2016-2016

Traffic simulation

Report

Group members:

MOHEDANO Geronimo

HENRY Romain

EL HILALI Houda

Summary

1. Introduction

This project aims to resume the knowledge of agents oriented programing learnt in the IA51.

The project begins first with collecting all the information needed in order to have an idea of which kind of simulation we'll be doing. Afterwards we've decided on how many agents we will be having for our Traffic simulation.

To achieve this project, we've used SARL programing language in order to program all the agents and the different parts of the simulation such as the different model classes and the GUI controller, whereas the GUI part is done using Javafx.

For this project, we chose to use Javafx due to the improved GUI components that the graphical library offers. And as we were asked to have a running SARL application SARL was the language used under eclipse IDE.

Our work was managed with the use of the git tool in order to share the improvement of each of us on the project.

2. Importance of such project

Computer simulations reproduce the behavior of a system using a mathematical model. Computer simulations have become a useful tool for the mathematical modeling of many natural systems not only in physics but also in human in all the different fields. Simulation of a system is represented as the running of the system's model. It can be used to explore and gain new insights into new technology and to estimate the performance of systems too complex for analytical solutions.

Computer simulations are computer programs that can be either small, running almost instantly on small devices, or large-scale programs that run for hours or days on network-based groups of computers. The scale of events being simulated by computer simulations has far exceeded anything possible (or perhaps even imaginable) using traditional paper-and-pencil mathematical modelling.

Computer simulation developed hand-in-hand with the rapid growth of the computer, following its first large-scale deployment during the Manhattan Project in World War II to model the process of nuclear detonation. It was a simulation of 12 hard spheres using a Monte Carlo algorithm. Computer simulation is often used as an adjunct to, or substitute for, modelling systems for which simple closed form analytic solutions are not possible. There are many types of computer simulations; their common feature is the attempt to generate a sample of representative scenarios for a model in which a complete enumeration of all possible states of the model would be prohibitive or impossible.

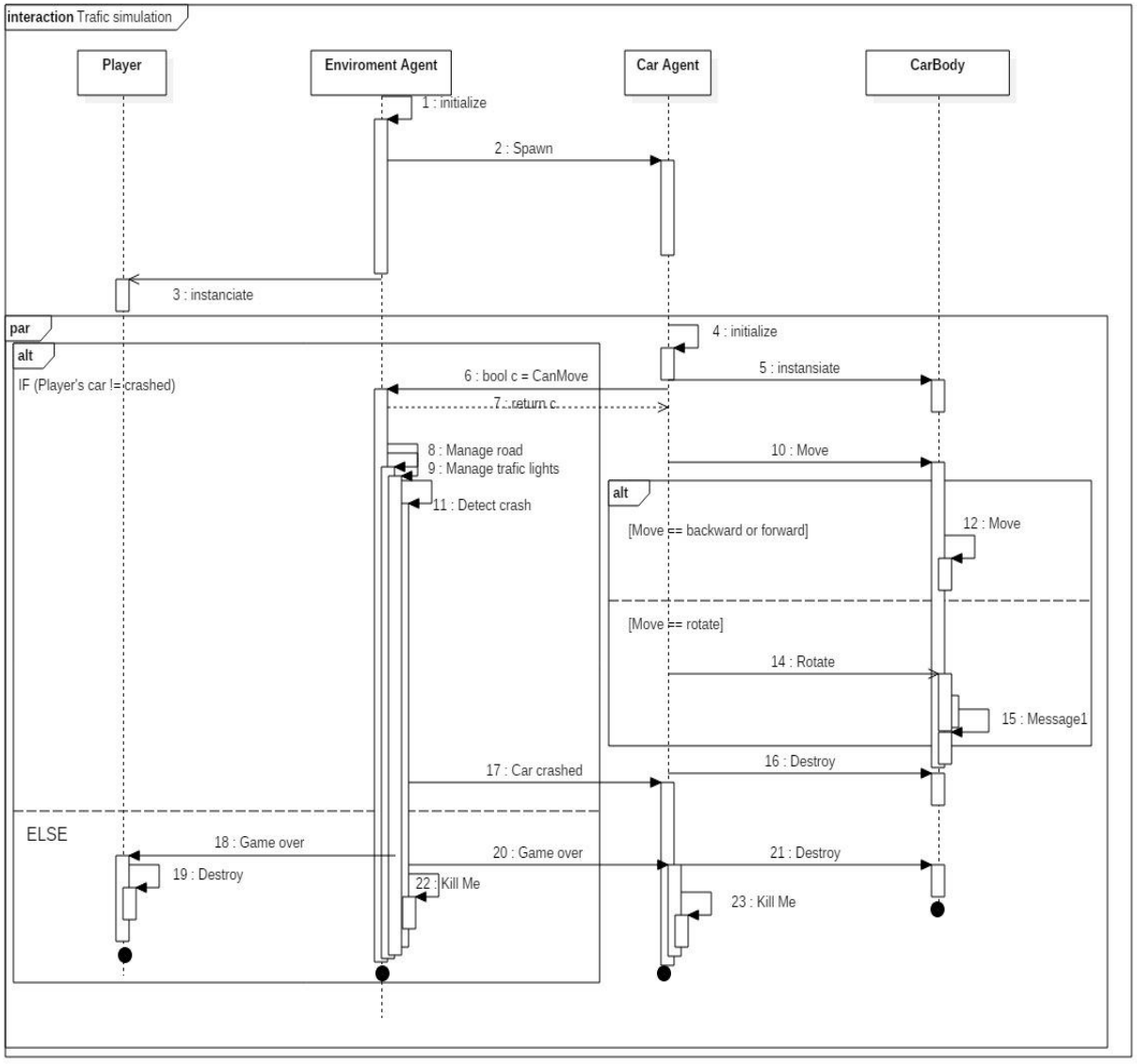
Our traffic simulation has also an undenied goal that is to simulate cars behaviour on a road taking in consideration different constraint such as traffic lights and the respect of the different traffic laws.

