Verkeerssimulatie

Documentsoort:	Behoeftespecificatie
Versie:	2.0
Datum:	28 maart 2019
Auteurs:	Brent van Bladel
Status:	In development

1 Samenvatting

Dit document bevat de specificaties voor een informaticasysteem ter ondersteuning van een verkeerssimulatie. Het is geschreven in het kader van het vak "Project Software Engineering"(1ste bachelor informatica - Universiteit Antwerpen).

2 Context

Op 25 juni 2018 heeft het politiek stuurcomité van Antwerpen 18 projecten geselecteerd die voor een overkapping van de Antwerpse ring zullen zorgen. De eerste werken beginnen in de zomer van 2019 en zullen een hoop verkeershinder met zich mee brengen. Om deze hinder zo goed mogelijk in te perken, heeft het Departement Mobiliteit en Openbare Werken geopteerd om een simulatiemodel te laten ontwikkelen dat het verkeer kan simuleren.

De Universiteit Antwerpen is gevraagd dit systeem te ontwikkelen. In de eerste bachelor informatica zal onder de vakken "Computer Graphicsën "Project Software Engineering"gewerkt worden aan dit project. Tijdens de practica Computer Graphics zal de visualisatie van de simulatie ontwikkeld worden, tijdens de practica Project Software Engineering zal gewerkt worden aan de simulatie applicatie zelf.

3 Legende

De behoeftespecificatie is opgesteld aan de hand van zogenaamde use-cases. Elke use-case beschrijft een klein gedeelte van de gewenste functionaliteit. Het is de bedoeling dat tijdens elke fase van het project verschillende van die use cases geïmplementeerd worden. Een typische use-case bevat de volgende onderdelen:

- **Refertenummer & titel:**
  Wordt gebruikt om naar een bepaalde use-case te verwijzen.

- **Prioriteit:**
  De specificatie van een systeem vraagt meer dan wat binnen de voorziene tijd op te leveren is. Vandaar dat we per use-case aangeven in hoeverre die functionaliteit belangrijk is. In volgorde van belangrijkheid kan hier staan: VERPLICHT (deze use-case moet opgeleverd worden), BELANGRIJK (niet essentieel maar bij voorkeur toch opleveren), NUTTIG (interessant maar kan weggelaten worden).

- **Doel:**
  Summiere beschrijving van het waarom van de use-case, t.t.z. wat de use-case bijdraagt tot de gehele functionaliteit.

- **Preconditie:**
  Summiere beschrijving van de uitgangspunten bij aanvang van de use-case.

- **Succesvol einde:**
  Summiere beschrijving van wat opgeleverd zal worden als er niks fout is gegaan.

- **Stappen:**
  Een sequentiële beschrijving van hoe de use-case precies zal verlopen als alles goed gaat (het zogenaamde "happy day scenario"). De stappen zijn genummerd en kunnen controle instructies (WHILE, IF, ...) bevatten.

- **Uitzonderingen:**
  Een lijst van mogelijke probleemgevallen en hoe die behandeld zullen worden. Een probleem geval (a) verwijst naar het nummer van de stap waar het probleem kan optreden, (b) bevat een conditie die aangeeft wanneer het probleemgeval optreedt, (c) omschrijft heel kort (een lijn) hoe het probleem behandeld zal worden.

- **Voorbeeld:**
  Een voorbeeld van wat in- of uitgevoerd kan worden.

Soms is een use-case een uitbreiding van een andere use-case, en dan zijn volgende onderdelen relevant:

- **Uitbreiding:**
  Een referte naar de use-case waarvan deze een uitbreiding is.

- **Stappen:**
  Een lijst van extra en/of aangepaste stappen t.o.v de use-case waarvan deze een uitbreiding is.
  Een uitbreiding (a) verwijst naar het nummer van de stap die uitgebreid wordt, (b) zegt of de uitbreiding voor, na of tijdens de normale stap zal gebeuren, (c) omschrijft wat precies in de uitbreiding zal gebeuren.

