Subject: Re: railML 3.x: Data Modelling Patterns
Posted by Joerg von Lingen on Wed, 12 Dec 2018 05:02:13 GMT
View Forum Message <> Reply to Message

concerning 1) the views in IL are different. You shall think about the different levels in interlocking:
control level with HMI and traffic management systems - Controller
interlocking level with safety related logic - SignalBox
field element level with physical assets - AssetsForIL
SpecificIM is an addition to view the specific rules

concerning 2) I still see the preambles useful. There might be other ways but would require extensive rework of the schema. This may be done anyway concerning polymorphism as stated below.

concerning 3) we have in-between decided to have always the suffix in case of name collisions in different domains. The suffix shall be used in both domains if developed during one release. Otherwise only the later coming gets the suffix. Thus we will have SwitchIS and SwitchIL etc.

With MoveableObject we have another issue. It is not SwitchIL but the abstract class SwitchIL is based on. At the moment the inheritence is used by xsi:type, however, I understood now this may not be the best way in XML world.

Best regards,
Joerg v. Lingen - Interlocking Coordinator

On 11.12.2018 13:48, Martin Karlsson wrote:
> Looking at the Interlocking domain in the v 3.1 release
> candidate, I have observed a number of deviations from these
> rules. As the rules are quite new, it is still an open
> discussion whether to change the rules or the schema, or
> even keep the rules but motivate a deviation. Either way, I
> would like to highlight my findings.
>
> 1) Hierarchy. IL has introduced a "super-container" level
> (e.g. AssetsForILs) between the Domain (Interlocking) and
> View (AssetsForIL). So you can have multiple Views of the
> same type in one file.
>
> The reason for this is apparently that Controller, SignalBox
> and SpecificIM are considered to be Views (and they
> obviously exist in multiplicity). But in my opinion, they
> are Objects. A view should represent a certain type of
> information, not an individual entity.
>
> I believe this should be aligned so that it is the same
> everywhere. Either follow the rules, or change them.

>
> 2) Naming of containers. According to the rules, container
> elements are named from the objects they contain, but in
> plural. In addition to this, IL has added a preamble,
> qualifying the type of relation. E.g. "knowsRoutes" instead
> of just "routes".
>
> I'm not convinced that this preamble adds any value to the
> developers. As far as I have seen, there are not multiple
> containers of the same object type anyway. But also here, an
> alternative is to apply this pattern consistently.
>
> 3) Naming of Objects. This is not really covered by the
> rules, but anyway... The principle used in IL has been to
> first try to find a name that is different from what is used
> in IS (e.g. MoveableObject instead of Switch). If this has
> not been possible, a suffix of "IL" has been added to the
> name (e.g. LevelCrossingIL).
>
> I would prefer to use the suffix variant consistently. That
> would make it apparent that the IS and IL objects are
> different views of the same thing. In earlier drafts of the
> IS domain, this principle was used to distinguish between
> functional and physical assets.
>
>
