DCS PROJECT

By: Marghitas Răzvan–Nicolae
Pietrar Florentin-Mircea

Group: 30342

Date: 12.01.2022

Content

Introduction

Component Diagram

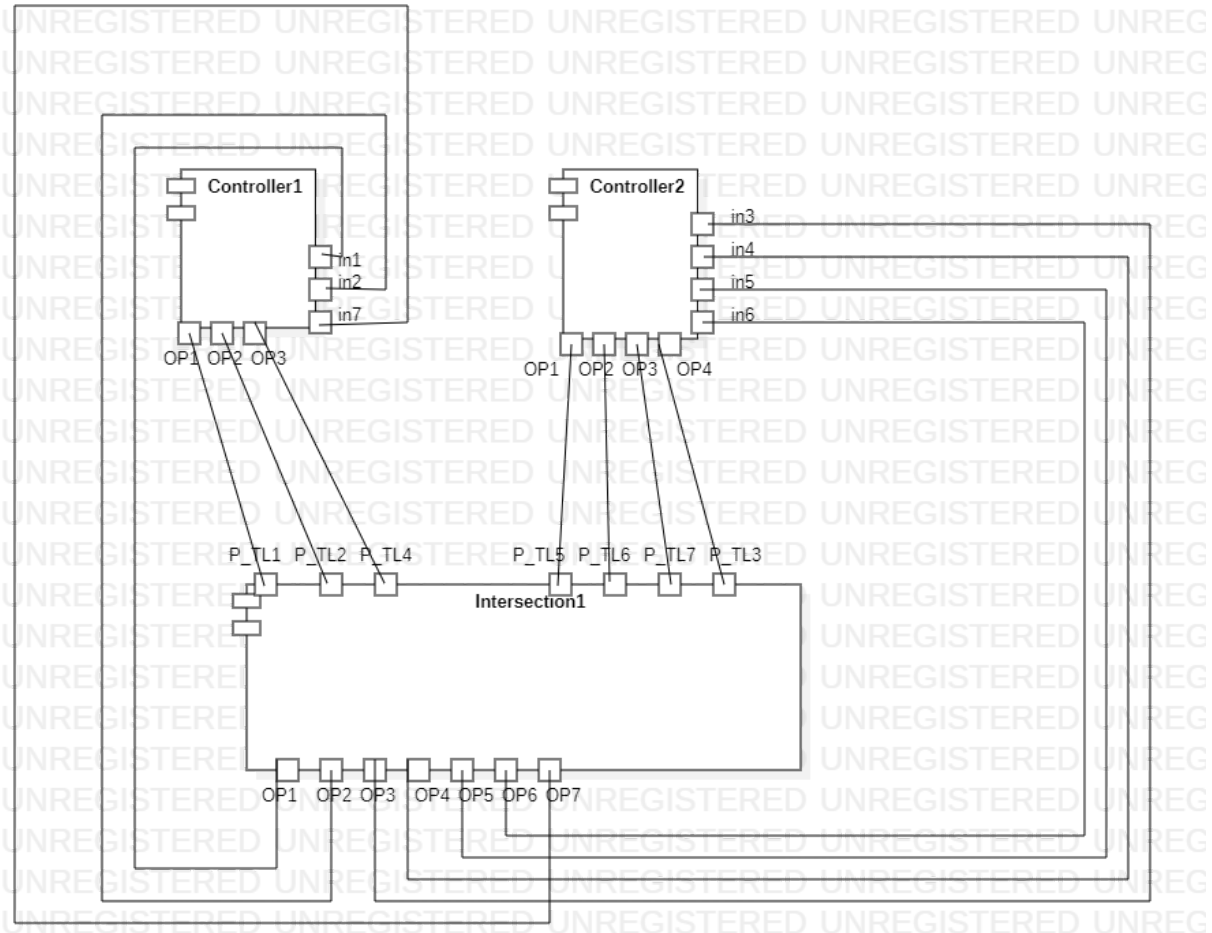


Photo_1. Component diagram

Problem description

We have two intersections with independent controllers and two lanes going in between those intersections so we have to control the semaphores for those two intersections separately and synchronously. The first intersection has 4 exit lanes and 3 entering lanes and the second one has 3 exit lanes and 4 entering lanes from which two of each are connected in between.

OETPN lane model

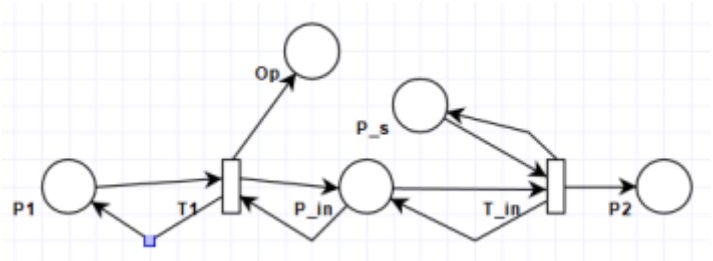


Photo_2. Lane model

t1: grd1: P1 not null && P_in CanAddCars;
map1: P1 ⇒ P_in
grd2: P1 not null && P_in CanNotAddCars;
map2: P1 = P1 && Op = full

t2: grd1: P_s equal green && P_in HaveCar
map1: P_in ⇒ P2 && P_s = P_s

Intersection model

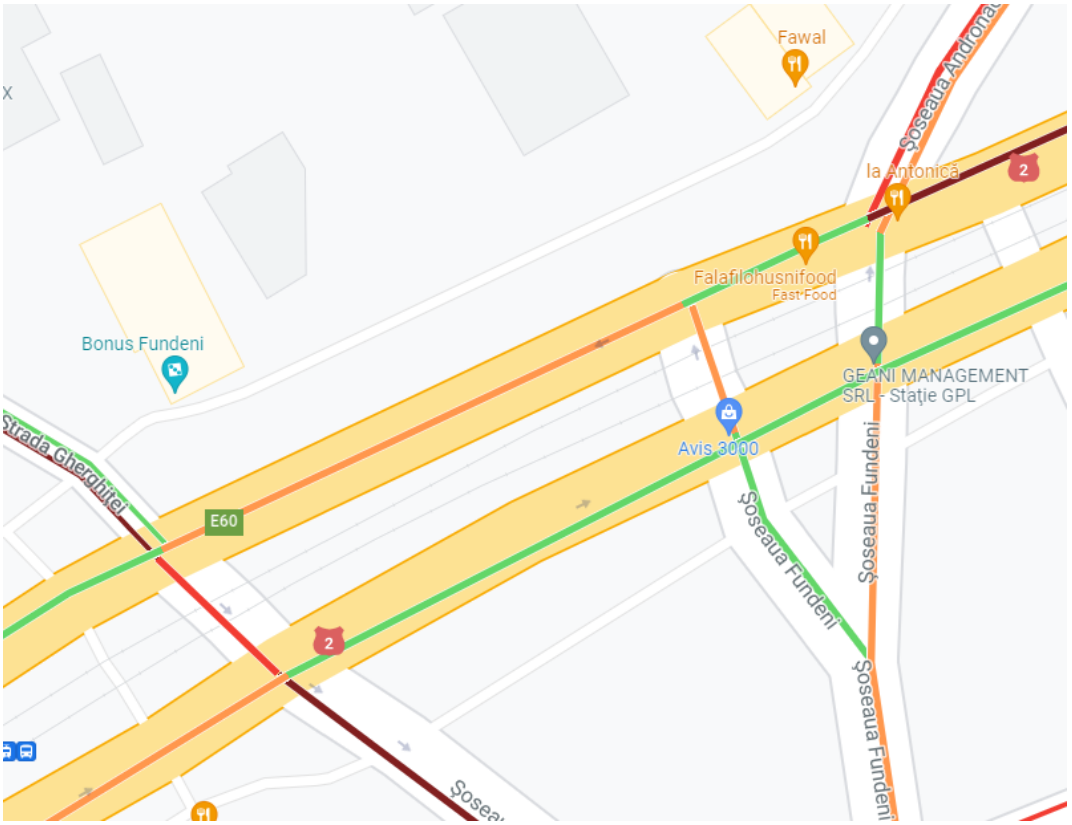


Photo_3. Left = First Intersection, Right = Second Intersection

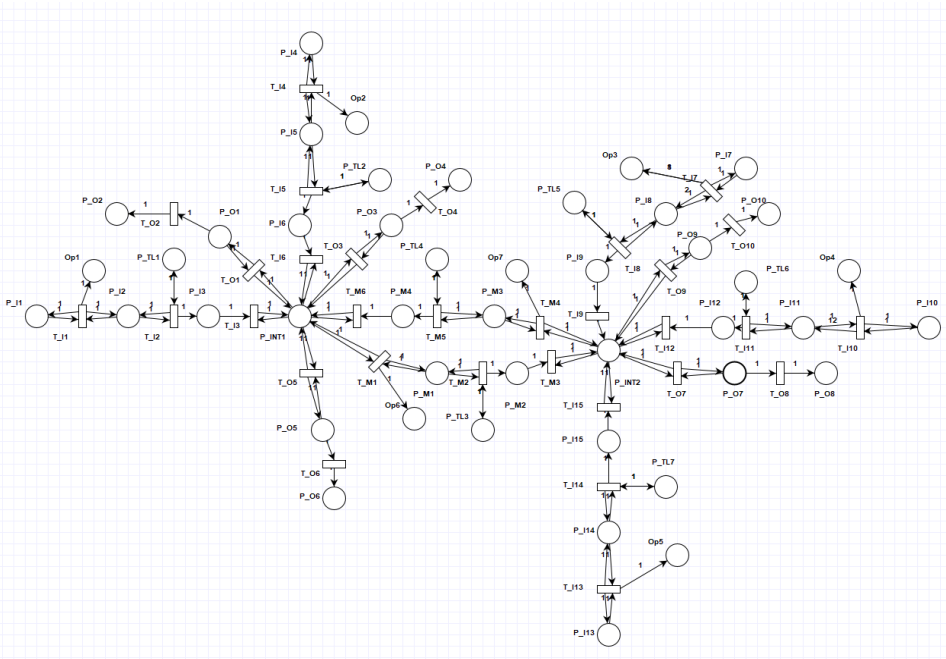


Photo_4. Left = First Intersection, Right = Second Intersection

First Intersection:

T_I1: grd1: P_I1 NotNull && P_I2 CanAddCars
map1: P_I1 => P_I2
grd2: P_I1 NotNull && P_I2 CanNotAddCars
map2: P_I1 = P_I1 && Op1 = full (SendOverNetwork)

T_I2: grd1: P_TL1 equal green && P_I2 HaveCar
map1: P_I2 => P_I3 && P_TL1 = P_TL1

T_I3: grd1: P_I3 NotNull && P_INT1 CanAddCars
map1: P_I3 => P_INT1

T_I4: grd1: P_I4 NotNull && P_I5 CanAddCars
map1: P_I4 => P_I5
grd2: P_I4 NotNull && P_I5 CanNotAddCars
map2: P_I4 = P_I4 && Op2 = full (SendOverNetwork)

T_I5: grd1: P_TL2 equal green && P_I5 HaveCar
map1: P_I5 => P_I6 && P_TL2 = P_TL2

T_I6: grd1: P_I6 NotNull && P_INT1 CanAddCars
map1: P_I6 => P_INT1

T_O1: grd1: P_INT1 HaveCarForMe && P_O1 CanAddCars
map1: P_INT1 PopElementWithTargetToQueue P_O1

T_O2: grd1: P_O1 NotNull && P_O2 CanAddCars
map1: P_O1 => P_O2

T_O3: grd1: P_INT1 HaveCarForMe && P_O3 CanAddCars
map1: P_INT1 PopElementWithTargetToQueue P_O3

T_O4: grd1: P_O3 NotNull && P_O4 CanAddCars
map1: P_O3 => P_O4

T_O5: grd1: P_INT1 HaveCarForMe && P_O5 CanAddCars
map1: P_INT1 PopElementWithTargetToQueue P_O5

T_O6: grd1: P_O5 NotNull && P_O6 CanAddCars
map1: P_O5 => P_O6

Middle lanes:

T_M1: grd1: P_INT1 HaveCarForMe && P_M1 CanAddCars
map1: P_INT1 => P_M1 (PopElementWithTargetToQueue
grd2: P_INT1 HaveCarForMe && P_M1 CanNotAddCars
map2: P_INT1 = P_INT1 && Op6 = full (SendOverNetwork)

T_M2: grd1: P_TL3 equal green && P_M1 HaveCar
map1: P_M1 => P_M2 && P_TL3 = P_TL3

T_M3: grd1: P_M2 NotNull && P_INT2 CanAddCars
map1: P_M2 => P_INT2

T_M4: grd1: P_INT2 HaveCarForMe && P_M3 CanAddCars
map1: P_INT2 => P_M3 (PopElementWithTargetToQueue)
grd2: P_INT2 HaveCarForMe && P_M3 CanNotAddCars
map2: P_INT2 = P_INT2 && Op7 = full (SendOverNetwork)

T_M5: grd1: P_TL4 equal green && P_M3 HaveCar
map1: P_M3 => P_M4 && P_TL4 = P_TL4

T_M6: grd1: P_M4 NotNull && P_INT1 CanAddCars
map1: P_M4 => P_INT1

Second Intersection:

T_I7: grd1: P_I7 NotNull && P_I8 CanAddCars
map1: P_I7 => P_I8
grd2: P_I7 NotNull && P_I8 CanNotAddCars

map2: P_I7 = P_I7 && Op3 = full (SendOverNetwork)

T_I8: grd1: P_TL5 equal green && P_I8 HaveCar
map1: P_I8 => P_I9 && P_TL5 = P_TL5

T_I9: grd1: P_I9 NotNull && P_INT2 CanAddCars
map1: P_I9 => P_INT2

T_I10: grd1: P_I10 NotNull && P_I11 CanAddCars
map1: P_I10 => P_I11
grd2: P_I10 NotNull && P_I11 CanNotAddCars
map2: P_I10 = P_I10 && Op4 = full (SendOverNetwork)

T_I11: grd1: P_TL6 equal green && P_I11 HaveCar
map1: P_I11 => P_I12 && P_TL6 = P_TL6

T_I12: grd1: P_I12 NotNull && P_INT2 CanAddCars
map1: P_I12 => P_INT2

T_I13: grd1: P_I13 NotNull && P_I14 CanAddCars
map1: P_I13 => P_I14
grd2: P_I13 NotNull && P_I14 CanNotAddCars
map2: P_I13 = P_I13 && Op5 = full (SendOverNetwork)

T_I14: grd1: P_TL7 equal green && P_I14 HaveCar
map1: P_I14 => P_I15 && P_TL7 = P_TL7

T_I15: grd1: P_I15 NotNull && P_INT2 CanAddCars
map1: P_I15 => P_INT2

T_O7: grd1: P_INT2 HaveCarForMe && P_O7 CanAddCars
map1: P_INT2 PopElementWithTargetToQueue P_O7

