MINISTERUL EDUCAȚIEI
UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA
FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

PROJECT REPORT

Distributed Control Systems

Authors: **Trif Diana-Nicoleta (30343)**
Oprea Sergiu-Daniel (30343)
Pleșca Evelyn-Iulia (30342)

Instructor: **As. Dr. Ing. Dahlia AL-JANABI**

2024

Cuprins

1 Introduction

1.1 General Context

The purpose of this project is modelling and controlling two real-life intersections (our Plant) randomly assigned per team with the use of OETPNs which communicate via network.

The intersections which were modelled in our project are: https://maps.app.goo.gl/BaDmMauP535nbKxy9 and https://maps.app.goo.gl/5ntoHkVLeHPCPEFp6.

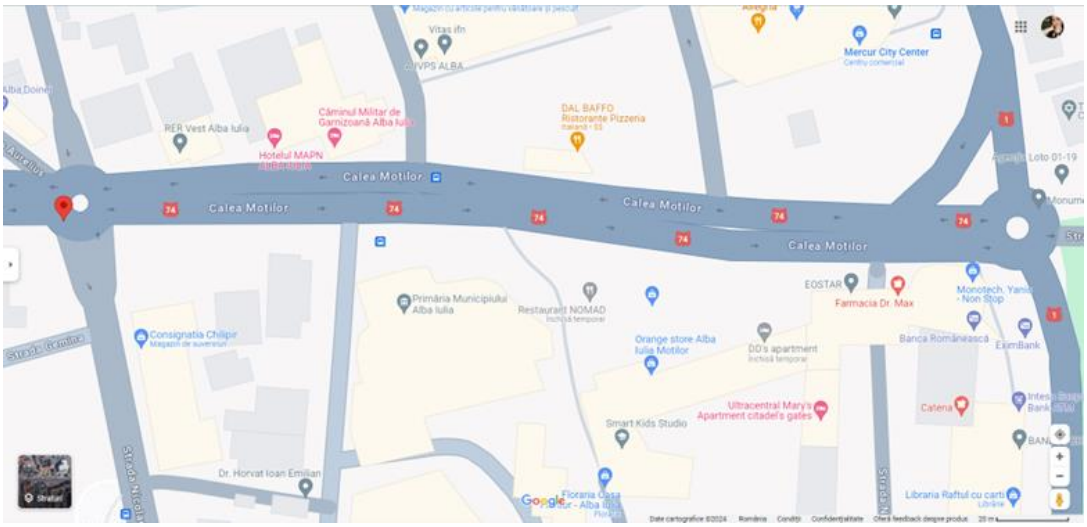

Figure 1 - Both intersections and the middle road connecting them, as seen on Google Maps

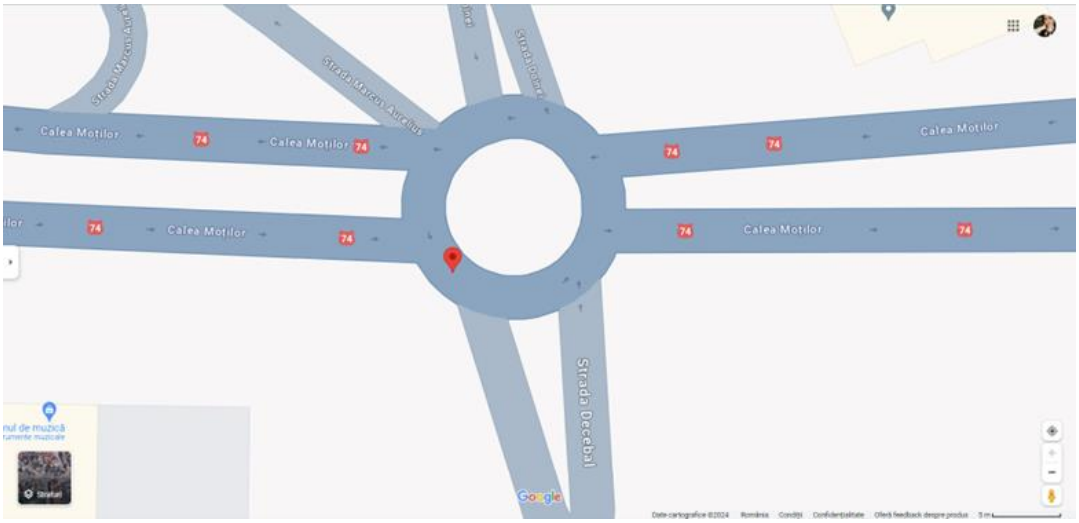

Figure 2 - First intersection, as seen on Google Maps

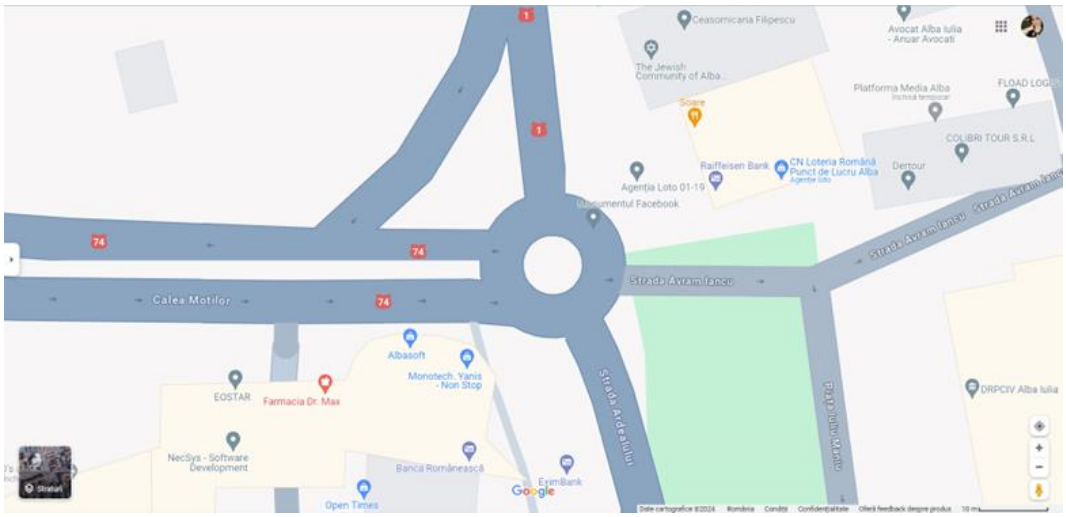

Figure 3 - Second intersection, as seen on Google Maps

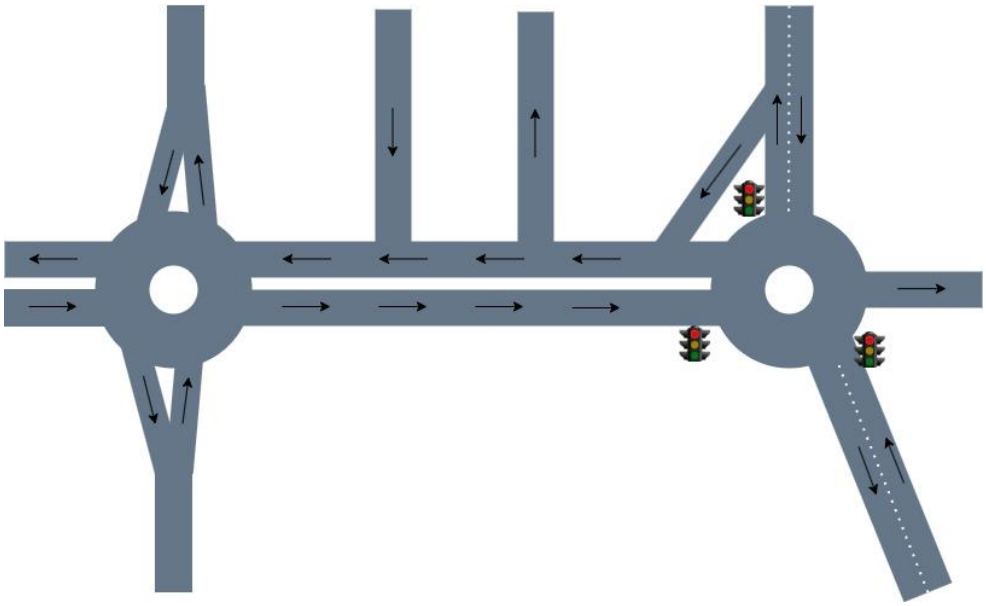

Figure 4 – Simplified map, graphical model of the two intersections and the middle lane that connects them

The Intersections & Traffic modelling chapter includes the OETPN model, with all the places, transitions, guards and maps included in the project and diagrams and Petri Nets that we modelled after our system.

