



VRIJE
UNIVERSITEIT
BRUSSEL



PROGRAMMEERPROJECT 2

Documentatie fase 3

Gérard Lichterttt
gerard.Lichtert@vub.be
0557513

9 mei 2021

Inhoudsopgave

1	Inleiding	2
2	Functionele vereisten	2
3	ADT's	3
3.1	Track ADT	3
3.1.1	Toelichting	3
3.2	Switch ADT	3
3.2.1	Toelichting	4
3.3	Dblock ADT	4
3.3.1	Toelichting	4
3.4	Locomotive ADT	5
3.4.1	Toelichting	5
3.5	Railway ADT	5
3.5.1	Toelichting	5
3.6	Infrabel ADT	5
3.6.1	Toelichting	5
3.7	Gui ADT	5
3.7.1	Toelichting	5
3.8	NMBS ADT	5
3.8.1	Toelichting	5
4	Afhankelijkheids diagram	5
5	Beschrijving van API tussen infrabel- en NMBS-component	5
6	Planning	5
7	Logboek	5

1 Inleiding

Dit document bevat het verslag van fase 3 van het vak "Programmeerproject 2". Het behandelt de eerste functionele vereisten en de 3 uitgebreide vereisten. Vervolgens zal het over de gebruikte datastructuren gaan en het afhankelijkheids diagram. Deze datastructuren zullen er voor zorgen dat alle functionaliteiten werken en vanuit de GUI aangeroepen kunnen worden. Verder staat in dit verslag een beschrijving van de API tussen de infrabel- en NMBS-component, de planning en het logboek.

2 Functionele vereisten

In dit verslag worden de functionele vereisten per fase besproken. In de eerste fase werd de GUI gemaakt alsook de command and control. De code van de command en control moeten we zelf niet schrijven maar de code die de hardware aanstuurt wel. De code moet locomotieven kunnen laten starten en stoppen, hun snelheid en rijrichting aflezen en veranderen. Verder moeten we ook de de stand van de wissels kunnen uitlezen en verzetten. Via de detectieblokken detecteren we waar een trein zich bevindt.

De GUI laat de eindgebruiker de toestand van het spoornetwerk en de locomotieven zien. Het laat ook interactie toe met de wissels en locomotieven.

In de tweede fase was het de bedoeling dat we de eerste twee uitgebreide vereisten implementeren. Voor dit project werd gekozen voor botsingpreventie aan de hand van een reservatie en bezettingssysteem. Dit zal ervoor zorgen dat een trein zijn snelheid op nul gezet wordt als het pad dat die gaat afleggen al gereserveerd of bezet is. Aangezien de stand van de wissels op eender welk moment gewijzigd kan worden, worden er veel spoorsegmenten gereserveerd tot de aanliggende detectieblokken. Dit is wel met de voorwaarde dat ze op het pad zijn van de richting van de trein. Een andere uitgebreide vereiste waar voor gekozen werd is het automatisch trajectbeheer. De manier waarop dit geïmplementeerd is dat de eindbestemming ingevoerd wordt en het pad berekend word. Dit wordt slechts gedaan als de trein detecteerbaar is. Anders zal het pad nog eens herberekend worden wanneer die detecteerbaar is. Er wordt bij het trajectbeheer ook rekening gehouden met het reservatiesysteem om botsingen te voorkomen.

In de 3e fase is het de bedoeling dat we de laatste uitgebreide vereiste implementeren. Bij dit project is dit de Raspberry Pi vereiste. Dit houdt in dat het mogelijk moet zijn om infrabel te runnen op de Raspberry Pi en NMBS op de computer. Dit wordt behaald door het externe IP adres (of lokaal) mee te geven bij het aanmaken van het NMBS object.

3 ADT's

3.1 Track ADT

Het track ADT houdt bij welke spoorsegmenten verbonden zijn. Het houdt ook bij of het spoorsegment al dan niet gereserveerd is. De reservatiestand kan ook aangepast worden.

Naam	Signatuur	Beschrijving
new	(symbol, list \rightarrow track%)	Maakt een track object aan. Verwacht de ID van het spoorsegment en een lijst met de ID's van de verbonden spoorsegmenten.
get-track-id	(/ \rightarrow symbol)	Geeft de ID van het spoorsegment terug.
track-links	(/ \rightarrow vector)	Geeft de vector terug met de ID's van de verbonden spoorsegmenten.
set-links!	(list \rightarrow /)	Verandert de verbonden spoorsegmenten.
links-map	((symbol \rightarrow any) \rightarrow vector)	Voert een procedure uit op de ID's van de verbonden spoorsegmenten en geeft de opgespannen vector terug.
reserved?	(/ \rightarrow symbol \cup false)	Geeft de ID van de locomotief die het spoorsegment gereserveerd heeft of false.
reserve!	(symbol \rightarrow /)	Verandert de reservatiestatus naar het meegegeven ID.
cancel-reservation!	(/ \rightarrow /)	Zet het reservatiestatus op false
nr-of-links	(/ \rightarrow integer)	Geeft het aantal verbonden spoorsegmenten terug