3. Diagrams

3.1. Global sequence diagram

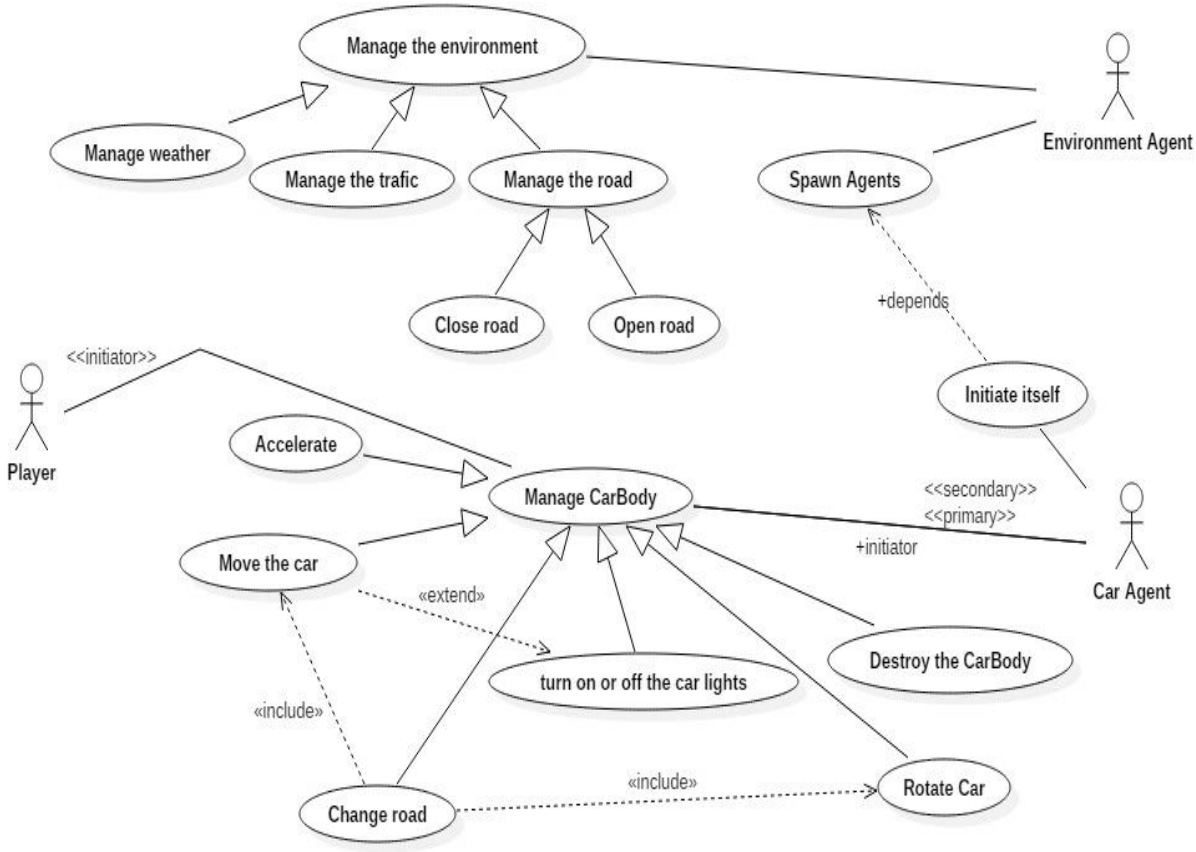
For our simulation, we've decided that it would be also a game which enables the user to move a car in the environment. In fact, the simulation starts when the users click on the button start on the home page. Then the environment is initialized and spawn the cars agents which create the car on the GUI and control its movement. The simulation ends once the player crashes it's car, this event is spread to the other agents by the environment to let the agents kill themselves.