Intersection Traffic Controllers includes the controller used for modelling the behaviour of the traffic lights.

Implementation includes code snippets from the project.

Logs and Screenshots include the logs and the screenshots which prove the functionality of our project, as seen after the testing of the application.

1.2 Objectives

Our objectives for this projects were simplifying the real-life intersections, providing a graphical representation of them and modelling the intersections using Petri nets and Java and the OETPN Framework provided during the laboratories.

Our Plant will have input and output channels used for modelling it, as well as a controller which interacts with the Plant and determines the evolution of the system as required by the specification requirements. The model is created afterwards, formally describing the required behaviour of the Plant.

2 Intersections & Trafic behaviour

2.1 Petri Nets

Below are attached the Petri Nets representing the intersections.

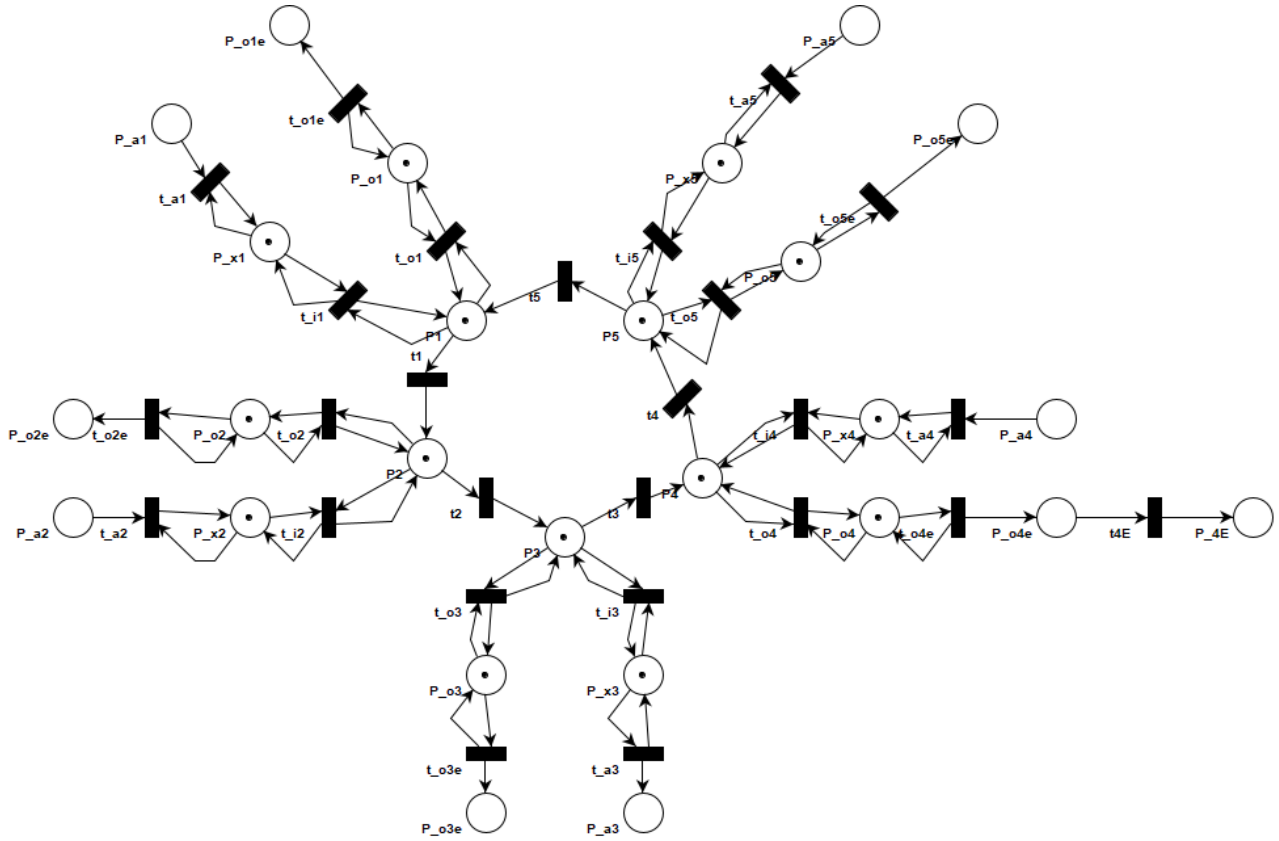

Figure 5 - Intersection 1 modelled using Petri Nets

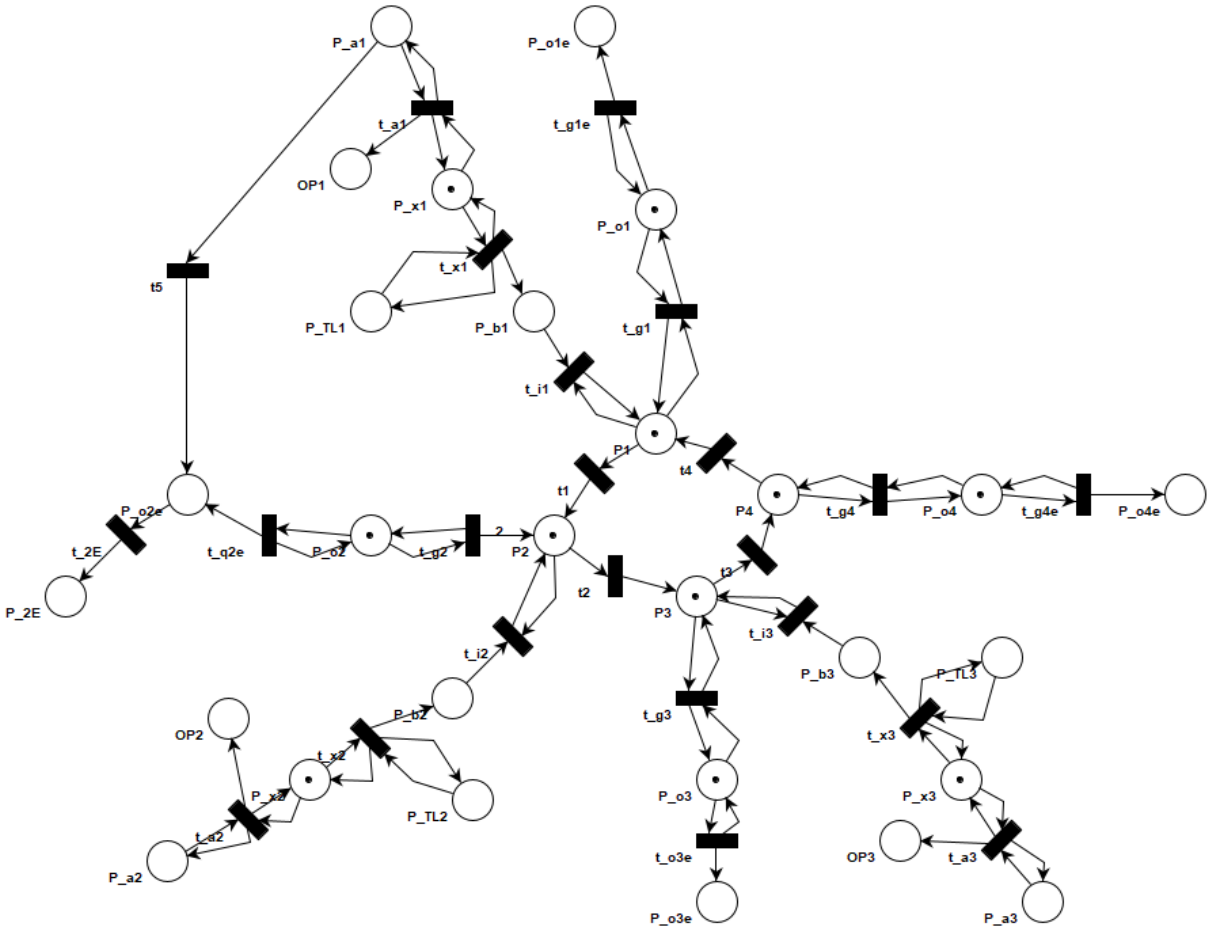

Figure 6 - Intersection 2 modelled using Petri Nets

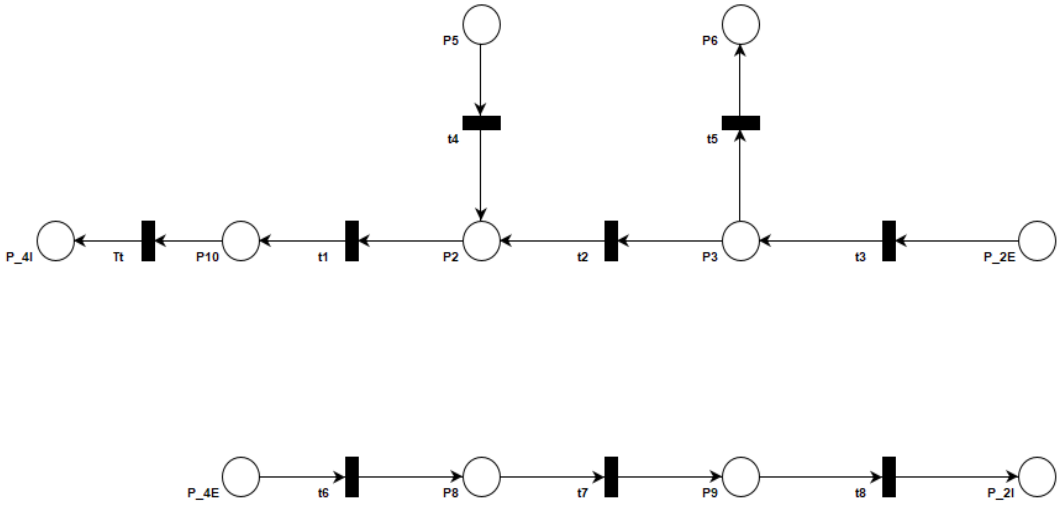

Figure 7 - Middle road, modelled using Petri Nets

2.2 OETPN model

Intersection 1:

In order to model the behavior of the traffic, we chose the following places and transitions for Intersection 1.

Places:

- Data Car types: P_a1, P_a2, P_a3, P_a4, P_a5, P_o1e, P_o2e, P_o3e, P_o4e, P_o5e
- Data Transfer type: P_4E
- Data Car Queue type: P_o1, P_o2, P_o3, P_o4, P_o5, P_x1, P_x2, P_x3, P_x4, P_x5, P1, P2, P3, P4, P5

Transitions:

- t1:
    - Input places: P1
    - grd: (m(P1).HaveCarForMe) AND (m(P2).CanAddCars)
    - map: m(P1).PopElementWithTargetToQueue (m(P2))
- t2:
    - Input places: P2
    - grd: (m(P2).HaveCarForMe) AND (m(P3).CanAddCars)
    - map: m(P2).PopElementWithTargetToQueue (m(P3))
- t3:
    - Input places: P3
    - grd: (m(P3).HaveCarForMe) AND (m(P4).CanAddCars)
    - map: m(P3).PopElementWithTargetToQueue (m(P4))
- t4:
    - Input places: P4
    - grd: (m(P4).HaveCarForMe) AND (m(P5).CanAddCars)
    - map: m(P4).PopElementWithTargetToQueue (m(P5))
- t5:
    - Input places: P5
    - grd: (m(P5).HaveCarForMe) AND (m(P1).CanAddCars)
    - map: m(P5).PopElementWithTargetToQueue (m(P1))
- t_i1:
    - Input places: P1, P_x1
    - grd (m(P_x1).HaveCarForMe) AND (m(P1).CanAddCars)
    - map: (m(P_x1).PopElementWithTargetToQueue (m(P1))
- t_i2:
    - Input places: P2, P_x2
    - grd: (m(P_x2).HaveCarForMe) AND (m(P2).CanAddCars)
    - map: (m(P_x2).PopElementWithTargetToQueue (m(P2))
- t_i3:
    - Input places: P3, P_x3
    - grd: (m(P_x3).HaveCarForMe) AND (m(P3).CanAddCars)

    - map: (m(P_x3).PopElementWithTargetToQueue (m(P3))
- t_i4:
    - Input places: P4, P_x4
    - grd: (m(P_x4), HaveCarForMe) AND (m(P4), CanAddCars)
    - map: (m(P_x4), PopElementWithTargetToQueue (m(P4))