T_O8: grd1: P_O7 NotNull && P_O8 CanAddCars
map1: P_O7 => P_O8

T_O9: grd1: P_INT2 HaveCarForMe && P_O9 CanAddCars

map1: P_INT2 PopElementWithTargetToQueue P_O9

T_O10: grd1: P_O9 NotNull && P_O10 CanAddCars

map1: P_O9 => P_O10

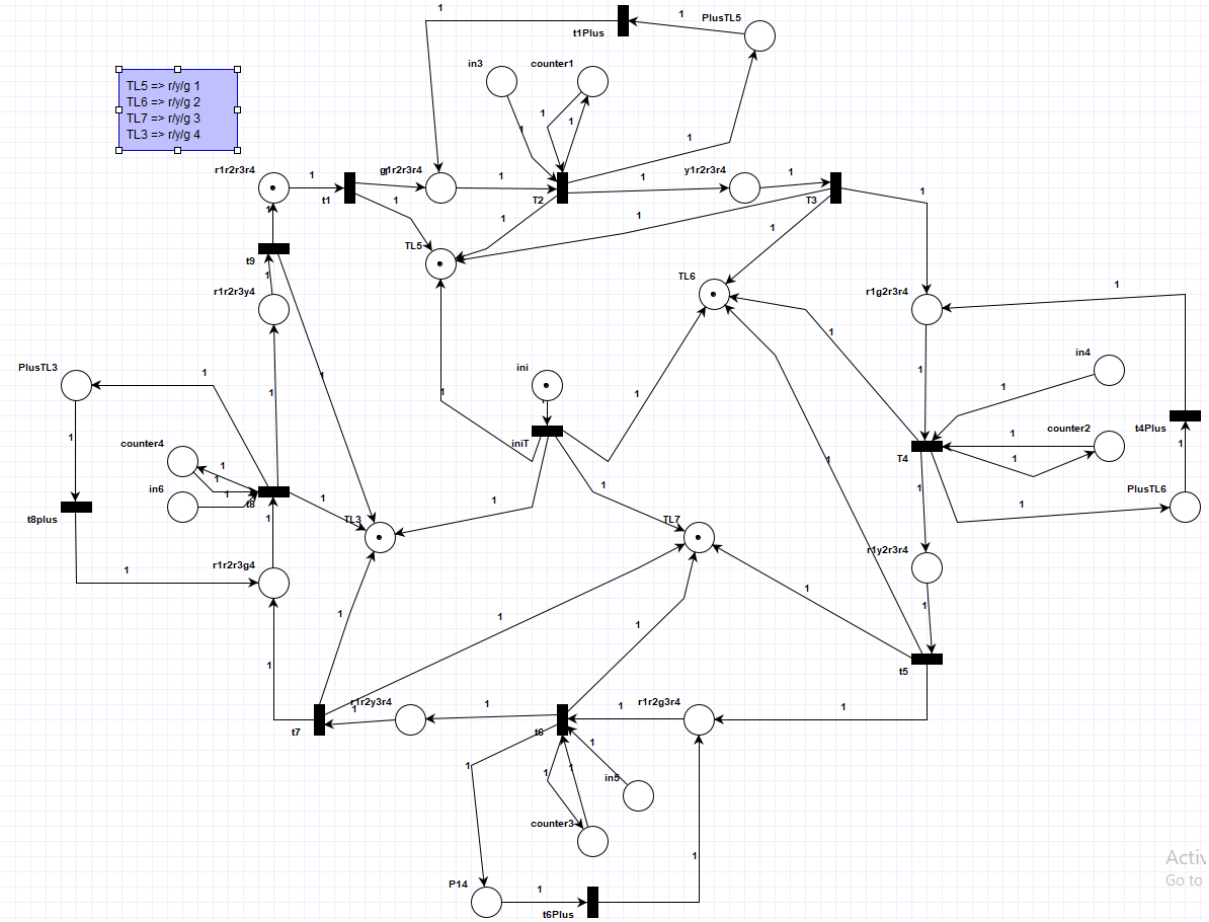


Photo_5. Controller 2 diagram

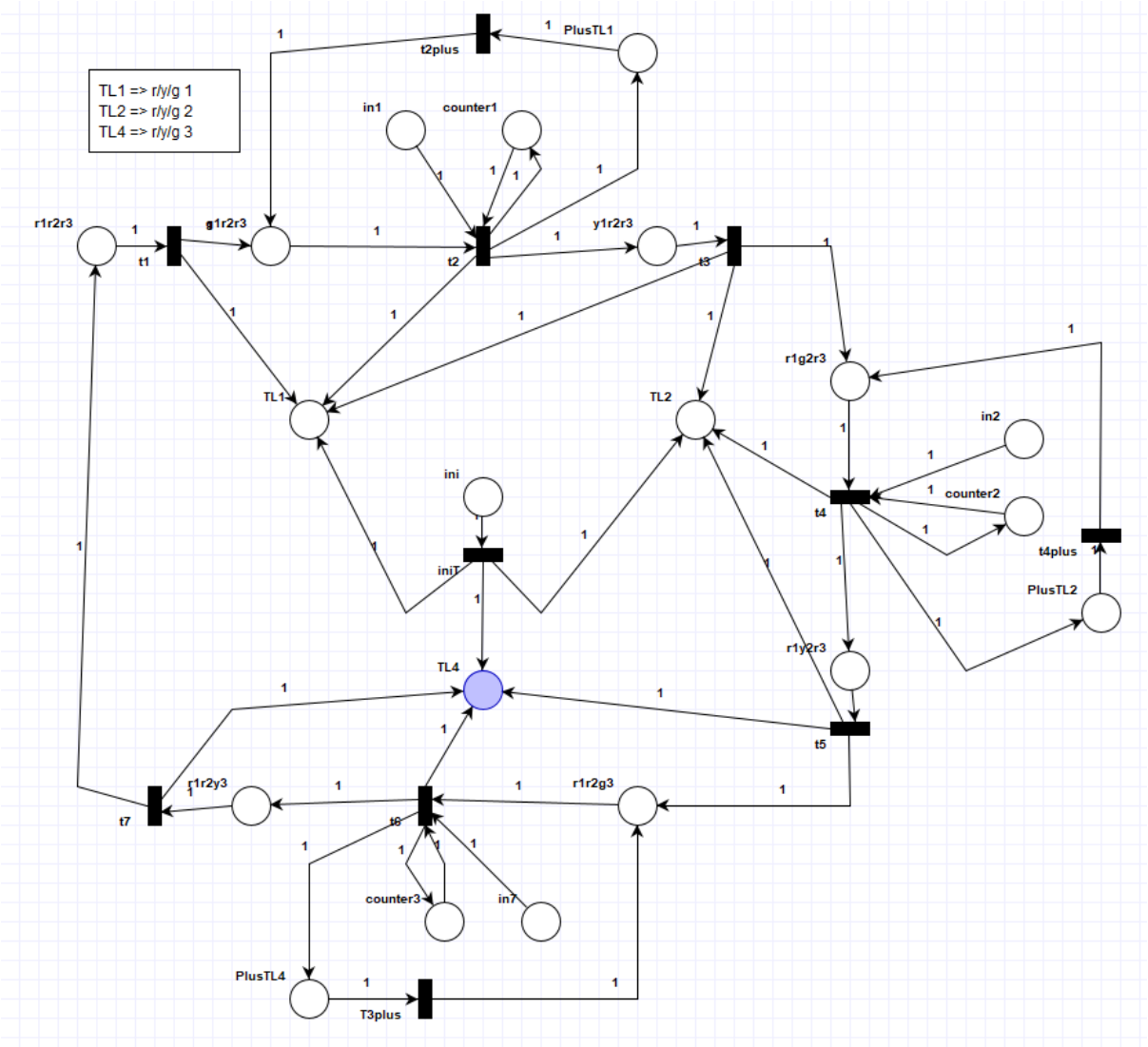
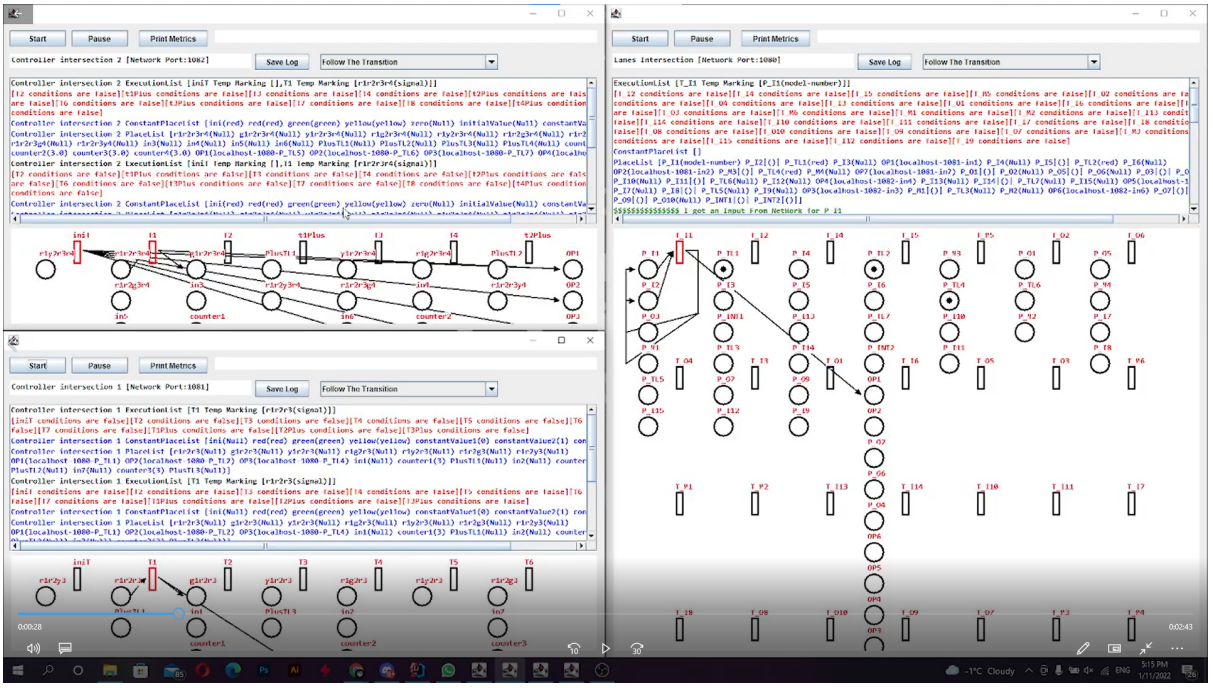
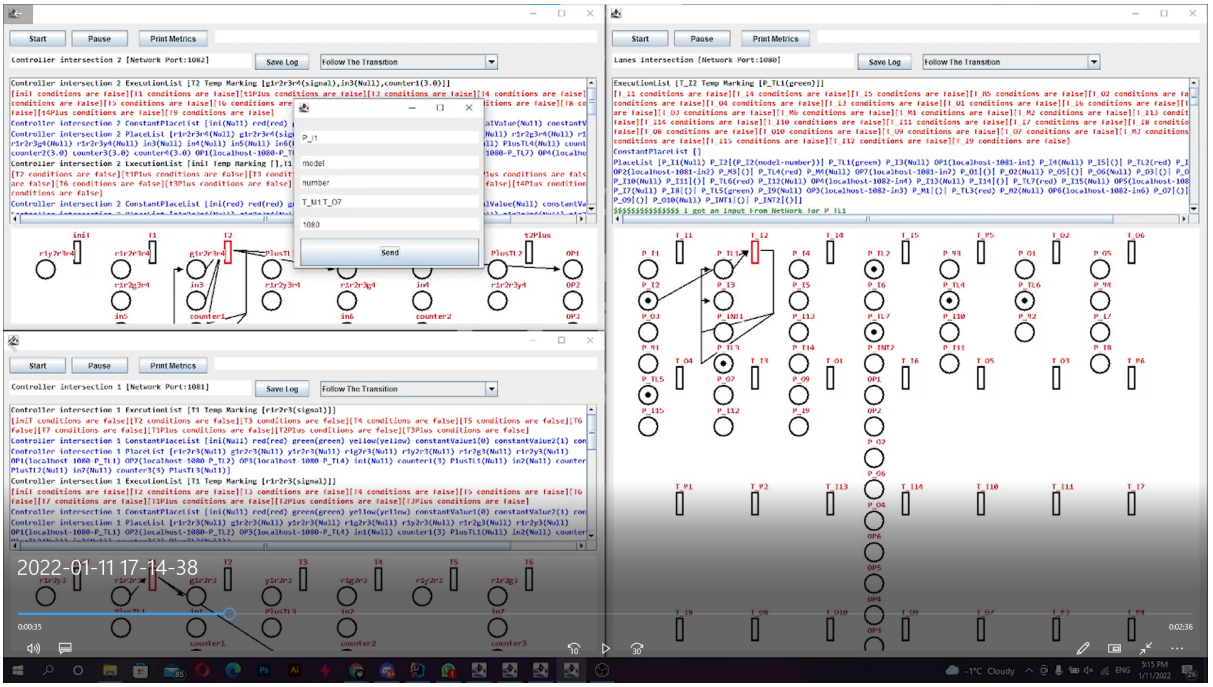
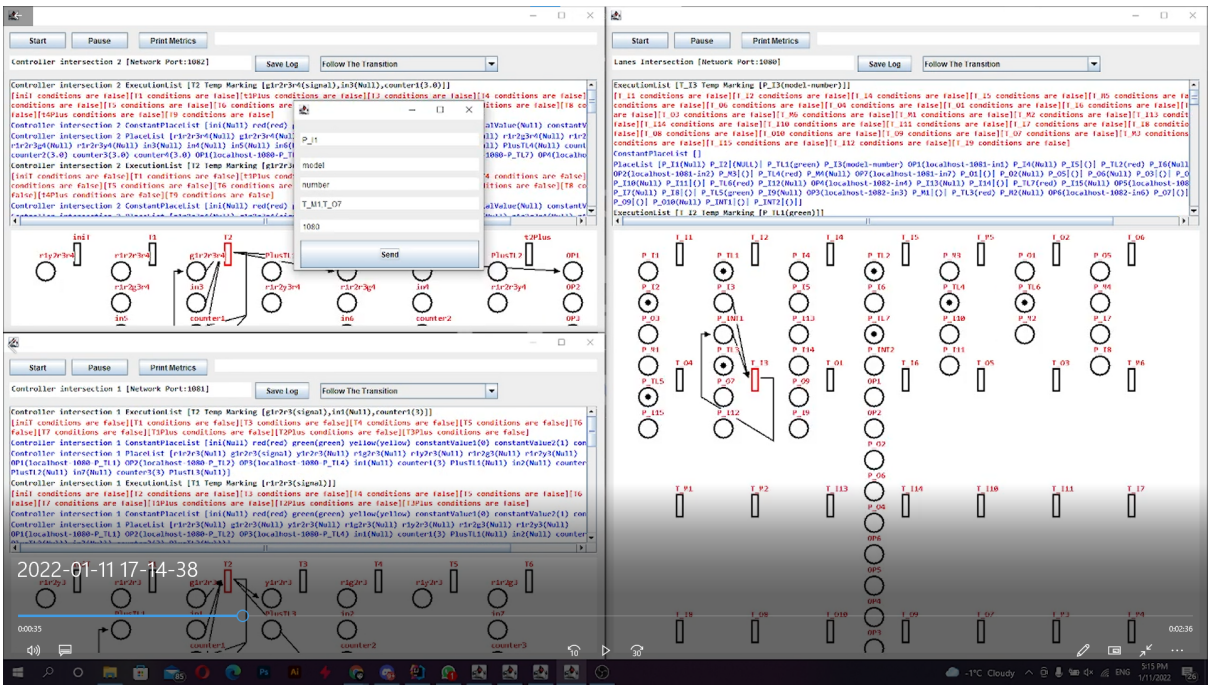