The image below describes the different communications between different actors of the simulation.



3.2. Use case diagram

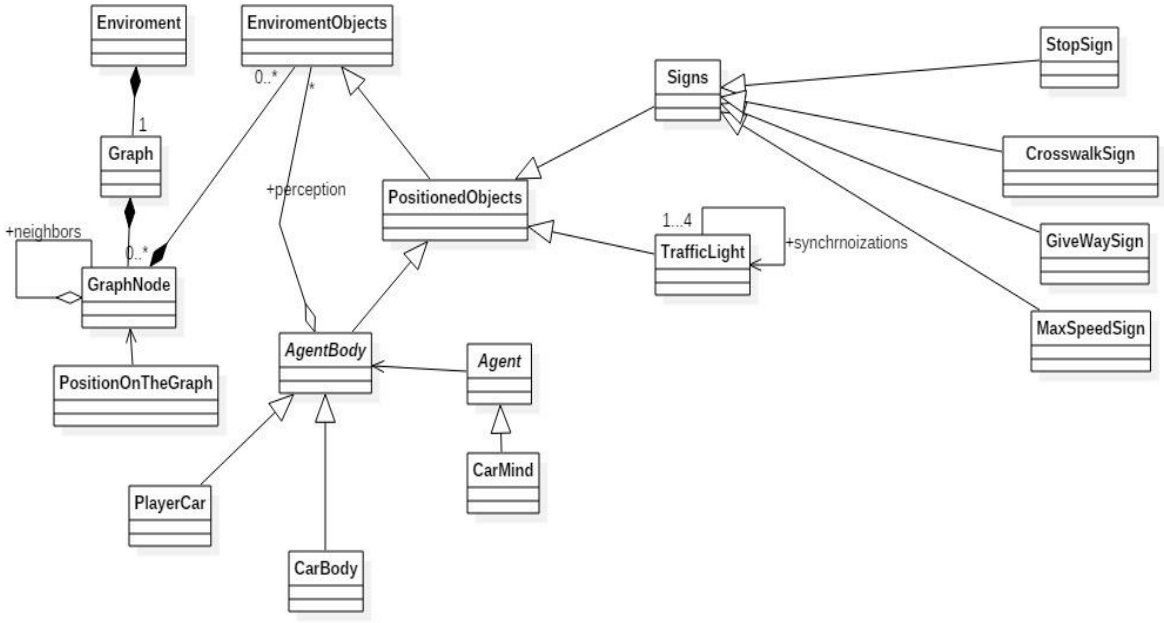
For our simulation, the game aspect we added to the simulation. That decision was taken in order to give a better experience to the user and make him a better conductor. The diagram below describes the different activities of the simulation and the dependencies between the different cases.



3.3. UML diagram

To better manage the code, we've chosen a MVC design pattern. The UML diagram below describes the different classes that we have in our project.

As in any based agent simulation, we've put an environment object that is composed of a graph which is composed of different nodes. The environment agent is the one initializing the road graph. The car object is a positioned object that the CarAgent controls.



4. Code explanation

The package controllers is the one having all the classes controlling the GUI it also has the agents because they do too control the car body. The package models is the one having the models of the object such as the car object or the road model. And the last package is the one containing all the classes related to the GUI such as the FXML file that has all the view components.

4.1. Controllers

This package contains the Car agent and the environment package and the traffic simulation SARL class that controls the GUI. The controllers don't only contain regular SARL classes but also agents that are controlling the cars and the environment.

