Subject: Some problems with/questions about the infrastructure schema...
Posted by Wolfgang Keller on Wed, 16 Jun 2004 11:03:21 GMT
View Forum Message <> Reply to Message

Hello,

as a total XML illiterate I'm currently trying to build a database (ER) schema for timetable and other operation data based on the RailML XML schemas. During this work, I ran into some problems/unclear points/potential improvements:

(My work is based on the files
http://www.linder-rail-consult.de/railml/docs/infrastructure /v094/Infrastructure_V095_Concept_DE_Rel1_2004-03-28.pdf
http://www.linder-rail-consult.de/railml/docs/infrastructure /v094/Infrastructure_V094_Reference_EN_2004-03-28.pdf
and
http://www.linder-rail-consult.de/railml/docs/infrastructure /v094/infrastructure_V094_18.xsd)

1. Generally I propose that identifiers such as "type", "length", "value" etc. should not be used at all, as they risk to collide with typical reserved keywords. It would be best imho if all identifiers were unique within the entire schema in order to avoid confusion.

2. Why are the <x>ChangeType definitions not based on the corresponding <x>Type definitions (by reference, inclusion of a subelement or whatever method)? This would imho avoid redundancies as well as incoherencies. One of these incoherencies appears to be that the electrificationChangeType definition in the .xsd file contains vMax and isolatedSection attributes, while the electrificationType definition does not.

3. The part of the schema about connections appears to be especially "unstable" at the moment. When can a somehow "settled" version of this part be expected? It also appears to me as a not-database-developer that this part is not obvious to map to a relational schema due to the "kinks" in the relationships, which leads to the next point:

4. Wouldn't it be useful/would it be impossible to include such considerations as technology-independence in the design of the schema, so that the logical structuring can also be used for plain-ASCII data exchange (such as datagrams sent over narrow-bandwidth wireless connections etc.), for relational databases and maybe also other implementations...?

TIA,

Best regards,

Wolfgang Keller

Subject: Re: Some problems with/questions about the infrastructure schema...
Posted by Volker Knollmann on Wed, 16 Jun 2004 15:49:26 GMT
View Forum Message <> Reply to Message

Wolfgang Keller wrote:
> 1. Generally I propose that identifiers such as "type", "length", "value"
> etc. should not be used at all, as they risk to collide with typical
> reserved keywords.

Do you speak of reserved keywords within your database-dialect (SQL, etc) or your programming language? If so, it should always be possible to escape such keywords with single / double /triple quotes, backslash and other means. And regarding SQL: I don't think that a statement like

SELECT * from switches WHERE type="ordinarySwitchRight"

causes problems, does it?

> It would be best imho if all identifiers were unique
> within the entire schema in order to avoid confusion.

Hmmm, I would prefer the opposite syntax. For example, I would appreciate a common attribute "elemID" for all elements. Right now, we have "ID", "elemID", "ocpID", "connectionID", etc. "elemID" is used in many children of <trackData>, why not everywhere?

> 2. Why are the <x>ChangeType definitions not based on the corresponding
> <x>Type definitions (by reference, inclusion of a subelement or whatever
> method)?

IMHO, this is a good suggestion for the sake of consistency. A xChangeType could be a combination of xType and something like "positionType", which encapsulates the typical attribute pos, absPos, dir and (occasionally) a track- and line-ID.

> 3. The part of the schema about connections appears to be especially
> "unstable" at the moment.

I plead "not guilty", your honor! ;)

BTW: I spent the last days implementing a fictional example track in RailML and I came across a bunch of missing elements, attributes and structures. I will create a summary of my "problems" and post it to this newsgroup in the next days.

> 4. Wouldn't it be useful/would it be impossible to include such
> considerations as technology-independence in the design of the schema, so
> that the logical structuring can also be used for plain-ASCII data exchange
> (such as datagrams sent over narrow-bandwidth wireless connections etc.),

> for relational databases and maybe also other implementations...?

What exactly do you mean? A formal description like ER-Diagrams or similar?
Imho XML is the best choice for our needs:
* structured and hierachical
* extensible without compatibility problems
* human readable
* can be edited with a simple editor
* supports encodings and complex character sets, but...
* ... can be used with 7-bit ASCII as well (works ALWAYS!)
* no bothering about linefeed, newlines and special characters when exchanging data between different platforms
* can be formaly validated used DTDs or XSDs
* easy to parse, libraries available for every language and platform
* not proprietary at all

I agree, that due to a certain redundancy within the file (opening / closing tags, ...) the file size is not optimal.
But for the transmission over bandwidth- or volume-critical connections, you can apply $FAVOURITE_COMPRESSION_UTILITY to your data and you should get satisfying results...

