Design Document

City Traffic Simulation Software Solution

v2…………………………………………………………………..14/05/2019
V3…………………………………………………..................27/05/2019
V4 ………………………………………………………………… 08-06-2019

Context:

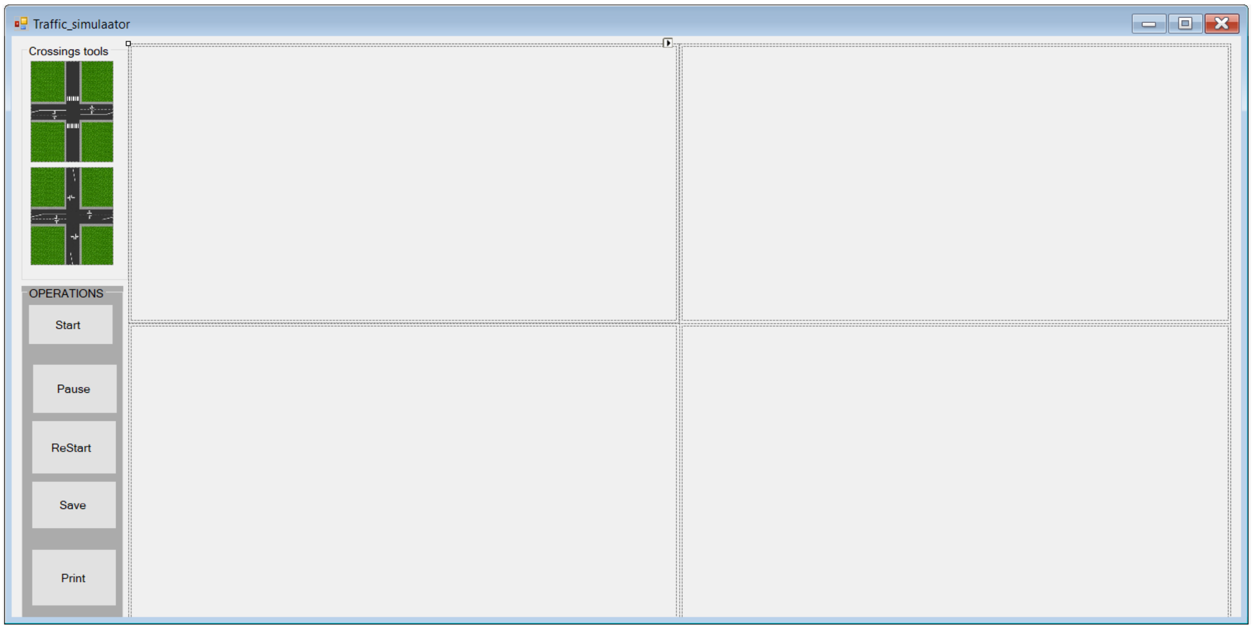
Traffic Problem is a big problem in some countries especially in populated countries. Every day millions of people use transportation to go places. But if there is no such thing that controls traffic then it becomes almost impossible to provide a fast communication service. Therefore, we are giving a solution by providing with a simulation app called “Traffic Simulation Application”.

Goal:

The goal of this application is to eradicate the problem of traffic congestion.

Description of the GUI:

The GUI of the application looks like this:



The user will be able to drag and drop the tool which are the templates in this interface and will be able to start the simulation. It is possible to drag and drop 4 crossings to create a city. Once the city is created, the user can start the simulation.

Class Description

Car:

The car class is the main class in our application.A car has a location. Which is a point, and a destination . The destination can be left, straight or right.The cars in this case will be generated using the method Draw which draws a rectangle as a car on a given location.Cars will be able to move from their location to some other destination. Cars uses the lane, by this we mean cars can be inlane or not. In other words, the cars

Traffic Light:

Traffic light class, a traffic light has a point , that is the location, a duration during which the light can shine , The light can be green which mean that the car can pass across the crossing,also the traffic light can be ilane or not. That is lane in .The traffic light is created using the Drawing method from the System.Drawing namespace. The light can also be made red. Which indicates that no car should go over the crossing point.

