Project Report

Distributed Control Systems

Authors: **Trif Diana-Nicoleta (30343)**

**Oprea Sergiu-Daniel (30343)**

**Pleșca Evelyn-Iulia (30342)**

Conducător științific: **As. Dr. Ing. Dahlia AL-JANABI**

Cuprins

[1 Introduction 2](#_Toc156080034)

[1.1 General Context 2](#_Toc156080035)

[1.2 Objectives 4](#_Toc156080036)

[2 Intersections & Trafic behaviour 4](#_Toc156080037)

[2.1 OETPN model 4](#_Toc156080038)

[2.2 Petri Nets 7](#_Toc156080039)

[3 Intersection Trafic Controllers 8](#_Toc156080040)

[4 Implementation 10](#_Toc156080041)

[5 Testing, Logs and Screenshots 11](#_Toc156080042)

[5.1 Formatarea paginii 11](#_Toc156080043)

[5.2 Titluri și stiluri 11](#_Toc156080044)

[5.3 Figuri, tabele și ecuații 11](#_Toc156080045)

[6 Conclusion 12](#_Toc156080046)

# Introduction

## 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>.

O imagine care conține captură de ecran, șubler, cântar

Descriere generată automat

Figure 1 - Simplified, graphical model of the two intersections

O imagine care conține captură de ecran, text

Descriere generată automat

Figure 2 - First intersection, as seen on Google Maps

O imagine care conține text, captură de ecran, hartă

Descriere generată automat

Figure 3 - Second intersection, as seen on Google Maps

O imagine care conține text, hartă

Descriere generată automat

Figure 4 - Both intersections and the middle road connecting them, as seen on Google Maps

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

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.

## 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.

# Intersections & Trafic behaviour

## Petri Nets

Below are attached the Petri Nets representing the intersections.

O imagine care conține schiță, diagramă, hartă

Descriere generată automat

Figure 5 - Intersection 1 modelled using Petri Nets

O imagine care conține schiță, desen, diagramă, hartă

Descriere generată automat

Figure 6 - Intersection 2 modelled using Petri Nets

O imagine care conține diagramă, linie

Descriere generată automat

Figure 7 - Middle road, modelled using Petri Nets

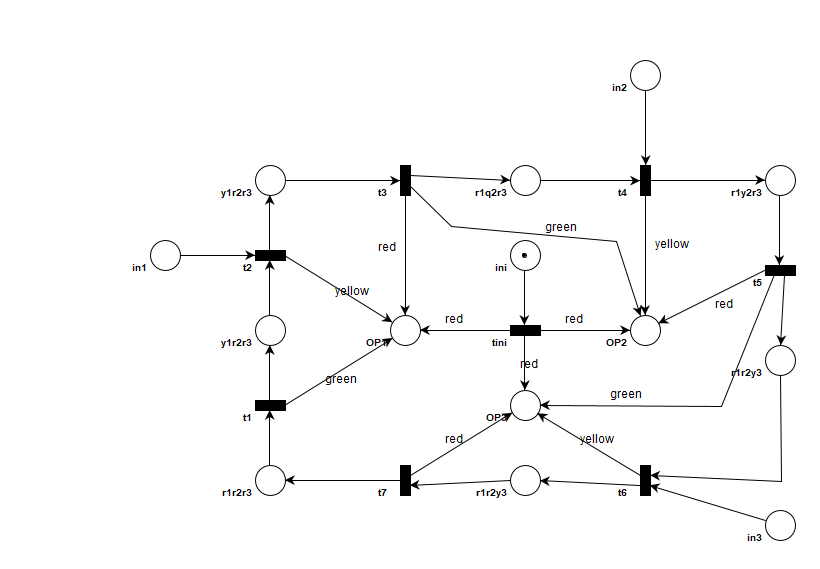


Figure 8 - Controller, modelled using Petri Nets

## OETPN model

In order to model the behaviour of the traffic, we chose the following places and transitions.

Places:

