Subject: Re: [Request for railML3] Different station tracks in one <ocpTT> for different <operatingPeriod>s
Posted by on Tue, 22 May 2018 13:42:37 GMT
View Forum Message <> Reply to Message

Dear all,

since quite some time, there is an open demand for a functional “add-on” of railML <timetable>: It should be allowed to encode different platforms at different operating days of one <trainPart> without splitting it into two or more <trainPart>s.

The matter has been discussed lengthy during the last <TT> developer meetings and telephone conferences. The pros and cons of splitting <trainPart>s vs. introducing new elements have been discussed.

A short summary of the main cons is:
- <trainPart>s are intended to be the basic atom of trains and therefore should not be divided furtherly.
- There is already a solution for the problem by using more than one <trainPart>. Any additional solution would be a redundancy.
- This redundancy leads by tendency to higher effort and therefore higher costs for import interfaces (if there would be a need to support all possibilities).

A short summary of the main pros is:
- It seems to be state of the art to enumerate different platforms at _one_ train since several common software programs show this feature.
- There are already other sub-elements of <trainPart> with an operatingPeriodRef.
- For passenger information, it may be a too demanding effort to re-merge information in an import interface which only have been splitted before in an export interface. This applies especially for passenger information with “aggregated” information over more than one operating day.
- This additional effort leads by tendency to higher costs and possibly lower acceptance of railML at our customers where nobody is interested in.

However, at the last <TT> developer meeting on 19.04.2018 at Berlin there has been a suggestion as a compromise. This would allow to enumerate several <trackInfo> (working title) elements at <trainPart>.<ocpTT>.<stopDescription>. Each <trackInfo> would have one @operatingPeriodRef and one @description. All such @operatingPeriodRef’s must be disjunctive and must cover (but not exceed) the @operatingPeriodRef of the <trainPart>.

It should be mentioned that this is a minimum solution which in any case needs a usage description or use case. Therefore, concerning the main “con”, in my opinion, there is no general need to support all possible (redundant) solutions in one import interface. For instance, we (iRFP) will support both technologies on export and therefore, allow the import partner to select the best solution for it’s demand. So, we regard this redundancy rather as more flexibility and better acceptance of railML.

The new “add-on” is up to be decided for railML 2.4 in near future. An example and suggestion for the scheme change can be found in the Wolke at

https://cloud.railml.org/remote.php/webdav/TT%20working%20group/railML%202.4/Vorschlag%20Saisonierte%20Gleisbelegung.pdf

With best regards,
Dirk.
