Traffic Simulation

Document:	System Specification
Version:	1.0
Date:	16 December 2022
Author:	John Businge
Status:	Delivered

1 Summary

This document contains the specification for a computer system that can execute a traffic simulation. It has been written for the course “472 - Software Product Design and Development I” (Bachelor of computer science - UNLV).

2 Context

On 25 June 2018, the political steering committee of Las Vegas selected 18 projects that will cover the Las Vegas Strip roads. The first works started in the summer of 2019 and have caused a lot of traffic disruption. In order to limit this nuisance as much as possible during the further course of the project, the Department of Public Works has opted to develop a simulation model that can simulate the traffic.

Department of Public Works has asked UNLV to develop this simulation. The computer science bachelor students will be tasked to work on this project in the “Software Product Design and Development I” and “Computer Graphics” courses.

3 Legend

The requirements specification has been drafted on the basis of use-cases. Each use-case describes a small part of the desired functionality. The intention is that during each phase of the project different use-cases are implemented. A typical use-case contains the following components:

- **Reference number & title:**
  Used to identify or refer to a use-case.
- **Priority:**
  The specification of the system demands more than what can be delivered within the foreseen time. That is why we use the priority of a use-case to indicate to what extent its functionality is important. The priority can be (in order of importance): REQUIRED (this use case must be completed), IMPORTANT (not essential but preferably deliver), USEFUL (interesting but can be omitted).
- **Goal:**
  Brief description of the goal of the use case, i.e. what the use case contributes to the entire functionality.
- **Precondition:**
  Brief description of the required properties at the start of the use-case.
- **Postcondition:**
  Brief description of the required properties at the end of the use-case.
- **Steps:**
  A sequential description of how the use-case executes if everything goes well (the so-called "happy day scenario "). The steps are numbered and may include control instructions (WHILE, IF, ...).
- **Exceptions:**
  A list of possible deviations from the happy day scenario and how they should be treated. An exception (a) refers to the number of the step where the exception may occur, (b) contains a condition that indicates when the exception occurs, and (c) describes very briefly how the exception will be treated.
- **Example:**
  An example of the input or output.

Sometimes a use-case is an extension of another use-case. Then the following components are relevant:

- **Extension:**
  A reference to the use-case that is being extended.

- **Steps:**
  A list of additional and / or modified steps with regard to the use-case that is being extended.
  An extension (a) refers to the step number being extended, (b) states whether the extension is before, after, or during the step, and (c) describes what exactly will happen in the extension.

4 Overview

Use-Case	**Priority**
1: Input	
1.1 Read traffic network	REQUIRED
1.2 Read in vehicle generator	IMPORTANT
2: Output	
2.1 Simple ouput	REQUIRED
3: Simulation	
3.1 Moving vehicle	REQUIRED
3.2 Traffic light simulation	REQUIRED
3.3 Automatic simulation	REQUIRED
3.4 Simulation with vehicle generator	IMPORTANT

1.1. Read traffic network

Priority:
REQUIRED

Goal:
Read of the schedule of the traffic network: the different roads and the different vehicles.

Precondition:
An ASCII file containing a description of the roads and vehicles. (See Appendix 4 for more information about the XML format)

Post condition:
The system contains a schedule with the different roads, and information about all vehicles.

Steps:

```
1. Open input file
2. WHILE Not at end of file
2.1. Detect the type of element (VEHICLE, ROAD, TRAFFIC LIGHT)
2.2. Read data of the element
2.3. IF Verify data is valid
2.3.1. THEN Add the element to the virtual traffic network
2.3.1. ELSE Error message + go to next element in the file
3. Verify consistency of the traffic network
4. Close input file
```

Exceptions:

2.1. [Unrecognized element] Error message + go to next element in the file ⇒ continue from step 2

2.2. [Invalid data] Error message + go to next element in the file ⇒ continue from step 2

3. [Inconsistent subway network] Error message ⇒ continue from step 4

Example:
A road with two cars and one traffic light:

```
<ROAD>
    <name>Rochelle</name>
    <length>500</length>
</ROAD>
<TRAFFIC LIGHT>
    <road>Rochelle</road>
    <position>400</position>
    <cycle>30</cycle>
</TRAFFIC LIGHT>
<VEHICLE>
    <road>Rochelle</road>
    <position>20</position>
</VEHICLE>
<VEHICLE>
    <road>Rochelle</road>
    <position>0</position>
</VEHICLE>
```

1.2. Read in vehicle generator

Priority:
IMPORTANT

Goal:
Read in how vehicles should be generated.