4 Overzicht

Use-Case	**Prioriteit**
1: Invoer	
1.1. Wegen en voertuigen inlezen	VERPLICHT
1.2. Wegennetwerk inlezen	BELANGRIJK
1.3. Voertuig met type inlezen	VERPLICHT
1.4. Wegen met verkeerstekens inlezen	BELANGRIJK
1.5. Wegen met meerdere rijstroken inlezen	NUTTIG
2: Uitvoer	
2.1. Simpele uitvoer	VERPLICHT
2.2. Grafische impressie	BELANGRIJK
2.3. Integratie met graphics	BELANGRIJK
2.4. Statistieken van het verkeer	NUTTIG
3: Simulatie	
3.1. Rijden van voertuigen	VERPLICHT
3.2. Automatische simulatie	VERPLICHT
3.3. Rijden van voertuigen met type	VERPLICHT
3.4. Simulatie van baan met zones	BELANGRIJK
3.5. Simulatie van baan met bushaltes	BELANGRIJK
3.6. Simulatie van baan met verkeerslichten	BELANGRIJK
3.7. Inhalen van voertuigen	NUTTIG
3.8. Slimme verkeerslichten	NUTTIG
4: Gebruikersinterface	
4.1. GUI voor simulatie	NUTTIG
4.2. GUI voor verkeerstekens	NUTTIG

1.1. Wegen en voertuigen inlezen

Prioriteit:
VERPLICHT

Doel:
Inlezen van het schema van de verkeerssituatie: de verschillende wegen en de verschillende voertuigen.

Preconditie:
Een ASCII bestand met daarop een beschrijving van de wegen en voertuigen. (Zie Appendix A voor meer informatie over het XML formaat)

Succesvol einde:
Het systeem bevat een schema met de verschillende wegen, en informatie over alle voertuigen.

Stappen:
1. Open invoerbestand
2. WHILE Bestand niet ingelezen
2.1. Herken het soort element (VOERTUIG, BAAN)
2.2. Lees verdere informatie voor het element
2.3. IF Verifieer geldige informatie
2.3.1. THEN Voeg element toe aan de simulatie
2.3.1. ELSE Foutboodschap + positioneer op volgende element in het bestand
3. Verifieer consistentie van de verkeerssituatie
4. Sluit invoerbestand

Uitzonderingen:
2.1. [Onherkenbaar element] Foutboodschap + positioneer op volgende element in het bestand ⇒ verdergaan vanaf stap 2
2.2. [Ongeldige informatie] Foutboodschap + positioneer op volgende element in het bestand ⇒ verdergaan vanaf stap 2
3. [Inconsistente verkeerssituatie] Foutboodschap ⇒ verdergaan vanaf stap 4

Voorbeeld:
Een baan met twee auto's die stilstaand vertrekken 10 meter van elkaar:

```
<BAAN>
    <naam>E19</naam>
    <snelheidslimiet>100</snelheidslimiet>
    <lengte>2000</lengte>
</BAAN>

<VOERTUIG>
    <type>AUTO</type>
    <nummerplaat>1THK180</nummerplaat>
    <baan>E19</baan>
    <positie>10</positie>
    <snelheid>0<snelheid>
</VOERTUIG>

<VOERTUIG>
    <type>AUTO</type>
    <nummerplaat>651BUF</nummerplaat>
    <baan>E19</baan>
    <positie>0</positie>
    <snelheid>0<snelheid>
</VOERTUIG>
```

1.2. Wegennetwerk inlezen

Prioriteit:
BELANGRIJK

Doel:
Inlezen van hoe de verschillende wegen met elkaar verbonden zijn.

Preconditie:
Een ASCII bestand met daarop een beschrijving van de wegen en voertuigen. (Zie Appendix A voor meer informatie over het XML formaat)

Succesvol einde:
Het systeem bevat een schema met de verschillende wegen, hoe deze verbonden zijn, en informatie over alle voertuigen.

