City Traffic Simulator

Requirements Analysis Document

Team 14
Mike Senior
Cal Leising
Louis Shakya
Nathan Luchsinger

Introduction

Purpose

This system aims to simulate the flow of traffic in the city of Pacopolis. Specifically, it will allow a user to change the arrangement of traffic lights and stop signs at intersections, then simulate the flow of traffic so the user can see the affect the arrangement has on how quickly the cars reach their destination. Using this information, a user can determine the optimal layout that will get the most cars to their destinations the fastest.

Scope

For this system, we will have to create a map of Pacopolis, vehicles to traverse it, starting locations and destinations, and stop sign and traffic light intersections. The map must be mapped out based on the dots in the Pac-man maze image provided. For the vehicles, we will have to make it so that they can pathfind to their destination, stop at intersections and wait when needed, and move at appropriate speeds. The starting and stopping points must be able to be established by the user. All intersections should be able to be set to several modes within the application, including traffic lights and the three different types of stop sign arrangements. The stop sign intersections must be able to be set to have stop signs in every direction, in north and south directions only, or in east and west directions only. These stop signs should keep a queue of every car that has arrived at the intersection and allow them to leave the intersection one at a time, in order of arrival, when it is safe. Traffic light intersections should essentially function as stop signs that switch from east and west to north and south every two minutes. Finally, there should be a timer that tracks the current time spent on a simulation and the total time taken for all cars to reach their destination.

Objectives and Success Criteria

The goal of this project is to make it possible for a user to test out different combinations of start and end points for vehicles, and different sets of stop signs and traffic lights for intersections, in order to find the most efficient combination of traffic lights and stop signs that gets most cars to their destinations the fastest. In order for this to be successful, we must have it be possible to change intersection types within the application. We must also make

it possible for the user to alter and create start and end points for vehicles. Lastly, we must make it so that Vehicles take consistent routes to their end points.

Definitions, Acronyms, and Abbreviations

Destination- End point of the car.

Dots- Points on the map, the space between each dot represents ⅛ mile.

Intersection- Any location where a car has more than one option for its next direction.

Pacopolis- The city we are using the map of for our program.

(incomplete, add more as they are needed)

References

City Traffic Simulator problem statement from increment 1 section 6.

Overview

Overall, we will create a system capable of managing traffic on a static map using an arrangement of traffic lights and stop signs, as well as an arrangements of start and stop points for each vehicle. The goal of this system is to allow the user to try out different combinations of traffics lights and stop signs in order to find the most efficient set of intersection for a given set of start end point points for vehicles.

Current System

Overview

The operator will be able to determine the start and end points of each vehicle and add more vehicles outside of the application. The operator will then be able to set what each intersections setting is by clicking on a button on each intersection that cycles through traffic lights and each type of stop sign. They may then run the simulation and start a timer. When all cars reach their destination, the timer will stop and the time taken will be recorded. The operator may then change the types of intersections and run the simulation again, and the new time will be recorded as well. All times within the current session of the application will be recorded.

Functional Requirements

Be sure to provide unique requirement identifiers for traceability

F1. The operator must be able to set intersections to be all way stop signs, north-south stop signs, east-west stop signs, or traffic lights.

F2. The operator must be able to set start and end points at certain dots on the map for each vehicle.

F3. The operator must be able to reset the simulation, putting all vehicles back at the starting location. The simulation will then wait for another input, after the operator has finished altering the intersections, to start the simulation again.

Nonfunctional Requirements

Be sure to provide unique requirement identifiers for traceability

N1. The system must support several vehicles moving at once.

N2. Vehicles should take the same route to their destination as long as their start and end points remain unchanged.

N3. All inputs from the user should be processed within 1 second.

N4. Times taken for all vehicles to reach their destinations are tracked throughout the session.

N5. Intersections cannot be changed in the middle of a simulation.