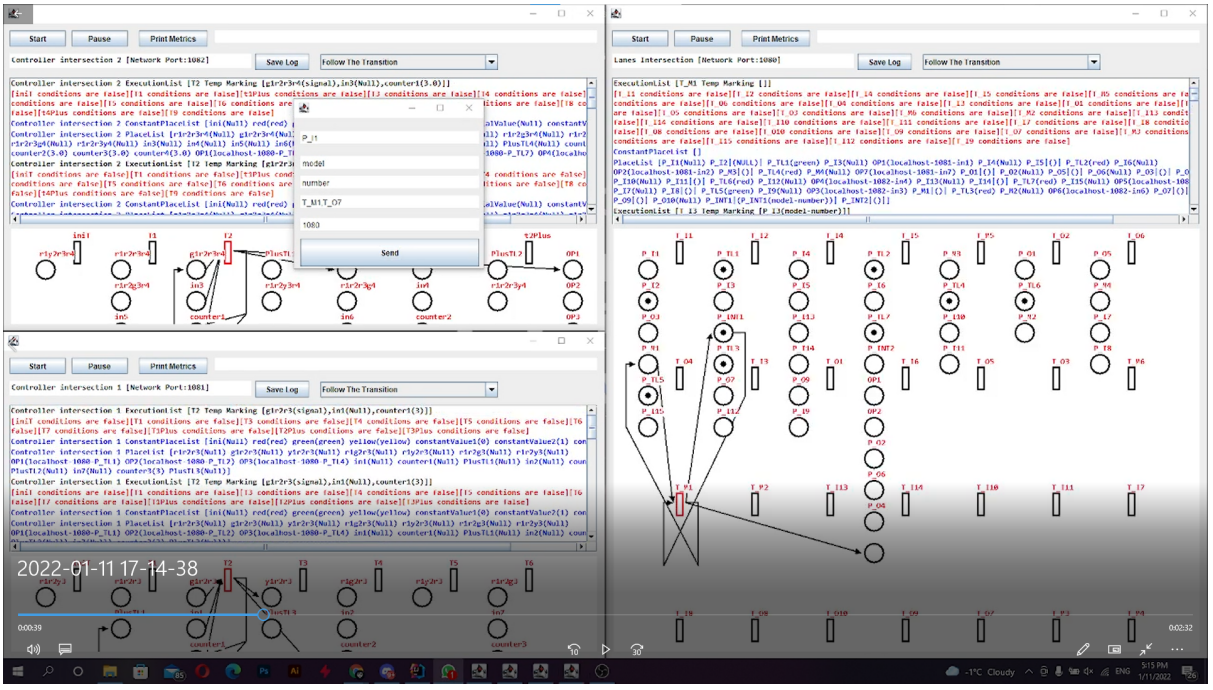
Photo_6. Controller 1 diagram

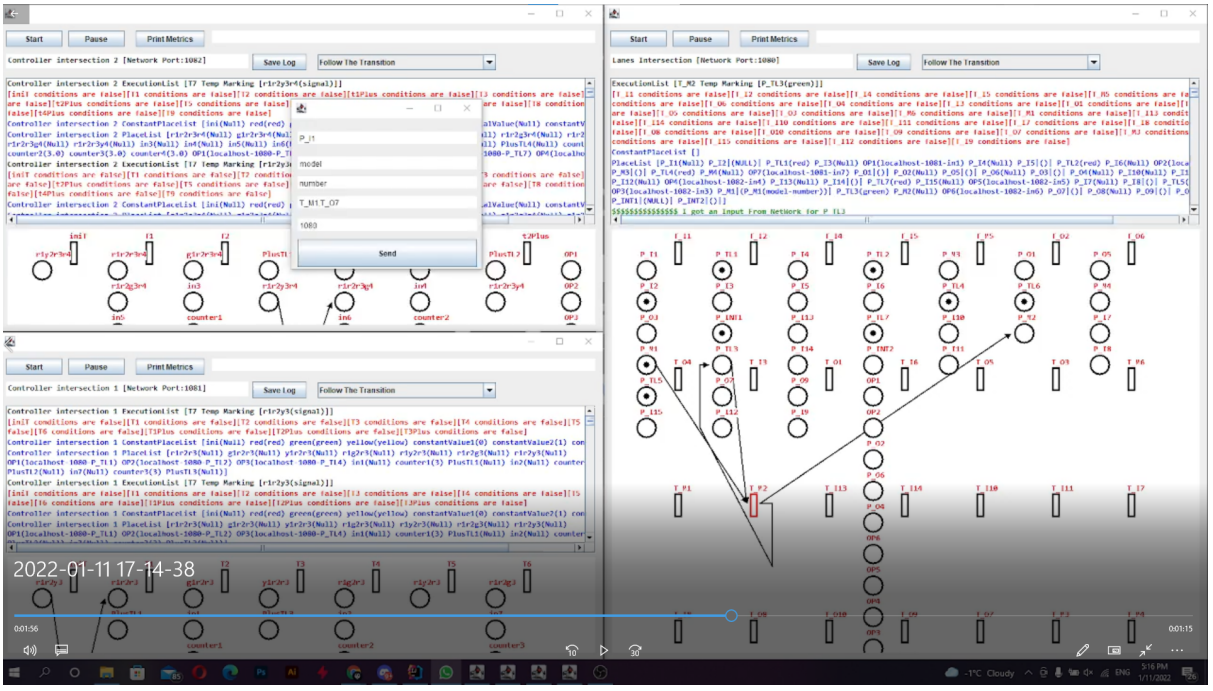
Working examples

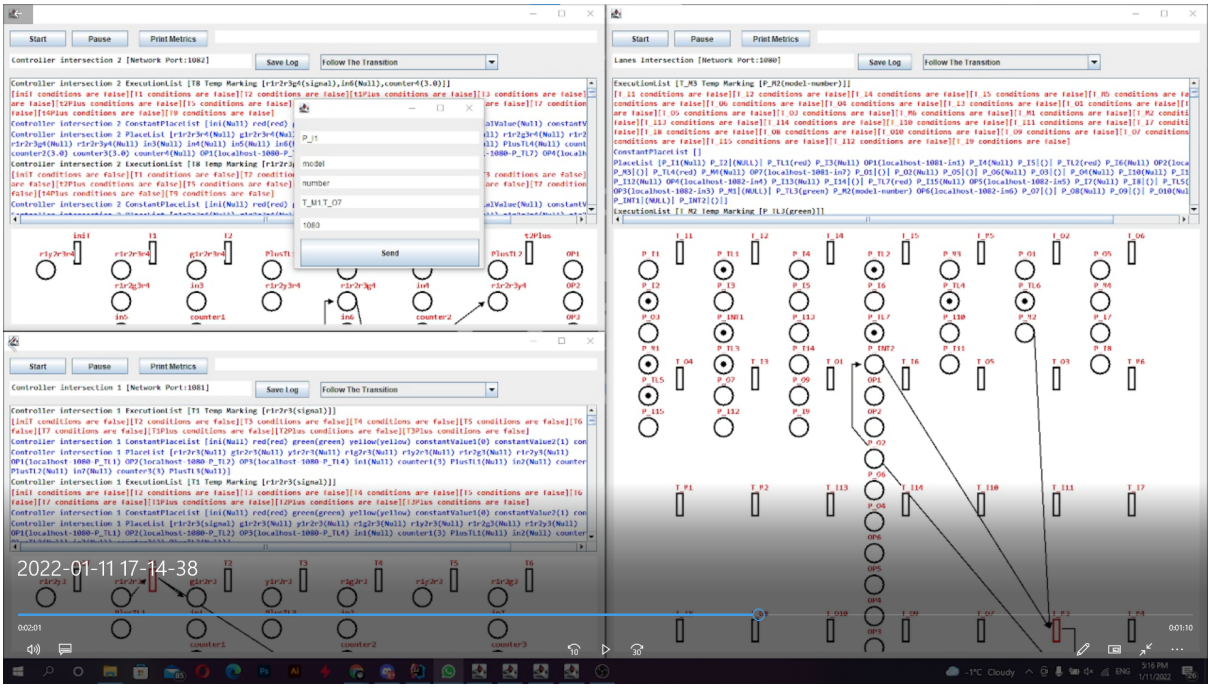


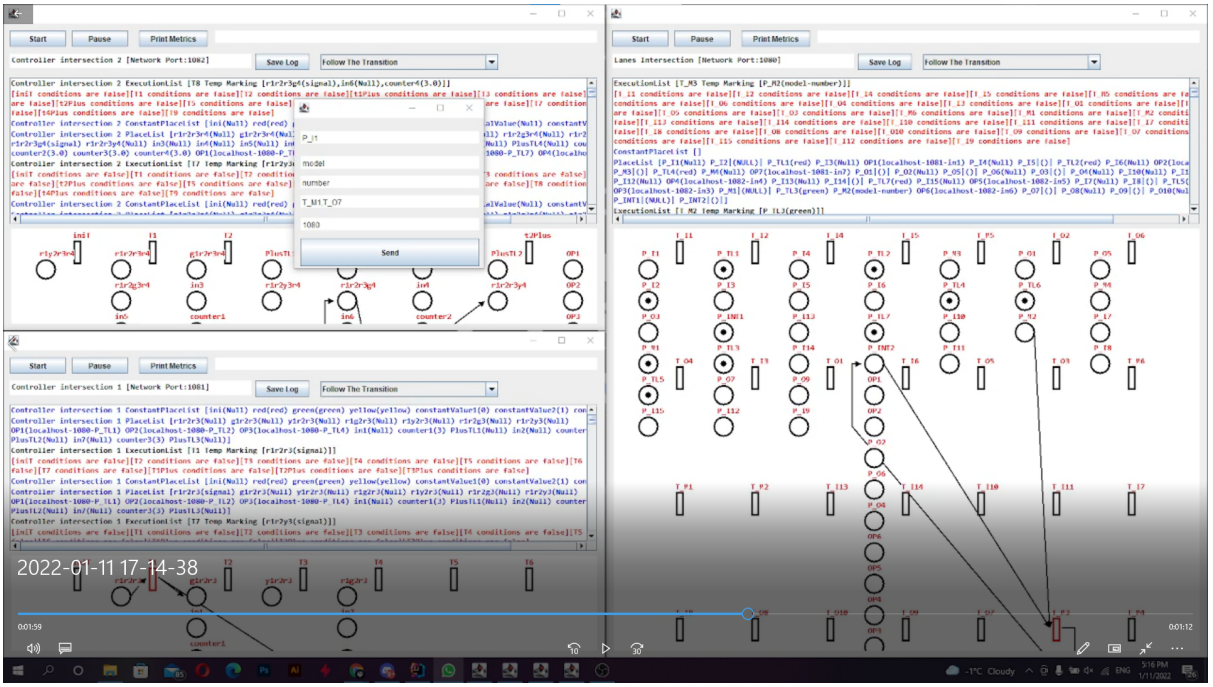


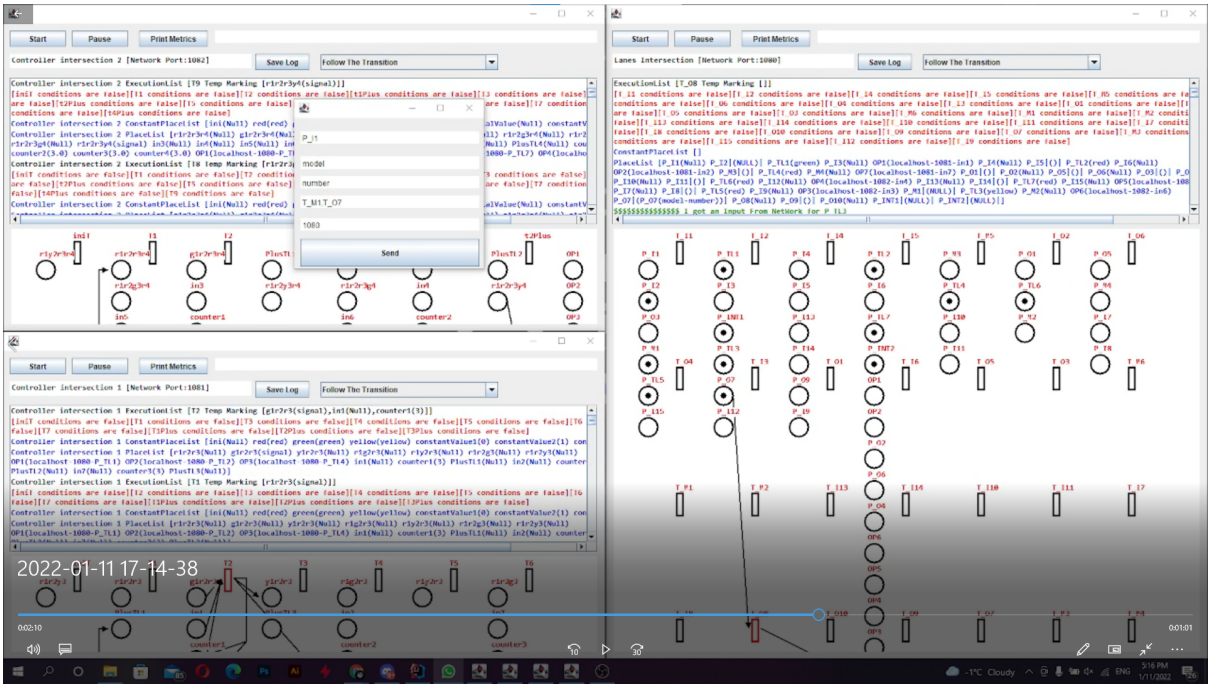

Start
Pause
Print Metrics
Save Log
Follow The Transition
Send
2022-01-11 17-14-38


Start
Pause
Print Metrics
Save Log
Follow The Transition
Send
2022-01-11 17-14-38


Start
Pause
Print Metrics
Save Log
Follow The Transition
Send
2022-01-11 17-14-38


Start
Pause
Print Metrics
Save Log
Follow The Transition
Send
2022-01-11 17-14-38


Start
Pause
Print Metrics
Save Log
Follow The Transition
Send
2022-01-11 17-14-38


Start
Pause
Print Metrics
Save Log
Follow The Transition
Send
2022-01-11 17-14-38

