Subject: [railML2] Extensions of triggers for passenger information within trains
Posted by Thomas Kabisch on Thu, 11 Mar 2021 16:42:13 GMT
View Forum Message <> Reply to Message

(ENG below)

Hallo,

als weitere Ergänzung für die Fahrgastinformation im Zug folgt hier die Beschreibung der Anforderungen und eines Modellierungsvorschlags für die Erweiterung des Trigger-Konzepts.

Anforderungen
Das Fahrgastinformationssystem im Zug soll Informationen gezielt in Abhängigkeit vom Eintreten bestimmter Ereignisse während der Fahrt präsentieren, beispielsweise sollen die Anzeigen und Ansagen "nächster Halt" 1000m vor vor dem Erreichen des Haltes ausgelöst werden.
Ein Fahrtverlauf kann dabei als eine Folge von Fahrtzuständen relativ zu den passierten Halten (bspw. "in Annäherung auf einen Halt", "direkt vor Halt", "am Halt", "nach Abfahrt", "nach Verlassen" und "Fahrt auf freier Strecke") verstanden werden.
Wechselt das Fahrzeug zwischen zwei derartigen Zuständen ist dieser Wechsel ein Auslösepunkt/Triggerpunkt.
Das häufigste Modell für die Umsetzung dieser Anforderungen ist die Defintion von sog. Sichtbarkeitsbereichen oder "Fenstern" konzentrisch um die einzelnen Halte (bspw. äußerer Bereich, innerer Bereich, Fangbereich).
Grundsätzlich soll sowohl eine allgemeine Definition der Sichtbarkeitsbereiche für alle Halte (als Default-Wert) sowie eine spezifische pro Halt möglich sein.
Für die Definition der Sichbarkeitsbereiche sollen verschieden Modelle verwendet werden können:
Für eine GPS-basierte Ansteuerung sollen GPS-Punkte begrenzte Polygone oder Kreisradien definierbar sein.
Für eine odometrische Ansteuerung sollen Distanzen verwendet werden können.
Für eine zeitbasierte Ansteuerung sollen Zeitoffsets (relativ zu Haltezeiten) modelliert werden können.

Modellierungsvorschlag
Die vorgestellte Modellierung baut wiederum auf den Konzepten der Fahrgastinformation am Bahnhof auf. Zentral ist hier das Konzept <trigger>. Die Sichtbarkeitsbereiche erweitern die Trigger um einen weiteren Triggertyp <scopeTrigger> welche auf einen Sichtbarkeitsbereich referenziert. Es wird vorgeschlagen, die Strukturdefintion der Sichtbarkeitsbereiche (<scopes>) unabhängig von der eigentlichen Trigger-Modellierung durchzuführen, da erstere nur von Infrastrukturelementen (i.d.R. Halten) abhängig sind und eine Modellierung bspw. unterhalb von <trainPart> eine hohe Redundanz zur Folge hätte.

Für die Definition der Sichtbarkeitsbereiche selbst wird folgende Modellierung vorgeschlagen:
Die Struktur der Bereicheslogik wird im neuen Element <scope> abgelegt welches die Attribute
@id : Identifier zur Referenzierung im Trigger
@type: Art des Bereichs, Enum (outer, inner, snap)

@ocpRef: Referenz zu einem <ocp> (Halt), kann entfallen wenn Modellierung direkt am <ocp>

Innerhalb des Elements <scope> werden die Detaildefinitionen abgelegt. Aufgrund der unterschiedlichen Attribute je nach Art der Bereichsdefinition werden verschiedene Detail-Elemente vorgeschlagen:
1) <scopeDistance> Distanz-Bereiche: Attribut @value für die Wegmeterangabe
2) <scopeRadius> Kreisförmige Bereiche: Attribut @value für die Angabe des Radius
3) <scopeRectangle> Rechteckige Bereiche: Angabe von zwei Eckpunkten als Geo-Koordinaten
4) <scopePolygon> Polygonbereiche: Angabe von 3..n Eckpunkten als Geo-Koordination
5) <scopeTime> Zeitbasierte Bereichsdefinition: Attribut @offset als Minutenangabe

Für die Platzierung der Bereichsdefinitionen werden zwei Modellierungsvarianten zur Diskussion gestellt:
i) als zentrale Liste am Beginn von Infrastruktur- oder Timetable-Schema, oder
ii) alternativ aufgeteilt auf allgemeine Sichtbarkeitsbereiche (zentrale Liste) und spezifische Sichtbarkeitsbereiche unterhalb der <ocp>.
Da die Sichtbarkeitsbereiche primär Halten (<ocp>) zugeordnet werden können, wird die Variante ii) bevorzugt.