System Models

Use Case Model

Use Case: Change Intersection

Precondition: Simulation is reset

1) User selects the button at the intersection they would like to change
2) The system gets the intersection that the user chose
3) User chooses the type of intersection they would like to implement, either one with north-south stop signs, east-west stop signs, all-way stop signs, or one with traffic lights
4) The intersection updates to match the user's choice

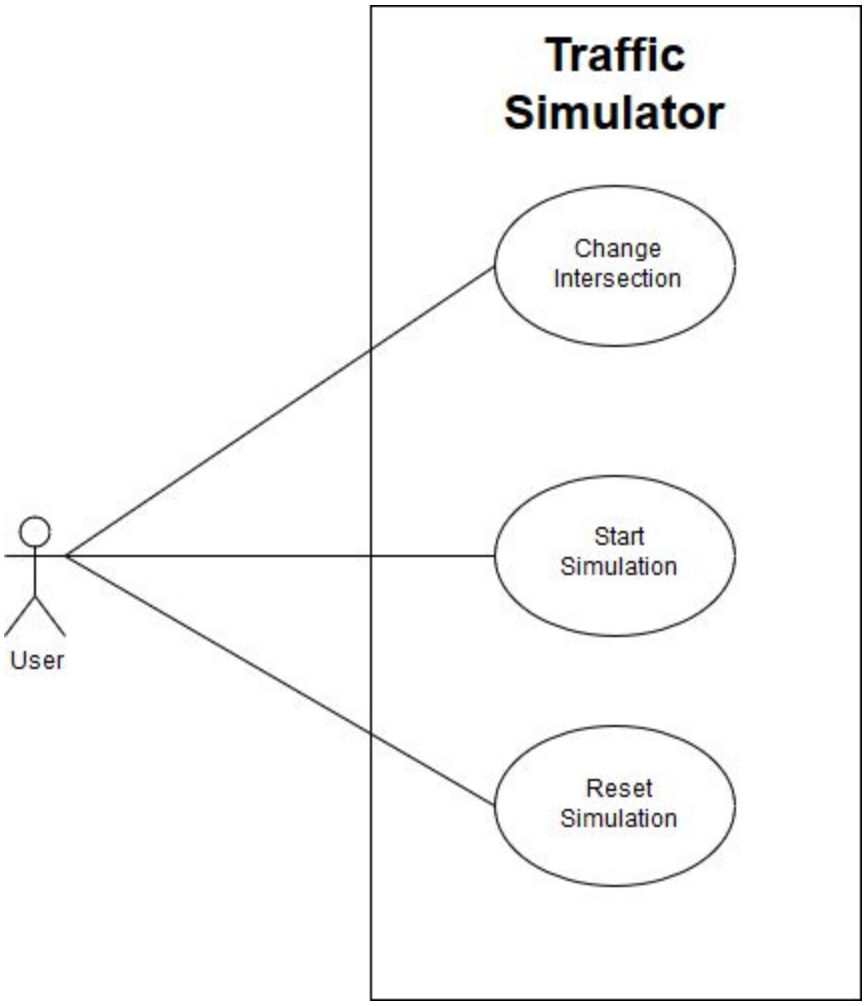
Use Case: Reset Simulation

1) User selects the reset simulation button from the user interface
2) The timer will be reset to zero
3) All vehicles will be returned to their starting points
4) The simulation will allow the user to update intersections, as well as change destinations for vehicles