Uitbreiding:
Use Case 1.1

Stappen:
[2.2, tijdens] Hou rekening met extra attributen tijdens het parsen van 'BAAN' elementen

Uitzonderingen:
Geen

Voorbeeld:

```
<BAAN>
    <naam>E19</naam>
    <snelheidslimiet>100</snelheidslimiet>
    <lengte>2000</lengte>
    <verbinding>E313</verbinding>
</BAAN>

<BAAN>
    <naam>E313</naam>
    <snelheidslimiet>120</snelheidslimiet>
    <lengte>5000</lengte>
</BAAN>
```

1.3. Voertuig met type inlezen

Prioriteit:
VERPLICHT

Doel:
Een specifiek voertuig zal zich anders gedragen afhangend van zijn type. Om dit in de simulatie te kunnen opnemen, moet deze data ook ingelezen worden. Zie Appendix B voor meer informatie over de verschillende soorten voertuigen.

Uitbreiding:
Use Case 1.1

Stappen:
[2, tijdens] Hou rekening met extra mogelijkheden voor 'type'

Uitzonderingen:
Geen

Voorbeeld:
Voorbeeld van verschillende types.

```
<VOERTUIG>
    <type>MOTORFIETS</type>
    <nummerplaat>1NXE881</nummerplaat>
    <baan>E19</baan>
    <positie>10</positie>
    <snelheid>0<snelheid>
</VOERTUIG>

<VOERTUIG>
    <type>BUS</type>
    <nummerplaat>1LML935</nummerplaat>
    <baan>Middelheimlaan</baan>
    <positie>0</positie>
    <snelheid>0<snelheid>
</VOERTUIG>
```

1.4. Verkeerstekens inlezen

Prioriteit:
BELANGRIJK

Doel:
Een baan kan verschillende verkeerstekens bevatten. Om dit in de simulatie te kunnen opnemen, moet deze data ook ingelezen worden.

Uitbreiding:
Use Case 1.1

Stappen:
[2, tijdens] Hou rekening met extra element

Uitzonderingen:
Geen

Voorbeeld:
Voorbeeld van verkeerstekens.

```
<BAAN>
    <naam>Middelheimlaan</naam>
    <snelheidslimiet>50</snelheidslimiet>
    <lengte>1000</lengte>
</BAAN>

<VERKEERSTEKEN>
    <type>BUSHALTE</type>
    <baan>Middelheimlaan</baan>
    <positie>250</positie>
</VERKEERSTEKEN>

<VERKEERSTEKEN>
    <type>ZONE</type>
    <baan>Middelheimlaan</baan>
    <positie>500</positie>
    <snelheidslimiet>30</snelheidslimiet>
</VERKEERSTEKEN>

<VERKEERSTEKEN>
    <type>BUSHALTE</type>
    <baan>Middelheimlaan</baan>
    <positie>750</positie>
</VERKEERSTEKEN>
```

1.5. Wegen met meerdere rijstroken inlezen

Prioriteit:
NUTTIG

Doel:
Een baan kan meerdere rijstroken hebben. Om dit in de simulatie te kunnen opnemen, moet deze data ook ingelezen worden.

Uitbreiding:
Use Case 1.1

Stappen:
[2, tijdens] Hou rekening met extra attributen

Uitzonderingen:
Geen

Voorbeeld:
Voorbeeld van een baan met 3 rijstroken.

```
<BAAN>
    <naam>E19</naam>
    <snelheidslimiet>100</snelheidslimiet>
    <lengte>2000</lengte>
    <rijstroken>3</rijstroken>
</BAAN>
```

2.1. Simpele uitvoer

Prioriteit:
VERPLICHT

Doel:
Uitvoer van alle informatie in de simulatie.

Preconditie:
Het systeem bevat een schema van de virtuele verkeerssituatie.

Succesvol einde:
Het systeem heeft een tekstbestand (ASCII) uitgevoerd, waarin de informatie over de virtuele verkeerssituatie netjes is uitgeschreven.