Grid:

Grid divides the panel into equal cells in which we can put our crossings in. It also serves as a class to enable car flow from particular direction.

Cell:

Cell class serves to determine which cell is occupied in the whole grid, meaning, to determine in which cell we can put the crossings in. It has properties such as location, to determine which cell we are talking about, a property that tells us whether cell is taken or not and crossing to know if the crossing was put into one of the cells.

Crossing:

Crossing serves to initialize whole crossings and cars on it, as well as traffic lights (and pedestrians). Besides that, it also controls car flow, meaning it determines how a car steers right, left or keeps going straight.

Lane:

The lane represent where the cars actually moves. A lane therefore have a list of cars, height, width and direction and also end points location and the end of the lane and beside the location of the lane.The height and the width of the lane helps us to defined the boundaries of the lane.
The lane can also have and end of lane and a stop point.
Cars are the for move across the the lane on which the belong to .

Entity :

The Class Entity is basically to act as a parent class for cars and pedestrians. We didn’t implement the pedestrians in the app, we kept this class as a parent class for the Car class.

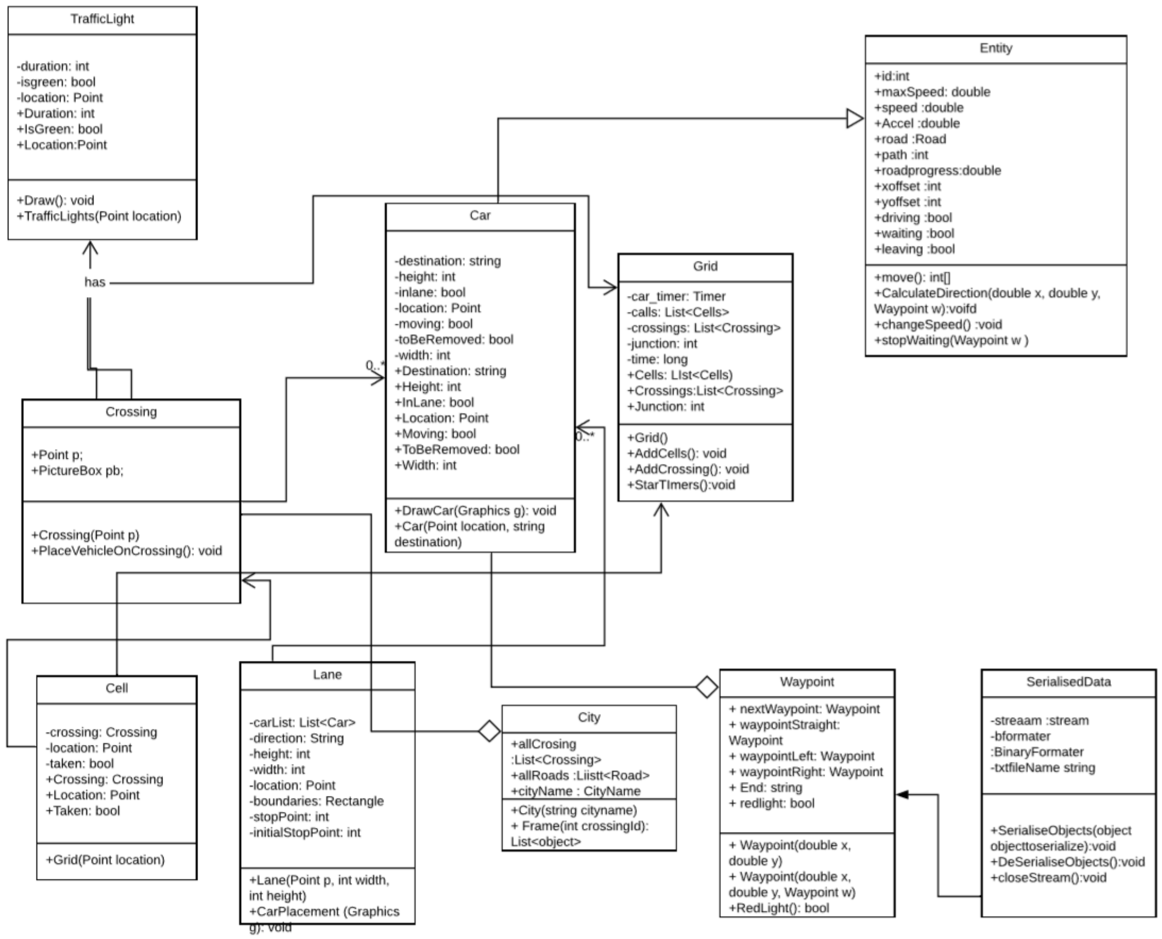
Waypoint :