- t_i5:
    - Input places: P5, P_x5
    - grd: (m(P_x5), HaveCarForMe) AND (m(P5), CanAddCars)
    - map: (m(P_x5), PopElementWithTargetToQueue(m(P5))
- t_o1:
    - Input places: P_o1, P1
    - grd: (m(P1).HaveCarForMe) AND (m(P_o1).CanADDCars)
    - map: (m(P1).PopElementWithTargetToQueue(m(P_o1))
- t_o2:
    - Input places: P_o2, P2
    - grd: (m(P2).HaveCarForMe) AND (m(P_o2).CanADDCars)
    - map: (m(P2).PopElementWithTargetToQueue(m(P_o2))
- t_o3:
    - Input places: P_o3, P3
    - grd: (m(P3).HaveCarForMe) AND (m(P_o3).CanADDCars)
    - map: (m(P3).PopElementWithTargetToQueue(m(P_o3))
- t_o4:
    - Input places: P_o4, P4
    - grd: (m(P4).HaveCarForMe) AND (m(P_o4).CanADDCars)
    - map: (m(P4).PopElementWithTargetToQueue(m(P_o4))
- t_o5:
    - Input places: P_o5, P5
    - grd: (m(P5).HaveCarForMe) AND (m(P_o5).CanADDCars)
    - map: (m(P5).PopElementWithTargetToQueue(m(P_o5))
- t_a1:
    - Input places: P_a1, P_x1
    - grd1: (m(P_a1) ≠ Ø) AND (m(P_x1).CanAddCars)
    - map1: (m(P_a1).addElement(m(p_x1))
    - grd2: (m(P_a1) ≠ Ø) AND (m(P_x1).CanNotAddCars)
    - map2: (m(P_a1).Move(m(P_a1))
- t_a2:
    - Input places: P_a2, P_x2
    - grd1: (m(P_a2) ≠ Ø) AND (m(P_x2).CanAddCars)
    - map1: (m(P_a2).addElement(m(p_x2))
    - grd2: (m(P_a2) ≠ Ø) AND (m(P_x2).CanNotAddCars)
    - map2: (m(P_a2).Move(m(P_a2))
- t_a3:
    - Input places: P_a3, P_x3
    - grd1: (m(P_a3) ≠ Ø) AND (m(P_x3).CanAddCars)
    - map1: (m(P_a3).addElement(m(p_x3))

    - grd2: (m(P_a3) ≠ Ø) AND (m(P_x3).CanNotAddCars)
    - map2: (m(P_a3).Move(m(P_a3))
- t_a4:
    - Input places: P_a4, P_x4
    - grd1: (m(P_a4) ≠ Ø) AND (m(P_x4).CanAddCars)
    - map1: (m(P_a4).addElement(m(p_x4))
    - grd2: (m(P_a4) ≠ Ø) AND (m(P_x4).CanNotAddCars)
    - map2: (m(P_a4).Move(m(P_a4))

- t_a5:
    - Input places: P_a5, P_x5
    - grd1: (m(P_a5) ≠ Ø) AND (m(P_x5).CanAddCars)
    - map1: (m(P_a5).addElement(m(p_x5))
    - grd2: (m(P_a5) ≠ Ø) AND (m(P_x5).CanNotAddCars)
    - map2: (m(P_a5).Move(m(P_a5))
- t_o1e:
    - Input places: P_o1
    - grd_21: (m(P_o1).HaveCar)
    - map_21: m(P_o1).PopElementWithoutTarget(m(P_o1e))
- t_o2e:
    - Input places: P_o2
    - grd_22: (m(P_o2).HaveCar)
    - map_22: m(P_o2).PopElementWithoutTarget(m(P_o2e))
- t_o3e:
    - Input places: P_o3
    - grd_23: (m(P_o3).HaveCar)
    - map_23: m(P_o3).PopElementWithoutTarget(m(P_o3e))
- t_o4e:
    - Input places: P_o4
    - grd_24: (m(P_o4).HaveCar)
    - map_24: m(P_o4).PopElementWithoutTarget(m(P_o4e))
- t_o5e:
    - Input places: P_o5
    - grd_25: (m(P_o5).HaveCar)
    - map_25: m(P_o5).PopElementWithoutTarget(m(P_o5e))
- t_4E:
    - Input place: P_o4e
    - grd_27: (m(P_o4E) ≠ Ø)
    - map_27: m(P_o4E).SendOverNetwork(), m(P_4E) = m(P_4E)

Intersection 2:

The Intersection 2 transitions are mostly similar to the ones of the first Intersection with the exception of the transitions on the lanes that have traffic lights.