Stappen:
1. Open uitvoerbestand
2. WHILE Nog banen beschikbaar
2.1. Schrijf baan-gegevens uit
3. WHILE Nog voertuigen beschikbaar
3.1. Schrijf voertuig-gegevens uit
4. Sluit uitvoerbestand

Uitzonderingen:
Geen

Voorbeeld:
Gegeven de input van 1.1

```
Baan: E19
  -> snelheidslimiet: 100
  -> lengte: 2000

Voertuig: auto (1THK180)
 -> baan: E19
 -> positie: 10
 -> snelheid: 0

 Voertuig: auto (651BUF)
 -> baan: E19
 -> positie: 0
 -> snelheid: 0
```

2.2. Grafische impressie

Prioriteit:
BELANGRIJK

Doel:
De toestand van de verkeerssituatie wordt grafisch weergegeven.

Preconditie:
Het systeem is correct geïnitialiseerd.

Succesvol einde:
Het systeem heeft een tekstbestand (ASCII) uitgevoerd, waarin de toestand van de verkeerssituatie staat beschreven.

Variant:
Het systeem heeft een tekstbestand (HTML) uitgevoerd, waarin de toestand van de verkeerssituatie staat beschreven.

Stappen:
1. Open uitvoerbestand
2. Teken gegevens uit voor de toestand van de verkeerssituatie
3. Sluit uitvoerbestand

Uitzonderingen:
Geen

Voorbeeld:
Indien twee banen (E19 en E313), een auto (A) en een vrachtwagen (V).

```
E19  | ====A===============
E313 | =V==============================================
```

2.3 Integratie met graphics

Prioriteit:
BELANGRIJK

Doel:
Onze klant had graag een 3D visualisatie gehad van het verkeerssituatie in de simulatie. Hiervoor kan je een standaard interface voor je graphics engine gebruiken.

Preconditie:
Het systeem is correct geïnitialiseerd.

Succesvol einde:
Elke beweging van voertuigen wordt weergegeven in een 3D omgeving.

2.4. Statistieken van het verkeer

Prioriteit:
NUTTIG

Doel:
Statistieken van het verkeer worden na de simulatie weergegeven.

Preconditie:
Het systeem bevat een schema van de virtuele verkeerssituatie.

Succesvol einde:
Het systeem heeft een tekstbestand (ASCII) uitgevoerd, waarin de statistieken over de simulatie netjes zijn uitgeschreven.

Variant:
Het systeem heeft een tekstbestand (CSV) uitgevoerd, waarin de statistieken over de simulatie netjes zijn uitgeschreven. Meer informatie over het CSV type vind je *hier*.

Opmerkingen:
De volgende statistieken worden uitgeschreven voor elk voertuig:

- Hoeveel seconden het voertuig in de simulatie zat.
- Hoeveel seconden het voertuig stil stond (snelheid gelijk aan nul).
- Hoeveel seconden het voertuig onderweg was (snelheid niet gelijk aan nul).
- Gemiddelde snelheid dat het voertuig heeft aangehouden.
- Maximale snelheid dat het voertuig heeft aangehouden.
- Afstand dat het voertuig heeft afgelegd.

3.1. Rijden van voertuigen

Prioriteit:
VERPLICHT

Doel:
Simuleren van het rijden van een voertuig. Zie Appendix B voor meer informatie over de formules.

Preconditie:
Het systeem bevat een schema van de virtuele verkeerssituatie. Er is een voertuig op een baan.

Succesvol einde:
Het voertuig heeft een nieuwe positie.

Stappen:
1. Bereken nieuwe positie van voertuig
2. Bereken nieuwe snelheid van voertuig
3. Bereken nieuwe versnelling van voertuig
4. IF nieuwe positie valt buiten huidige baan
4.1. IF huidige baan heeft verbinding
4.1.1 Zet voertuig op verbindingsbaan
4.2. ELSE
4.2.1. Verwijder voertuig uit simulatie

Uitzonderingen:
Geen

3.2. Automatische simulatie

Prioriteit:
VERPLICHT

Doel:
Simulatie automatisch laten lopen.

Preconditie:
Het systeem bevat een schema van de virtuele verkeerssituatie.

Succesvol einde:
Alle voertuigen hebben het wegennetwerk verlaten en de simulatie stopt.

