Subject: Identification in the XML list files and its references (was: small issues on "register" and "tLineInfrastructureManagerCode")
Posted by Susanne Wunsch railML on Thu, 06 Dec 2012 10:29:03 GMT
View Forum Message <> Reply to Message

Dear Dirk, Christian and others,

I would like to discuss the general content structure of the separate XML list file in the 'misc' forum, because it concerns all sub-schemas.

The separate XML list files should be an easier to maintain replacement for schema-internal enumeration lists.

I will change the already proposed example a bit taking care of the parallel discussion about attribute- or element-centric XML styles. Please don't discuss this issue here. The following XML structure may be easy changed into an attribute-centric one, if that comes as consensus from the neighbouring thread.

```
<registers xmlns="http://www.railml.org/lists">
  <register id="d1e3">
    <version code="ENEE">
      <name>European Railway Location Database</name>
      <validity/>
      <remarks/>
    </version>
  </register>
  <register id="d1e51">
    <version code="RL100">
      <name>Richtlinie 100</name>
      <validity begin="xxxx"/>
      <remarks/>
    </version>
    <version code="DS100">
      <name>Drucksache 100</name>
      <validity begin="1951" end="xxxx"/>
      <remarks/>
    </version>
    <version code="DV100">
      <name>Dienstvorschrift 100</name>
      <validity end="1951"/>
      <remarks/>
    </version>
  </register>
</registers>
```

* Each register would be identified by a file-wide unique 'id' attribute.

* Each version of a register would be identified by a domain-specific
  'code' attribute. This is not a unique string.

The purpose of this list is to constraint the strings for the same
register name, e.g.

  recommend "RL100",
  not to use "Ril100", "KoRil100", "Ril 100"...

But how to use it?

For the following code snippets the <ocp> context is assumed.

* Should a railML file be meaningful without this list file?

  That would mean to refer to the /meaningful/ 'code' value, that is not
  unique.

  <designator register="RL100" entry="..."/>

* Should a railML file be meaningful only with knowledge of the list
  file, only in cases, where its attributes are used?

  That would mean to refer to the _unique_ but not /meaningful/ 'id'
  value.

  <designator registerRef="registers.xml#d1e51" entry="..."/>

* Should both possibilities be provided? If the list file is present, it
  may be looked up for further details, if not, the value is
  /meaningful/ anyway.

  That would mean to refer to both values.

  <designator register="RL100" registerRef="registers.xml#d1e51" entry="..."/>

Any comments, questions, concerns, +1, -1 ... appreciated.

Kind regards...
Susanne

Crosspost & Followup-To: railML.misc

--
Susanne Wunsch
Schema Coordinator: railML.common

Subject: Re: Identification in the XML list files and its references
Posted by on Mon, 10 Dec 2012 15:49:08 GMT
View Forum Message <> Reply to Message

Dear Susanne and all others,

> The separate XML list files should be an easier to maintain replacement
> for schema-internal enumeration lists.

+1
I think it is a compromise between a free string and an XML enumeration type.

> * Should a railML file be meaningful without this list file?
>
> That would mean to refer to the /meaningful/ 'code' value, that is not
> unique.

Between the contradictory aspects of "uniqueness" and "easy to read at text file level", I would prefer a technical uniqueness in case of doubt. This means: Rather refer to /id/ than to /code/ in favour of consequence.

If anybody wants to create an "easy to read" XML file he may use a meaningful /id/ for test purposes.

> * Should a railML file be meaningful only with knowledge of the list
> file, only in cases, where its attributes are used?
>
> That would mean to refer to the _unique_ but not /meaningful/ 'id'
> value.

Again the same answer from my side.

> * Should both possibilities be provided? If the list file is present, it
> may be looked up for further details, if not, the value is
> /meaningful/ anyway.
>
> That would mean to refer to both values.

> <designator register="RL100" registerRef="registers.xml#d1e51" entry="..."/>

As we have already agreed to avoid redundancies as far as possible, this should not be an option. What should a reading software do if there are to contradictory references at one element? I. e. an <ocp> refers to a <register> using a /registerRef/ but the attribute /register/ of the <ocp> is different than the appropriate attribute in the additional XML file?

I think RailML is more a technical data exchange format than a text format for intuitive reading. So, the readability has to step back in cases of doubt. There is still the possibility to create "easy readable" files using meaningful /id/s. This possibility is good, but it should be the left to explicit test cases. For all-day work, interoperability should be the main aim.

With best regards,
Dirk.

Subject: Re: Identification in the XML list files and its references
Posted by Christian Rahmig on Fri, 04 Jan 2013 16:58:31 GMT
View Forum Message <> Reply to Message

Dear Susanne,

Am 06.12.2012 11:29, schrieb Susanne Wunsch:
> Dear Dirk, Christian and others,
>
> I would like to discuss the general content structure of the separate
> XML list file in the 'misc' forum, because it concerns all sub-schemas.
>
> The separate XML list files should be an easier to maintain replacement
> for schema-internal enumeration lists.
>
> I will change the already proposed example a bit taking care of the
> parallel discussion about attribute- or element-centric XML
> styles. Please don't discuss this issue here. The following XML
> structure may be easy changed into an attribute-centric one, if that
> comes as consensus from the neighbouring thread.
>
> <registers xmlns="http://www.railml.org/lists">
> <register id="d1e3">
> <version code="ENEE">
> <name>European Railway Location Database</name>
> <validity/>
> <remarks/>
> </version>
> </register>
> <register id="d1e51">
> <version code="RL100">
> <name>Richtlinie 100</name>
> <validity begin="xxxx"/>
> <remarks/>
> </version>
> <version code="DS100">
> <name>Drucksache 100</name>

> <validity begin="1951" end="xxxx"/>
> <remarks/>
> </version>
> <version code="DV100">
> <name>Dienstvorschrift 100</name>
> <validity end="1951"/>
> <remarks/>
> </version>
> </register>
> </registers>

I agree with that structure, but what about Dirk's remark that there hasn't been any naming brake from DS100 to RL100. In order to keep the disjunctive relation, we cannot leave the dates empty although we do not know them.

However, if the OCP only refers to the "not readable" ID of the register entry, its versions do not have to be disjunct since they just define different versions of the same register. Depending on the user, when refering to "d1e51" he/she may think of the "DV100" or the "DS100" or "RL100", but this does not affect the data exchange or the refered ID in particular. Consequently the element <validity> could be removed at all.

> [...]
>
> * Should both possibilities be provided? If the list file is present, it
> may be looked up for further details, if not, the value is
> /meaningful/ anyway.
>
> That would mean to refer to both values.
>
> <designator register="RL100" registerRef="registers.xml#d1e51" entry="..."/>

That is not a good solution. The reference to the ("not readable") register's ID should be enough.

Regards

--
Christian Rahmig
railML.infrastructure coordinator