Precondition:
An ASCII file containing a description of a vehicle generator. (See Appendix A for more information about the XML format)

Post condition:
The system includes a schema of a vehicle generator .

Extension:
Use Case 1.1

Steps:
[2.1, during] Take into account extra element VEHICLE GENERATOR

Exceptions:
None

Example:

```
<VEHICLEGENERATOR>
    <name>Rochelle</name>
    <frequency>5</frequency>
</VEHICLEGENERATOR>
```

2.1. Simple output

Priority:
REQUIRED

Goal:
Output all data from the virtual traffic system.

Precondition:
The system contains a diagram of virtual traffic network.

Postcondition:
The system generated an ASCII file that contains all data from the virtual traffic network.

Steps:

```
1. Write out current simulation time
2. WHILE Vehicles still in simulation
2.1. Write out vehicle data
```

Exceptions:
None

Example:
Given the input from 1.1

```
Time 0
Vehicle 1
    -> road: Rochelle
    -> position: 20
    -> speed: 16.6

Vehicle 2
    -> road: Rochelle
    -> position: 0
    -> speed: 16.6
```

3.1. Moving vehicle

Priority:
REQUIRED

Goal:
Simulate driving a vehicle. See Appendix B for more information about the formulas.

Precondition:
The system contains a diagram of the virtual traffic network. There is a vehicle on a road.

Postcondition:
The position, speed, and acceleration of the vehicle have been recalculated.

Steps:

1. Calculate new speed and position of vehicle
2. Calculate new acceleration of vehicle
3. IF new position is outside current road

3.1. Remove vehicle from simulation

Exceptions:
None

3.2. Traffic light simulation

Priority:
REQUIRED

Goal:
Simulating traffic lights

Precondition:
The system contains a diagram of the virtual road network. There is a traffic light on a road.

Postcondition:
Vehicles adapt depending on the state of the traffic light.

Steps:

```
1. IF time since last change > cycle
1.1 THEN change the color of the light (green $\iff$ red)
2. IF traffic light is green
2.1 THEN vehicles in front of the traffic light may accelerate back up
3.1 IF traffic light is red
3.1.1 THEN IF the first vehicle in front of the light is in the deceleration distance
3.1.1.1 THEN apply the deceleration factor to the vehicle
3.1.2 ELSE IF the first vehicle in front of the light is in the first half of the stopping distance
3.1.2.1 THEN stop the vehicle
```

3.3. Automatic simulation

Priority:
REQUIRED

Goal:
Run simulation automatically.

Precondition:
The system contains a diagram of the virtual road network.

Postcondition:
The traffic in the road network is simulated.

Steps:
1. FOR any vehicle in the road network
1.1 Execute use-case 3.1 out on the vehicle
2. FOR any traffic light in the road network
2.1 Execute use-case 3.2 out at the traffic light

3.4. Simulation with vehicle generator

Priority:
IMPORTANT

Goal:
Simulation with vehicle generator.

Precondition:
The system contains a diagram of the virtual road network. There is a vehicle generator on a road.

Postcondition:
Vehicles are added automatically during the simulation.

Uitbreiding:
Use Case 3.3

Steps:

```
3. FOR any vehicle generator
3.1 IF time since last vehicle > frequency
3.1.1 IF no vehicle on road between positions 0 and $2l$
3.1.1.1 THEN add vehicle to road at position 0
```

Appendix A - Input format

The input format for the virtual traffic network has been chosen in such a way that new attributes and elements can easily be added.

```
TrafficSystem = { Element }
Element = "<" ElementType ">" AttributeList "</" ElementType ">"
ElementType = "VEHICLE" | "TRAFFIC LIGHT" | "ROAD" | "VEHICLE GENERATOR"
AttributeList = Attribute { Attribute }
Attribute = "<" AttributeType ">" AttributeValue "</" AttributeType ">"
AttributeType = "name" | "length" | "road" | "position" | "cycle" | "frequency"
AttributeValue = Primitive
Primitive = Integer | String
Integer = Digit { Digit }
Digit = "0" ... "9"
String = Character { Character }
Character = "a" ... "z" | "A" ... "Z"
```

Note that the AttributeList has a relatively free format which will strongly depend on the type of element defined. The following table shows the attributes for each element:

ElementType	Attribute
ROAD	name, length
TRAFFIC LIGHT	road, position, cycle
VEHICLE	road, position
VEHICLE GENERATOR	road, frequency

In addition, depending on the AttributeType, only one specific AttributeValue is allowed:

AttributeType	AttributeValue
name, road	String
length, position, cycle, frequency	Integer

In addition, the opening tag must always correspond to the closing tag. This is why it is necessary to check whether or not the input is valid during parsing.