Places:

- Data Car types: P_a1, P_a2, P_a3, P_o1e, P_o2e, P_o3e, P_o4e, P_b1, P_b2, P_b3
- Data Transfer type: P_2E, OP1, OP2, OP3
- Data String type: P_TL1, P_TL2, P_TL3
- Data Car Queue type: P_o1, P_o2, P_o3, P_o4, P_x1, P_x2, P_x3, P1, P2, P3, P4

Transitions:

- t_a1 (same t_a2, t_a3):
    - Input places: P_TL1, P_x1
    - grd1: (m(P_a1) ≠ Ø) AND (m(P_x1).CanAddCars)
    - map1: (m(P_a1).addElement(m(p_x1))
    - grd2: (m(P_a1) ≠ Ø) AND (m(OP1).CanNotAddCars)
    - map2: (m(P_a1).Move(m(P_a1)), m(OP1).SendOverNetwork(full)
- t_x1(same t_x2, t_x3):
    - Input places: P_a1, P_x1
    - grd1: (m(P_x1).HaveCar And m(P_TL1)=green)
    - map: m(P_x1).PopElementWithoutTarget() (m(P_b1)); m(P_TL1).Move() m(P_TL1)
- t5:
    - Input places: P_a1
    - grd: (m(P_a1) ≠ Ø)
    - map: (m(P_a1).addElement(m(P_o2e))

Middle street:

Places:

- Data Car types: P5, P6, P10, P_2E, P9, P_4E
- Data Transfer type: P_4I, P_2I
- Data Car Queue type: P2, P3, P8

Transitions:

- t1:
    - Input places: P2
    - grd_1: (m(P2), HaveCar))
    - map_1: m(P2), PopElementWithoutTarget(P10)
- tt:
    - Input places: P10
    - grd_2: (m(P10), NotNull))

    - map_2: m(P10).SendOverNetwork, m(P_4I) = m(P_4I)
- t2:
    - Input places: P3
    - grd_3: (m(P3), HaveCarForMe AND m(P2), CanAddCars))
    - map_3: m(P3), PopElementWithTargetToQueue(P2)
- t3:
    - Input places: P_2E
    - grd_4: (m(P_2E), NotNull AND m(P3), CanAddCars))
    - map_4: m(P_2E).AddElement(m(P3))
- t4:
    - Input places: P5
    - grd_5: (m(P5), NotNull AND m(P2), CanAddCars))
    - map_5: m(P5).AddElement(m(P2))
- t5:
    - Input places: P3
    - grd_6: (m(P3), HaveCar))
    - map_6: m(P3), PopElementWithoutTarget(P6)
- t6:
    - Input places: P_4E
    - grd_7: (m(P_4E), NotNull AND m(P8), CanAddCars))
    - map_7: m(P_4E).AddElement(m(P8))
- t7:
    - Input places: P8
    - grd_8: (m(P8), HaveCar))
    - map_8: m(P8), PopElementWithoutTarget(P9)
- t8:
    - Input places: P9
    - grd_9: (m(P9), NotNull)
    - map_9: m(P9).SendOverNetwork, m(P_2I) = m(P_2I)

3 Intersection Traffic Controllers

3.1 Petri Nets

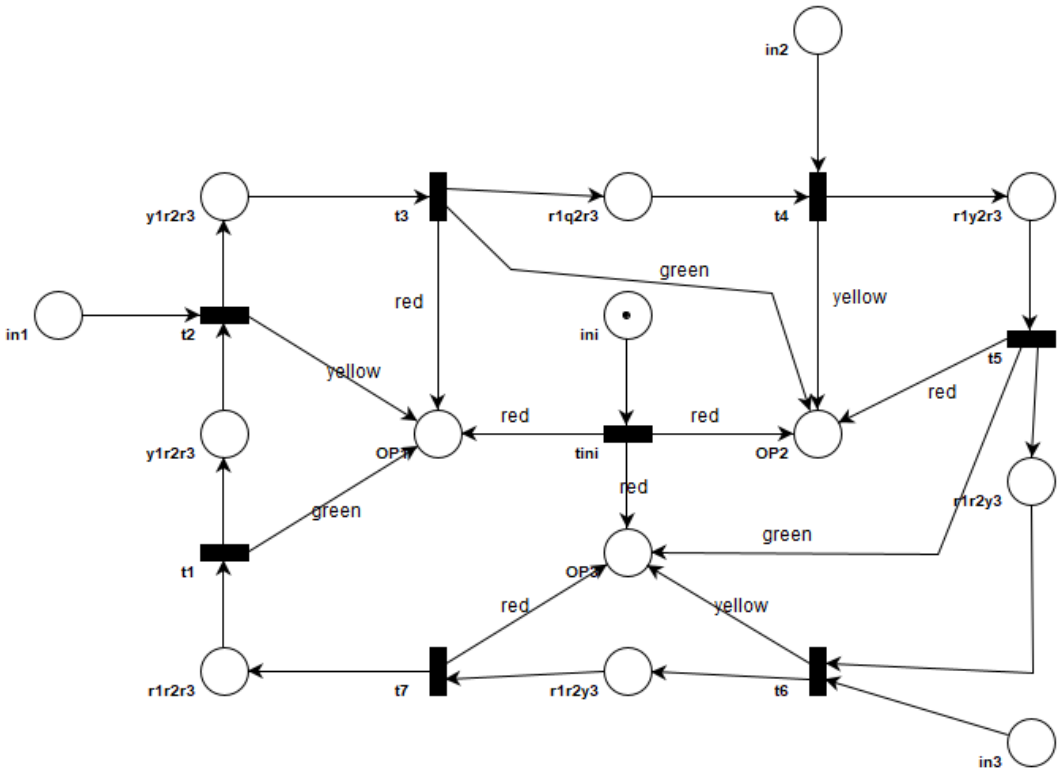

Figure 8 - Controller, modelled using Petri Nets

3.2 OETPN model

Places:

- Data String types: ini, red, green, yellow, in1, in2, in3, r1r2r3, g1r2r3, r1g2r3, r1r2g3, r1r2y3, y1r2r3, r1y2r3
- Data Integer: Five, Ten
- Data Transfer type: OP1, OP2, OP3

Transitions:

- tini:
    - Input places: ini
    - grd_1: (m(ini) ≠ Ø)
    - map_1: m(ini).SendOverNetwork(), m(OP1) = m(OP1),

m(ini).SendOverNetwork(), m(OP2) = m(OP2),
m(ini).SendOverNetwork(), m(OP3) = m(OP3),
m(ini).MakeNull(ini)

- t1:
  - Input places: r1r2r3
  - grd_2: (m(r1r2r3) ≠ Ø)
  - map_2: m(r1r2r3).Move(m(g1r2r3)),
    m(r1r2r3).SendOverNetwork(), m(OP1) = m(OP1)
- t2:
  - Input places: g1r2r3, in1
  - grd_3: (m(g1r2r3) ≠ Ø AND m(in1) = Ø)
  - map_3: m(g1r2r3).Move(m(y1r2r3)),
    yellow.SendOverNetwork(), m(OP1) = m(OP1),
    Five.DynamicDelay()
  - grd_4: (m(g1r2r3) ≠ Ø AND m(in1) ≠ Ø)
  - map_4: m(g1r2r3).Move(m(y1r2r3))
    yellow.SendOverNetwork(), m(OP1) = m(OP1),
    Ten.DynamicDelay()
- t3:
  - Input places: y1r2r3
  - grd_5: (m(y1r2r3) ≠ Ø)
  - map_5: m(y1r2r3).Move(m(r1g2r3))
    red.SendOverNetwork(), m(OP1) = m(OP1),
    green.sendOverNetwork(), m(OP2) = m(OP2)
- t4:
  - Input places: r1g2r3, in2
  - grd_6: (m(r1g2r3) ≠ Ø AND m(in2) = Ø)
  - map_6: m(r1g2r3).Move(m(r1y2r3)),
    yellow.SendOverNetwork(), m(OP2) = m(OP2),
    Five.DynamicDelay()
  - grd_7: (m(r1g2r3) ≠ Ø AND m(in2) ≠ Ø)
  - map_7: m(r1g2r3).Move(m(r1y2r3))
    yellow.SendOverNetwork(), m(OP2) = m(OP2),
    Ten.DynamicDelay()
- t5:
  - Input places: r1y2r3
  - grd_5: (m(r1y2r3) ≠ Ø)
  - map_5: m(r1y2r3).Move(m(r1r2g3))
    red.SendOverNetwork(), m(OP1) = m(OP2),
    green.sendOverNetwork(), m(OP2) = m(OP3)
- t6:
  - Input places: r1r2g3, in3
  - grd_6: (m(r1r2g3) ≠ Ø AND m(in3) = Ø)
  - map_6: m(r1r2g3).Move(m(r1r2y3)),
    yellow.SendOverNetwork(), m(OP3) = m(OP3),