Source code

Lane Intersections:

```
package NullPointerException;

import Components.*;
import DataObjects.DataCar;
import DataObjects.DataCarQueue;
import DataObjects.DataString;
import DataObjects.DataTransfer;
import DataOnly.TransferOperation;
import Enumerations.LogicConnector;
import Enumerations.TransitionCondition;
import Enumerations.TransitionOperation;

public class Lanes_intersection {
   public static void main(String[] args) {
        PetriNet pn = new PetriNet();
        pn.PetriNetName = "Lanes Intersection";

        pn.NetworkPort = 1080;

        DataString green = new DataString();
        green.Printable = false;
        green.SetName("green");
        green.SetValue("green");
        pn.ConstantPlaceList.add(green);

        DataString full = new DataString();
        full.Printable = false;
        full.SetName("full");
        full.SetValue("full");
        pn.ConstantPlaceList.add(full);

        // ------------------------------------------------------------------
        // -------------------------------Int1-------------------------------
        //
------------------------------------------------------------------


        // ------------------------------------------------------------------
        //
-------------------------------Lane1-------------------------------
        //
------------------------------------------------------------------

        DataCar P_I1 = new DataCar();
```

```
        P_I1.SetName("P_I1");
        pn.PlaceList.add(P_I1);

        DataCarQueue P_I2 = new DataCarQueue();
        P_I2.Value.Size = 3;
        P_I2.SetName("P_I2");
        pn.PlaceList.add(P_I2);

        DataString P_TL1 = new DataString();
        P_TL1.SetName("P_TL1");
        pn.PlaceList.add(P_TL1);

        DataCar P_I3 = new DataCar();
        P_I3.SetName("P_I3");
        pn.PlaceList.add(P_I3);

        DataTransfer OP1 = new DataTransfer();
        OP1.Value=new TransferOperation("localhost","1081","in1");
        OP1.SetName("OP1");
        pn.PlaceList.add(OP1);

        // ------------------------------------------------------------------
        //
-------------------------------Lane2---------------------------------
        //
-----------------------------------------------------------------------

        DataCar P_I4 = new DataCar();
        P_I4.SetName("P_I4");
        pn.PlaceList.add(P_I4);

        DataCarQueue P_I5 = new DataCarQueue();
        P_I5.Value.Size = 3;
        P_I5.SetName("P_I5");
        pn.PlaceList.add(P_I5);

        DataString P_TL2 = new DataString();
        P_TL2.SetName("P_TL2");
        pn.PlaceList.add(P_TL2);

        DataCar P_I6 = new DataCar();
        P_I6.SetName("P_I6");
        pn.PlaceList.add(P_I6);

        DataTransfer OP2 = new DataTransfer();
        OP2.SetName("OP2");
        OP2.Value=new TransferOperation("localhost","1081","in2");
        pn.PlaceList.add(OP2);

        // ------------------------------------------------------------------
        //
-------------------------------Lane3---------------------------------
```

```
        //
-------------------------------------------------------------------

        DataCarQueue P_M3 = new DataCarQueue();
        P_M3.Value.Size = 3;
        P_M3.SetName("P_M3");
        pn.PlaceList.add(P_M3);

        DataString P_TL4 = new DataString();
        P_TL4.SetName("P_TL4");
        pn.PlaceList.add(P_TL4);

        DataCar P_M4 = new DataCar();
        P_M4.SetName("P_M4");
        pn.PlaceList.add(P_M4);

        DataTransfer OP7 = new DataTransfer();
        OP7.SetName("OP7");
        OP7.Value=new TransferOperation("localhost","1081","in7");
        pn.PlaceList.add(OP7);

        //
-------------------------------------------------------------------------------
        // ----------------------------Exit lane
1-----------------------------------
        //
-------------------------------------------------------------------------------

        DataCarQueue P_O1 = new DataCarQueue();
        P_O1.Value.Size = 3;
        P_O1.SetName("P_O1");
        pn.PlaceList.add(P_O1);

        DataCar P_O2 = new DataCar();
        P_O2.SetName("P_O2");
        pn.PlaceList.add(P_O2);

        //
-------------------------------------------------------------------------------
        // ----------------------------Exit lane
2-----------------------------------
        //
-------------------------------------------------------------------------------

        DataCarQueue P_O5 = new DataCarQueue();
        P_O5.Value.Size = 3;
        P_O5.SetName("P_O5");
        pn.PlaceList.add(P_O5);

        DataCar P_O6 = new DataCar();
        P_O6.SetName("P_O6");
        pn.PlaceList.add(P_O6);
```

```
        //
-------------------------------------------------------------------------------
        // ----------------------------Exit lane
3-----------------------------------
        //
-------------------------------------------------------------------------------

        DataCarQueue P_O3 = new DataCarQueue();
        P_O3.Value.Size = 3;
        P_O3.SetName("P_O3");
        pn.PlaceList.add(P_O3);

        DataCar P_O4 = new DataCar();
        P_O4.SetName("P_O4");
        pn.PlaceList.add(P_O4);

        // ---------------------------------------------------------------------
        // --------------------------------Int2---------------------------------
        //
-----------------------------------------------------------------------

        // ---------------------------------------------------------------------
        //
------------------------------Lane1----------------------------------
        //
-----------------------------------------------------------------------

        DataCar P_I10 = new DataCar();
        P_I10.SetName("P_I10");
        pn.PlaceList.add(P_I10);

        DataCarQueue P_I11 = new DataCarQueue();
        P_I11.Value.Size = 3;
        P_I11.SetName("P_I11");
        pn.PlaceList.add(P_I11);

        DataString P_TL6 = new DataString();
        P_TL6.SetName("P_TL6");
        pn.PlaceList.add(P_TL6);

        DataCar P_I12 = new DataCar();
        P_I12.SetName("P_I12");
        pn.PlaceList.add(P_I12);

        DataTransfer OP4 = new DataTransfer();
        OP4.Value=new TransferOperation("localhost","1082","in4");
        OP4.SetName("OP4");
        pn.PlaceList.add(OP4);

        // ---------------------------------------------------------------------
        //
------------------------------Lane2----------------------------------
```

```
        //
-------------------------------------------------------------------------

        DataCar P_I13 = new DataCar();
        P_I13.SetName("P_I13");
        pn.PlaceList.add(P_I13);

        DataCarQueue P_I14 = new DataCarQueue();
        P_I14.Value.Size = 3;
        P_I14.SetName("P_I14");
        pn.PlaceList.add(P_I14);

        DataString P_TL7 = new DataString();
        P_TL7.SetName("P_TL7");
        pn.PlaceList.add(P_TL7);

        DataCar P_I15 = new DataCar();
        P_I15.SetName("P_I15");
        pn.PlaceList.add(P_I15);

        DataTransfer OP5 = new DataTransfer();
        OP5.Value=new TransferOperation("localhost","1082","in5");
        OP5.SetName("OP5");
        pn.PlaceList.add(OP5);

        // ---------------------------------------------------------------------
        //
-------------------------------Lane3-----------------------------------
        //
-------------------------------------------------------------------------

        DataCar P_I7 = new DataCar();
        P_I7.SetName("P_I7");
        pn.PlaceList.add(P_I7);

        DataCarQueue P_I8 = new DataCarQueue();
        P_I8.Value.Size = 3;
        P_I8.SetName("P_I8");
        pn.PlaceList.add(P_I8);

        DataString P_TL5 = new DataString();
        P_TL5.SetName("P_TL5");
        pn.PlaceList.add(P_TL5);

        DataCar P_I9 = new DataCar();
        P_I9.SetName("P_I9");
        pn.PlaceList.add(P_I9);

        DataTransfer OP3 = new DataTransfer();
        OP3.Value=new TransferOperation("localhost","1082","in3");
        OP3.SetName("OP3");
        pn.PlaceList.add(OP3);
```

```
        // ------------------------------------------------------------------
        //
-------------------------------Lane4---------------------------------
        //
-----------------------------------------------------------------------

        DataCarQueue P_M1 = new DataCarQueue();
        P_M1.SetName("P_M1");
        pn.PlaceList.add(P_M1);

        DataString P_TL3 = new DataString();
        P_TL3.SetName("P_TL3");
        pn.PlaceList.add(P_TL3);

        DataCar P_M2 = new DataCar();
        P_M2.SetName("P_M2");
        pn.PlaceList.add(P_M2);

        DataTransfer OP6 = new DataTransfer();
        OP6.Value=new TransferOperation("localhost","1082","in6");
        OP6.SetName("OP6");
        pn.PlaceList.add(OP6);


        //
-------------------------------------------------------------------------------
        // ----------------------------Exit lane
1-----------------------------------
        //
-------------------------------------------------------------------------------

        DataCarQueue P_O7 = new DataCarQueue(); //p17.Printable = false;
        P_O7.Value.Size = 3;
        P_O7.SetName("P_O7");
        pn.PlaceList.add(P_O7);

        DataCar P_O8 = new DataCar(); //p18.Printable = false;
        P_O8.SetName("P_O8");
        pn.PlaceList.add(P_O8);

        //
-------------------------------------------------------------------------------
        // ----------------------------Exit lane
2-----------------------------------
        //
-------------------------------------------------------------------------------

        DataCarQueue P_O9 = new DataCarQueue(); //p17.Printable = false;
        P_O9.Value.Size = 3;
        P_O9.SetName("P_O9");
        pn.PlaceList.add(P_O9);

        DataCar P_O10 = new DataCar(); //p18.Printable = false;
```

```
        P_O10.SetName("P_O10");
        pn.PlaceList.add(P_O10);


        //
-------------------------------------------------------------------------------
--------------
        //
--------------------------------Intersection1---------------------------------
---------------
        //
-------------------------------------------------------------------------------
--------------

        DataCarQueue P_INT1 = new DataCarQueue();
        P_INT1.Value.Size = 3;
        P_INT1.SetName("P_INT1");
        pn.PlaceList.add(P_INT1);

        // T_I1 ------------------------------------------------
        PetriTransition T_I1 = new PetriTransition(pn);
        T_I1.TransitionName = "T_I1";
        T_I1.InputPlaceName.add("P_I1");
        T_I1.InputPlaceName.add("P_I2");

        Condition T1Ct1 = new Condition(T_I1, "P_I1",
TransitionCondition.NotNull);
        Condition T1Ct2 = new Condition(T_I1, "P_I2",
TransitionCondition.CanAddCars);
        T1Ct1.SetNextCondition(LogicConnector.AND, T1Ct2);

        GuardMapping grdT1 = new GuardMapping();
        grdT1.condition = T1Ct1;
        grdT1.Activations.add(new Activation(T_I1, "P_I1",
TransitionOperation.AddElement, "P_I2"));
        T_I1.GuardMappingList.add(grdT1);

        Condition T1Ct3 = new Condition(T_I1, "P_I1",
TransitionCondition.NotNull);
        Condition T1Ct4 = new Condition(T_I1, "P_I2",
TransitionCondition.CanNotAddCars);
        T1Ct3.SetNextCondition(LogicConnector.AND, T1Ct4);

        GuardMapping grdT1_1 = new GuardMapping();
        grdT1_1.condition = T1Ct3;
        grdT1_1.Activations.add(new Activation(T_I1, "full",
TransitionOperation.SendOverNetwork, "OP1"));
        grdT1_1.Activations.add(new Activation(T_I1, "P_I1",
TransitionOperation.Copy, "P_I1"));
        T_I1.GuardMappingList.add(grdT1_1);

        T_I1.Delay = 0;
        pn.Transitions.add(T_I1);
```

```
        // T_I2 ------------------------------------------------
        PetriTransition T_I2 = new PetriTransition(pn);
        T_I2.TransitionName = "T_I2";
        T_I2.InputPlaceName.add("P_I2");
        T_I2.InputPlaceName.add("P_TL1");

        Condition T2Ct1 = new Condition(T_I2, "P_TL1",
TransitionCondition.Equal, "green");
        Condition T2Ct2 = new Condition(T_I2, "P_I2",
TransitionCondition.HaveCar);
        T2Ct1.SetNextCondition(LogicConnector.AND, T2Ct2);

        GuardMapping grdT2 = new GuardMapping();
        grdT2.condition = T2Ct1;
        grdT2.Activations.add(new Activation(T_I2, "P_I2",
TransitionOperation.PopElementWithoutTarget, "P_I3"));
        grdT2.Activations.add(new Activation(T_I2, "P_TL1",
TransitionOperation.Move, "P_TL1"));

        T_I2.GuardMappingList.add(grdT2);

        T_I2.Delay = 0;
        pn.Transitions.add(T_I2);

        // T_I4 ------------------------------------------------
        PetriTransition T_I4 = new PetriTransition(pn);
        T_I4.TransitionName = "T_I4";
        T_I4.InputPlaceName.add("P_I4");
        T_I4.InputPlaceName.add("P_I5");

        Condition T3Ct1 = new Condition(T_I4, "P_I4",
TransitionCondition.NotNull);
        Condition T3Ct2 = new Condition(T_I4, "P_I5",
TransitionCondition.CanAddCars);
        T3Ct1.SetNextCondition(LogicConnector.AND, T3Ct2);

        GuardMapping grdT3 = new GuardMapping();
        grdT3.condition = T3Ct1;
        grdT3.Activations.add(new Activation(T_I4, "P_I4",
TransitionOperation.AddElement, "P_I5"));