Stappen:
1. WHILE voertuigen in het wegennetwerk
1.1 FOR elk voertuig in het wegennetwerk
1.1.1 voer use case 3.1 uit op het voertuig

3.3. Rijden van voertuigen met type

Prioriteit:
VERPLICHT

Doel:
Een specifiek voertuig zal zich anders gedragen afhangend van zijn type. Zie Appendix B voor meer informatie over de verschillende soorten voertuigen.

Uitbreiding:
Use Case 3.1

Stappen:
[2, tijdens] Hou rekening met de grenswaarden van het type voertuig
[3, tijdens] Hou rekening met de grenswaarden en lengte van het type voertuig

Uitzonderingen:
Geen

3.4. Simulatie van baan met zones

Prioriteit:
BELANGRIJK

Doel:
Een baan kan verschillende verkeerstekens bevatten, waaronder zones (bijvoorbeeld een zone 30 of een zone 50). Wanneer een auto voorbij een verkeersteken rijdt dat de start van een nieuwe zone specifieert, zal de auto zich houden aan de nieuwe snelheidslimiet. Hiervoor moet je ook use case 1.4 implementeren.

Preconditie:
Het systeem bevat een schema van de virtuele verkeerssituatie. Dit schema bevat minstens 1 zone.

Succesvol einde:
Voertuigen houden zich aan de snelheidlimiet van de zone.

Uitbreiding:
Use case 3.1

Stappen:
[3, tijdens] Hou rekening met de variabele snelheidslimiet van de baan

Uitzonderingen:
Geen

3.5. Simulatie van baan met bushaltes

Prioriteit:
BELANGRIJK

Doel:
Een baan kan verschillende verkeerstekens bevatten, waaronder bushaltes. Wanneer een voertuig van type bus op 100 meter van een bushalte rijdt, zal het voertuig zijn versnelling aanpassen om aan de bushalte te stoppen (Zie Appendix A voor meer informatie over de formule voor de versnelling). Hiervoor moet je ook use case 1.4 implementeren.

Preconditie:
Het systeem bevat een schema van de virtuele verkeerssituatie. Dit schema bevat minstens 1 bushalte en 1 bus.

Succesvol einde:
De bus stopt aan de bushalte.

Uitbreiding:
Use case 3.1

Stappen:
[3, tijdens] IF type van voertuig is bus AND afstand tot volgende bushalte $<$ 100 meter
[3, tijdens] THEN bereken de versnelling om te stoppen aan de bushalte
[4, tijdens] IF nieuwe positie is gelijk aan de bushalte
[4, tijdens] THEN het voertuig wacht 30 seconden op deze positie

Uitzonderingen:
[4, tijdens] IF nieuwe positie is gelijk aan de bushalte AND snelheid is $>$ 0
$\Rightarrow$ Foutboodschap + ga over naar Use case 3.1

3.6. Simulatie van baan met verkeerslichten

Prioriteit:
BELANGRIJK

Doel:
Een baan kan verschillende verkeerstekens bevatten, waaronder verkeerslichten. Wanneer een voertuig op 2 maal de ideale volgafstand van een rood of oranje verkeerslicht rijdt, zal het voertuig zijn versnelling aanpassen om aan het verkeerslicht te stoppen (Zie Appendix A voor meer informatie over de formule voor de versnelling). Hiervoor moet je ook use case 1.4 implementeren.

Preconditie:
Het systeem bevat een schema van de virtuele verkeerssituatie. Dit schema bevat minstens 1 verkeerslicht.

Succesvol einde:
Voertuigen stoppen aan het verkeerslicht.