Five.DynamicDelay()

- grd_7: (m(r1r2g3) ≠ Ø AND m(in3) ≠ Ø)
- map_7: m(r1r2g3).Move(m(r1r2y3))
  yellow.SendOverNetwork(), m(OP3) = m(OP3),
  Ten.DynamicDelay()

- t7:
  - Input places: r1r2y3
  - grd_5: (m(r1r2y3) ≠ Ø)
  - map_5: m(r1r2y3).Move(m(r1r2r3))
    red.SendOverNetwork(), m(OP3) = m(OP3)

4 Component Diagram

The component diagram that shows the way the two intersections communicate with each other through input/output ports and how the Controller interacts with Intersection 2, is presented bellow:

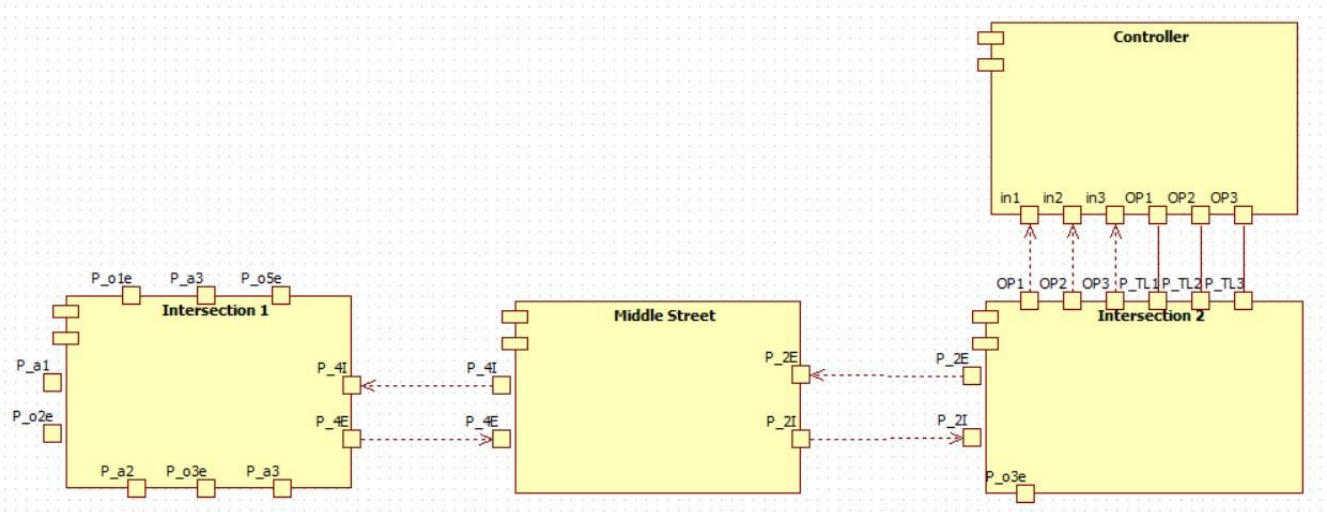

Figure 9 - Component diagram of the entire system

5 Implementation

For our implementation, we extended the functionality of the existing OETPN framework and added our own classes which were needed for solving the given problem.

We added the Controller_Intersection2, Intersection1, Intersection2 and MiddleStreet classes. All of these are added into the FINAL_PROJECT folder.

We used Github for source control and the implementation of our project can be found at the following repository link: https://github.com/DianaT08/DCS_Project.

5.1 Controller_Intersection2 class

We implemented the controller that communicates with the second intersection in this class.

We implemented the transitions presented in Chapter 2 of this document here.

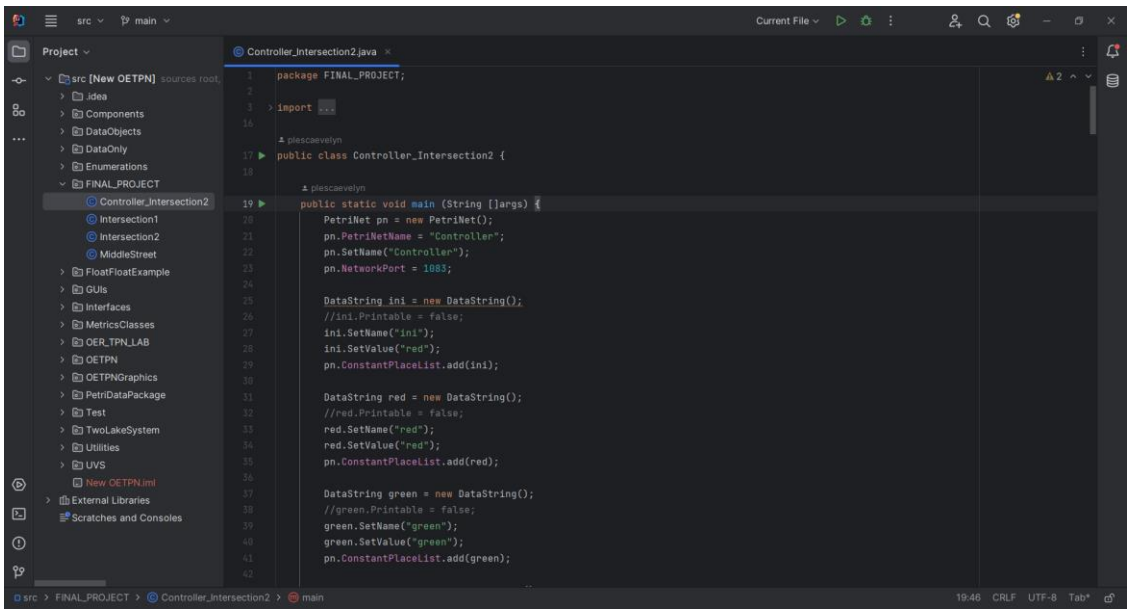

Figure 10 - Screenshot from the implementation of Controller_Intersection2 class in IntelliJ

5.2 Intersection1 class

We implemented all the transitions related to the first intersection in the Intersection1 class. We added comments in the code in order to separate the code implementation of the places and transitions related to lanes, exit lanes and the intersection.

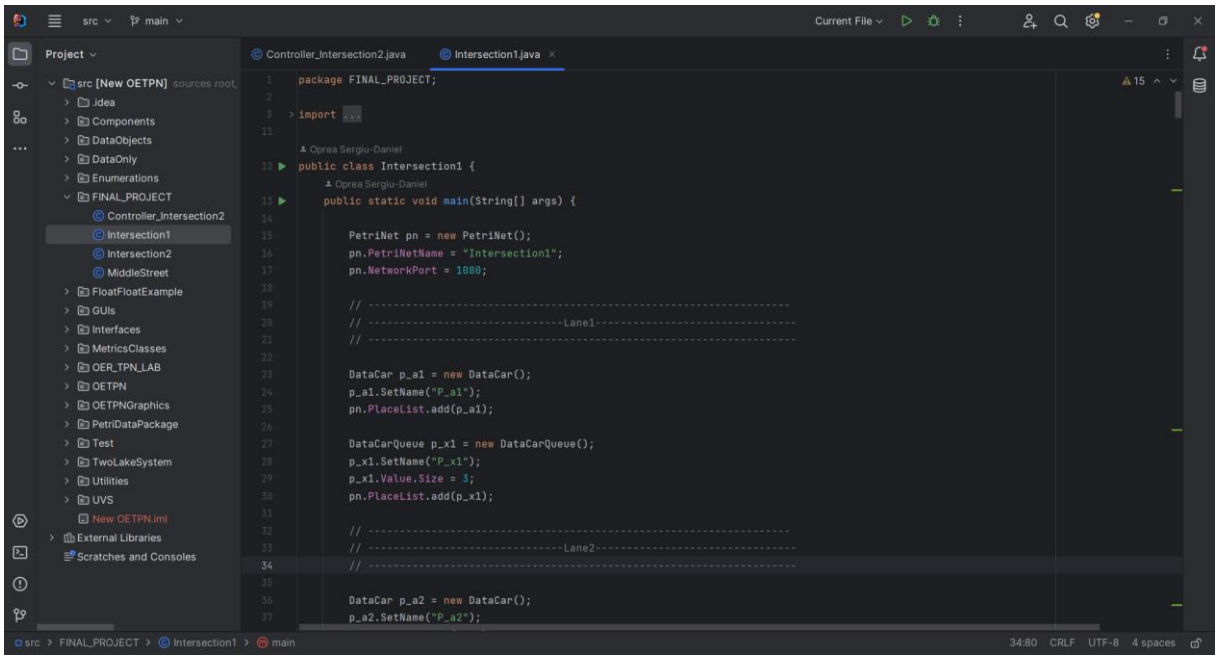

Figure 11 - Screenshot from the implementation of Intersection1 class in IntelliJ

5.3 Intersection2 class

We implemented the second intersection in this class, in a similar manner to the implementation of the first intersection.

