Subject: To use or not to use? - default values in XSDs
Posted by on Mon, 08 Dec 2014 09:40:08 GMT
View Forum Message <> Reply to Message

(English below)

Liebe Mitstreiter,

In "trac-"Fahrkarten liest man gelegentlich:
Please, don't apply default values to attributes.

Scheinbar besteht hierzu schon seit längerem ein Glaubenskrieg. Ich versuche mal eine Moderation, bevor hier durch vollendete Tatsachen Schaden angerichtet wird.

Die Befürworter von Standard-Werten sehen darin wohl eine vermeintlich elegante Möglichkeit, die Größe der RailML-Daten zu reduzieren – worauf ich die Gegner sagen höre, dass XML eines der schlechtesten Formate für reduzierte Größe ist… Wenn es hier darum ginge, Platz zu sparen, hätte man besser gar kein XML als Basis genommen.

Die Bedenken der „Standard-Gegner“ hingegen sind wohl, dass durch Definition von Standardwerten die Möglichkeit genommen wird, durch Weglassen des Attributs auszudrücken, dass man den Wert schlichtweg nicht kennt. Ließe man ein Attribut mit Standard-Wert weg, würde ja automatisch der Standard-Wert gelten. Es wäre dann nicht mehr unterscheidbar, ob der Standard-Wert tatsächlich gelten soll oder doch eher „unbekannt“.

Diese Bedenken erscheinen völlig gerechtfertigt – aber doch nur, wenn „unbekannt“ überhaupt in Frage kommt!

Wie wäre es denn damit:
*1. Bei Attributen, die semantisch auch unbekannt sein können, sollten Standard-Werte tatsächlich vermieden werden.
*2. Aus Attributen, die semantisch Pflichtfelder sind, können durch Standardwerte optionale Attribute werden – hier kommt „unbekannt“ ja sowieso nicht in Frage.

Standardwerte können danach durchaus zulässig sein, nämlich dort, wo sie keinen Schaden anrichten können. Sehr sinnvoll und empfehlenswert sind sie, wo ein semantisches Pflichtfeld mit hoher Wahrscheinlichkeit/Häufigkeit einen bestimmten Wert annimmt, also praktisch nur selten vom Standard-Wert abweicht.

Ein gutes Beispiel sind die Attribute <ocpTT>.arrival/departureDay: Sie sind optional mit default-Wert 0, und das ist gut so. Sie nicht anzugeben heißt eben: garantiert =0, und garan-tiert nicht: „Ich weiß

nicht, ob der Zug über Mitternacht gefahren ist“.

Hier haben wir ein semantisches „Muss-Attribut“ (ein schreibendes Programm muss wissen, ob und wann sein Zug über Mitternacht fährt), das durch Definition eines default-Werts (hier: 0) zu einem optionalen Attribut geworden ist. Es würde die Lesbarkeit der RailML-Daten unnötig erschweren, wenn solche Attribute wie arrival/departureDay keinen default-Wert hätten und man sie daher an jeder Betriebsstelle wiederholen müsste, obwohl die meisten Züge gar nichts mit Mitternachtsübergängen am Hut haben. Und schließlich ist Lesbarkeit (im Gegensatz zu Platzersparnis) der einzige Grund, warum wir uns für ein _text_basiertes Dateiformat entschieden haben.
(Für inhaltlich Interessierte:
http://www.wiki.railml.org/index.php?title=TT:Midnight_overr un).

Man könnte vielleicht noch anführen, dass man „unbekannt“ immer auch durch einen expliziten Wert (für „unbekannt“) ausdrücken kann. Das wäre dann allerdings auch nicht gerade platzsparend und bei Integer/Dezimal-Werten denkbar unpraktisch. Und vor allem: Es wird bei <Elementen> nicht konsequent durchzuhalten sein. Bei <Elementen> werden wir wohl immer „unbekannt“ (auch) durch Weglassen ausdrücken. Dann sollten wir m. E. bei Attributen nicht anders vorgehen. Aber wir könnten ja Kompromisse machen:

*3. Wenn ein sehr selten benutztes Attribut (also eines, das mit hoher Wahrscheinlichkeit zum Standardwert tendiert) leider kein semantisches Pflichtfeld ist, kann trotzdem ein Standard-Wert definiert werden – wenn gleichzeitig ein expliziter Wert für „unbekannt“ definiert wird.

Das würde dann allerdings bedeuten, dass der Standardwert deutlich häufiger vorkommt (wahrscheinlicher ist) als „unbekannt“ – und das will gut begründet sein.