The inputfile containing the traffic network is written by hand. In order to simulate the subway system, the information must be consistent.

The traffic system is consistent if:

- Each vehicle is on an existing road.
- Each traffic light is on an existing road.
- Each vehicle generator is on an existing road.
- The position of each vehicle is less than the length of the road.
- The position of each traffic light is less than the length of the road.
- There is a maximum of one vehicle generator on each road.
- A traffic light must not be within the deceleration distance of another traffic light (see Appendix B).

Comments:

- Length, position, cycle, and frequency must always be positive.
- The name is used to uniquely identify a road.

Appendix B - Simulation model

B.1 - Variables

The table below contains an overview of the variables used in the simulation model.

Abbreviation	Name	Description
l	length	Length of a vehicle.
x	position	Current position of a vehicle, measured from the front bumper.
v	speed	Current speed of a vehicle.
v_{max}	maximum speed	The desired maximum speed of a vehicle. Can change due to external factors (such as a traffic light).
V_{max}	maximum speed	The absolute maximum speed that a vehicle can travel.
a	acceleration	Current acceleration of a vehicle.
a_{max}	maximum acceleration	The maximum acceleration of a vehicle.
b_{max}	maximum braking factor	The maximum braking factor of a vehicle.
f_{min}	minimum following distance	The minimum desired following distance of a vehicle.
Δt	simulation time	The time between two steps of the simulation.
Δx_s	deceleration distance	The distance to a traffic light in which a vehicle must slow down, including the stopping distance.
Δx_{s0}	stopping distance	The distance from a traffic light in which a vehicle must stop.
s	delay factor	Factor by which a vehicle must slow down at a traffic light.

B.2 - Position, speed, and formulas

We distinguish between two networks when adjusting the position and speed of a vehicle:

- If $v + a\Delta t$ is less than zero, the velocity would become negative. This is not allowed in our model. In this case we adjust the position as follows:

$$x = x - \frac{v^2}{2a} \tag{1}$$

Then we set the speed equal to zero

- If this is not the case, we first adjust the speed:

$$v = v + a\Delta t \tag{2}$$

Then we adjust the position:

$$x = x + v\Delta t + a\frac{\Delta t^2}{2} \tag{3}$$

B.3 - Acceleration Formulas

To adjust the acceleration of a vehicle i, we first calculate δ, the interaction term with vehicle $i-1$ (the vehicle driving in front of vehicle i).

We first calculate the following distance $x\Delta$:

$$\Delta x = x_{i-1} - x_i - l_{i-1} \tag{4}$$

We then calculate the speed difference Δv:

$$\Delta v = v_i - v_{i-1} \tag{5}$$

With this we calculate δ:

$$\delta = \frac{f_{min} + max(0, v + \frac{v\Delta v}{2\sqrt{a_{max}b_{max}}})}{\Delta x} \tag{6}$$

If there is no vehicle in front, so vehicle i is the first vehicle on the road, then we set δ equal to zero.

We can then calculate the acceleration as follows:

$$a = a_{max}(1 - \left(\frac{v}{v_{max}}\right)^4 - \delta^2) \tag{7}$$

B.4 - Slowing down and accelerating

To slow down the first vehicle on the road, we set v_{max} equal to sV_{max}, with s the deceleration factor. Vehicles behind this vehicle will also automatically slow down.

If the first vehicle on the road is allowed to accelerate back up, then we set v_{max} equal to V_{max}. Vehicles behind this vehicle will automatically accelerate as well.

B.5 Stopping

If the first vehicle on the road has to come to a stop, we adjust the acceleration in each simulation step as follows.

$$a = -\frac{b_{max}v}{v_{max}} \tag{8}$$

Vehicles behind this vehicle will also automatically come to a stop

B.6 - Default values

The table below contains an overview of the default values used in the simulation model.

Abbreviation	Name	Description
l	length	4
V_{max}	maximum speed	16.6
a_{max}	maximum acceleration	1.44
b_{max}	maximum braking factor	4.61
f_{min}	minimum following distance	4
Δt	simulation time	0.0166
Δx_s	deceleration distance	50
Δx_{s0}	stopping distance	15
s	delay factor	0.4