In the following we'll explain the work of each of them with more details.

a. The Car Agent

This agent is the one controlling the car body. In fact, when initialized the agent creates a car body instance and place the car on the road interface.

At first the agent starts moving the car randomly taking in account the possible directions that he has been given to him by the environment and while moving it sends the influence to the environment in order to let the other agents know about the change that happened at the level of the environment. Also, while moving the car agent respects the speed regulations on the road, this information is given to him by the environment that sets those regulations and also it respects the traffic lights existing on the road.

This car agent is a typical good conductor that respects the road laws, this decision was taken in order to give the player who's a future conductor the perfect road to make him able to improve his driving skills. This aspect could help to give the best experience to the player.

The car agent has a car object as an attribute which inherits from the Positioned objects class that has severed attributes such as the position, the influence and the UUID which enables the environment to differentiate between all the positioned object.

When the car body crashes the car agent emits a notification to all agents that he is dead and then kill his self.

b. The environment agent

The environment agent is the one controlling the road with all its components besides the cars. The environment agent at its initialization creates the city and sets the listener on the GUI in order to make possible the visualization of all the changes made by the environment or the car agent.

The city is composed not only from the road and different city components such as buildings, houses, trees, etc but also it has traffic lights which controls and organizes the traffic to avoid accidents on the road while moving from a node to another; Those traffic lights are also controlled by the environment agent that sets the sets the lights respecting the different constraints.

In fact, the environment agent has a graph object as an attribute, this graph object is composed of a set of graph nodes. Each graph node has information about its neighbours which are also nodes. The graph node contains a list of positioned objects and also a set of nodes that may create a conflict in a road. The city which is controlled by the environment agent is created thanks to the graph creator class that sets all the right values of the city.

Furthermore, every time step the environment trigger a perception event and applies the influences with a function that takes each object influence and apply it of its possible then give him the ok to move. And each time this agent gets a notification that there's a car agent who's dead it spawns a new one. This agent also tests if there are any collisions and solves them.

The environment agent is also the one that updates the GUI thanks to the environment listener. And also, it's the one deciding how many cars will be on the road.

c. The player

This class is the one handling all the player's movement request such as turning right or left or going forward. It reads the keyboard inputs and sends the orders to the car body which applies those orders by moving on the road according to them. The player's car has a different colour in order to enable the player differentiate its car from the others.

Also, the player is the one basically launching the environment when he presses the start button. As said before this simulation has as a goal to give the player the best experience that he can have and make him a better conductor that's why as in real life once the player crashes his car or doesn't respect a traffic light he dies, that how he will learn that the road isn't a joke and learn to respect the traffic lights and the road laws.

d. The boot agent

The boot is in fact type of the environment that spawns the environment who does the rest.