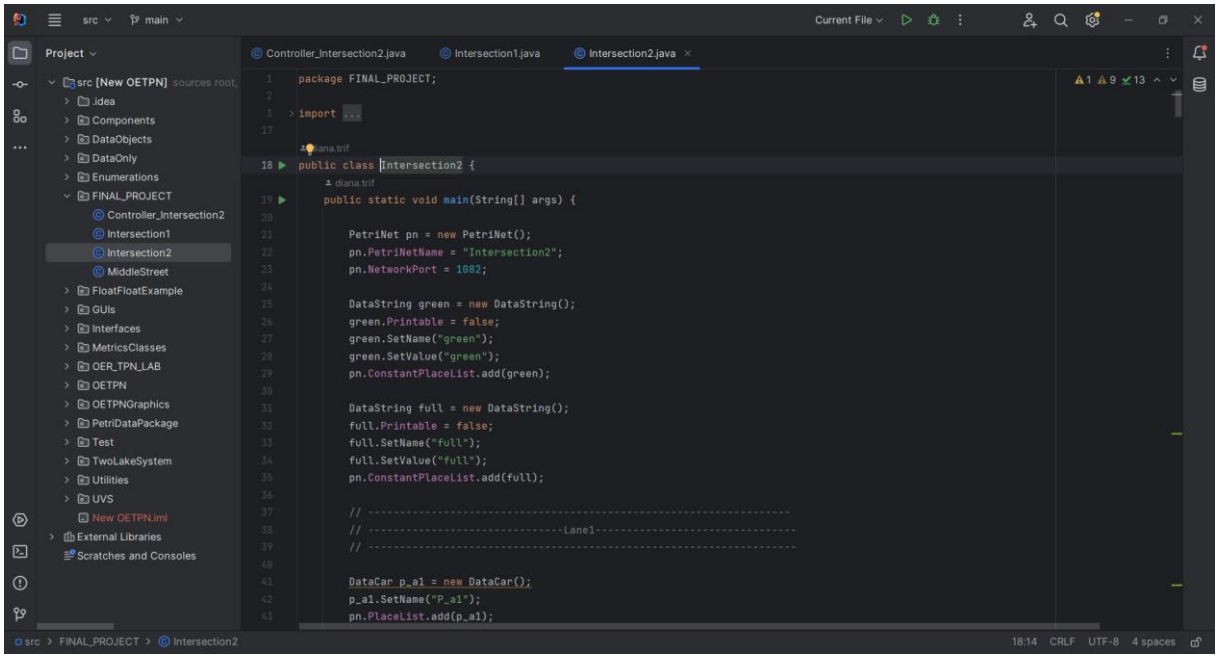

Figure 12 - Screenshot from the implementation of Intersection2 class in IntelliJ

5.4 MiddleStreet class

The middle street connecting the two intersections was implemented in this class.

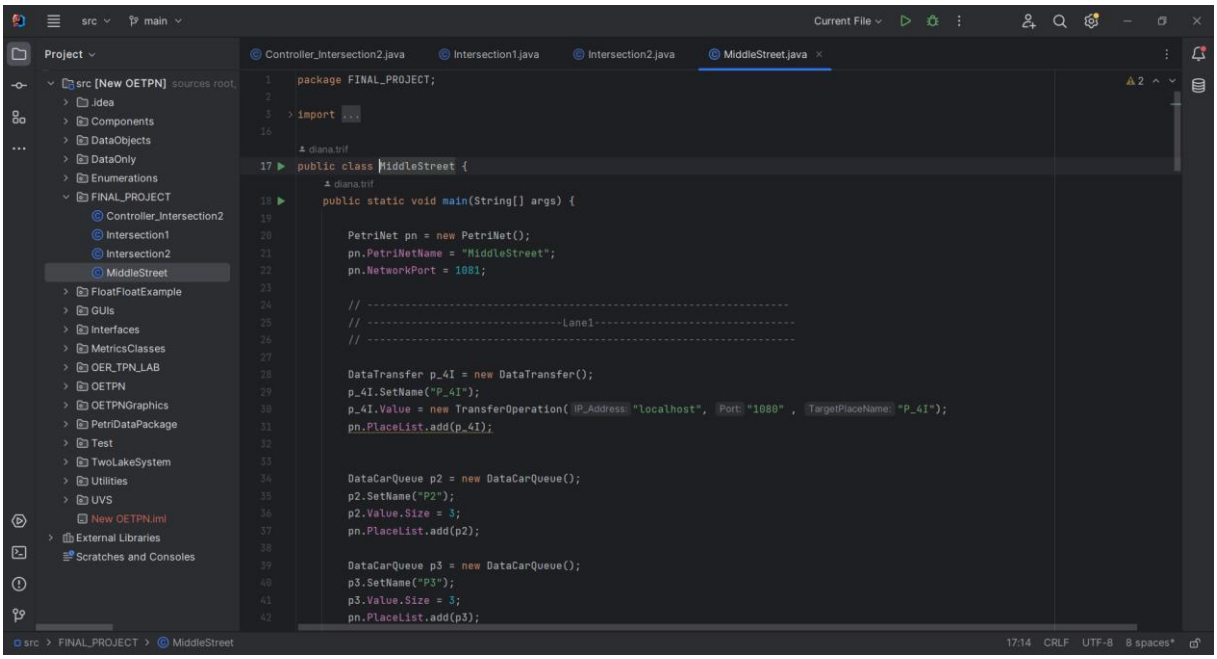

Figure 13 - Screenshot from the implementation of Intersection2 class in IntelliJ

6 Testing

6.1 Test 1

For this case, the car entered lane 1 from intersection 1, exited through lane exit 4, entered the middle street, after which it entered intersection 2 through lane 2 and exited through lane 4.

Screenshots of the execution can be found below.

