Subject: Re: Platform @belongsToParent and <ownsPlatformEdge>
Posted by Thomas Nygreen JBD on Mon, 18 Feb 2019 19:55:45 GMT
View Forum Message <> Reply to Message

Dear Christian,

christian.rahmig wrote on Mon, 18 February 2019 16:12
> Option 1 is my favourite.

I assumed so, but the other options are also valid (although the last one is weird).

> In case we want to remove the attribute @ownsPlatformEdge, the related
> platform edges have to reference their parent platform via @belongsToParent.

Yes. Nothing strange about that.

>> Also, in all these examples, the lower levels of the
>> hierarchy can be skipped. There are also multiple ways to
>> assign <linearLocation>s: either all elements have them (and
>> platforms have one for each track), or only the lower levels
>> have them, or something in between.
>
> I would expect platform edges to be modelled / located as linear
> elements, while a platform is more likely seen as an areal object. For
> operational purposes it is necessary to have linear coordinates (mileage
> values) for the platform edges.

Or you can have no platform edges at all, but linear coordinates on two different tracks on the same platform. I.e. with the lower levels of the hierarchy skipped. <stoppingPlace>s on both tracks can reference this same platform.

>> christian.rahmig wrote on Mon, 11 February 2019 15:29
>>> we need to think about alternatives of how to
>>> distinguish between
>>> platforms and platform edges.
>>>
>>> Alternative 1:
>>> Introduce a boolean flag in <platform>: @isPlatformEdge
>>>
>>> Alternative 2:
>>> Re-Introduce the functional infrastructure element
>>> <platformEdge>
>
> Alternative 3:
> <platformEdge> as child element of <platform>
> In that case, a <platformEdge> cannot exist alone (without a platform).

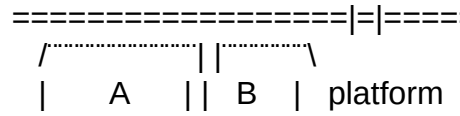
Alternative 4: no explicit platform edge concept.

If the user systems use this separation between platforms and platform edges, they can be separated by, for instance, regarding <platform>s with <areaLocation>s as platforms and <platform>s with <linearLocation>s as platformEdges. But if this separation is a strong requirement, I strongly prefer the @isPlatformEdge attribute (as optional).

>> If we need or want a strict hierarchy, considering platforms
>> and platform edges as separate entities, then I think it
>> would make the most sense to have them as separate elements.
>> But do we really need to? I would rather keep using one type
>> and leave the hierarchy up to the writing system. One thing
>> we could do to make life a bit easier for the reading
>> systems is to introduce one semantic constraint: If
>> attributes describing a parent <platform> such as length and
>> height are given, they must include other <platform>s that
>> @belongsToParent.
>
> I don't understand how you mean this, sorry.

In the first half, I was just saying what I have reiterated above, that I do not see a strong need to separate the concepts platform and platform edge.

The second half was an attempt to rule out the weird option from my list of hierarchies. (The one where the platform edges just reference each other and not a parent platform.) My example is probably too weird, but a slightly more realistic one is this: Let's say we have a platform of 250 metres physically divided in two parts A (150 m) and B (100 m) (e.g. by a level crossing or change in height or surface).



Maybe we regard A as the main part of the platform, and would use @belongsToParent on B to reference A. But this would be confusing if A@length="150" and B@length="100" and B@belongsToParent="A". What is then the total length of the platform? What I suggested, in more plain text, was to require that whatever B@belongsToParent references must be something that B is a part of.

[b]Summing up:[b] I do not need a way to separate platforms from platform edges (alternative 4). But if someone else needs a way to do this explicitly, I prefer to add an optional attribute (alternative 1).