Hier ein Beispiel: Nehmen wir einmal an, es gäbe ein Element <vehicleBrake> mit den Attributen „brakeType“ und „brakeMass“. Das Attribut „brakeType“ sei von einem Aufzählungstyp und könne die Ausprägungen „unknown“, „compressedAir“ und „vacuum“ annehmen. Mal Hand
auf’s Herz: Wer hat heute noch Saugluftbremsen? Oder anders gefragt: Wozu bitte soll jemand, der eine (Museumsbahn mit) Saugluftbremse hat, RailML brauchen? Und dass jemand die Bremsmasse kennt, aber nicht die Bremsbauform, dürfte sehr unwahrscheinlich sein. Wer die Bremsen nicht kennt, soll das ganze Element weglassen. Also könnte man hier vielleicht wirklich einmal den Standardwert auf „compressedAir“ setzen: Wenn das Element verwendet wird, dann nahezu ausschließlich mit „compressedAir“.

Viele Grüße,
Eurer Dirk.

Dear all,

Sometimes one reads in forum posts lines as such:
Please, don't apply default values to attributes.

Apparently there is a kind of holy war on that for a long time. Let me try mediation before a fait accompli is created.

It seems to me that the supporters of default values suppose an elegant possibility in them to reduce the size of railML data. Whereupon I hear the opponents say that XML is one of the worst formats for reduced size… If this would be about saving space, one should rather not have chosen XML as a basis.

On the contrary, the concerns of the opponents of ‘defaults’ seem to be that with them there is no possibility to express “unknown” anymore by omitting the attribute. If one would omit an attribute which has a default value, the default value would come into effect automatically. It would not be possible to find out whether the default value is actually meant or rather “unknown”.

These concerns seem to be entitled - but only if “unknown” comes into consideration at all!

How about this:
*1. Attributes which could be „unkown“ by semantics shall indeed have no default value.
*2. Attributes which are mandatory by semantics can be made optional by default values - here “unknown” cannot be possible anyway.

So, default values may absolutely be allowed – namely where they cannot do harm. They are very reasonable and recommended where an attribute mandatory by semantics has a certain value with a high probability/occurrence, so very seldom differing from the default value.

Good examples are the attributes <ocpTT>.arrival/departureDay: They are optional with default value 0, and quite right so. To omit them means surely =0 but means surely not “I don’t know whether the train did run over midnight”.

Here we have a mandatory attribute by semantics: A writing software has to know whether and when its train runs over midnight. This mandatory attribute has become optional by the default value. It would complicate the readability of a railML file unnecessarily if such attributes wouldn’t have default values and one would have to repeat them at each station. And most of the trains do not even run at midnight! And,

readability is (in contrast to saving space) the only reason why we could have chosen a _text_ based file format.
(More on the contents at
http://www.wiki.railml.org/index.php?title=TT:Midnight_overr un).

On could possibly add that „unknown“ could be expressed by a certain explicit value (a value meaning “unknown”). But this would by far not being space-saving. And impractical with integer/decimal values. And last but not least: It could probably not be done with <elements>. I guess we will have to express “unknown” for <elements> by omitting them too. So why do it different for attributes? But we can make compromises:

*3. If, for a seldom used attribute which is unfortunately not a mandatory one, a default value can be defined anyway - if an explicit value for “unknown” is defined as well.

This would mean that the default value is much more probable than “unknown”, and this will to be justified very well.

Example: Let’s assume there is an element <vehicleBrake> with the attributes “brakeType” and “brakeMass”. The attribute “brakeType” shall be an enumeration with the possible values „unknown“, „compressedAir“ and „vacuum“. But who has still vacuum brakes nowadays? And it is highly improbably that somebody knows the baking mass but does not know the brake type. If someone does not know the brakes at all, he should omit the whole element. So: Here we could possibly define the default value “compressedAir” indeed.

So:
Please, don't write general sentences on default values! ;-)

Best regards,
Dirk.

Subject: Re: To use or not to use? - default values in XSDs
Posted by on Fri, 09 Jun 2017 09:03:42 GMT
View Forum Message <> Reply to Message

Dear all,

the matter of default values was again raised during <TT> developer meeting on 1st June 2017 in Berlin. The result from the <TT> developers is:

- In general, default values may be used in railML.

- The default values must not be country-specific.

- Each default value is an individual, isolated case and has to be documented concerning its meaning and function (at least in Wiki). There is no rule such as "default means unknown".

Typical examples for reasonable default values - which do NOT mean "unknown" - are:
- <TT>..<times> @arrivalDay=0 / departureDay=0
- <TT>..<ocpTT> @guaranteedPass=false

A fine example for a context-sensitive default value is:
- <IS>..<speedChange> @trainRelation: =headOfTrain when effectively decreasing, =endOfTrain when effectively increasing permitted speed, whereas increasing/decreasing can possibly only be determined from merging different speed profiles.

With best regards,
Dirk.