Uitbreiding:
use case 3.1

Stappen:
[3, tijdens] IF afstand tot volgende verkeerslicht $< 2 * \Delta d_{ideal}$
[3, tijdens] 1. IF verkeerslicht is rood OR verkeerslicht is oranje
[3, tijdens] 1.1. bereken de versnelling om te stoppen aan het verkeerslicht

Uitzonderingen:
[4, tijdens] IF nieuwe positie is gelijk aan het verkeerslicht AND snelheid is > 0 AND verkeerslicht is rood
$\Rightarrow$ Foutboodschap + ga over naar Use case 3.1

Opmerking:
Gebruik de volgende timings voor het verkeerslicht:

- 30 seconden groen
- 5 seconden oranje
- 30 seconden rood

3.7. Inhalen van voertuigen

Prioriteit:
NUTTIG

Doel:
Voertuigen die op een baan rijden met meerdere rijstroken zullen tragere voertuigen willen inhalen. Hiervoor moet je ook use case 1.5 implementeren.

Preconditie:
Het systeem bevat een schema van de virtuele verkeerssituatie. Dit schema bevat minstens 1 baan met meerdere rijstroken.

Succesvol einde:
Snellere voertuigen halen tragere voertuigen in.

Uitbreiding:
Use case 3.1

Uitzonderingen:
IF een voertuig op een rijstrook rijdt op een baan met verbinding
AND de verbinding bevat deze rijstrook niet
AND het voertuig bereikt het einde van de huidige baan
⇒ Foutboodschap + verwijder dit voertuig uit de simulatie

Opmerkingen:

- Voertuigen beginnen altijd op de meest rechtse rijstrook.
- Een voertuig zal 1 rijstrook naar links opschuiven indien volgende stellingen allemaal gelden:
    1. Het voertuig rijdt trager dan de snelheidslimiet van de baan of zone.
    2. Het voertuig rijdt trager dan zijn maximaal haalbare snelheid.
    3. Het voertuig heeft 5 seconden op rij een versnelling van 0.
    4. Het voertuig rijdt volgens de regels in Use case 3.1, en is dus niet aan het vertragen voor een verkeersteken.
    5. Het voertuig is van type auto of motorfiets.
    6. Er is geen voertuig op de nieuwe rijstrook in een straal van de ideale volgafstand (dus zowel voor als achter het voertuig).
- Voertuigen proberen altijd op de meest rechtse rijstrook te rijden wanneer mogelijk. Wanneer een voertuig niet op de meest rechtse rijstrook rijdt, en stellingen 3-6 gelden, zal het voertuig een rijstrook naar rechts opschuiven.

- Het wisselen van rijstrook duurt 5 seconden. Tijdens deze wisseltijd staat het voertuig op beide rijstroken: voertuigen op beide stroken zullen rekening moeten houden met het wisselend voertuig in hun versnellingsberekening.

3.8. Slimme verkeerslichten

Prioriteit:
NUTTIG

Doel:
Verkeerslichten houden rekening met voertuigen op de baan om de doorstroming te verbeteren.

Preconditie:
Het systeem bevat een schema van de virtuele verkeerssituatie. Dit schema bevat minstens 1 verkeerslicht.

Succesvol einde:
Verkeerslichten houden rekening met voertuigen.

Uitbreiding:
Use case 3.6

Opmerkingen:
In plaats van een vast patroon aan te houden, zullen verkeerslichten proberen om voertuigen zo veel mogelijk groen te geven. Je mag zelf kiezen hoe je dit algoritme implementeert. Wel moet het zich aan de volgende condities houden:

- Wanneer het verkeerslicht van groen naar rood verandert zal het altijd eerst 5 seconden oranje zijn.
- Het verkeerslicht moet minstens 15 seconden zijn kleur (groen of rood) behouden voordat het terug mag veranderen.
- Het verkeerslicht mag maximaal 90 seconden zijn kleur (groen of rood) behouden voordat het terug moet veranderen.
- Er verkeerslicht moet eerlijk zijn: hou bij hoe lang het verkeerslicht groen en rood was in totaal, en minimalizeer het verschil.

4.1. GUI voor simulatie

Prioriteit:
NUTTIG

Doel:
Een gebruiksvriendelijke userinterface voor het uitvoeren van de simulatie.

Preconditie:
Het systeem bevat een schema van de virtuele verkeerssituatie.

Succesvol einde:
De simulatie kan beheerd worden aan de hand van een grafische userinterface.