        T_I4.GuardMappingList.add(grdT3);

        Condition T3Ct3 = new Condition(T_I4, "P_I4",
TransitionCondition.NotNull);
        Condition T3Ct4 = new Condition(T_I4, "P_I5",
TransitionCondition.CanNotAddCars);
        T3Ct3.SetNextCondition(LogicConnector.AND, T3Ct4);

        GuardMapping grdT3_1 = new GuardMapping();
        grdT3_1.condition = T3Ct3;
        grdT3_1.Activations.add(new Activation(T_I4, "full",
TransitionOperation.SendOverNetwork, "OP2"));
```

```
        grdT3_1.Activations.add(new Activation(T_I4, "P_I4",
TransitionOperation.Copy, "P_I4"));
        T_I4.GuardMappingList.add(grdT3_1);

        T_I4.Delay = 0;
        pn.Transitions.add(T_I4);

        // T_I5 ------------------------------------------------
        PetriTransition T_I5 = new PetriTransition(pn);
        T_I5.TransitionName = "T_I5";
        T_I5.InputPlaceName.add("P_I5");
        T_I5.InputPlaceName.add("P_TL2");

        Condition T4Ct1 = new Condition(T_I5, "P_TL2",
TransitionCondition.Equal, "green");
        Condition T4Ct2 = new Condition(T_I5, "P_I5",
TransitionCondition.HaveCar);
        T4Ct1.SetNextCondition(LogicConnector.AND, T4Ct2);

        GuardMapping grdT4 = new GuardMapping();
        grdT4.condition = T4Ct1;
        grdT4.Activations.add(new Activation(T_I5, "P_I5",
TransitionOperation.PopElementWithoutTarget, "P_I6"));
        grdT4.Activations.add(new Activation(T_I5, "P_TL2",
TransitionOperation.Move, "P_TL2"));
        T_I5.GuardMappingList.add(grdT2);

        T_I5.Delay = 0;
        pn.Transitions.add(T_I5);

        // T_M5 ------------------------------------------------
        PetriTransition T_M5 = new PetriTransition(pn);
        T_M5.TransitionName = "T_M5";
        T_M5.InputPlaceName.add("P_M3");
        T_M5.InputPlaceName.add("P_TL4");

        Condition T6Ct1 = new Condition(T_M5, "P_TL4",
TransitionCondition.Equal, "green");
        Condition T6Ct2 = new Condition(T_M5, "P_M3",
TransitionCondition.HaveCar);
        T6Ct1.SetNextCondition(LogicConnector.AND, T6Ct2);

        GuardMapping grdT6 = new GuardMapping();
        grdT6.condition = T6Ct1;
        grdT6.Activations.add(new Activation(T_M5, "P_M3",
TransitionOperation.PopElementWithoutTarget, "P_M4"));
        grdT6.Activations.add(new Activation(T_M5, "P_TL4",
TransitionOperation.Move, "P_TL4"));
        T_M5.GuardMappingList.add(grdT6);

        T_M5.Delay = 0;
        pn.Transitions.add(T_M5);
```

```
        //
T_O2-----------------------------------------------------------------

        PetriTransition T_O2 = new PetriTransition(pn);
        T_O2.TransitionName = "T_O2";
        T_O2.InputPlaceName.add("P_O1");

        Condition T9Ct1 = new Condition(T_O2, "P_O1",
TransitionCondition.HaveCar);

        GuardMapping grdT9 = new GuardMapping();
        grdT9.condition = T9Ct1;
        grdT9.Activations.add(new Activation(T_O2, "P_O1",
TransitionOperation.PopElementWithoutTarget, "P_O2"));
        T_O2.GuardMappingList.add(grdT9);

        T_O2.Delay = 0;
        pn.Transitions.add(T_O2);

        //
T_O6-----------------------------------------------------------------

        PetriTransition T_O6 = new PetriTransition(pn);
        T_O6.TransitionName = "T_O6";
        T_O6.InputPlaceName.add("P_O5");

        Condition T10Ct1 = new Condition(T_O6, "P_O5",
TransitionCondition.HaveCar);

        GuardMapping grdT10 = new GuardMapping();
        grdT10.condition = T10Ct1;
        grdT10.Activations.add(new Activation(T_O6, "P_O5",
TransitionOperation.PopElementWithoutTarget, "P_O6"));
        T_O6.GuardMappingList.add(grdT10);

        T_O6.Delay = 0;
        pn.Transitions.add(T_O6);

        // T_O4-------------------------------------------------

        PetriTransition T_O4 = new PetriTransition(pn);
        T_O4.TransitionName = "T_O4";
        T_O4.InputPlaceName.add("P_O3");

        Condition T_O4Ct1 = new Condition(T_O4, "P_O3",
TransitionCondition.HaveCar);

        GuardMapping grdT_O4 = new GuardMapping();
        grdT_O4.condition = T_O4Ct1;
        grdT_O4.Activations.add(new Activation(T_O4, "P_O3",
TransitionOperation.PopElementWithoutTarget, "P_O4"));
        T_O4.GuardMappingList.add(grdT_O4);
```

```
        T_O4.Delay = 0;
        pn.Transitions.add(T_O4);

        // T_I3-----------------------------------------------
        PetriTransition T_I3 = new PetriTransition(pn);
        T_I3.TransitionName = "T_I3";
        T_I3.InputPlaceName.add("P_I3");
        T_I3.InputPlaceName.add("P_INT1");

        Condition T13Ct1 = new Condition(T_I3, "P_I3",
TransitionCondition.NotNull);
        Condition T13Ct2 = new Condition(T_I3, "P_INT1",
TransitionCondition.CanAddCars);
        T13Ct1.SetNextCondition(LogicConnector.AND, T13Ct2);

        GuardMapping grdT13 = new GuardMapping();
        grdT13.condition = T13Ct1;
        grdT13.Activations.add(new Activation(T_I3, "P_I3",
TransitionOperation.AddElement, "P_INT1"));
        T_I3.GuardMappingList.add(grdT13);

        T_I3.Delay = 0;
        pn.Transitions.add(T_I3);

        // T_O1-----------------------------------------------------------------
        PetriTransition T_O1 = new PetriTransition(pn);
        T_O1.TransitionName = "T_O1";
        T_O1.InputPlaceName.add("P_INT1");
        T_O1.InputPlaceName.add("P_O1");

        Condition T14Ct1 = new Condition(T_O1, "P_INT1",
TransitionCondition.HaveCarForMe);
        Condition T14Ct2 = new Condition(T_O1, "P_O1",
TransitionCondition.CanAddCars);
        T14Ct1.SetNextCondition(LogicConnector.AND, T14Ct2);

        GuardMapping grdT14 = new GuardMapping();
        grdT14.condition = T14Ct1;
        grdT14.Activations.add(new Activation(T_O1, "P_INT1",
TransitionOperation.PopElementWithTargetToQueue, "P_O1"));
        T_O1.GuardMappingList.add(grdT14);

        T_O1.Delay = 2;
        pn.Transitions.add(T_O1);

        // T_I6-----------------------------------------------
        PetriTransition T_I6 = new PetriTransition(pn);
        T_I6.TransitionName = "T_I6";
        T_I6.InputPlaceName.add("P_I6");
        T_I6.InputPlaceName.add("P_INT1");

        Condition T15Ct1 = new Condition(T_I6, "P_I6",
TransitionCondition.NotNull);
```

```
        Condition T15Ct2 = new Condition(T_I6, "P_INT1",
TransitionCondition.CanAddCars);
        T15Ct1.SetNextCondition(LogicConnector.AND, T15Ct2);

        GuardMapping grdT15 = new GuardMapping();
        grdT15.condition = T15Ct1;
        grdT15.Activations.add(new Activation(T_I6, "P_I6",
TransitionOperation.AddElement, "P_INT1"));
        T_I6.GuardMappingList.add(grdT15);

        T_I6.Delay = 0;
        pn.Transitions.add(T_I6);

        // T_O5------------------------------------------------------------
        PetriTransition T_O5 = new PetriTransition(pn);
        T_O5.TransitionName = "T_O5";
        T_O5.InputPlaceName.add("P_INT1");
        T_O5.InputPlaceName.add("P_O5");

        Condition T16Ct1 = new Condition(T_O5, "P_INT1",
TransitionCondition.HaveCarForMe);
        Condition T16Ct2 = new Condition(T_O5, "P_O5",
TransitionCondition.CanAddCars);
        T16Ct1.SetNextCondition(LogicConnector.AND, T16Ct2);

        GuardMapping grdT16 = new GuardMapping();
        grdT16.condition = T16Ct1;
        grdT16.Activations.add(new Activation(T_O5, "P_INT1",
TransitionOperation.PopElementWithTargetToQueue, "P_O5"));
        T_O5.GuardMappingList.add(grdT16);

        T_O5.Delay = 2;
        pn.Transitions.add(T_O5);

        // T_O3------------------------------------------------------------
        PetriTransition T_O3 = new PetriTransition(pn);
        T_O3.TransitionName = "T_O3";
        T_O3.InputPlaceName.add("P_INT1");
        T_O3.InputPlaceName.add("P_O3");

        Condition T18Ct1 = new Condition(T_O3, "P_INT1",
TransitionCondition.HaveCarForMe);

        GuardMapping grdT18 = new GuardMapping();
        grdT18.condition = T18Ct1;
        grdT18.Activations.add(new Activation(T_O3, "P_INT1",
TransitionOperation.PopElementWithTarget, "P_O3"));
        T_O3.GuardMappingList.add(grdT18);

        T_O3.Delay = 2;
        pn.Transitions.add(T_O3);

        // T_M6------------------------------------------------
```

```
        PetriTransition T_M6 = new PetriTransition(pn);
        T_M6.TransitionName = "T_M6";
        T_M6.InputPlaceName.add("P_M4");
        T_M6.InputPlaceName.add("P_INT1");

        Condition T_M6_Ct1 = new Condition(T_M6, "P_M4",
TransitionCondition.NotNull);
        Condition T_M6_Ct2 = new Condition(T_M6, "P_INT1",
TransitionCondition.CanAddCars);
        T_M6_Ct1.SetNextCondition(LogicConnector.AND, T_M6_Ct2);

        GuardMapping grdT_M6 = new GuardMapping();
        grdT_M6.condition = T_M6_Ct1;
        grdT_M6.Activations.add(new Activation(T_M6, "P_M4",
TransitionOperation.AddElement, "P_INT1"));
        T_M6.GuardMappingList.add(grdT_M6);

        T_M6.Delay = 0;
        pn.Transitions.add(T_M6);

        // T_M1-------------------------------------------------------

        PetriTransition T_M1 = new PetriTransition(pn);
        T_M1.TransitionName = "T_M1";
        T_M1.InputPlaceName.add("P_INT1");
        T_M1.InputPlaceName.add("P_M1");

        Condition T_M1_Ct1 = new Condition(T_M1, "P_INT1",
TransitionCondition.HaveCarForMe);
        Condition T_M1_Ct2 = new Condition(T_M1, "P_M1",
TransitionCondition.CanAddCars);
        T_M1_Ct1.SetNextCondition(LogicConnector.AND, T_M1_Ct2);

        GuardMapping grdT_M1 = new GuardMapping();
        grdT_M1.condition = T_M1_Ct1;
        grdT_M1.Activations.add(new Activation(T_M1, "P_INT1",
TransitionOperation.PopElementWithTargetToQueue, "P_M1"));
        T_M1.GuardMappingList.add(grdT_M1);

        Condition T_M1_Ct3 = new Condition(T_M1, "P_INT1",
TransitionCondition.HaveCarForMe);
        Condition T_M1_Ct4 = new Condition(T_M1, "P_M1",
TransitionCondition.CanNotAddCars);
        T_M1_Ct3.SetNextCondition(LogicConnector.AND, T_M1_Ct4);

        GuardMapping grdT_M1_1 = new GuardMapping();
        grdT_M1_1.condition = T_M1_Ct3;
        grdT_M1_1.Activations.add(new Activation(T_M1, "P_INT1",
TransitionOperation.Copy, "P_INT1"));
        grdT_M1_1.Activations.add(new Activation(T_M1, "full",
TransitionOperation.SendOverNetwork, "OP6"));
        T_M1.GuardMappingList.add(grdT_M1_1);
```

```
        T_M1.Delay = 2;
        pn.Transitions.add(T_M1);

        //
-------------------------------------------------------------------------------
--------------
        //
--------------------------------Intersection2--------------------------------
---------------
        //
-------------------------------------------------------------------------------
--------------

        DataCarQueue P_INT2 = new DataCarQueue();
        P_INT2.Value.Size = 3;
        P_INT2.SetName("P_INT2");
        pn.PlaceList.add(P_INT2);

        // T_M2 ------------------------------------------------
        PetriTransition T_M2 = new PetriTransition(pn);
        T_M2.TransitionName = "T_M2";
        T_M2.InputPlaceName.add("P_M1");
        T_M2.InputPlaceName.add("P_TL3");

        Condition T_M2_Ct1 = new Condition(T_M2, "P_TL3",
TransitionCondition.Equal, "green");
        Condition T_M2_Ct2 = new Condition(T_M2, "P_M1",
TransitionCondition.HaveCar);
        T_M2_Ct1.SetNextCondition(LogicConnector.AND, T_M2_Ct2);

        GuardMapping grdT_M2 = new GuardMapping();
        grdT_M2.condition = T_M2_Ct1;
        grdT_M2.Activations.add(new Activation(T_M2, "P_M1",
TransitionOperation.PopElementWithoutTarget, "P_M2"));
        grdT_M2.Activations.add(new Activation(T_M2, "P_TL3",
TransitionOperation.Move, "P_TL3"));
        T_M2.GuardMappingList.add(grdT_M2);

        T_M2.Delay = 0;
        pn.Transitions.add(T_M2);

        // T_I13 -----------------------------------------------
        PetriTransition T_I13 = new PetriTransition(pn);
        T_I13.TransitionName = "T_I13";
        T_I13.InputPlaceName.add("P_I13");
        T_I13.InputPlaceName.add("P_I14");

        Condition T_I13_Ct1 = new Condition(T_I13, "P_I13",
TransitionCondition.NotNull);
        Condition T_I13_Ct2 = new Condition(T_I13, "P_I14",
TransitionCondition.CanAddCars);
        T_I13_Ct1.SetNextCondition(LogicConnector.AND, T_I13_Ct2);
```

```
        GuardMapping grdT_I13 = new GuardMapping();
        grdT_I13.condition = T_I13_Ct1;
        grdT_I13.Activations.add(new Activation(T_I13, "P_I13",
TransitionOperation.AddElement, "P_I14"));
        T_I13.GuardMappingList.add(grdT_I13);

        Condition T_I13_Ct3 = new Condition(T_I13, "P_I13",
TransitionCondition.NotNull);
        Condition T_I13_Ct4 = new Condition(T_I13, "P_I14",
TransitionCondition.CanNotAddCars);
        T_I13_Ct3.SetNextCondition(LogicConnector.AND, T_I13_Ct4);

        GuardMapping grdT_I13_1 = new GuardMapping();
        grdT_I13_1.condition = T_I13_Ct3;
        grdT_I13_1.Activations.add(new Activation(T_I13, "full",
TransitionOperation.SendOverNetwork, "OP5"));
        grdT_I13_1.Activations.add(new Activation(T_I13, "P_I13",
TransitionOperation.Copy, "P_I13"));
        T_I13.GuardMappingList.add(grdT_I13_1);

        T_I13.Delay = 0;
        pn.Transitions.add(T_I13);

        // T_I14 ------------------------------------------------
        PetriTransition T_I14 = new PetriTransition(pn);
        T_I14.TransitionName = "T_I14";
        T_I14.InputPlaceName.add("P_I14");
        T_I14.InputPlaceName.add("P_TL7");