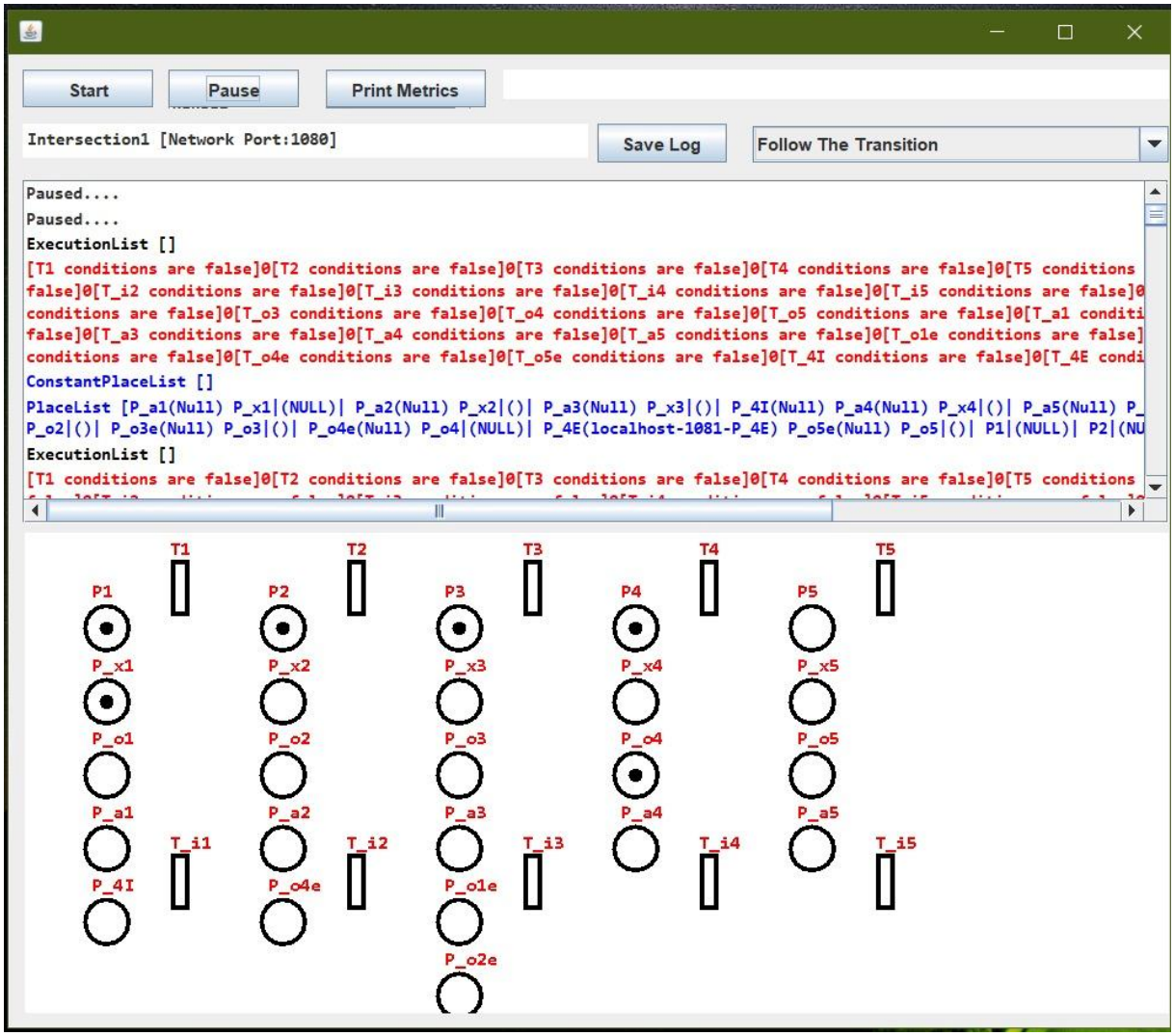

Figure 14 - Execution screenshot of test 1

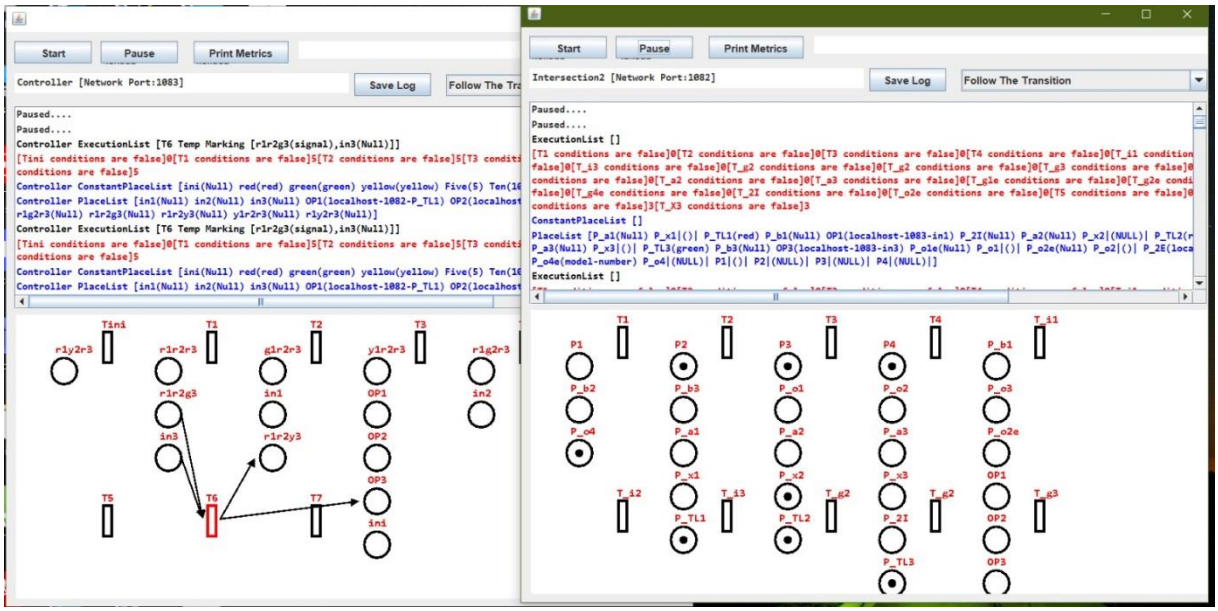

Figure 15 - Execution screenshot of test 1

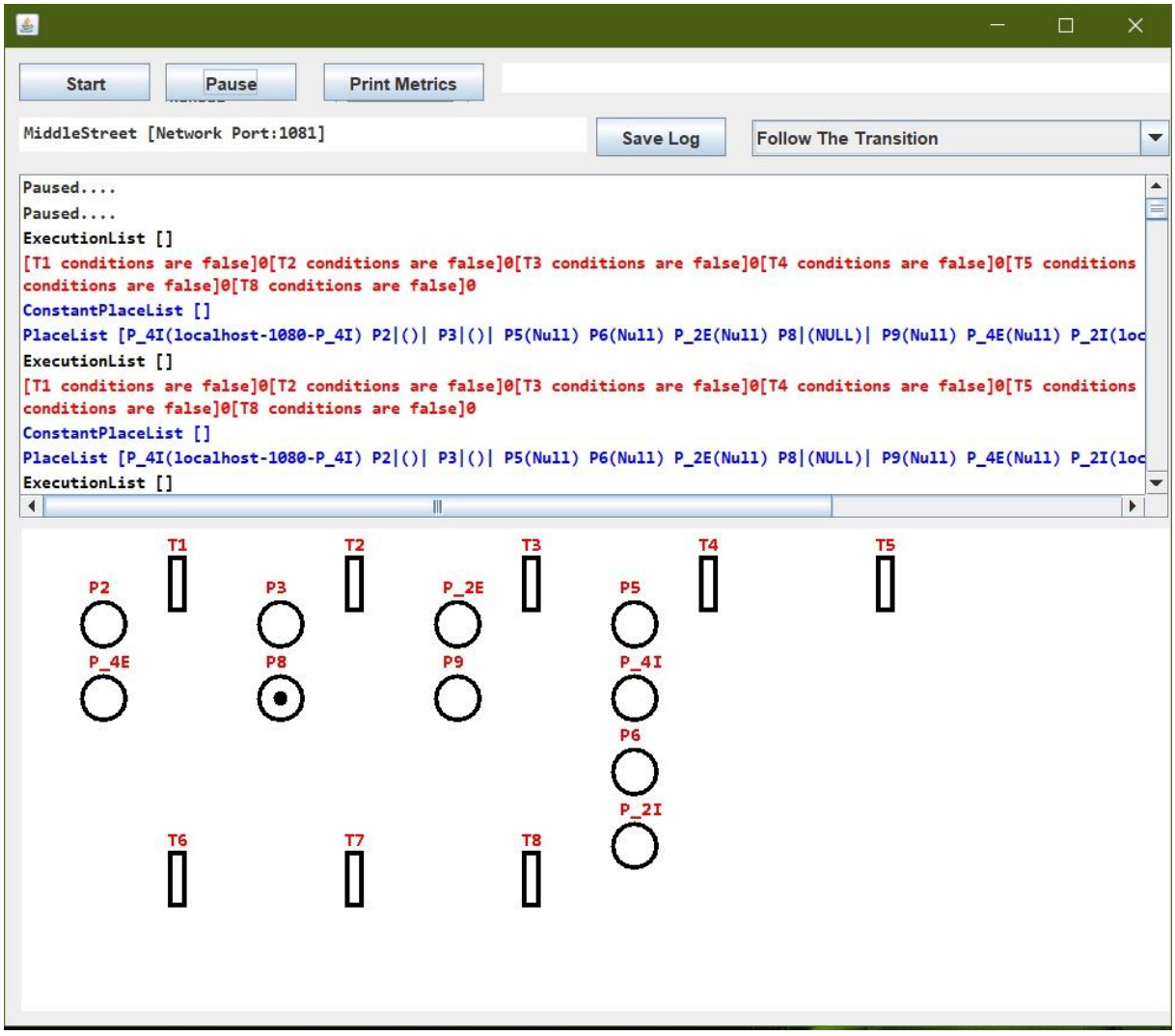

Figure 16 - Execution screenshot of test 1

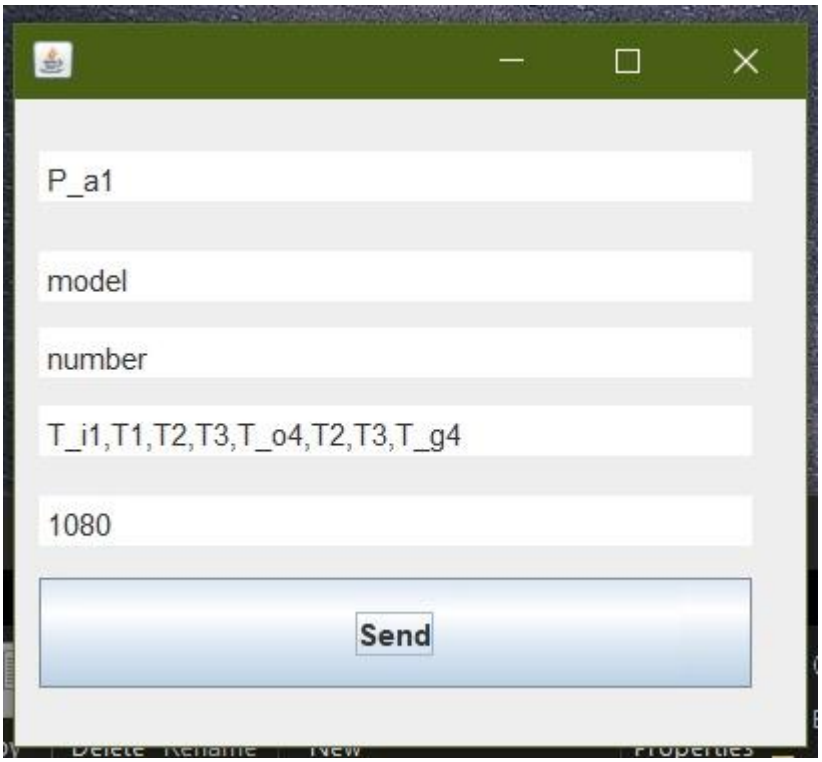

Figure 17 - Execution screenshot of test 1

6.2 Test 2 – Traffic Jam

For the second intersection, we created a traffic jam case by sending the maximum number of cars (three) to the input lane of the intersection, started the controller, then sent the last car.

The controller received a signal from the plant (intersection) and the transition that is responsible for sending a yellow light to the lane where the cars have been input changed the delay to 10 sec.

The controller OETPN ran until it reached the same transition (2 loops) to show that the delay is changed back to 5 sec.

Screenshots from the logs saved at this moment can be seen below.

```
[Tini conditions are false]0[T1 conditions are false]5[T2 conditions are false]10[T4 conditions are fa
Controller ConstantPlaceList [ini(Null)  red(red)  green(green)  yellow(yellow)  Five(5)  Ten(10)]
Controller PlaceList [in2(Null)  in3(Null)  OP1(localhost-1082-P_TL1)  OP2(localhost-1082-P_TL2)  OP3(
Controller ExecutionList [T3 Temp Marking [y1r2r3(signal)]]
[Tini conditions are false]0[T1 conditions are false]5[T2 conditions are false]10[T4 conditions are fa
Controller ConstantPlaceList [ini(Null)  red(red)  green(green)  yellow(yellow)  Five(5)  Ten(10)]
Controller PlaceList [in2(Null)  in3(Null)  OP1(localhost-1082-P_TL1)  OP2(localhost-1082-P_TL2)  OP3(
Controller ExecutionList [T3 Temp Marking [y1r2r3(signal)]]
[Tini conditions are false]0[T1 conditions are false]5[T2 conditions are false]10[T4 conditions are fa
Controller ConstantPlaceList [ini(Null)  red(red)  green(green)  yellow(yellow)  Five(5)  Ten(10)]
Controller PlaceList [in2(Null)  in3(Null)  OP1(localhost-1082-P_TL1)  OP2(localhost-1082-P_TL2)  OP3(
Controller ExecutionList [T3 Temp Marking [y1r2r3(signal)]]
[Tini conditions are false]0[T1 conditions are false]5[T2 conditions are false]10[T4 conditions are fa
Controller ConstantPlaceList [ini(Null)  red(red)  green(green)  yellow(yellow)  Five(5)  Ten(10)]
Controller PlaceList [in2(Null)  in3(Null)  OP1(localhost-1082-P_TL1)  OP2(localhost-1082-P_TL2)  OP3(
Controller ExecutionList [T3 Temp Marking [y1r2r3(signal)]]
[Tini conditions are false]0[T1 conditions are false]5[T2 conditions are false]10[T4 conditions are fa
Controller ConstantPlaceList [ini(Null)  red(red)  green(green)  yellow(yellow)  Five(5)  Ten(10)]
Controller PlaceList [in2(Null)  in3(Null)  OP1(localhost-1082-P_TL1)  OP2(localhost-1082-P_TL2)  OP3(
Controller ExecutionList [T2 Temp Marking [g1r2r3(signal),in1(full)]]