Tabel 1: Signaturen van track%

3.1.1 Toelichting

Het track ADT is eigenlijk een superklasse van de volgende ADT's. Dit is zodat de volgende ADT's de methoden erven van het track ADT. New maakt een nieuwe track ADT aan. Dit kan ook vervangen worden door (make-object track% < argumenten >). *get-track-id* dient om de ID op te vragen. *track-links* dient om de vector met verbonden spoorsegmenten op te vragen. *set-links!* wijzigt de verbonden spoorsegmenten. *links-map* voert een procedure uit op de vector-elementen. *reserved?* geeft of de ID van het locomotief terug die het spoorsegment gereserveerd heeft of false. *cancel-reservation!* zet de reservatiestand naar false. *nr-of-links* geeft het aantal verbindingen terug. Classen zoals switch en detectieblokken zullen de methoden erven van deze klasse.

3.2 Switch ADT

Het switch ADT erft de methoden van het Track ADT. Het zal dus bovenop de onderstaande methoden ook de methoden van het Track ADT bevatten. Het switch ADT houdt de stand van een wissel bij, welk spoorsegment verbonden is met de huidige stand en het staat toe om de stand te veranderen.

Naam	Signatuur	Beschrijving
new	(symbol, list \rightarrow switch%)	Maakt een nieuwe switch object aan
get-status	(/ \rightarrow integer)	Geeft de stand van de wissel terug.
set-status!	(integer \rightarrow integer)	Veranert de stand van de wissel.
get-merge-track	(/ \rightarrow symbol)	Geeft de ID van het spoorsegment die niet met de stand van de wissels verandert.
get-linked-track	(/ \rightarrow symbol)	Geeft de ID van het spoorsegment die verbonden is met de stand van de wissel.

Tabel 2: Signaturen van switch%

3.2.1 Toelichting

Het switch ADT is een subklasse van het track ADT. Het kan dus dezelfde methoden gebruik als het track ADT. De extra methoden zijn *new* dat dezelfde argumenten verwacht als het track ADT (De ID van het spoorsegment en een lijst met de ID's van de verbonden spoorsegmenten.) Let op!: Zet de ID op de juiste index in de lijst. Als de stand van een wissel 1 is en het verbindt met het spoorsegment 1-5, dan zou de lijst '(U-1 1-5 ... etc...) moeten zijn. Dit is zodat de interne index van de stand van de wissels analoog is aan de stand van de wissels van de hardware. *get - status* geeft de stand van de wissel terug als een nummer. *set - status!* verandert de stand van de wissel naar het gegeven stand. Zorg er wel voor dat de index correct overeenkomt omdat het afleest van de vector die de verbonden spoorsegmenten bijhoudt. *get - merge - track* geeft de ID van het spoorsegment die niet met de stand van de wissel verandert. *get - linked - track* geeft de ID terug van de huidige verbonden spoorsegment aan de hand van de stand van de wissel.

3.3 Dblock ADT

Het detection block ADT is een subklasse van het track ADT. Het erft dus net zoals het switch ADT de methoden van het track ADT. Bovendien houdt het ook bij of dat er zich een locomotief op bevindt.

Naam	Signatuur	Beschrijving
new	(symbol, list \rightarrow dblock%)	Maakt een nieuw detectie blok object aan.
occupied?	(/ \rightarrow symbol \cup false)	Geeft ID van de locomotief terug die momenteel de detectieblok bezet.
occupy!	(symbol \rightarrow /)	Bezet de detectieblok met het gegeven ID.
vacant!	(/ \rightarrow /)	Maakt de detectieblok vrij.

Tabel 3: Signaturen van dblock%

3.3.1 Toelichting

De methoden die bruikbaar zijn voor het detectieblok ADT zijn de methoden die het ADT erft van het track ADT. Dit is ook niet zonder de hierboven genoemde methoden die het arsenaal vervullen. *new* maakt een nieuwe detectieblok object aan. Het verwacht dezelfde argumenten

zoals *new* van het track ADT. *occupied?* geeft de ID van de locomotief die het detectieblok bezet of false indien die niet bezet is. *occupy!* bezet het detectieblok met het gegeven ID van een locomotief. *vacant!* zal er voor zorgen dat het detectieblok terug vrij is.

3.4 Locomotive ADT

Het locomotief ADT houdt de interne data van een locomotief bij zoals snelheid, richting, vorige locatie, huidige locatie, manuele modus, het pad en de eindbestemming. Dit komt natuurlijk ook met methoden om ze te veranderen. Het pad en eindbestemming wordt gebruikt voor het automatisch trajectbeheer.

Naam	Signatuur	Beschrijving
new	(symbol, symbol, symbol, symbol → locomotive%)	Maakt een nieuw locomotief object aan.

Tabel 4: Signaturen van locomotive%

3.4.1 Toelichting

3.5 Railway ADT

3.5.1 Toelichting

3.6 Infrabel ADT

3.6.1 Toelichting

3.7 Gui ADT

3.7.1 Toelichting

3.8 NMBS ADT

3.8.1 Toelichting

4 Afhankelijkheids diagram

5 Beschrijving van API tussen infrabel- en NMBS-component

6 Planning

7 Logboek