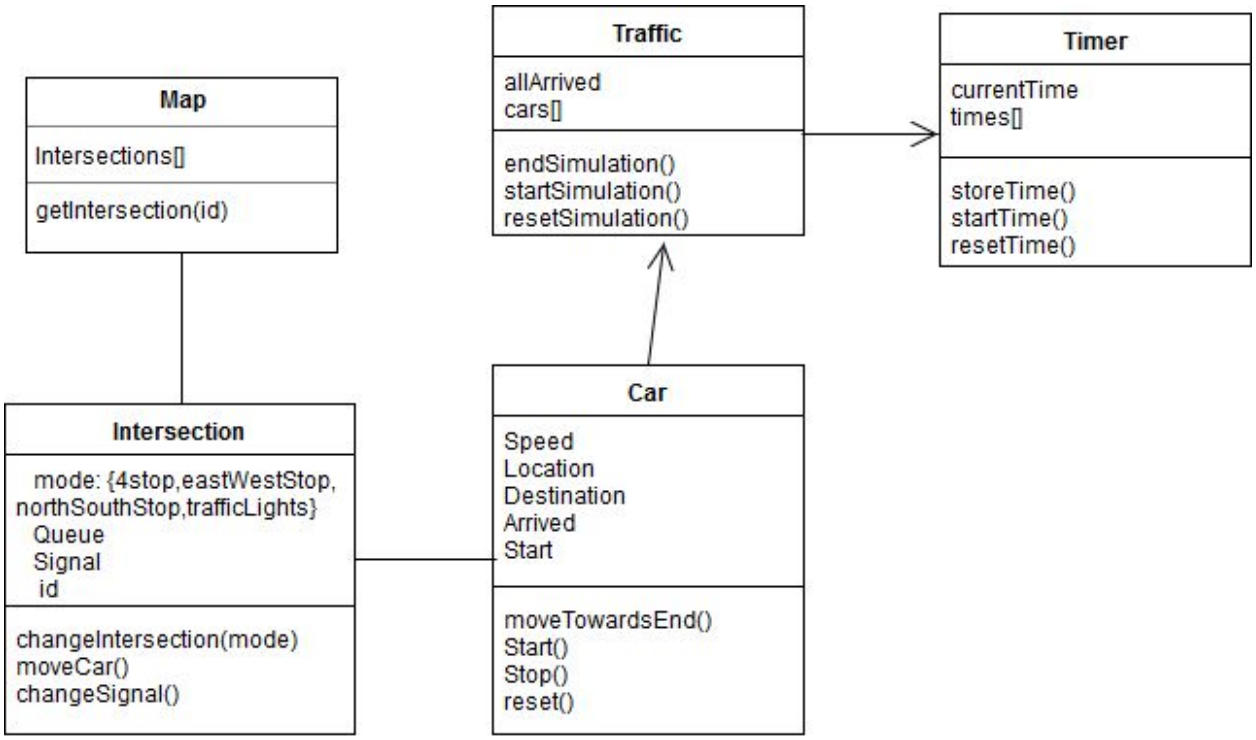
Use Case: Start Simulation

1) User selects the start simulation button from the user interface
2) The timer will begin, and vehicles will move towards their destinations

