Subject: More standard attributes for objects
Posted by Claus Feyling on Mon, 04 Feb 2019 18:27:31 GMT
View Forum Message <> Reply to Message

Dear all!

We are using the editor RailCOMPLETE®, a graphical and table-based editor based on AutoCAD®, in order to enter data into a network-oriented railway infrastructure model. Such models are usually geographic and comprises areas (zones), alignments, point objects with XYZ position (and with a 3D orientation in space).

A simple model in this CAD editor is just a AutoCAD DWG file with objects. The DWG file structure of AutoCAD is a proprietary to Autodesk but widely used as a database format. One DWG file may reference another DWG file. The referenced file may reference another DWG file etc.. We therefore need an identification which is unique within a context which changes every time a new DWG file with objects is referenced.

On the other hand, it can be argued that two teams may produce models of the same infrastructure, and the objects being modelled will usually be given different id’s by these teams, even though the objects being modelled are “the same” in a certain way.

Suggestion 1: id - unique within all contexts

We suggest that the existing railML attribute 'id' is defined as it is at present, allowing the same id to be found in different railML models. However, its general usage shall be encouraged to be as follows: The id should be unique within all foreseeable contexts and generated once at the time the object was created.

We suggest that the following should always hold true for a well-formed railML file:

a) A globally unique ID (guid) shall be computed and assigned to each object's id attribute at the time the object is created.

b) The id will never change.

Immediate consequences:

If the same object shall exist in several construction phases, then this object must contain information about the phases it belongs to.

Two objects may share identical data except their id. They are still two separate objects in the model. Sharing the same 'code' and/or 'name' values does not make the two objects the same object in a strict sense.

Practical adaptions:

Even though two objects with different id's are strictly speaking different objects, a user may still decide, within his or her context, that when for instance 'ocp' and 'code' have the same value for tow objects of the same object class, then these two are to be interpreted as 'the same' within that model.

Suggestion 2: tag - unique within one administration's context

A guid is not human-readable. Most real systems in operation have a TAG system, which constitutes a unique and human-readable identification of an object within their context. Bane NOR, as an example, has a tag system called 'BANEDATA OBJEKT ID', consisting of code of the form "XX-YYY-nnnnnn", for instance 'SA-TEL-002349'. SA means "Signalanlegg" (signalling system) in this context, and "TEL" means Axle counter sensor ('Tellepunkt'). The six digits are a unique number within the SA-TEL category. Other railway administrations might have similar but differently specified tag encoding systems. We assume that one administration has only one tag system.

We suggest that the attribute 'tag' is added to all objects in railML.

Suggestion 3: code - unique within the context of a given station and object type

Railway objects such as switches are usually numbered using a numbering system which is unique within the station they are part of. These numbers are then re-used at the next station or at the next railway line. The same occurs for signal object numbers, which might be unique within one line but the numbers then re-appear for another line. Many objects are related to each other in a parent-child relationship where the child will be referred to using a name formed from its parent and from some numbering scheme for such child objects, ensuring uniqueness within the current context. For instance, if a switch is numbered ‘501’ and this switch has three point machines, then these point machines could be identified using the names “501.1”, “501.2” and “503.1”.

We suggest that the railML attribute 'code' should preferably be used as a compact, context-sensitive, human-readable and easily remembered encoding of the object’s identity. It should use the ‘tribe language’ of those who plan, build and maintain these objects. The 'code' field should lend itself well as a basis for other, related, objects when such related objects need a value for their own 'code' attribute. The 'code' attribute is usually shown in a context where the object type, the station name, the line name, the administration, the country etc are already known, for instance as a number appearing in a column in a table.

Suggestion 4: name - an ornamented and context-dependent identification of an object

With the 'id' and the 'code' attributes used as recommended above, we still have a need for a human-readable text which may appear in drawings or in references to an object mentioned somewhat outside their limited context. As an example, referring to switch number 501 inside Sandvika station on the Drammensbanen line may be presented in a text or in a PDF drawing as either "501", "V.501", "SV-V.501", “SV-V.501 (DB)", “Sandvika sporveksel 501 (Drammenbanen)" or "Sandvika (Norway), switch 501 on the Drammen line" etc – the amount of information provide will depend on the assumed knowledge of the context that the intended audience has.