4.2. GUI voor verkeerstekens

Prioriteit:
NUTTIG

Doel:
Een gebruiksvriendelijke userinterface hebben voor het beheren van verkeerstekens: tussen simulatiestappen kan je de snelheidslimiet van een zone aanpassen en de staat van verkeerslichten.

Preconditie:
Het systeem bevat een schema van de virtuele verkeerssituatie.

Succesvol einde:
Verkeerstekens kunnen manueel aangepast worden aan de hand van een grafische userinterface.

A Invoer formaat

Het invoerformaat voor de virtuele verkeerssituatie is zodanig gekozen dat nieuwe attributen en elementen makkelijk kunnen worden toegevoegd.

```
Verkeerssimulatie = { Element }
Element = "<" ElementType ">" AttributeList "</" ElementType ">"
ElementType = "VOERTUIG" | "BAAN" | "VERKEERSTEKEN"
AttributeList = Attribute { Attribute }
Attribute = "<" AttributeType ">" AttributeValue "</" AttributeType ">"
AttributeType = "naam" | "snelheidslimiet" | "lengte" | "type"
                | "nummerplaat" | "baan" | "positie" | "snelheid"
                | "verbinding" | "rijstroken"
AttributeValue = Integer | String
Integer = Digit { Digit }
Digit = "0" ... "9"
String = Character { Character }
Character = "a" ... "z" | "A" ... "Z" | Digit
```

Merk op dat de attribuutlijst een relatief vrij formaat heeft wat sterk zal afhangen van het soort element dat gedefinieerd wordt. De volgende tabel toont de verplichte en optionele attributen voor elk element:

Element	Attribuut (verplicht)	Attribuut (optioneel)
BAAN	naam, snelheidslimiet, lengte	verbinding, rijstroken
VOERTUIG	type, nummerplaat, baan, positie, snelheid	-/-
VERKEERSTEKEN	type, baan, positie	snelheidslimiet

Bovendien zal afhankelijk van het attribuuttype slechts een bepaalde attribuutwaarde toegelaten zijn:

Attribuut	Waarde
naam, type, nummerplaat, baan, verbinding	String
snelheidslimiet, lengte, positie, snelheid, rijstroken	Integer

Bovendien moet de openings tag steeds overeenkomen met de sluitingstag. Vandaar dat tijdens de invoer moet gecontroleerd worden of de invoer al dan niet geldig is.

Het bestand met de in te lezen verkeerssituatie wordt met de hand geschreven. Om de ingelezen verkeerssituatie te kunnen simuleren moet de informatie consistent zijn.

Een verkeerssituatie is consistent als:

- Elk voertuig en elk verkeersteken staat op een bestaande baan.
- De positie van elk voertuig en elk verkeersteken is kleiner dan de lengte van de baan.
- Er zijn geen 2 voertuigen op minder dan 5 meter van elkaar.
- Indien een baan een verbinding heeft, is deze verbinding een bestaande baan.

Opmerkingen:

- Snelheden worden in kilometer per uur genoteerd.
- Lengtes en posities worden in meter genoteerd.
- Lengtes en snelheden zijn altijd positief.
- De types voertuigen die ondersteund worden zijn 'MOTORFIETS', 'AUTO', 'BUS', en 'VRACHTWAGEN'.
- De types verkeerstekens die ondersteund worden zijn 'BUSHALTE', 'VERKEERSLICHT', en 'ZONE'.
- De nummerplaat wordt gebruikt als unieke identificatie van een voertuig.
- De naam wordt gebruikt als unieke identificatie van een baan.

B Simulatie

B.1 Positie

De positie van een voertuig wordt berekend aan de hand van de snelheid van het voertuig. We gaan uit van een eenparige rechtlijnige beweging gedurende 1 stap van de simulatie. De positie p op stap i van de simulatie wordt dan gegeven door

$$p_i = v.t + p_{i-1}$$

waarbij v de snelheid en t de duur van 1 simulatiestap. We veronderstellen dat de simulatie gebeurt in stappen van 1 seconde. Dit zorgt ervoor dat we in de formule $t = 1$ kunnen stellen. We krijgen

$$p_i = v + p_{i-1}$$

Merk op dat de positie hier uitgedrukt wordt in meter, en de snelheid in meter per seconde.