The class Waypoint has some certain characteristics such as, creating the waypoints which the cars will follow. It also has a method for creating the traffic light in the crossing which can turn green from red and vice versa after some while.

SerializeData :

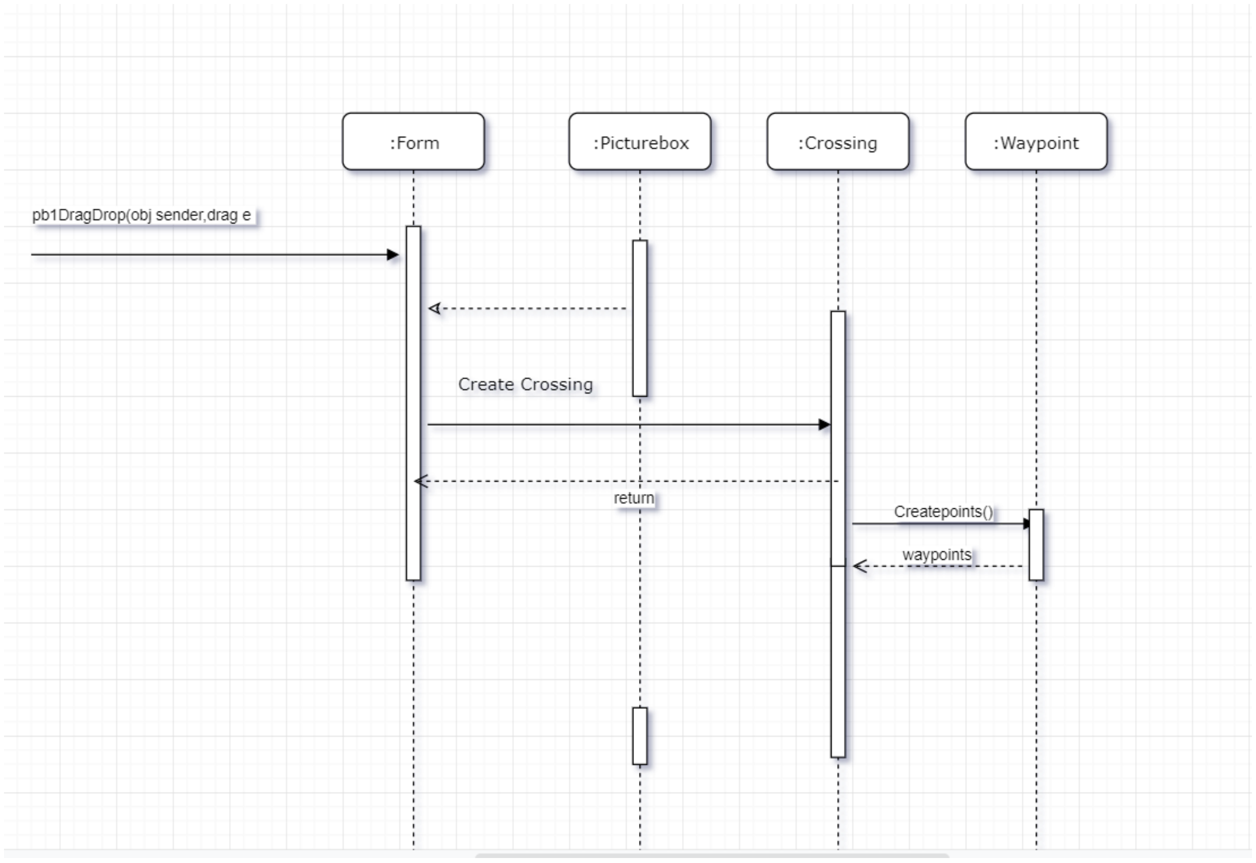
This class is created to save and load data. This class has serialization method which means that it will convert the data into binary form and store it in a file.