We suggest that the railML attribute 'name' should represent the object's human-readable name with sufficient extra contextual information, intended for a reader which does not a priori have enough contextual knowledge to understand the object’s identity just from the ‘code’ attribute.

Best regards,
--
Claus Feyling
Daglig leder
CEO

Railcomplete AS
Kontor: Vestfjordgaten 4, N-1338 Sandvika
Firmapost: Brageveien 4a, N-0358 OSLO
+47 908 24 018
www.railcomplete.com

Subject: Re: More standard attributes for objects
Posted by Thomas Nygreen JBD on Wed, 06 Feb 2019 16:23:32 GMT
View Forum Message <> Reply to Message

Dear Claus,

Are these suggestions for railML2.x or railML3.x?

> Suggestion 1: id - unique within all contexts
>
> We suggest that the existing railML attribute 'id' is defined as it is
> at present, allowing the same id to be found in different railML models.
> However, its general usage shall be encouraged to be as follows: The id
> should be unique within all foreseeable contexts and generated once at

> the time the object was created.

railML2.x uses the XML Schema ID datatype, and IDREF for references to other objects. As a consequence it is not possible to reference an object that is not contained in the same file. The datatype also dictates that the id must be unique within the file and must start with a letter or an underscore. Apart from this, the writing system can decide freely on how to generate IDs.

railML3.x allows both XML Schema IDs/IDREFs and UUIDs for all id and reference attributes. This provides the possibility to identify and reference objects uniquely across files, contexts and time. How and when the UUIDs are generated is up to the owners of the data. railML3.x also allows separately listing multiple external identifiers and/or references for an object.

In my opinion the possibilities provided in railML3.x are sufficient. I think it is a good idea to encourage the use of UUIDs, but I do not think it can be made a requirement.

> We suggest that the following should always hold true for a well-formed
> railML file:
>
> a) A globally unique ID (guid) shall be computed and assigned to each
> object's id attribute at the time the object is created.[/quote]
>
> b) The id will never change.

This is the normal usage of UUIDs, but not for file-internal XML ids.
This is normally useful in detailed planning and asset management

> Suggestion 2: tag - unique within one administration's context
> (...)
> Suggestion 3: code - unique within the context of a given station and
> object type
> (...)
> Suggestion 4: name - an ornamented and context-dependent identification
> of an object
> (...)

To me, these suggestions seem to be a perfect match against the following already existing railML2.x attributes (with description from the wiki):

@code (for the suggested @tag): "Machine-interpretable string (e.g. an abbreviation) used for identification of the object across exchange partners, usecase specific uniqueness constraints may apply." In railML3.x one can use the <designator> subelement.

@name (for the suggested @code): Established, human-readable short string, giving the object a name. Usually not intended for machine interpretation. In railML3.x this is a repeatable <name> subelement, so that e.g. names in different languages can be provided.

@description (for the suggested @name): Human-readable, more detailed description as addition

to the name. It should give additional explanations or hints to the contents of this data set. Usually not intended for machine interpretation. In railML3.x this is an attribute of the <name> subelement.

Subject: Re: More standard attributes for objects
Posted by Jörg von Lingen on Sat, 09 Feb 2019 09:10:30 GMT
View Forum Message <> Reply to Message

Dear Claus,

from interlocking schema view:

> Suggestion 1: id - unique within all contexts
Almost every element has to possibility to use the attribute 'id' which is a union of xsd:ID and rail3:tUUID. This will
fulfil your suggestion.

> Suggestion 2: tag - unique within one administration's context
Beside the attribute 'id' one can use the child 'designator' with its mandatory attributes 'register' and 'entry'. This
will fulfil your suggestion.

> Suggestion 3: code - unique within the context of a given station and
> object type
> Suggestion 4: name - an ornamented and context-dependent identification
> of an object
Considering that IL is refering to the physical objects in IS there is no sense to have them in interlocking schema.

Regards,
Jörg von Lingen - Interlocking Coordinator