Beispiel folgt nach dem englischen Text.

(ENG)

Refinement of trigger-concept by station-specific scopes

Requirements
The passenger information system within a train distributes information with respect to particular events during the journey.
For example, an interior information display should activate the "next station"-screen 1000 metres before the train arrives at the station. We may understand a journey as sequence of consecutive train-states that reflect the position of the train relatively to the passed stations, for example "approaching stop", "directly before stop", "stopped", "departed", "shortly after departure", "leaving station", "beteen stations".
A transition between two consecutive states is a trigger point for the PIS. The most common model for the implementation of these requirements is the definion of concentric so called scope-windows surrounding a station such as "outer scope", "inner scope", "snap scope" of the station. Principially such scopes should either be globally valid (for all stations) or specifically for a particular station.

Different options for the definition of scopes are required:
- GPS-Polygons for a GPS-based triggering
- distances for a odometric triggering
- time offsets for a triggering based on the timetable

Modelling suggestion
The introduced modelling approach builds up on concepts of the already introduced use case

passenger information at stations.
The central concept is <trigger>.
Scopes extent trigger by an additional trigger type <scopeTrigger>. This trigger type mainly references to a scope. We suggest a separated modelling of scopes and triggers. Scopes depend only on infrastructure elements, thus a modelling within <trainpart> will lead to a high degree of redundancy.

The new suggested element <scope> will have the following attributes:
@id : Identifier for later referencing at <trigger>
@type: Type of scope, Enum (outer, inner, snap)
@ocpRef: Reference to <ocp>, may be ommited if defined directly below <ocp>

Within <scope> the detail definitions are stored.
We suggest multiple detail elements due to the different structures of them:

1) <scopeDistance> odometric scopes: attribute @value for distance in metres
2) <scopeRadius> cirlic scopes: attribute @value for definition of radius
3) <scopeRectangle> rectangular scopes: notation of two geo-coordinates
4) <scopePolygon> polygon scopes: notation of 3..n corner points as geo-coordinates
5) <scopeTime> time based scopes: attribute @offset on a minute-base

We would like to suggest two alternative locations within the overall RailML-Schema for further discussions:

i) centralized: s single list at the beginning of either infrastrukture- or timetable-schema
ii) separted: general scopes in a central list and specific scopes within <ocp>.

Due to the high dependency of scopes to <ocp> we prefer option ii).

Beispiel / Example

```
<!-- Beispiel-Ocp mit Bereichsangaben -->
<ocp code="HH" id="ocp_HH" name="Hannover Hauptbahnhof">
 <scopes>
  <!-- Fangbereich als Radius -->
  <scope id="sc_HH_snap" type="snap">
   <scopeRadius value="75"/>
  </scope>
  <!-- Innerer Bereich als Rechteck -->
  <scope id="sc_HH_inner" type="inner">
   <scopeRectangle>
    <geoCoord coord="50.376762 10.741021"/>
    <geoCoord coord="52.536462 9.741021"/>
   </scopeRectangle>
  <!-- Äußerer Bereich als Zeitoffset -->
```

```
  <scope id="sc_HH_outer" type="outer">
   <scopeTime value ="10min"/>
  </scope>
 </scopes>
 ...
</ocp>
...
<!-- Auslösesteuerung einer Ansage mit Scope-Trigger (bspw. direkt innerhalb der ocp oder im trainPart) -->
<announcementRef ref="announce-2019">
 <trigger>
  <scopeTrigger event = "arrival" scopeRef="sc_HH_inner"/>
 </trigger>
</announcementRef>
```