Class Diagram:

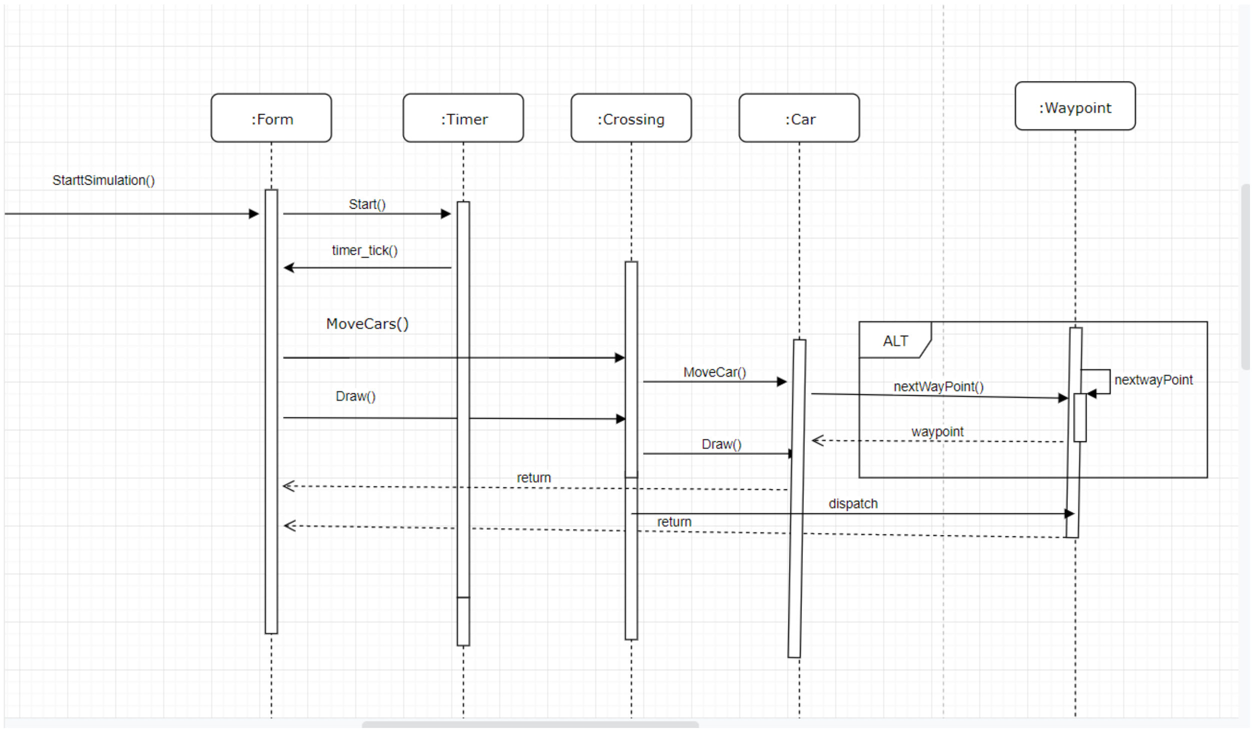


Sequence Diagram

case 1: Drag dropping the people



case 2: Start Simulation


:Form
:Timer
:Crossing
:Car
:Waypoint
StarttSimulation()
Start()
timer_tick()
MoveCars()
MoveCar()
ALT
nextWayPoint()
nextwayPoint
Draw()
waypoint
Draw()
return
dispatch
return