Subject: [railML 3] Areas in railML 3
Posted by Thomas Langkamm on Tue, 21 Sep 2021 10:09:15 GMT
View Forum Message <> Reply to Message

Dear all,

we have discussed the introduction of areas in railML 2 (https://www.railml.org/forum/index.php?t=msg&th=813, https://trac.railml.org/ticket/393). In the IS working group we had some discussions on whether we should introduce a similar concept in railML 3. More precisely:

railML 3 has 2 existing concepts which could be used to define an area, which are (1) IS:operationalPoint and (2) aggregation if netElements in higher level topologies (that is, a netElement in a macroscopic topology is an aggregation of microscopic netElements). However, the netElement aggregation is probably not a good tool to model areas, because we may have many different types of areas (see the list in the ticket linked above) and we would have to subdivide netElements whenever an area does not contain all of the netElement. This could lead to a combinatorial explosion with a huge number of netElements.

So basically, we have the option to use operationalPoint or to introduce a new concept similar to the one we use in railML 2.5. Personally, I think it would be a good idea to mirror the changes from railML 2.5 in railML 3 and introduce a new area object, for the following reasons:

operationalPoint is a very generic concept. Although this gives us a high degree of flexibility, in my experience it's better to have concepts that are a bit less generic as it makes the model more descriptive and consistent. Basically a very generic concept will probably lead to a large number of very different uses and perhaps different interpretations on how to use it.

Many of the areas that we discussed are well defined, for example the areas used by an interlocking to display train occupations, or the part of a network that is controlled by one interlocking. I believe it would help the schema if we would use more specific types here instead of a very generic type.

Some other areas are only references to external software systems, whose data we don't want to model in railML. An example would be maintainance systems, where we have track areas allowing certain types of work or access to certain machines. Here we need to define the part of the rail network belonging to these areas, but the only other additional data would be a reference to the external software system.

Looking at the schema, a minimal and generic definition of an area would be basically the use of the location types (areaLocation, linearLocation and spotLocation) plus the optional "external" reference, without most of the other attributes that we can add to operationalPoint.

I don't have a strong preference on whether we should introduce additional types for specific areas (like information area, track sections used in interlockings or the parts of the network controlled by an interlocking), or if we should add a type attribute to the area definition. What do you think?