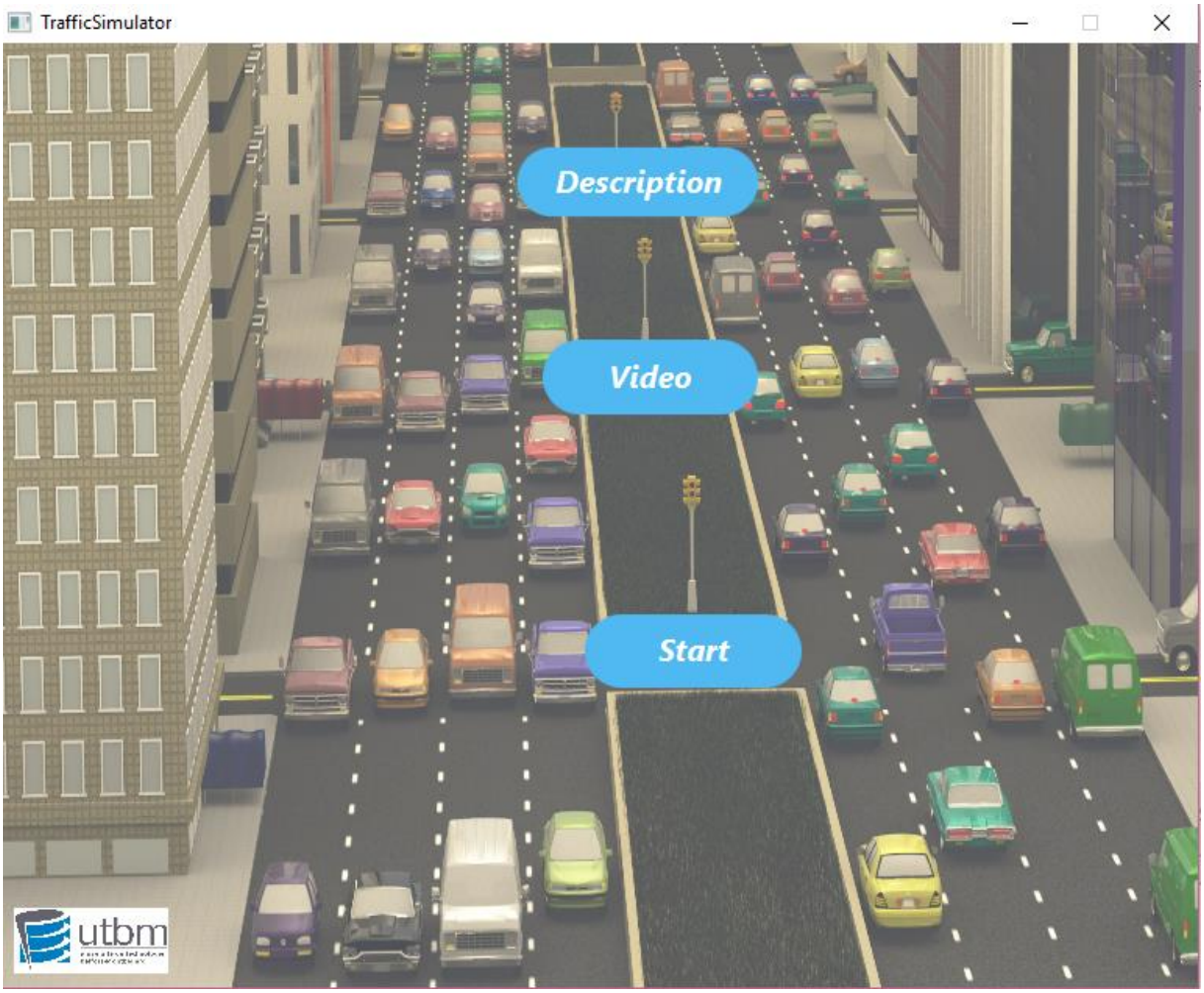
The boot class is necessary to launch the Janus kernel the main SARL agent. It contains the static main function that has as an argument the Controller of the GUI.