B.2 Snelheid

De snelheid van een voertuig wordt berekend aan de hand van de versnelling van het voertuig. De snelheid v op stap i van de simulatie wordt gegeven door

$$v_i = a.t + v_{i-1}$$

waarbij a de versnelling en t de duur van 1 simulatiestap. We veronderstellen dat de simulatie gebeurt in stappen van 1 seconde. Dit zorgt ervoor dat we in de formule $t = 1$ kunnen stellen. We krijgen

$$v_i = a + v_{i-1}$$

Merk op dat de snelheid hier uitgedrukt wordt in meter per seconde, en de versnelling in meter per seconde kwadraat.

B.3 Versnelling

De versnelling van een voertuig wordt door zijn chauffeur bepaald op elke simulatiestap. We veronderstellen dat de chauffeur probeert om de ideale volgafstand te bekomen.

De ideale volgafstand in meter wordt gedefinieerd als 75% van de snelheid in kilometer per uur. Omdat we de positie meten als het voorste punt van het voertuig, moet de lengte van het voertuig hierbij opgeteld worden. Verder willen we ook een minimale afstand van 2 meter opleggen. We kunnen dan de ideale volgafstand berekenen als

$$\Delta d_{ideal} = \frac{3}{4}v + l_{prev} + 2$$

waarbij l de lengte van het voorgaande voertuig en v de snelheid van het voertuig in kilometer per uur.

De eigenlijke volgafstand in meter wordt berekend als

$$\Delta d_{actual} = p_{prev} - l_{prev} - p$$

waarbij l_{prev} de lengte van het voorgaande voertuig, p_{prev} de positie van het voorgaande voertuig in meter, en p de positie in meter.

De versnelling in meter per seconde kwadraat wordt elke simulatiestap berekend als

$$a = \frac{1}{2}(\Delta d_{actual} - \Delta d_{ideal})$$

Merk op dat, indien het voertuig de snelheidslimiet overschrijdt, de versnelling nooit groter dan nul mag zijn. Merk ook op dat, indien er geen voertuig voor rijdt, de maximale versnelling genomen wordt.

In het geval dat een voertuig tot stilstand wil komen op een bepaalde positie (bijvoorbeeld aan een verkeerslicht), kunnen we de voorgaande formules omrekenen. We krijgen dan dat de versnelling in meter per seconde kwadraat berekend wordt als

$$a = -\frac{v^2}{\Delta p}$$

waarbij v de huidige snelheid in meter per seconde en Δp de afstand waarin men moet stoppen (bijvoorbeeld de afstand tot het verkeerslicht).

B.4 Grenswaarden

Om onze simulatie realistisch te houden, gaan we een aantal grenswaarden respecteren. De volgende tabel geeft een overzicht van deze grenswaarden:

Variabele	Minimale Waarde	Maximale Waarde
positie	0	lengte van baan
snelheid MOTORFIETS (km/u)	0	180
snelheid AUTO (km/u)	0	150
snelheid BUS (km/u)	0	70
snelheid VRACHTWAGEN (km/u)	0	90
versnelling MOTORFIETS (m/s^2)	-10	4
versnelling AUTO (m/s^2)	-8	2
versnelling BUS (m/s^2)	-7	1
versnelling VRACHTWAGEN (m/s^2)	-6	1

Merk op dat de snelheid niet negatief mag worden. Een negatieve snelheid zou theoretisch wel mogelijk zijn indien het voertuig achteruit rijdt, maar dit is niet gewenst in onze simulatie. De versnelling kan wel negatief zijn: in dit geval remt het voertuig.

B.5 Lengte

De volgende tabel geeft een overzicht van de lengte van een voertuig:

Voertuig	Lengte (m)
MOTORFIETS	1
AUTO	3
BUS	10
VRACHTWAGEN	15