> should be an "atomic" structure for "elementary" information. To keep RailML consistent, we should not "soften" this too much, e. g. not to move too much (elementary?) information away from the <trainPart>.

Best regards,
Dirk.