e. The GUI controller

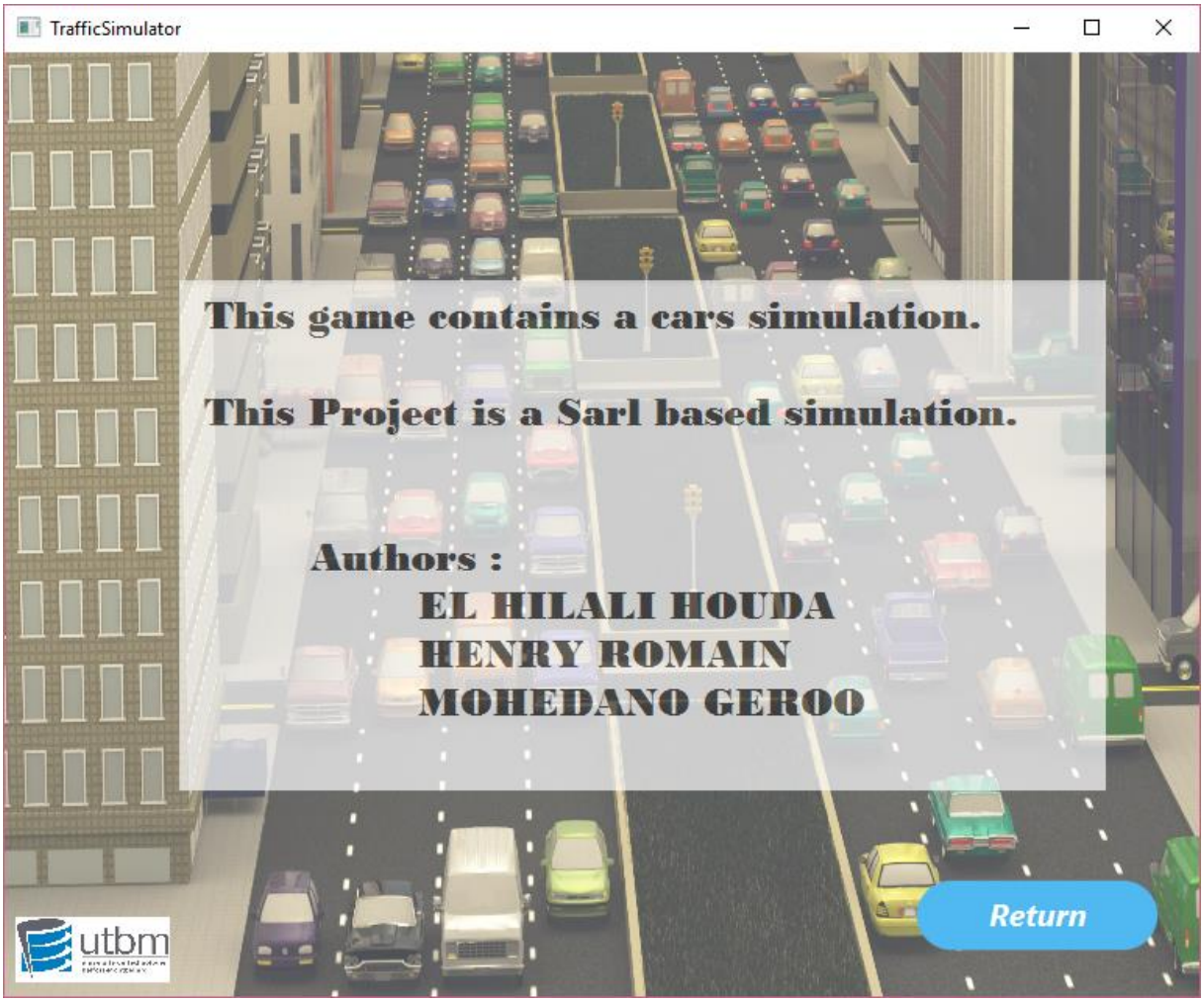
This is a class that controls the graphical user interface and implements the event listener interface. This class contains the functions that proceed the click events for the buttons at the home interface and also creates the different components on the GUI. It manages visualizing the environment changes. This class listens and draws the changes on the GUI.

4.2. The Graphical user interfaces

As we used javafx our GUI is an fxml file controlled by the GUI controller. This file contains four main views. The first view is the home view having tree buttons as shown below:



The first button, the description button, takes the user to a briefing about the simulation/game and its contributors and also opens this report. This view has also a return button that enables the users to get back to the home view.

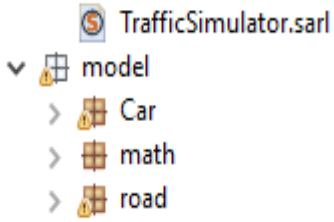


The third one is a view containing a video, this video gives a general idea to the user about how the simulation works and makes him learn more about it. It also has a return button:

The last view is the game view where all the magic happens. It where the simulation/game really starts. This view contains the city with its road and the cars on it and the other components as shown below:

4.3. The Models

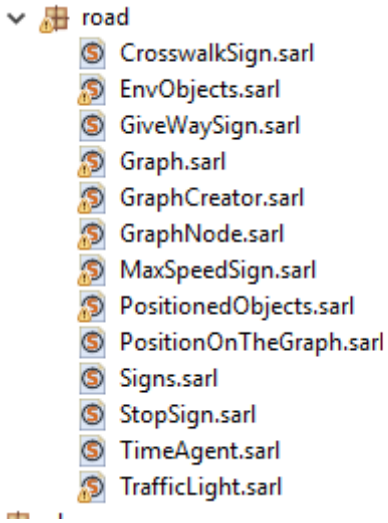
In this package, there's all the models needed for this simulation. This package has tree sub-packages:



The Car package has all the car's related models that are:

- Car object package contains the Car object SARL class and the directions java class.
- The math package has the Point2i and the Point2f SARL classes.

- Lastly, the road package has all the classes related to the road that are:



5. Conclusion

In this project, we successfully built a traffic simulation with a game aspect. We had several steps to achieve our project.

This project was an interesting subject to work on. It enabled us to get used to SARL language and to the agent programming logic. This simulation project helped us also to accentuate what we learnt during this semester about agents and their decision making. We were allowed to see each part of a basic simulation system and know the concept to monitor and code it altogether.