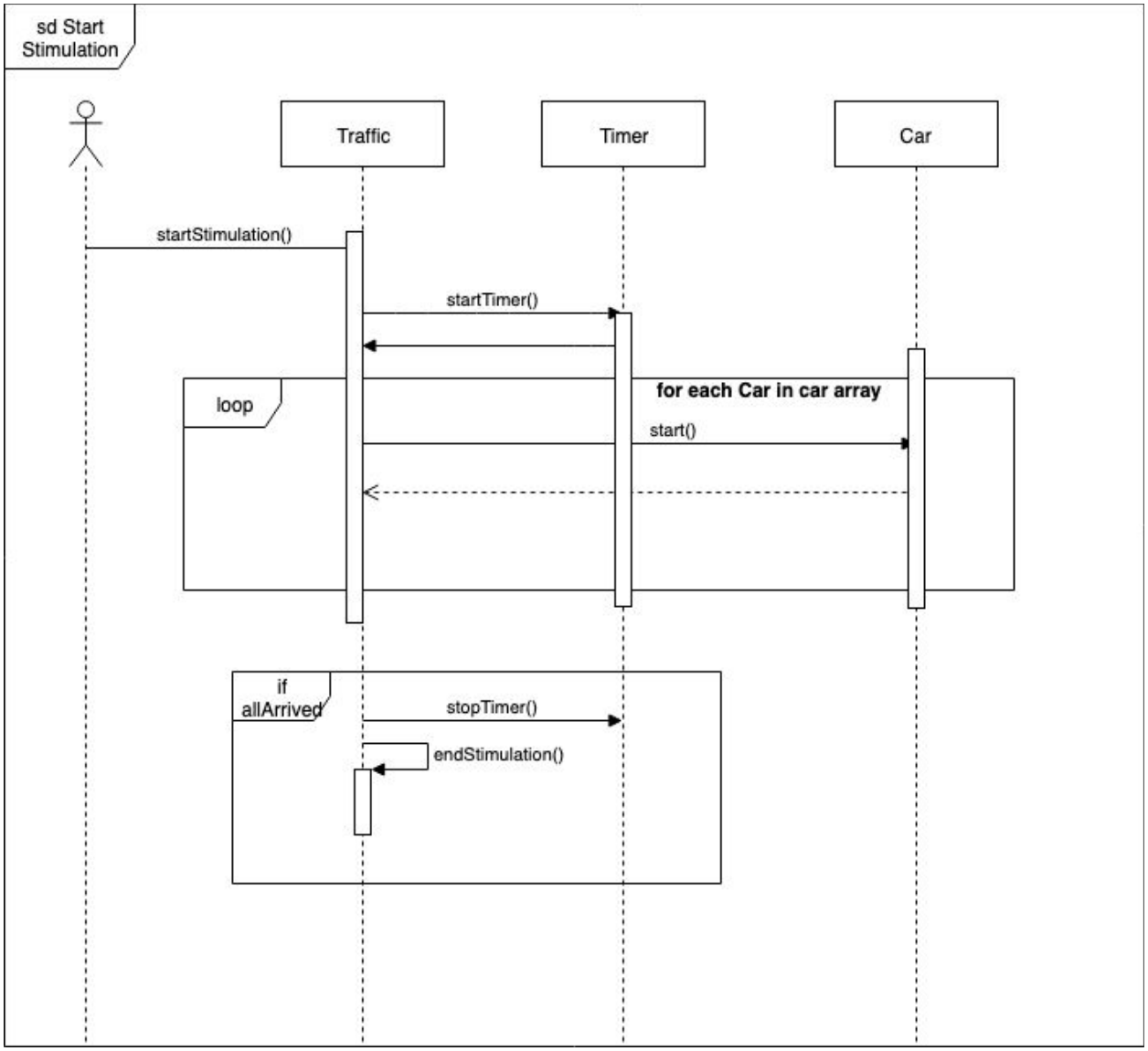
3) The simulation will continue until all cars have reached their destinations, or the user has selected to reset the simulation
4) The timer will record the time at the end of the simulation