        Condition T_I14_Ct1 = new Condition(T_I14, "P_TL7",
TransitionCondition.Equal, "green");
        Condition T_I14_Ct2 = new Condition(T_I14, "P_I14",
TransitionCondition.HaveCar);
        T_I14_Ct1.SetNextCondition(LogicConnector.AND, T_I14_Ct2);

        GuardMapping grdT_I14 = new GuardMapping();
        grdT_I14.condition = T_I14_Ct1;
        grdT_I14.Activations.add(new Activation(T_I14, "P_I14",
TransitionOperation.PopElementWithoutTarget, "P_I15"));
        grdT_I14.Activations.add(new Activation(T_I14, "P_TL7",
TransitionOperation.Move, "P_TL7"));
        T_I14.GuardMappingList.add(grdT_I14);

        T_I14.Delay = 0;
        pn.Transitions.add(T_I14);

        // T_I10 ------------------------------------------------
        PetriTransition T_I10 = new PetriTransition(pn);
        T_I10.TransitionName = "T_I10";
        T_I10.InputPlaceName.add("P_I10");
        T_I10.InputPlaceName.add("P_I11");
```

```
        Condition T7Ct1 = new Condition(T_I10, "P_I10",
TransitionCondition.NotNull);
        Condition T7Ct2 = new Condition(T_I10, "P_I11",
TransitionCondition.CanAddCars);
        T7Ct1.SetNextCondition(LogicConnector.AND, T7Ct2);

        GuardMapping grdT7 = new GuardMapping();
        grdT7.condition = T7Ct1;
        grdT7.Activations.add(new Activation(T_I10, "P_I10",
TransitionOperation.AddElement, "P_I11"));
        T_I10.GuardMappingList.add(grdT7);

        Condition T7Ct3 = new Condition(T_I10, "P_I10",
TransitionCondition.NotNull);
        Condition T7Ct4 = new Condition(T_I10, "P_I11",
TransitionCondition.CanNotAddCars);
        T7Ct3.SetNextCondition(LogicConnector.AND, T7Ct4);

        GuardMapping grdT7_1 = new GuardMapping();
        grdT7_1.condition = T7Ct3;
        grdT7_1.Activations.add(new Activation(T_I10, "full",
TransitionOperation.SendOverNetwork, "OP4"));
        grdT7_1.Activations.add(new Activation(T_I10, "P_I10",
TransitionOperation.Copy, "P_I10"));
        T_I10.GuardMappingList.add(grdT7_1);

        T_I10.Delay = 0;
        pn.Transitions.add(T_I10);

        // T_I11 ------------------------------------------------
        PetriTransition T_I11 = new PetriTransition(pn);
        T_I11.TransitionName = "T_I11";
        T_I11.InputPlaceName.add("P_I11");
        T_I11.InputPlaceName.add("P_TL6");

        Condition T8Ct1 = new Condition(T_I11, "P_TL6",
TransitionCondition.Equal, "green");
        Condition T8Ct2 = new Condition(T_I11, "P_I11",
TransitionCondition.HaveCar);
        T8Ct1.SetNextCondition(LogicConnector.AND, T8Ct2);

        GuardMapping grdT8 = new GuardMapping();
        grdT8.condition = T8Ct1;
        grdT8.Activations.add(new Activation(T_I11, "P_I11",
TransitionOperation.PopElementWithoutTarget, "P_I12"));
        grdT8.Activations.add(new Activation(T_I11, "P_TL6",
TransitionOperation.Move, "P_TL6"));
        T_I11.GuardMappingList.add(grdT8);

        T_I11.Delay = 0;
        pn.Transitions.add(T_I11);

        // T_I7 -------------------------------------------------
```

```
        PetriTransition T_I7 = new PetriTransition(pn);
        T_I7.TransitionName = "T_I7";
        T_I7.InputPlaceName.add("P_I7");
        T_I7.InputPlaceName.add("P_I8");

        Condition T_I7_Ct1 = new Condition(T_I7, "P_I7",
TransitionCondition.NotNull);
        Condition T_I7_Ct2 = new Condition(T_I7, "P_I8",
TransitionCondition.CanAddCars);
        T_I7_Ct1.SetNextCondition(LogicConnector.AND, T_I7_Ct2);

        GuardMapping grdT_I7 = new GuardMapping();
        grdT_I7.condition = T_I7_Ct1;
        grdT_I7.Activations.add(new Activation(T_I7, "P_I7",
TransitionOperation.AddElement, "P_I8"));
        T_I7.GuardMappingList.add(grdT_I7);

        Condition T_I7_Ct3 = new Condition(T_I7, "P_I7",
TransitionCondition.NotNull);
        Condition T_I7_Ct4 = new Condition(T_I7, "P_I8",
TransitionCondition.CanNotAddCars);
        T_I7_Ct3.SetNextCondition(LogicConnector.AND, T_I7_Ct4);

        GuardMapping grdT_I7_1 = new GuardMapping();
        grdT_I7_1.condition = T_I7_Ct3;
        grdT_I7_1.Activations.add(new Activation(T_I7, "full",
TransitionOperation.SendOverNetwork, "OP3"));
        grdT_I7_1.Activations.add(new Activation(T_I7, "P_I7",
TransitionOperation.Copy, "P_I7"));
        T_I7.GuardMappingList.add(grdT_I7_1);

        T_I7.Delay = 0;
        pn.Transitions.add(T_I7);

        // T_I8 ------------------------------------------------
        PetriTransition T_I8 = new PetriTransition(pn);
        T_I8.TransitionName = "T_I8";
        T_I8.InputPlaceName.add("P_I8");
        T_I8.InputPlaceName.add("P_TL5");

        Condition T_I8_Ct1 = new Condition(T_I8, "P_TL5",
TransitionCondition.Equal, "green");
        Condition T_I8_Ct2 = new Condition(T_I8, "P_I8",
TransitionCondition.HaveCar);
        T_I8_Ct1.SetNextCondition(LogicConnector.AND, T_I8_Ct2);

        GuardMapping grdT_I8 = new GuardMapping();
        grdT_I8.condition = T_I8_Ct1;
        grdT_I8.Activations.add(new Activation(T_I8, "P_I8",
TransitionOperation.PopElementWithoutTarget, "P_I9"));
        grdT_I8.Activations.add(new Activation(T_I8, "P_TL5",
TransitionOperation.Move, "P_TL5"));
        T_I8.GuardMappingList.add(grdT_I8);
```

```
        T_I8.Delay = 0;
        pn.Transitions.add(T_I8);

        // T_O8--------------------------------------------------------------------
        PetriTransition T_O8 = new PetriTransition(pn);
        T_O8.TransitionName = "T_O8";
        T_O8.InputPlaceName.add("P_O7");

        Condition T_O8_Ct1 = new Condition(T_O8, "P_O7",
TransitionCondition.HaveCar);

        GuardMapping grdT_O8 = new GuardMapping();
        grdT_O8.condition = T_O8_Ct1;
        grdT_O8.Activations.add(new Activation(T_O8, "P_O7",
TransitionOperation.PopElementWithoutTarget, "P_O8"));
        T_O8.GuardMappingList.add(grdT_O8);

        T_O8.Delay = 0;
        pn.Transitions.add(T_O8);

        //
T_O10-------------------------------------------------------------------
        PetriTransition T_O10 = new PetriTransition(pn);
        T_O10.TransitionName = "T_O10";
        T_O10.InputPlaceName.add("P_O9");

        Condition T_O10_Ct1 = new Condition(T_O10, "P_O9",
TransitionCondition.HaveCar);

        GuardMapping grdT_O10 = new GuardMapping();
        grdT_O10.condition = T_O10_Ct1;
        grdT_O10.Activations.add(new Activation(T_O10, "P_O9",
TransitionOperation.PopElementWithoutTarget, "P_O10"));
        T_O10.GuardMappingList.add(grdT_O10);

        T_O10.Delay = 0;
        pn.Transitions.add(T_O10);

        // T_O9--------------------------------------------------------------
        PetriTransition T_O9 = new PetriTransition(pn);
        T_O9.TransitionName = "T_O9";
        T_O9.InputPlaceName.add("P_INT2");
        T_O9.InputPlaceName.add("P_O9");

        Condition T_O9_Ct1 = new Condition(T_O9, "P_INT2",
TransitionCondition.HaveCarForMe);
        Condition T_O9_Ct2 = new Condition(T_O9, "P_O9",
TransitionCondition.CanAddCars);
        T_O9_Ct1.SetNextCondition(LogicConnector.AND, T_O9_Ct2);

        GuardMapping grdT_O9 = new GuardMapping();
        grdT_O9.condition = T_O9_Ct1;
```

```
        grdT_O9.Activations.add(new Activation(T_O9, "P_INT2",
TransitionOperation.PopElementWithTargetToQueue, "P_O9"));
        T_O9.GuardMappingList.add(grdT_O9);

        T_O9.Delay = 2;
        pn.Transitions.add(T_O9);

        // T_O7-------------------------------------------------------------
        PetriTransition T_O7 = new PetriTransition(pn);
        T_O7.TransitionName = "T_O7";
        T_O7.InputPlaceName.add("P_INT2");
        T_O7.InputPlaceName.add("P_O7");

        Condition T_O7_Ct1 = new Condition(T_O7, "P_INT2",
TransitionCondition.HaveCarForMe);
        Condition T_O7_Ct2 = new Condition(T_O7, "P_O7",
TransitionCondition.CanAddCars);
        T_O7_Ct1.SetNextCondition(LogicConnector.AND, T_O7_Ct2);

        GuardMapping grdT_O7 = new GuardMapping();
        grdT_O7.condition = T_O7_Ct1;
        grdT_O7.Activations.add(new Activation(T_O7, "P_INT2",
TransitionOperation.PopElementWithTargetToQueue, "P_O7"));
        T_O7.GuardMappingList.add(grdT_O7);

        T_O7.Delay = 2;
        pn.Transitions.add(T_O7);

        // T_M3-------------------------------------------------
        PetriTransition T_M3 = new PetriTransition(pn);
        T_M3.TransitionName = "T_M3";
        T_M3.InputPlaceName.add("P_M2");
        T_M3.InputPlaceName.add("P_INT2");

        Condition T_M3_Ct1 = new Condition(T_M3, "P_M2",
TransitionCondition.NotNull);
        Condition T_M3_Ct2 = new Condition(T_M3, "P_INT2",
TransitionCondition.CanAddCars);
        T_M3_Ct1.SetNextCondition(LogicConnector.AND, T_M3_Ct2);

        GuardMapping grdT_M3 = new GuardMapping();
        grdT_M3.condition = T_M3_Ct1;
        grdT_M3.Activations.add(new Activation(T_M3, "P_M2",
TransitionOperation.AddElement, "P_INT2"));
        T_M3.GuardMappingList.add(grdT_M3);

        T_M3.Delay = 0;
        pn.Transitions.add(T_M3);

        // T_M4-------------------------------------------------------------
        PetriTransition T_M4 = new PetriTransition(pn);
        T_M4.TransitionName = "T_M4";
        T_M4.InputPlaceName.add("P_INT2");
```

```
        T_M4.InputPlaceName.add("P_M3");

        Condition T_M4_Ct1 = new Condition(T_M4, "P_INT2",
TransitionCondition.HaveCarForMe);
        Condition T_M4_Ct2 = new Condition(T_M4, "P_M3",
TransitionCondition.CanAddCars);
        T_M4_Ct1.SetNextCondition(LogicConnector.AND, T_M4_Ct2);

        GuardMapping grdT_M4 = new GuardMapping();
        grdT_M4.condition = T_M4_Ct1;
        grdT_M4.Activations.add(new Activation(T_M4, "P_INT2",
TransitionOperation.PopElementWithTargetToQueue, "P_M3"));
        T_M4.GuardMappingList.add(grdT_M4);

        Condition T_M4_Ct3 = new Condition(T_M4, "P_INT2",
TransitionCondition.HaveCarForMe);
        Condition T_M4_Ct4 = new Condition(T_M4, "P_M3",
TransitionCondition.CanNotAddCars);
        T_M4_Ct3.SetNextCondition(LogicConnector.AND, T_M4_Ct4);

        GuardMapping grdT_M4_1 = new GuardMapping();
        grdT_M4_1.condition = T_M4_Ct3;
        grdT_M4_1.Activations.add(new Activation(T_M4, "P_INT2",
TransitionOperation.Move, "P_INT2"));
        grdT_M4_1.Activations.add(new Activation(T_M4, "full",
TransitionOperation.SendOverNetwork, "OP7"));
        T_M4.GuardMappingList.add(grdT_M4_1);

        T_M4.Delay = 2;
        pn.Transitions.add(T_M4);



        // T_I15-------------------------------------------------
        PetriTransition T_I15 = new PetriTransition(pn);
        T_I15.TransitionName = "T_I15";
        T_I15.InputPlaceName.add("P_I15");
        T_I15.InputPlaceName.add("P_INT2");

        Condition T17Ct1 = new Condition(T_I15, "P_I15",
TransitionCondition.NotNull);
        Condition T17Ct2 = new Condition(T_I15, "P_INT2",
TransitionCondition.CanAddCars);
        T17Ct1.SetNextCondition(LogicConnector.AND, T17Ct2);

        GuardMapping grdT17 = new GuardMapping();
        grdT17.condition = T17Ct1;
        grdT17.Activations.add(new Activation(T_I15, "P_I15",
TransitionOperation.AddElement, "P_INT2"));
        T_I15.GuardMappingList.add(grdT17);

        T_I15.Delay = 0;
        pn.Transitions.add(T_I15);
```

```
        // T_I12-----------------------------------------------
        PetriTransition T_I12 = new PetriTransition(pn);
        T_I12.TransitionName = "T_I12";
        T_I12.InputPlaceName.add("P_I12");
        T_I12.InputPlaceName.add("P_INT2");

        Condition T19Ct1 = new Condition(T_I12, "P_I12",
TransitionCondition.NotNull);
        Condition T19Ct2 = new Condition(T_I12, "P_INT2",
TransitionCondition.CanAddCars);
        T19Ct1.SetNextCondition(LogicConnector.AND, T19Ct2);

        GuardMapping grdT19 = new GuardMapping();
        grdT19.condition = T19Ct1;
        grdT19.Activations.add(new Activation(T_I12, "P_I12",
TransitionOperation.PopElementWithoutTarget, "P_INT2"));
        T_I12.GuardMappingList.add(grdT19);

        T_I12.Delay = 0;
        pn.Transitions.add(T_I12);

        // T_I9------------------------------------------------
        PetriTransition T_I9 = new PetriTransition(pn);
        T_I9.TransitionName = "T_I9";
        T_I9.InputPlaceName.add("P_I9");
        T_I9.InputPlaceName.add("P_INT2");

        Condition T_I9_Ct1 = new Condition(T_I9, "P_I9",
TransitionCondition.NotNull);
        Condition T_I9_Ct2 = new Condition(T_I9, "P_INT2",
TransitionCondition.CanAddCars);
        T_I9_Ct1.SetNextCondition(LogicConnector.AND, T_I9_Ct2);

        GuardMapping grdT_I9 = new GuardMapping();
        grdT_I9.condition = T_I9_Ct1;
        grdT_I9.Activations.add(new Activation(T_I9, "P_I9",
TransitionOperation.PopElementWithoutTarget, "P_INT2"));
        T_I9.GuardMappingList.add(grdT_I9);

        T_I9.Delay = 0;
        pn.Transitions.add(T_I9);

        //
--------------------------------------------------------------------------------
--------
        //
----------------------------PNStart---------------------------------------------
-------
        //
--------------------------------------------------------------------------------
--------
```

```
        System.out.println("Lanes intersection started \n
------------------------------");
        pn.Delay = 2000;
        // pn.Start();

        PetriNetWindow frame = new PetriNetWindow(false);
        frame.petriNet = pn;
        frame.setVisible(true);
    }
}
```