Best regards,
Volker Knollmann

Subject: Re: Some problems with/questions about the infrastructure schema...
Posted by Wolfgang Keller on Thu, 17 Jun 2004 10:05:52 GMT
View Forum Message <> Reply to Message

Am Wed, 16 Jun 2004 17:49:26 +0200 schrieb Volker Knollmann:

> Wolfgang Keller wrote:
>> 1. Generally I propose that identifiers such as "type", "length", "value"
>> etc. should not be used at all, as they risk to collide with typical
>> reserved keywords.
>
> Do you speak of reserved keywords within your database-dialect (SQL,
> etc) or your programming language?

Reserved keywords in as many contexts as the RailML definitions will be typically used. Both databases and programming languages, and maybe also others.

>> It would be best imho if all identifiers were unique
>> within the entire schema in order to avoid confusion.
>
> Hmmm, I would prefer the opposite syntax. For example, I would

> appreciate a common attribute "elemID" for all elements. Right now, we
> have "ID", "elemID", "ocpID", "connectionID", etc. "elemID" is used in
> many children of <trackData>, why not everywhere?

Easy answer. >:-> For the sake of inobfuscation. Or, why make things more difficult than absolutely necessary. RailML is (considered to be) an open specification. Within bulkloads of code, it's easy to get lost, especially but not only for others than the individual who wrote the code. Instantly figuring out what all that code is about can be pretty difficult when everything is identified by an elemID. However, if you read trackID, you instantly know that the element in question is a track element. The same applies for all other identifiers. Having unique identifiers everywhere avoids lots of mistakes.

>> 4. Wouldn't it be useful/would it be impossible to include such
>> considerations as technology-independence in the design of the schema, so
>> that the logical structuring can also be used for plain-ASCII data exchange
>> (such as datagrams sent over narrow-bandwidth wireless connections etc.),
>> for relational databases and maybe also other implementations...?
>
> What exactly do you mean? A formal description like ER-Diagrams or similar?

No. I just propose to take aspects of technology-independence into consideration in the design process. This does not require actually creating and maintaining a unique relational database (or whatever else) schema, just designing the XML schemas in such a way that it is easily possible to do so if someone wants to.

Just like XML, which was unknown ten years ago, other information representation technologies may come up in the future. If you take care of technology-independence now, you can take advantage of these later. Otherwise, most if not all the work invested may be lost.

> Imho XML is the best choice for our needs:
> * structured and hierachical

It's more "semi-structured", with very loose constraints on the structure, which makes it difficult to map XML schemas to more restrictively structured representations (such as for example, relational database schemas, but probably also others).

> I agree, that due to a certain redundancy within the file (opening /
> closing tags, ...) the file size is not optimal.
> But for the transmission over bandwidth- or volume-critical connections,
> you can apply $FAVOURITE_COMPRESSION_UTILITY to your data and you should
> get satisfying results...

Nope, sorry. The experts who are working on such topics know why they are

doing things they way they do them. And they don't use zipped XML for their datagrams.

Under different conditions, different information representation technologies have different advantages. If you want to allow for automatic processing of data without extensive handwork, then you will have to take this into account.

Best regards,

Wolfgang Keller

Subject: Re: Some problems with/questions about the infrastructure schema...
Posted by Volker Knollmann on Fri, 18 Jun 2004 09:08:47 GMT
View Forum Message <> Reply to Message

Wolfgang Keller wrote:
> Am Wed, 16 Jun 2004 17:49:26 +0200 schrieb Volker Knollmann:
>> Hmmm, I would prefer the opposite syntax. For example, I would
>> appreciate a common attribute "elemID" for all elements. Right now, we
>> have "ID", "elemID", "ocpID", "connectionID", etc. "elemID" is used in
>> many children of <trackData>, why not everywhere?
>
> Within bulkloads of code, it's easy to get lost, especially
> but not only for others than the individual who wrote the code. Instantly
> figuring out what all that code is about can be pretty difficult when
> everything is identified by an elemID. However, if you read trackID, you
> instantly know that the element in question is a track element. The same
> applies for all other identifiers. Having unique identifiers everywhere
> avoids lots of mistakes.