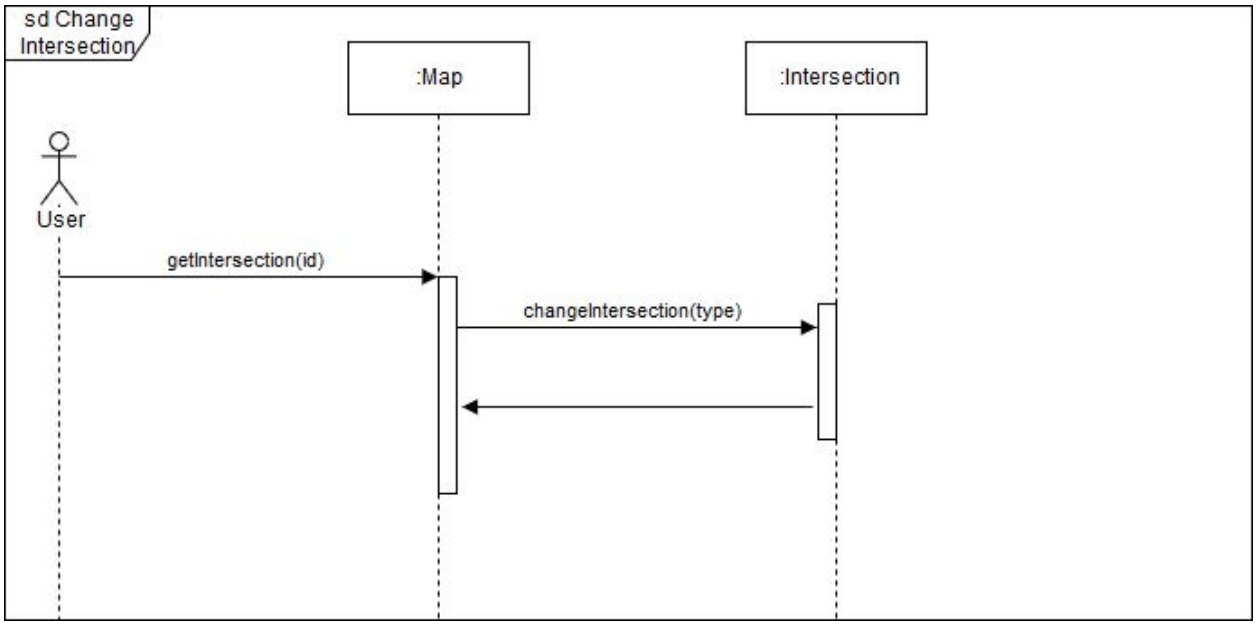


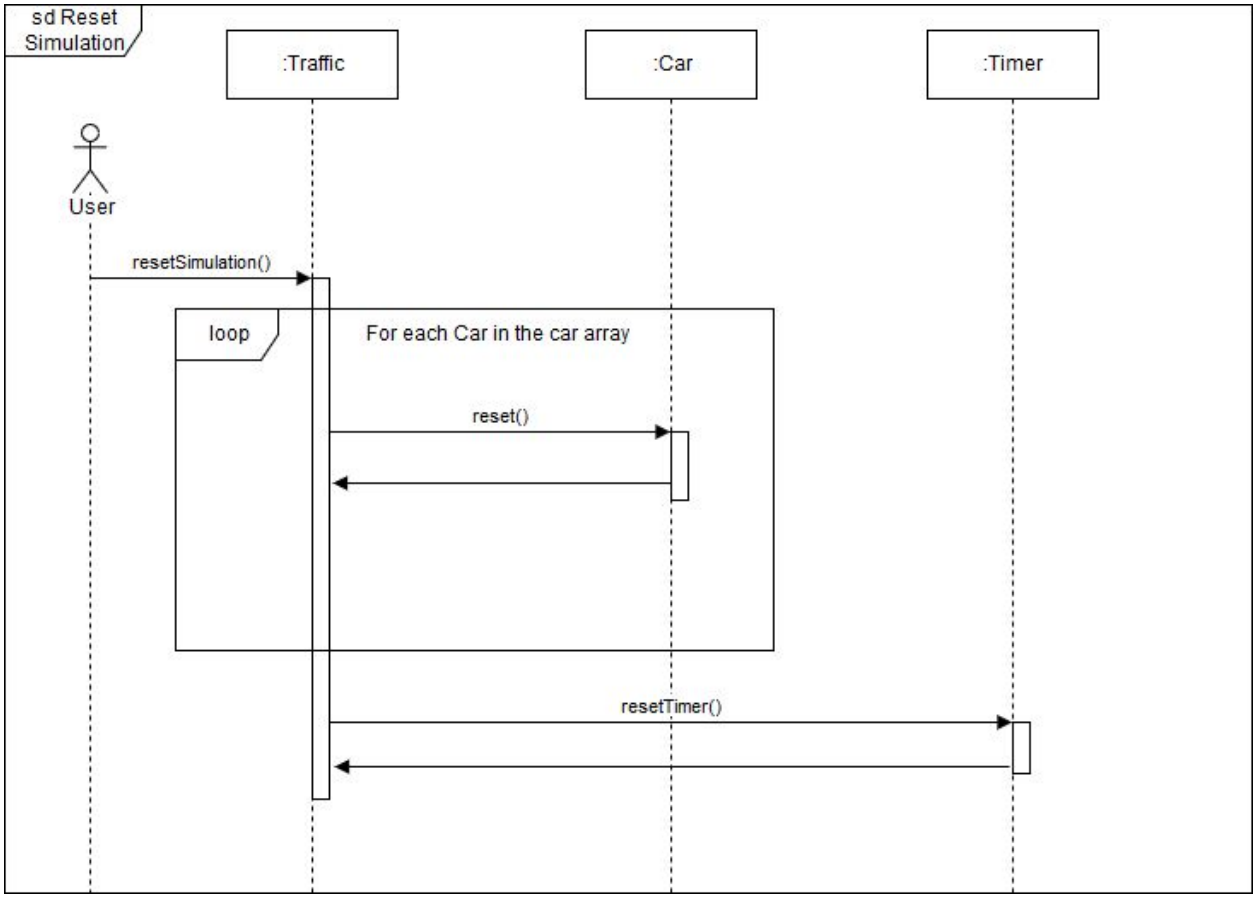
Structural Model



Behavioral Model


sd Start Stimulation
Traffic
Timer
Car
startStimulation()
startTimer()
loop
for each Car in car array
start()
if allArrived
stopTimer()
endStimulation()


sd Change Intersection
:Map
:Intersection
User
getIntersection(id)
changeIntersection(type)


sd Reset Simulation
:Traffic
:Car
:Timer
User
resetSimulation()
loop
For each Car in the car array
reset()
resetTimer()

Proposed System

Overview

The operator will be able to determine the start and end points of each vehicle and add more vehicles outside of the application. The operator will then be able to set what each intersections setting is by clicking on a button on each intersection that cycles through timer-based traffic lights, sensor-based traffic lights, and each type of stop sign. They may then run the simulation and start a timer. When all cars reach their destination, the timer will stop and the time taken will be recorded. The operator may then change the types of intersections and run the simulation again, and the new time will be recorded as well. The timer-based traffic lights will change every two minutes, while the sensor-based traffic lights will change dynamically depending on the flow of traffic at that particular intersection.

Functional Requirements

Be sure to provide unique requirement identifiers for traceability

F1. The operator must be able to set intersections to be all way stop signs, north-south stop signs, east-west stop signs, timer-based traffic lights, or sensor-based traffic lights.

F2. The operator must be able to set start and end points at certain dots on the map for each vehicle, which is accomplished via the Add Cars button.

F3. The operator must be able to reset the simulation, putting all vehicles back at the starting location. The simulation will then wait for another input, after the operator has finished altering the intersections, to start the simulation again.

F4. When the operator presses the Start button, the simulation will run, at each intersection the cars will obey the selected type of intersection, with timer-based traffic lights changing every 2 minutes, and sensor-based traffic lights changing based on the following conditions:

1. If the light is currently green, and there is no cross-traffic, the light will remain green
2. If the light is red for a car when it arrives at the intersection, and there is no cross-traffic, the light will change to green for that car
3. If there is traffic in both directions, the light will operate as a timer-based traffic light does

Nonfunctional Requirements

Be sure to provide unique requirement identifiers for traceability

N1. The system must support several vehicles moving at once.

N2. Vehicles should take the same route to their destination as long as their start and end points remain unchanged.

N3. All inputs from the user should be processed within 1 second.

N4. Times taken for all vehicles to reach their destinations are tracked throughout the session.

N5. Intersections cannot be changed in the middle of a simulation.

Glossary

Destination- End point of the car.

Dots- Points on the map, the space between each dot represents ⅛ mile.

Intersection- Any location where a car has more than one option for its next direction.

Pacopolis- The city we are using the map of for our program.