Controller 1

```
package NullPointerException;

import Components.*;
import DataObjects.DataInteger;
import DataObjects.DataString;
import DataObjects.DataTransfer;
import DataOnly.TransferOperation;
import Enumerations.LogicConnector;
import Enumerations.TransitionCondition;
import Enumerations.TransitionOperation;

import java.util.ArrayList;

public class Controller_int1 {
    public static void main(String[] args) {
        PetriNet pn = new PetriNet();
        pn.PetriNetName = "Controller intersection 1";
        pn.SetName("Controller intersection 1");
        pn.NetworkPort = 1081;

        DataString ini = new DataString();
        ini.SetName("ini");
        ini.SetValue("red");
        pn.ConstantPlaceList.add(ini);

        DataString red = new DataString();
        red.SetName("red");
        red.SetValue("red");
        pn.ConstantPlaceList.add(red);

        DataString green = new DataString();
        green.SetName("green");
        green.SetValue("green");
        pn.ConstantPlaceList.add(green);

        DataString yellow = new DataString();
```

```
yellow.SetName("yellow");
yellow.SetValue("yellow");
pn.ConstantPlaceList.add(yellow);

DataString p1 = new DataString();
p1.SetName("r1r2r3");
p1.SetValue("signal");
pn.PlaceList.add(p1);

DataString p2 = new DataString();
p2.SetName("g1r2r3");
pn.PlaceList.add(p2);

DataString p3 = new DataString();
p3.SetName("y1r2r3");
pn.PlaceList.add(p3);

DataString p4 = new DataString();
p4.SetName("r1g2r3");
pn.PlaceList.add(p4);

DataString p5 = new DataString();
p5.SetName("r1y2r3");
pn.PlaceList.add(p5);

DataString p6 = new DataString();
p6.SetName("r1r2g3");
pn.PlaceList.add(p6);

DataString p7 = new DataString();
p7.SetName("r1r2y3");
pn.PlaceList.add(p7);

DataTransfer p10 = new DataTransfer();
p10.SetName("OP1");
p10.Value = new TransferOperation("localhost", "1080" , "P_TL1");
pn.PlaceList.add(p10);

DataTransfer p11 = new DataTransfer();
p11.SetName("OP2");
p11.Value = new TransferOperation("localhost", "1080" , "P_TL2");
pn.PlaceList.add(p11);

DataTransfer p12 = new DataTransfer();
p12.SetName("OP3");
p12.Value = new TransferOperation("localhost", "1080" , "P_TL4");
pn.PlaceList.add(p12);

DataInteger constantValue1 = new DataInteger();
constantValue1.SetName("constantValue1");
constantValue1.SetValue(0);
pn.ConstantPlaceList.add(constantValue1);
```

```
        DataInteger constantValue2 = new DataInteger();
        constantValue2.SetName("constantValue2");
        constantValue2.SetValue(1);
        pn.ConstantPlaceList.add(constantValue2);

        DataInteger constantValue3 = new DataInteger();
        constantValue3.SetName("constantValue3");
        constantValue3.SetValue(3);
        pn.ConstantPlaceList.add(constantValue3);

        DataString in1 = new DataString();
        in1.SetName("in1");
        pn.PlaceList.add(in1);

        DataInteger counter1 = new DataInteger();
        counter1.SetName("counter1");
        counter1.Value = 3;
        pn.PlaceList.add(counter1);

        DataString PlusTL1 = new DataString();
        PlusTL1.SetName("PlusTL1");
        pn.PlaceList.add(PlusTL1);

        DataString in2 = new DataString();
        in2.SetName("in2");
        pn.PlaceList.add(in2);

        DataInteger counter2 = new DataInteger();
        counter2.SetName("counter2");
        counter2.Value = 3;
        pn.PlaceList.add(counter2);

        DataString PlusTL2 = new DataString();
        PlusTL2.SetName("PlusTL2");
        pn.PlaceList.add(PlusTL2);

        DataString in7 = new DataString();
        in7.SetName("in7");
        pn.PlaceList.add(in7);

        DataInteger counter3 = new DataInteger();
        counter3.SetName("counter3");
        counter3.Value = 3;
        pn.PlaceList.add(counter3);

        DataString PlusTL3 = new DataString();
        PlusTL3.SetName("PlusTL3");
        pn.PlaceList.add(PlusTL3);



//----------------------------iniT-------------------------------------
        PetriTransition iniT = new PetriTransition(pn);
```

```
        iniT.TransitionName = "iniT";

        Condition iniTCt1 = new Condition(iniT, "ini",
TransitionCondition.NotNull);

        GuardMapping grdiniT = new GuardMapping();
        grdiniT.condition= iniTCt1;

        grdiniT.Activations.add(new Activation(iniT, "ini",
TransitionOperation.SendOverNetwork, "OP1"));
        grdiniT.Activations.add(new Activation(iniT, "ini",
TransitionOperation.SendOverNetwork, "OP2"));
        grdiniT.Activations.add(new Activation(iniT, "ini",
TransitionOperation.SendOverNetwork, "OP3"));
        grdiniT.Activations.add(new Activation(iniT, "",
TransitionOperation.MakeNull, "ini"));

        iniT.GuardMappingList.add(grdiniT);

        iniT.Delay = 0;
        pn.Transitions.add(iniT);



        //----------------------------T1-----------------------------------
        PetriTransition t1 = new PetriTransition(pn);
        t1.TransitionName = "T1";
        t1.InputPlaceName.add("r1r2r3");


        Condition T1Ct1 = new Condition(t1, "r1r2r3",
TransitionCondition.NotNull);

        GuardMapping grdT1 = new GuardMapping();
        grdT1.condition= T1Ct1;
        grdT1.Activations.add(new Activation(t1, "r1r2r3",
TransitionOperation.Move, "g1r2r3"));
        grdT1.Activations.add(new Activation(t1, "green",
TransitionOperation.SendOverNetwork, "OP1"));

        t1.GuardMappingList.add(grdT1);

        t1.Delay = 5;
        pn.Transitions.add(t1);

        //----------------------------T2-----------------------------------
        PetriTransition t2 = new PetriTransition(pn);
        t2.TransitionName = "T2";
        t2.InputPlaceName.add("g1r2r3");
        t2.InputPlaceName.add("in1");
        t2.InputPlaceName.add("counter1");
```

```
        Condition T2Ct1 = new Condition(t2, "in1",
TransitionCondition.IsNull);
        Condition T2Ct2 = new Condition(t2, "g1r2r3",
TransitionCondition.NotNull);
        T2Ct1.SetNextCondition(LogicConnector.AND, T2Ct2);

        GuardMapping grdT2_1 = new GuardMapping();
        grdT2_1.condition= T2Ct1;
        grdT2_1.Activations.add(new Activation(t2, "g1r2r3",
TransitionOperation.Move, "y1r2r3"));
        grdT2_1.Activations.add(new Activation(t2, "yellow",
TransitionOperation.SendOverNetwork, "OP1"));
        grdT2_1.Activations.add(new Activation(t2, "counter1",
TransitionOperation.Move, "counter1"));
        t2.GuardMappingList.add(grdT2_1);


        Condition T2Ct3 = new Condition(t2, "in1",
TransitionCondition.NotNull);
        Condition T2Ct4 = new Condition(t2, "counter1",
TransitionCondition.MoreThan, "constantValue1");
        Condition T2Ct5 = new Condition(t2, "g1r2r3",
TransitionCondition.NotNull);
        T2Ct3.SetNextCondition(LogicConnector.AND, T2Ct4);
        T2Ct4.SetNextCondition(LogicConnector.AND, T2Ct5);

        GuardMapping grdT2_2 = new GuardMapping();
        grdT2_2.condition= T2Ct3;

        ArrayList<String> inputPlaces1 = new ArrayList<>();
        inputPlaces1.add("counter1");
        inputPlaces1.add("constantValue2");

        grdT2_2.Activations.add(new Activation(t2, "g1r2r3",
TransitionOperation.Move, "PlusTL1"));
        grdT2_2.Activations.add(new Activation(t2, inputPlaces1,
TransitionOperation.Sub, "counter1"));
        t2.GuardMappingList.add(grdT2_2);



        Condition T2Ct6 = new Condition(t2, "counter1",
TransitionCondition.Equal, "constantValue1");
        Condition T2Ct7 = new Condition(t2, "g1r2r3",
TransitionCondition.NotNull);
        T2Ct6.SetNextCondition(LogicConnector.AND, T2Ct7);

        GuardMapping grdT2_3 = new GuardMapping();
        grdT2_3.condition= T2Ct6;
        grdT2_3.Activations.add(new Activation(t2, "g1r2r3",
TransitionOperation.Move, "y1r2r3"));
        grdT2_3.Activations.add(new Activation(t2, "constantValue3",
TransitionOperation.Copy, "counter1"));
```

```
        grdT2_3.Activations.add(new Activation(t2, "yellow",
TransitionOperation.SendOverNetwork, "OP1"));
        t2.GuardMappingList.add(grdT2_3);


        t2.Delay = 5;
        pn.Transitions.add(t2);


        //----------------------------T3-----------------------------------
        PetriTransition t3 = new PetriTransition(pn);
        t3.TransitionName = "T3";
        t3.InputPlaceName.add("y1r2r3");



        Condition T3Ct1 = new Condition(t3, "y1r2r3",
TransitionCondition.NotNull);

        GuardMapping grdT3 = new GuardMapping();
        grdT3.condition= T3Ct1;
        grdT3.Activations.add(new Activation(t3, "y1r2r3",
TransitionOperation.Move, "r1g2r3"));
        grdT3.Activations.add(new Activation(t3, "red",
TransitionOperation.SendOverNetwork, "OP1"));
        grdT3.Activations.add(new Activation(t3, "green",
TransitionOperation.SendOverNetwork, "OP2"));

        t3.GuardMappingList.add(grdT3);

        t3.Delay = 5;
        pn.Transitions.add(t3);


        //----------------------------T4-----------------------------------
        PetriTransition t4 = new PetriTransition(pn);
        t4.TransitionName = "T4";
        t4.InputPlaceName.add("r1g2r3");
        t4.InputPlaceName.add("in2");
        t4.InputPlaceName.add("counter2");

        Condition T4Ct1 = new Condition(t4, "in2",
TransitionCondition.IsNull);
        Condition T4Ct2 = new Condition(t4, "r1g2r3",
TransitionCondition.NotNull);
        T4Ct1.SetNextCondition(LogicConnector.AND, T4Ct2);

        GuardMapping grdT4_1 = new GuardMapping();
        grdT4_1.condition= T4Ct1;
        grdT4_1.Activations.add(new Activation(t4, "r1g2r3",
TransitionOperation.Move, "r1y2r3"));
        grdT4_1.Activations.add(new Activation(t4, "yellow",
TransitionOperation.SendOverNetwork, "OP2"));
```

```
        grdT4_1.Activations.add(new Activation(t4, "counter2",
TransitionOperation.Move, "counter2"));
        t4.GuardMappingList.add(grdT4_1);


        Condition T4Ct3 = new Condition(t4, "in2",
TransitionCondition.NotNull);
        Condition T4Ct4 = new Condition(t4, "counter2",
TransitionCondition.MoreThan, "constantValue1");
        Condition T4Ct5 = new Condition(t4, "r1g2r3",
TransitionCondition.NotNull);
        T4Ct3.SetNextCondition(LogicConnector.AND, T4Ct4);
        T4Ct4.SetNextCondition(LogicConnector.AND, T4Ct5);

        GuardMapping grdT4_2 = new GuardMapping();
        grdT4_2.condition= T4Ct3;

        ArrayList<String> inputPlaces2 = new ArrayList<>();
        inputPlaces2.add("counter2");
        inputPlaces2.add("constantValue2");

        grdT4_2.Activations.add(new Activation(t4, "r1g2r3",
TransitionOperation.Move, "PlusTL2"));
        grdT4_2.Activations.add(new Activation(t4, inputPlaces2,
TransitionOperation.Sub, "counter2"));
        t4.GuardMappingList.add(grdT4_2);



        Condition T4Ct6 = new Condition(t4, "counter2",
TransitionCondition.Equal, "constantValue1");
        Condition T4Ct7 = new Condition(t4, "r1g2r3",
TransitionCondition.NotNull);
        T4Ct6.SetNextCondition(LogicConnector.AND, T4Ct7);

        GuardMapping grdT4_3 = new GuardMapping();
        grdT4_3.condition= T4Ct6;
        grdT4_3.Activations.add(new Activation(t4, "r1g2r3",
TransitionOperation.Move, "r1y2r3"));
        grdT4_3.Activations.add(new Activation(t4, "constantValue3",
TransitionOperation.Copy, "counter2"));
        grdT4_3.Activations.add(new Activation(t4, "yellow",
TransitionOperation.SendOverNetwork, "OP2"));
        t4.GuardMappingList.add(grdT4_3);


        t4.Delay = 5;
        pn.Transitions.add(t4);


        //----------------------------T5-----------------------------------
        PetriTransition t5 = new PetriTransition(pn);
        t5.TransitionName = "T5";
```

```
        t5.InputPlaceName.add("r1y2r3");


        Condition T5Ct1 = new Condition(t5, "r1y2r3",
TransitionCondition.NotNull);

        GuardMapping grdT5 = new GuardMapping();
        grdT5.condition= T5Ct1;
        grdT5.Activations.add(new Activation(t5, "r1y2r3",
TransitionOperation.Move, "r1r2g3"));
        grdT5.Activations.add(new Activation(t5, "red",
TransitionOperation.SendOverNetwork, "OP2"));
        grdT5.Activations.add(new Activation(t5, "green",
TransitionOperation.SendOverNetwork, "OP3"));


        t5.GuardMappingList.add(grdT5);

        t5.Delay = 5;
        pn.Transitions.add(t5);


        //----------------------------T6-----------------------------------
        PetriTransition t6 = new PetriTransition(pn);
        t6.TransitionName = "T6";
        t6.InputPlaceName.add("r1r2g3");
        t6.InputPlaceName.add("in7");
        t6.InputPlaceName.add("counter3");

        Condition T6Ct1 = new Condition(t6, "in7",
TransitionCondition.IsNull);
        Condition T6Ct2 = new Condition(t6, "r1r2g3",
TransitionCondition.NotNull);
        T6Ct1.SetNextCondition(LogicConnector.AND, T6Ct2);

        GuardMapping grdT6_1 = new GuardMapping();
        grdT6_1.condition= T6Ct1;
        grdT6_1.Activations.add(new Activation(t6, "r1r2g3",
TransitionOperation.Move, "r1r2y3"));
        grdT6_1.Activations.add(new Activation(t6, "yellow",
TransitionOperation.SendOverNetwork, "OP3"));
        grdT6_1.Activations.add(new Activation(t6, "counter3",
TransitionOperation.Move, "counter3"));
        t6.GuardMappingList.add(grdT6_1);


        Condition T6Ct3 = new Condition(t6, "in7",
TransitionCondition.NotNull);
        Condition T6Ct4 = new Condition(t6, "counter3",
TransitionCondition.MoreThan, "constantValue1");
        Condition T6Ct5 = new Condition(t6, "r1r2g3",
TransitionCondition.NotNull);
        T6Ct3.SetNextCondition(LogicConnector.AND, T6Ct4);
```

```
        T6Ct4.SetNextCondition(LogicConnector.AND, T6Ct5);

        GuardMapping grdT6_2 = new GuardMapping();
        grdT6_2.condition= T6Ct3;

        ArrayList<String> inputPlaces3 = new ArrayList<>();
        inputPlaces3.add("counter3");
        inputPlaces3.add("constantValue2");

        grdT6_2.Activations.add(new Activation(t6, "r1r2g3",
TransitionOperation.Move, "PlusTL3"));
        grdT6_2.Activations.add(new Activation(t6, inputPlaces3,
TransitionOperation.Sub, "counter3"));
        t6.GuardMappingList.add(grdT6_2);



        Condition T6Ct6 = new Condition(t6, "counter3",
TransitionCondition.Equal, "constantValue1");
        Condition T6Ct7 = new Condition(t6, "r1r2g3",
TransitionCondition.NotNull);
        T6Ct6.SetNextCondition(LogicConnector.AND, T6Ct7);

        GuardMapping grdT6_3 = new GuardMapping();
        grdT6_3.condition= T6Ct6;
        grdT6_3.Activations.add(new Activation(t6, "r1r2g3",
TransitionOperation.Move, "r1r2y3"));
        grdT6_3.Activations.add(new Activation(t6, "constantValue3",
TransitionOperation.Copy, "counter3"));
        grdT6_3.Activations.add(new Activation(t6, "yellow",
TransitionOperation.SendOverNetwork, "OP3"));
        t6.GuardMappingList.add(grdT6_3);

        t6.Delay = 5;
        pn.Transitions.add(t6);


        //----------------------------T7-----------------------------------
        PetriTransition t7 = new PetriTransition(pn);
        t7.TransitionName = "T7";
        t7.InputPlaceName.add("r1r2y3");


        Condition T7Ct1 = new Condition(t7, "r1r2y3",
TransitionCondition.NotNull);

        GuardMapping grdT7 = new GuardMapping();
        grdT7.condition= T7Ct1;
        grdT7.Activations.add(new Activation(t7, "r1r2y3",
TransitionOperation.Move, "r1r2r3"));
        grdT7.Activations.add(new Activation(t7, "red",
TransitionOperation.SendOverNetwork, "OP3"));
```

```
        t7.GuardMappingList.add(grdT7);

        t7.Delay = 5;
        pn.Transitions.add(t7);



//----------------------------T1Plus-------------------------------------
        PetriTransition t1plus = new PetriTransition(pn);
        t1plus.TransitionName = "T1Plus";
        t1plus.InputPlaceName.add("PlusTL1");


        Condition T1PlusCt1 = new Condition(t1plus, "PlusTL1",
TransitionCondition.NotNull);

        GuardMapping grdT1Plus = new GuardMapping();
        grdT1Plus.condition= T1PlusCt1;
        grdT1Plus.Activations.add(new Activation(t1plus, "PlusTL1",
TransitionOperation.Move, "g1r2r3"));

        t1plus.GuardMappingList.add(grdT1Plus);

        t1plus.Delay = 0;
        pn.Transitions.add(t1plus);


//----------------------------T2Plus-------------------------------------
        PetriTransition t2plus = new PetriTransition(pn);
        t2plus.TransitionName = "T2Plus";
        t2plus.InputPlaceName.add("PlusTL2");


        Condition T2PlusCt1 = new Condition(t2plus, "PlusTL2",
TransitionCondition.NotNull);

        GuardMapping grdT2Plus = new GuardMapping();
        grdT2Plus.condition= T2PlusCt1;
        grdT2Plus.Activations.add(new Activation(t2plus, "PlusTL2",
TransitionOperation.Move, "r1g2r3"));

        t2plus.GuardMappingList.add(grdT2Plus);

        t2plus.Delay = 0;
        pn.Transitions.add(t2plus);


//----------------------------T3Plus-------------------------------------
        PetriTransition t3plus = new PetriTransition(pn);
        t3plus.TransitionName = "T3Plus";
        t3plus.InputPlaceName.add("PlusTL3");
```

```
        Condition T3PlusCt1 = new Condition(t3plus, "PlusTL3",
TransitionCondition.NotNull);

        GuardMapping grdT3Plus = new GuardMapping();
        grdT3Plus.condition= T3PlusCt1;
        grdT3Plus.Activations.add(new Activation(t3plus, "PlusTL3",
TransitionOperation.Move, "r1r2g3"));

        t3plus.GuardMappingList.add(grdT3Plus);

        t3plus.Delay = 0;
        pn.Transitions.add(t3plus);


        //
-------------------------------------------------------------------------
----------
        // ----------------------------PN
Start-------------------------------------------------
        //
-------------------------------------------------------------------------
----------

        System.out.println("Controller1 started \n
------------------------------");
        pn.Delay = 2000;
        // pn.Start();

        PetriNetWindow frame = new PetriNetWindow(false);
        frame.petriNet = pn;
        frame.setVisible(true);
    }
}
```