Sorry again, but your arguments don't convince me ;)

If you are reading a XML-file, you can immediately figure out with which kind of element an "elemID" is associated - just take a look at the tag the "elemID" belongs to:

<track elemID="42" ....>,
<ocp elemID="kjdf" ....>,
<switch elemID="666" ...> and so on.

On the other hand if you use different names (trackID, switchID, trackRefID, lineID, branchID, whateverID) for the same thing (a ID), you get puzzled while writing XML or a parser. You always have to take look at the reference what the exact name of the ID-attribute for a certain element is. If you only have to remember "elemID", things are a lot easier and less vulnerable to mistakes.

> Just like XML, which was unknown ten years ago, other information
> representation technologies may come up in the future. If you take care of
> technology-independence now, you can take advantage of these later.
> Otherwise, most if not all the work invested may be lost.

How could a technology-indepent representation of the data be realized?
I can only think of diagrams like UML, ER or other formal methods (and
UML itself was unknown some years ago as well as XML).
And don't forget: as soon as you what to actually _USE_ the language /
structure / schema / system you invented, you have to make a decision
for a certain technology. AFAIK, railML had always the intention to be
applicable and usable from the beginning. And XML is a good compromise
between instantly usable technology on the one hand and a structured
describtion on the other hand.

>> I agree, that due to a certain redundancy within the file (opening /
>> closing tags, ...) the file size is not optimal.
>> But for the transmission over bandwidth- or volume-critical connections,
>> you can apply $FAVOURITE_COMPRESSION_UTILITY to your data and you should
>> get satisfying results...
>
> Nope, sorry. The experts who are working on such topics know why they are
> doing things they way they do them. And they don't use zipped XML for their
> datagrams.

"Datagrams"? That sounds like dynamic data (e. g. track occupation,
current train speed, etc) which is send periodically between network
nodes. Indeed, compressed XML is not the first choice for such purposes
(although I would like to hear the arguments of "the experts who are
working on such topics").
railML focuses on _STATIC_ data like infrastructure and timetables. And
this static information is not as time- and volume-critical as dynamic
process data which you need to update periodically or in "realtime"
(whatever "realtime" means in your context).

> Under different conditions, different information representation
> technologies have different advantages. If you want to allow for automatic
> processing of data without extensive handwork, then you will have to take
> this into account.

See above: railML's primary intention is the platform- and
vendor-independent representaion of static railway data. And for this
purpose, I still think XML is a good choice. I definitely prefer parsing
a structured text file over fiddling around with multiple binary tables
of a relational database for instance.
For XML, various libraries for various languages and platforms are
available. They simply work. And the file itself can be reduced to use

7-bit ASCII only. It simply works. Everywhere. Even with slow telnet-connections, VT100-Terminals or a Z80-machine.
Have you in contrast ever tried to access a MySQL-Server on Linux from a Windoze-Workstation and MS-Access via ODBC and SQL? I was close to biting my keyboard and sending a letter bomb to Redmond, USA. To cite Murphy: some systems are more compatible than others. You _NEVER_ have this kind trouble with XML. This is why I really like it.

Best regards and a nice weekend,
Volker Knollmann

Subject: Re: Some problems with/questions about the infrastructure schema...
Posted by Matthias Hengartner on Fri, 18 Jun 2004 12:08:45 GMT
View Forum Message <> Reply to Message

Hello,

For the moment I have only one comment regarding the discussion around the serveral ID-attributes.

> Hmmm, I would prefer the opposite syntax. For example, I would
> appreciate a common attribute "elemID" for all elements. Right now, we
> have "ID", "elemID", "ocpID", "connectionID", etc. "elemID" is used in
> many children of <trackData>, why not everywhere?

At least the attributes "ocpID" and "trackID" ar meant to be "close to reality", so they should represent the identifiers which are also used in reality. I don't know which elements of <trackData> and <trackTopology> do have also such an identifier in practice, but certainly not all of them (e.g. <trackBegin>/<trackEnd>, probably also <crossSection> have none).

So we have to deal with different types of IDs.

Further there was a discussion about globally unique IDs (GUIDs), which has still not come to an definite end.

Roughly, there are two possibilities:

1. We introduce GUIDs.
Then I think we should leave "real-ID-attributes" like ocpID and trackID (and keep also this names) and discard other ID-attributes with no reference to reality.

2. We do not introduce GUIDs.
So we surely keep the ID-attributes. And I think we should also keep the naming of most of them. Personally, I find it a little confusing to have a

connectionID and not an elemID in a <bufferstop>-element.

So far for the moment.

Have a nice weekend!
Matthias Hengartner