* Data Car types: P\_a1, P\_a2, P\_a3, P\_a4, P\_a5, P\_o1e, P\_o2e, P\_o3e, P\_o4e, P\_o5e, P\_4I
* 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\_1: (m(P1), HaveCarForMe) AND (m(P2), CanAddCars)
  + map\_1: m(P1), PopElementWithTargetToQueue (P2)
* t2:
  + Input places: P2
  + grd\_2: (m(P2), HaveCarForMe) AND (m(P3), CanAddCars)
  + map\_2: m(P2), PopElementWithTargetToQueue (P3)
* t3:
  + Input places: P3
  + grd\_3: (m(P3), HaveCarForMe) AND (m(P4), CanAddCars)
  + map\_3: m(P3), PopElementWithTargetToQueue (P4)
* t4:
  + Input places: P4
  + grd\_4: (m(P4), HaveCarForMe) AND (m(P5), CanAddCars)
  + map\_4: m(P4), PopElementWithTargetToQueue (P5)
* t5:
  + Input places: P5
  + grd\_5: (m(P5), HaveCarForMe) AND (m(P1), CanAddCars)
  + map\_5: m(P5), PopElementWithTargetToQueue (P1)
* t\_i1:
  + Input places: P1, P\_x1
  + grd\_6: (m(P\_x1), HaveCarForMe) AND (m(P1), CanAddCars)
  + map\_6: (m(P\_x1), PopElementWithTargetToQueue (P1))
* t\_i2:
  + Input places: P2, P\_x2
  + grd\_7: (m(P\_x2), HaveCarForMe) AND (m(P2), CanAddCars)
  + map\_7: (m(P\_x2), PopElementWithTargetToQueue (P2))
* t\_i3:
  + Input places: P3, P\_x3
  + grd\_8: (m(P\_x3), HaveCarForMe) AND (m(P3), CanAddCars)
  + map\_8: (m(P\_x3), PopElementWithTargetToQueue (P3))
* t\_i4:
  + Input places: P4, P\_x4
  + grd\_9: (m(P\_x4), HaveCarForMe) AND (m(P4), CanAddCars)
  + map\_9: (m(P\_x4), PopElementWithTargetToQueue (P4))
* t\_i5:
  + Input places: P5, P\_x5
  + grd\_10: (m(P\_x5), HaveCarForMe) AND (m(P5), CanAddCars)
  + map\_10: (m(P\_x5), PopElementWithTargetToQueue(P5))
* t\_o1:
  + Input places: P\_o1, P1
  + grd\_11: (m(P1), HaveCarForMe) AND (m(P\_o1), CanADDCars)
  + map\_11: (m(P1).PopElementWithTargetToQueue(m(P\_o1))
* t\_o2:
  + Input places: P\_o2, P2
  + grd\_12: (m(P2), HaveCarForMe) AND (m(P\_o2), CanADDCars)
  + map\_12: (m(P2).PopElementWithTargetToQueue(m(P\_o2))
* t\_o3:
  + Input places: P\_o3, P3
  + grd\_13: (m(P3), HaveCarForMe) AND (m(P\_o3), CanADDCars)
  + map\_13: (m(P3).PopElementWithTargetToQueue(m(P\_o3))
* t\_o4:
  + Input places: P\_o4, P4
  + grd\_14: (m(P4), HaveCarForMe) AND (m(P\_o4), CanADDCars)
  + map\_14: (m(P4).PopElementWithTargetToQueue(m(P\_o4))
* t\_o5:
  + Input places: P\_o5, P5
  + grd\_15: (m(P5), HaveCarForMe) AND (m(P\_o5), CanADDCars)
  + map\_15: (m(P5).PopElementWithTargetToQueue(m(P\_o5))
* t\_a1:
  + Input places: P\_a1, P\_x1
  + grd\_16: (m(P\_a1) ≠ Ø) AND (m(P\_x1), CanAddCars)
  + map\_16: (m(P\_a1).addElement(m(p\_x1))
* t\_a2:
  + Input places: P\_a2, P\_x2
  + grd\_17: (m(P\_a2) ≠ Ø) AND (m(P\_x2), CanAddCars)
  + map\_17: (m(P\_a2).addElement(m(p\_x2))
* t\_a3:
  + Input places: P\_a3, P\_x3
  + grd\_18: (m(P\_a3) ≠ Ø) AND (m(P\_x3), CanAddCars)
  + map\_18: (m(P\_a3).addElement(m(p\_x3))
* t\_a4:
  + Input places: P\_a4, P\_x4
  + grd\_19: (m(P\_a4) ≠ Ø) AND (m(P\_x4), CanAddCars)
  + map\_19: (m(P\_a4).addElement(m(p\_x4))
* t\_a5:
  + Input places: P\_a5, P\_x5
  + grd\_20: (m(P\_a5) ≠ Ø) AND (m(P\_x5), CanAddCars)
  + map\_20: (m(P\_a5).addElement(m(p\_x5))
* 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\_4I:
  + Input place: P\_41
  + grd\_26: (m(P\_4I) ≠ Ø) AND (m(P\_a4).CanAddCars)
  + map\_26: (m(P\_4I).addElement(m(p1))
* 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 Trafic Controllers

## Relation with the intersections

The second intersection has a closed-loop traffic controller that communicates with the model of the second intersection, as seen in *Figure 9* below.

O imagine care conține text, diagramă, linie, Plan

Descriere generată automat

Figure 9 - Component diagram of the intersections

In intersection 3, there are 3 traffic lights at the exit of the roundabout, which can be seen as ports: both the Controller and Intersection 2 have 6 ports. Controller has 3 input and 3 output ports and Intersection 2 has 3 output ports and 3 traffic light ports.

## OETPN model

# Implementation

# Testing, Logs and Screenshots

## Formatarea paginii

## Titluri și stiluri

## Figuri, tabele și ecuații

# Conclusion