Controller 2

```
package NullPointerException;

import Components.*;
import DataObjects.DataFloat;
import DataObjects.DataString;
import DataObjects.DataTransfer;
import DataOnly.TransferOperation;
import Enumerations.LogicConnector;
import Enumerations.TransitionCondition;
import Enumerations.TransitionOperation;

import java.util.ArrayList;
```

```
public class Controller_int2 {
   public static void main(String[] args) {
        PetriNet pn = new PetriNet();
        pn.PetriNetName = "Controller intersection 2";
        pn.SetName("Controller intersection 2");
        pn.NetworkPort = 1082;

        DataString ini = new DataString();
        //ini.Printable = false;
        ini.SetName("ini");
        ini.SetValue("red");
        pn.ConstantPlaceList.add(ini);

        DataString red = new DataString();
        //red.Printable = false;
        red.SetName("red");
        red.SetValue("red");
        pn.ConstantPlaceList.add(red);

        DataString green = new DataString();
        //green.Printable = false;
        green.SetName("green");
        green.SetValue("green");
        pn.ConstantPlaceList.add(green);

        DataString yellow = new DataString();
        //yellow.Printable = false;
        yellow.SetName("yellow");
        yellow.SetValue("yellow");
        pn.ConstantPlaceList.add(yellow);

        DataString p1 = new DataString();
        p1.SetName("r1r2r3r4");
        p1.SetValue("signal");
        pn.PlaceList.add(p1);

        DataString p2 = new DataString();
        p2.SetName("g1r2r3r4");
        pn.PlaceList.add(p2);

        DataString p3 = new DataString();
        p3.SetName("y1r2r3r4");
        pn.PlaceList.add(p3);

        DataString p4 = new DataString();
        p4.SetName("r1g2r3r4");
        pn.PlaceList.add(p4);

        DataString p5 = new DataString();
        p5.SetName("r1y2r3r4");
        pn.PlaceList.add(p5);
```

```
DataString p6 = new DataString();
p6.SetName("r1r2g3r4");
pn.PlaceList.add(p6);

DataString p7 = new DataString();
p7.SetName("r1r2y3r4");
pn.PlaceList.add(p7);

DataString p8 = new DataString();
p8.SetName("r1r2r3g4");
pn.PlaceList.add(p8);

DataString p9 = new DataString();
p9.SetName("r1r2r3y4");
pn.PlaceList.add(p9);

DataString in3 = new DataString();
in3.SetName("in3");
pn.PlaceList.add(in3);

DataString in4 = new DataString();
in4.SetName("in4");
pn.PlaceList.add(in4);

DataString in5 = new DataString();
in5.SetName("in5");
pn.PlaceList.add(in5);

DataString in6 = new DataString();
in6.SetName("in6");
pn.PlaceList.add(in6);

DataString PlusTL1 = new DataString();
PlusTL1.SetName("PlusTL1");
pn.PlaceList.add(PlusTL1);

DataString PlusTL2 = new DataString();
PlusTL2.SetName("PlusTL2");
pn.PlaceList.add(PlusTL2);

DataString PlusTL3 = new DataString();
PlusTL3.SetName("PlusTL3");
pn.PlaceList.add(PlusTL3);

DataString PlusTL4 = new DataString();
PlusTL4.SetName("PlusTL4");
pn.PlaceList.add(PlusTL4);

DataFloat counter1 = new DataFloat();
counter1.SetName("counter1");
counter1.SetValue(3.0f);
pn.PlaceList.add(counter1);
```

```
        DataFloat counter2 = new DataFloat();
        counter2.SetName("counter2");
        counter2.SetValue(3.0f);
        pn.PlaceList.add(counter2);

        DataFloat counter3 = new DataFloat();
        counter3.SetName("counter3");
        counter3.SetValue(3.0f);
        pn.PlaceList.add(counter3);

        DataFloat counter4 = new DataFloat();
        counter4.SetName("counter4");
        counter4.SetValue(3.0f);
        pn.PlaceList.add(counter4);

        DataFloat zero = new DataFloat();
        zero.SetName("zero");
        zero.SetValue(0);
        pn.ConstantPlaceList.add(zero);

        DataFloat initialValue = new DataFloat();
        initialValue.SetName("initialValue");
        initialValue.SetValue(3);
        pn.ConstantPlaceList.add(initialValue);

        DataFloat constantValue = new DataFloat();
        constantValue.SetName("constantValue");
        constantValue.SetValue(1);
        pn.ConstantPlaceList.add(constantValue);

        DataTransfer p10 = new DataTransfer();
        p10.SetName("OP1");
        p10.Value = new TransferOperation("localhost", "1080", "P_TL5");
        pn.PlaceList.add(p10);

        DataTransfer p11 = new DataTransfer();
        p11.SetName("OP2");
        p11.Value = new TransferOperation("localhost", "1080", "P_TL6");
        pn.PlaceList.add(p11);

        DataTransfer p12 = new DataTransfer();
        p12.SetName("OP3");
        p12.Value = new TransferOperation("localhost", "1080", "P_TL7");
        pn.PlaceList.add(p12);

        DataTransfer p13 = new DataTransfer();
        p13.SetName("OP4");
        p13.Value = new TransferOperation("localhost", "1080", "P_TL3");
        pn.PlaceList.add(p13);



//----------------------------iniT-------------------------------------
```

```
        PetriTransition iniT = new PetriTransition(pn);
        iniT.TransitionName = "iniT";

        Condition iniTCt1 = new Condition(iniT, "ini",
TransitionCondition.NotNull);

        GuardMapping grdiniT = new GuardMapping();
        grdiniT.condition = iniTCt1;

        grdiniT.Activations.add(new Activation(iniT, "ini",
TransitionOperation.SendOverNetwork, "OP1"));
        grdiniT.Activations.add(new Activation(iniT, "ini",
TransitionOperation.SendOverNetwork, "OP2"));
        grdiniT.Activations.add(new Activation(iniT, "ini",
TransitionOperation.SendOverNetwork, "OP3"));
        grdiniT.Activations.add(new Activation(iniT, "ini",
TransitionOperation.SendOverNetwork, "OP4"));
        grdiniT.Activations.add(new Activation(iniT, "",
TransitionOperation.MakeNull, "ini"));

        iniT.GuardMappingList.add(grdiniT);

        iniT.Delay = 5;
        pn.Transitions.add(iniT);


        //----------------------------T1-----------------------------------
        PetriTransition t1 = new PetriTransition(pn);
        t1.TransitionName = "T1";
        t1.InputPlaceName.add("r1r2r3r4");


        Condition T1Ct1 = new Condition(t1, "r1r2r3r4",
TransitionCondition.NotNull);

        GuardMapping grdT1 = new GuardMapping();
        grdT1.condition = T1Ct1;
        grdT1.Activations.add(new Activation(t1, "r1r2r3r4",
TransitionOperation.Move, "g1r2r3r4"));
        grdT1.Activations.add(new Activation(t1, "green",
TransitionOperation.SendOverNetwork, "OP1"));
        t1.GuardMappingList.add(grdT1);

        t1.Delay = 5;
        pn.Transitions.add(t1);

        //----------------------------T2-----------------------------------
        PetriTransition t2 = new PetriTransition(pn);
        t2.TransitionName = "T2";
        t2.InputPlaceName.add("g1r2r3r4");
        t2.InputPlaceName.add("in3");
        t2.InputPlaceName.add("counter1");
```

```
        Condition T2Ct1 = new Condition(t2, "in3",
TransitionCondition.IsNull);
        Condition T2Ct2 = new Condition(t2, "g1r2r3r4",
TransitionCondition.NotNull);
        T2Ct1.SetNextCondition(LogicConnector.AND, T2Ct2);

        GuardMapping grdT21 = new GuardMapping();
        grdT21.condition = T2Ct1;
        grdT21.Activations.add(new Activation(t2, "g1r2r3r4",
TransitionOperation.Move, "y1r2r3r4"));
        grdT21.Activations.add(new Activation(t2, "counter1",
TransitionOperation.Move, "counter1"));
        grdT21.Activations.add(new Activation(t2, "yellow",
TransitionOperation.SendOverNetwork, "OP1"));

        t2.GuardMappingList.add(grdT21);

        Condition T2Ct3 = new Condition(t2, "in3",
TransitionCondition.NotNull);
        Condition T2Ct4 = new Condition(t2, "counter1",
TransitionCondition.MoreThan, "zero");
        Condition T2Ct5 = new Condition(t2, "g1r2r3r4",
TransitionCondition.NotNull);
        T2Ct3.SetNextCondition(LogicConnector.AND, T2Ct4);
        T2Ct4.SetNextCondition(LogicConnector.AND, T2Ct5);

        GuardMapping grdT22 = new GuardMapping();
        grdT22.condition = T2Ct3;
        grdT22.Activations.add(new Activation(t2, "PlusTL1",
TransitionOperation.Move, "g1r2r3r4"));
        ArrayList<String> list = new ArrayList<String>();
        list.add("counter1");
        list.add("constantValue");
        grdT22.Activations.add(new Activation(t2, list,
TransitionOperation.Sub, "counter1"));

        t2.GuardMappingList.add(grdT22);

        Condition T2Ct6 = new Condition(t2, "counter1",
TransitionCondition.Equal,"zero");
        Condition T2Ct7 = new Condition(t2, "g1r2r3r4",
TransitionCondition.NotNull);
        T2Ct6.SetNextCondition(LogicConnector.AND, T2Ct7);

        GuardMapping grdT23 = new GuardMapping();
        grdT23.condition = T2Ct6;
        grdT23.Activations.add(new Activation(t2, "g1r2r3r4",
TransitionOperation.Move, "y1r2r3r4"));
        grdT23.Activations.add(new Activation(t2, "initialValue",
TransitionOperation.Move, "counter1"));
        grdT23.Activations.add(new Activation(t2, "yellow",
TransitionOperation.SendOverNetwork, "OP1"));
```

```
        t2.GuardMappingList.add(grdT23);

        t2.Delay = 5;
        pn.Transitions.add(t2);



//----------------------------t1Plus-------------------------------------
        PetriTransition t1Plus = new PetriTransition(pn);
        t1Plus.TransitionName = "t1Plus";
        t1Plus.InputPlaceName.add("PlusTL1");

        Condition T1pCt1 = new Condition(t1Plus, "PlusTL1",
TransitionCondition.NotNull);

        GuardMapping grdT1p = new GuardMapping();
        grdT1p.condition = T1pCt1;
        grdT1p.Activations.add(new Activation(t1Plus, "PlusTL1",
TransitionOperation.Move, "g1r2r3r4"));

        t1Plus.GuardMappingList.add(grdT1p);

        t1Plus.Delay = 0;
        pn.Transitions.add(t1Plus);


        //----------------------------T3-------------------------------------
        PetriTransition t3 = new PetriTransition(pn);
        t3.TransitionName = "T3";
        t3.InputPlaceName.add("y1r2r3r4");


        Condition T3Ct1 = new Condition(t3, "y1r2r3r4",
TransitionCondition.NotNull);

        GuardMapping grdT3 = new GuardMapping();
        grdT3.condition = T3Ct1;
        grdT3.Activations.add(new Activation(t3, "y1r2r3r4",
TransitionOperation.Move, "r1g2r3r4"));
        grdT3.Activations.add(new Activation(t3, "red",
TransitionOperation.SendOverNetwork, "OP1"));
        //grdT3.Activations.add(new Activation(t3, "red",
TransitionOperation.SendOverNetwork, "OP3"));
        grdT3.Activations.add(new Activation(t3, "green",
TransitionOperation.SendOverNetwork, "OP2"));
        //grdT3.Activations.add(new Activation(t3, "green",
TransitionOperation.SendOverNetwork, "OP4"));

        t3.GuardMappingList.add(grdT3);

        t3.Delay = 5;
        pn.Transitions.add(t3);
```

```
        //----------------------------T4-----------------------------------
        PetriTransition t4 = new PetriTransition(pn);
        t4.TransitionName = "T4";
        t4.InputPlaceName.add("r1g2r3r4");
        t4.InputPlaceName.add("in4");
        t4.InputPlaceName.add("counter2");


        Condition T4Ct1 = new Condition(t4, "in4",
TransitionCondition.IsNull);
        Condition T4Ct2 = new Condition(t4, "r1g2r3r4",
TransitionCondition.NotNull);
        T4Ct1.SetNextCondition(LogicConnector.AND, T4Ct2);

        GuardMapping grdT41 = new GuardMapping();
        grdT41.condition = T4Ct1;
        grdT41.Activations.add(new Activation(t4, "r1g2r3r4",
TransitionOperation.Move, "r1y2r3r4"));
        grdT41.Activations.add(new Activation(t4, "yellow",
TransitionOperation.SendOverNetwork, "OP2"));
        grdT41.Activations.add(new Activation(t4, "counter2",
TransitionOperation.Move, "counter2"));

        t4.GuardMappingList.add(grdT41);

        Condition T4Ct3 = new Condition(t4, "in4",
TransitionCondition.NotNull);
        Condition T4Ct4 = new Condition(t4, "counter2",
TransitionCondition.MoreThan, "zero");
        Condition T4Ct5 = new Condition(t4, "r1g2r3r4",
TransitionCondition.NotNull);
        T4Ct3.SetNextCondition(LogicConnector.AND, T4Ct4);
        T4Ct4.SetNextCondition(LogicConnector.AND, T4Ct5);

        GuardMapping grdT42 = new GuardMapping();
        grdT42.condition = T4Ct3;
        grdT42.Activations.add(new Activation(t4, "PlusTL2",
TransitionOperation.Move, "r1g2r3r4"));
        ArrayList<String> list2 = new ArrayList<String>();
        list2.add("counter2");
        list2.add("constantValue");
        grdT42.Activations.add(new Activation(t4, list2,
TransitionOperation.Sub, "counter2"));

        t4.GuardMappingList.add(grdT42);

        Condition T4Ct6 = new Condition(t4, "counter2",
TransitionCondition.Equal,"zero");
        Condition T4Ct7 = new Condition(t4, "r1g2r3r4",
TransitionCondition.NotNull);
        T4Ct6.SetNextCondition(LogicConnector.AND, T4Ct7);
```

```
        GuardMapping grdT43 = new GuardMapping();
        grdT43.condition = T4Ct6;
        grdT43.Activations.add(new Activation(t4, "r1g2r3r4",
TransitionOperation.Move, "r1y2r3r4"));
        grdT43.Activations.add(new Activation(t4, "initialValue",
TransitionOperation.Move, "counter2"));
        grdT43.Activations.add(new Activation(t4, "yellow",
TransitionOperation.SendOverNetwork, "OP2"));

        t4.GuardMappingList.add(grdT43);

        t4.Delay = 5;
        pn.Transitions.add(t4);



//----------------------------t2Plus-------------------------------------
        PetriTransition t2Plus = new PetriTransition(pn);
        t2Plus.TransitionName = "t2Plus";
        t2Plus.InputPlaceName.add("PlusTL2");

        Condition T2pCt1 = new Condition(t2Plus, "PlusTL2",
TransitionCondition.NotNull);

        GuardMapping grdT2p = new GuardMapping();
        grdT2p.condition = T2pCt1;
        grdT2p.Activations.add(new Activation(t2Plus, "PlusTL2",
TransitionOperation.Move, "r1g2r3r4"));

        t2Plus.GuardMappingList.add(grdT2p);

        t2Plus.Delay = 0;
        pn.Transitions.add(t2Plus);


        //----------------------------T5-------------------------------------
        PetriTransition t5 = new PetriTransition(pn);
        t5.TransitionName = "T5";
        t5.InputPlaceName.add("r1y2r3r4");


        Condition T5Ct1 = new Condition(t5, "r1y2r3r4",
TransitionCondition.NotNull);

        GuardMapping grdT5 = new GuardMapping();
        grdT5.condition = T5Ct1;
        grdT5.Activations.add(new Activation(t5, "r1y2r3r4",
TransitionOperation.Move, "r1r2g3r4"));
        grdT5.Activations.add(new Activation(t5, "red",
TransitionOperation.SendOverNetwork, "OP2"));
        grdT5.Activations.add(new Activation(t5, "green",
TransitionOperation.SendOverNetwork, "OP3"));
```

```
        t5.GuardMappingList.add(grdT5);

        t5.Delay = 5;
        pn.Transitions.add(t5);


        //----------------------------T6-----------------------------------
        PetriTransition t6 = new PetriTransition(pn);
        t6.TransitionName = "T6";
        t6.InputPlaceName.add("r1r2g3r4");
        t6.InputPlaceName.add("in5");
        t6.InputPlaceName.add("counter3");


        Condition T6Ct1 = new Condition(t4, "in5",
TransitionCondition.IsNull);
        Condition T6Ct2 = new Condition(t6, "r1r2g3r4",
TransitionCondition.NotNull);
        T6Ct1.SetNextCondition(LogicConnector.AND, T6Ct2);

        GuardMapping grdT61 = new GuardMapping();
        grdT61.condition = T6Ct1;
        grdT61.Activations.add(new Activation(t6, "r1r2g3r4",
TransitionOperation.Move, "r1r2y3r4"));
        grdT61.Activations.add(new Activation(t6, "yellow",
TransitionOperation.SendOverNetwork, "OP3"));
        grdT61.Activations.add(new Activation(t6, "counter3",
TransitionOperation.Move, "counter3"));

        t6.GuardMappingList.add(grdT61);

        Condition T6Ct3 = new Condition(t6, "in5",
TransitionCondition.NotNull);
        Condition T6Ct4 = new Condition(t6, "counter3",
TransitionCondition.MoreThan, "zero");
        Condition T6Ct5 = new Condition(t6, "r1r2g3r4",
TransitionCondition.NotNull);
        T6Ct3.SetNextCondition(LogicConnector.AND, T6Ct4);
        T6Ct4.SetNextCondition(LogicConnector.AND, T6Ct5);

        GuardMapping grdT62 = new GuardMapping();
        grdT62.condition = T6Ct3;
        grdT62.Activations.add(new Activation(t6, "PlusTL3",
TransitionOperation.Move, "r1r2g3r4"));
        ArrayList<String> list3 = new ArrayList<String>();
        list3.add("counter3");
        list3.add("constantValue");
        grdT62.Activations.add(new Activation(t6, list3,
TransitionOperation.Sub, "counter3"));

        t6.GuardMappingList.add(grdT62);
```

```
        Condition T6Ct6 = new Condition(t6, "counter3",
TransitionCondition.Equal,"zero");
        Condition T6Ct7 = new Condition(t6, "r1r2g3r4",
TransitionCondition.NotNull);
        T6Ct6.SetNextCondition(LogicConnector.AND, T6Ct7);

        GuardMapping grdT63 = new GuardMapping();
        grdT63.condition = T6Ct6;
        grdT63.Activations.add(new Activation(t6, "r1r2g3r4",
TransitionOperation.Move, "r1r2y3r4"));
        grdT63.Activations.add(new Activation(t6, "initialValue",
TransitionOperation.Move, "counter3"));
        grdT63.Activations.add(new Activation(t6, "yellow",
TransitionOperation.SendOverNetwork, "OP3"));

        t6.GuardMappingList.add(grdT63);

        t6.Delay = 5;
        pn.Transitions.add(t6);



//----------------------------t3Plus-------------------------------------
        PetriTransition t3Plus = new PetriTransition(pn);
        t3Plus.TransitionName = "t3Plus";
        t3Plus.InputPlaceName.add("PlusTL3");

        Condition T3pCt1 = new Condition(t3Plus, "PlusTL3",
TransitionCondition.NotNull);

        GuardMapping grdT3p = new GuardMapping();
        grdT3p.condition = T3pCt1;
        grdT3p.Activations.add(new Activation(t3Plus, "PlusTL3",
TransitionOperation.Move, "r1r2g3r4"));

        t3Plus.GuardMappingList.add(grdT3p);

        t3Plus.Delay = 0;
        pn.Transitions.add(t3Plus);


        //----------------------------T7-------------------------------------
        PetriTransition t7 = new PetriTransition(pn);
        t7.TransitionName = "T7";
        t7.InputPlaceName.add("r1r2y3r4");


        Condition T7Ct1 = new Condition(t7, "r1r2y3r4",
TransitionCondition.NotNull);

        GuardMapping grdT7 = new GuardMapping();
        grdT7.condition = T7Ct1;
```

```
        grdT7.Activations.add(new Activation(t7, "r1r2y3r4",
TransitionOperation.Move, "r1r2r3g4"));
        grdT7.Activations.add(new Activation(t7, "red",
TransitionOperation.SendOverNetwork, "OP3"));
        grdT7.Activations.add(new Activation(t7, "green",
TransitionOperation.SendOverNetwork, "OP4"));


        t7.GuardMappingList.add(grdT7);

        t7.Delay = 5;
        pn.Transitions.add(t7);


        //----------------------------T8-----------------------------------
        PetriTransition t8 = new PetriTransition(pn);
        t8.TransitionName = "T8";
        t8.InputPlaceName.add("r1r2r3g4");
        t8.InputPlaceName.add("in6");
        t8.InputPlaceName.add("counter4");

        Condition T8Ct1 = new Condition(t4, "in6",
TransitionCondition.IsNull);
        Condition T8Ct2 = new Condition(t8, "r1r2r3g4",
TransitionCondition.NotNull);
        T8Ct1.SetNextCondition(LogicConnector.AND, T8Ct2);

        GuardMapping grdT81 = new GuardMapping();
        grdT81.condition = T8Ct1;
        grdT81.Activations.add(new Activation(t8, "r1r2r3g4",
TransitionOperation.Move, "r1r2r3y4"));
        grdT81.Activations.add(new Activation(t8, "yellow",
TransitionOperation.SendOverNetwork, "OP4"));
        grdT81.Activations.add(new Activation(t8, "counter4",
TransitionOperation.Move, "counter4"));

        t8.GuardMappingList.add(grdT81);

        Condition T8Ct3 = new Condition(t8, "in6",
TransitionCondition.NotNull);
        Condition T8Ct4 = new Condition(t8, "counter4",
TransitionCondition.MoreThan, "zero");
        Condition T8Ct5 = new Condition(t8, "r1r2r3g4",
TransitionCondition.NotNull);
        T8Ct3.SetNextCondition(LogicConnector.AND, T8Ct4);
        T8Ct4.SetNextCondition(LogicConnector.AND, T8Ct5);

        GuardMapping grdT82 = new GuardMapping();
        grdT82.condition = T8Ct3;
        grdT82.Activations.add(new Activation(t8, "PlusTL4",
TransitionOperation.Move, "r1r2r3g4"));
        ArrayList<String> list4 = new ArrayList<String>();
        list4.add("counter4");
```

```
        list4.add("constantValue");
        grdT82.Activations.add(new Activation(t8, list4,
TransitionOperation.Sub, "counter4"));

        t8.GuardMappingList.add(grdT82);

        Condition T8Ct6 = new Condition(t8, "counter4",
TransitionCondition.Equal,"zero");
        Condition T8Ct7 = new Condition(t8, "r1r2r3g4",
TransitionCondition.NotNull);
        T8Ct6.SetNextCondition(LogicConnector.AND, T8Ct7);

        GuardMapping grdT83 = new GuardMapping();
        grdT83.condition = T8Ct6;
        grdT83.Activations.add(new Activation(t8, "r1r2r3g4",
TransitionOperation.Move, "r1r2r3y4"));
        grdT83.Activations.add(new Activation(t8, "initialValue",
TransitionOperation.Move, "counter4"));
        grdT83.Activations.add(new Activation(t8, "yellow",
TransitionOperation.SendOverNetwork, "OP4"));

        t8.GuardMappingList.add(grdT83);

        t8.Delay = 5;
        pn.Transitions.add(t8);



//----------------------------t4Plus-------------------------------------
        PetriTransition t4Plus = new PetriTransition(pn);
        t4Plus.TransitionName = "t4Plus";
        t4Plus.InputPlaceName.add("PlusTL4");

        Condition T4pCt1 = new Condition(t4Plus, "PlusTL4",
TransitionCondition.NotNull);

        GuardMapping grdT4p = new GuardMapping();
        grdT4p.condition = T4pCt1;
        grdT4p.Activations.add(new Activation(t4Plus, "PlusTL4",
TransitionOperation.Move, "r1r2r3g4"));

        t4Plus.GuardMappingList.add(grdT4p);

        t4Plus.Delay = 0;
        pn.Transitions.add(t4Plus);


        //----------------------------T9-------------------------------------
        PetriTransition t9 = new PetriTransition(pn);
        t9.TransitionName = "T9";
        t9.InputPlaceName.add("r1r2r3y4");
```

```
        Condition T9Ct1 = new Condition(t9, "r1r2r3y4",
TransitionCondition.NotNull);

        GuardMapping grdT9 = new GuardMapping();
        grdT9.condition = T9Ct1;
        grdT9.Activations.add(new Activation(t9, "r1r2r3y4",
TransitionOperation.Move, "r1r2r3r4"));
        grdT9.Activations.add(new Activation(t9, "red",
TransitionOperation.SendOverNetwork, "OP4"));


        t9.GuardMappingList.add(grdT9);

        t9.Delay = 5;
        pn.Transitions.add(t9);


        //
----------------------------------------------------------------------------
----------
        // ----------------------------PN
Start-------------------------------------------------
        //
----------------------------------------------------------------------------
----------

        System.out.println("Exp1 started \n
------------------------------");
        pn.Delay = 2000;
        // pn.Start();

        PetriNetWindow frame = new PetriNetWindow(false);
        frame.petriNet = pn;
        frame.setVisible(true);
    }
}
```