[Tini conditions are false]0[T1 conditions are false]5[T3 conditions are false]5[T4 conditions are fal
Controller ConstantPlaceList [ini(Null)  red(red)  green(green)  yellow(yellow)  Five(5)  Ten(10)]
Controller PlaceList [in2(Null)  in3(Null)  OP1(localhost-1082-P_TL1)  OP2(localhost-1082-P_TL2)  OP3(
Controller ExecutionList [T2 Temp Marking [g1r2r3(signal),in1(full)]]
```

Figure 18 - Screenshot from the log saved after executing test 2

```
Controller ExecutionList [T3 Temp Marking [y1r2r3(signal)]]
[Tini conditions are false]0[T1 conditions are false]5[T2 conditions are false]5[T4 conc
Controller ConstantPlaceList [ini(Null)  red(red)  green(green)  yellow(yellow)  Five(5)
Controller PlaceList [in2(Null)  in3(Null)  OP1(localhost-1082-P_TL1)  OP2(localhost-108
Controller ExecutionList [T3 Temp Marking [y1r2r3(signal)]]
[Tini conditions are false]0[T1 conditions are false]5[T2 conditions are false]5[T4 conc
Controller ConstantPlaceList [ini(Null)  red(red)  green(green)  yellow(yellow)  Five(5)
Controller PlaceList [in2(Null)  in3(Null)  OP1(localhost-1082-P_TL1)  OP2(localhost-108
Controller ExecutionList [T3 Temp Marking [y1r2r3(signal)]]
[Tini conditions are false]0[T1 conditions are false]5[T2 conditions are false]5[T4 conc
Controller ConstantPlaceList [ini(Null)  red(red)  green(green)  yellow(yellow)  Five(5)
Controller PlaceList [in2(Null)  in3(Null)  OP1(localhost-1082-P_TL1)  OP2(localhost-108
Controller ExecutionList [T3 Temp Marking [y1r2r3(signal)]]
[Tini conditions are false]0[T1 conditions are false]5[T2 conditions are false]5[T4 conc
Controller ConstantPlaceList [ini(Null)  red(red)  green(green)  yellow(yellow)  Five(5)
Controller PlaceList [in2(Null)  in3(Null)  OP1(localhost-1082-P_TL1)  OP2(localhost-108
Controller ExecutionList [T3 Temp Marking [y1r2r3(signal)]]
[Tini conditions are false]0[T1 conditions are false]5[T2 conditions are false]5[T4 conc
Controller ConstantPlaceList [ini(Null)  red(red)  green(green)  yellow(yellow)  Five(5)
Controller PlaceList [in2(Null)  in3(Null)  OP1(localhost-1082-P_TL1)  OP2(localhost-108
Controller ExecutionList [T2 Temp Marking [g1r2r3(signal),in1(Null)]]
[Tini conditions are false]0[T1 conditions are false]5[T3 conditions are false]5[T4 conc
Controller ConstantPlaceList [ini(Null)  red(red)  green(green)  yellow(yellow)  Five(5)
Controller PlaceList [in2(Null)  in3(Null)  OP1(localhost-1082-P_TL1)  OP2(localhost-108
Controller ExecutionList [T2 Temp Marking [g1r2r3(signal),in1(Null)]]
[Tini conditions are false]0[T1 conditions are false]5[T3 conditions are false]5[T4 conc
Controller ConstantPlaceList [ini(Null)  red(red)  green(green)  yellow(yellow)  Five(5)
Controller PlaceList [in2(Null)  in3(Null)  OP1(localhost-1082-P_TL1)  OP2(localhost-108
```

Figure 19 - Screenshot from the log saved after executing test 2