For ORC you are going to need the following FLRS:

for T0 (Reader)

**static** String *reader* = "" +

"{[<ZR,NL><ZR,NM><ZR,ZR><ZR,PM><ZR,PL>]" +

" [<ZR,NL><ZR,NM><ZR,ZR><ZR,PM><ZR,PL>]" +

" [<ZR,NL><ZR,NM><ZR,ZR><ZR,PM><ZR,PL>]" +

" [<ZR,NL><ZR,NM><ZR,ZR><ZR,PM><ZR,PL>]" +

" [<ZR,NL><ZR,NM><ZR,ZR><ZR,PM><ZR,PL>]}";

for T2out

**static** String *t2Table* = "{[<PL><PM><ZR><NM><NL>]}";

/\* So when the temperature is NL> the command should be PL

NM> pm

ZR stays the same

PM > it should be cooler so NM

PL> NL \*/

These are the fuzzy drivers:

**private** FuzzyPetriNet net;

**private** **int** p1OustideTemInp;

**private** **int** t2Out;

**private** FuzzyDriver outsideTempDriver;

**private** FuzzyDriver tankWaterTemeDriver;

**private** FullRecorder rec;

**private** AsyncronRunnableExecutor execcutor;

The constructor:

**public** OutsideReferenceCalculatorComponent(Plant plant, HeaterTankControllerComponent comp, **long** simPeriod) {

// Build the petri net here // your homework

outsideTempDriver = FuzzyDriver.*createDriverFromMinMax*(-30, 10);

tankWaterTemeDriver = FuzzyDriver.*createDriverFromMinMax*(45, 68);

net.addActionForOuputTransition(t2Out, tk -> { //<<<<<<<<<< this is the t2 action

comp.setWaterRefTemp(tankWaterTemeDriver.defuzzify(tk));//<<< that is connected to

//<<< The water tank

});

rec = **new** FullRecorder();

execcutor = **new** AsyncronRunnableExecutor(net, simPeriod);

execcutor.setRecorder(rec);

These are the start, stop, get net , and get recorder methods

**public** **void** start() {

(**new** Thread(execcutor)).start(); }

**public** **void** stop() { execcutor.stop(); }

**public** **void** setOutsideTemp(**double** waterRefTemp) {

Map<Integer, FuzzyToken> inps = **new** HashMap<Integer, FuzzyToken>();

inps.put(p1OustideTemInp, outsideTempDriver.fuzzifie(waterRefTemp));

execcutor.putTokenInInputPlace(inps); }

**public** FuzzyPetriNet getNet() { **return** net; }

**public** FullRecorder getRecorder() { **return** rec; }

The Simple main should be changed so you can see all the controllers:

**import** Main.FuzzyPVizualzer;

**import** Main.Plotter;

**import** View.MainView;

**public** **class** OutsideTermoMain {

**public** **static** **final** **long** ***SIM\_PERIOD*** = 10;

**public** **static** **void** main(String[] args) {

Scenario sceonario = Scenario.*winterDay*();

Plant plant = **new** Plant(***SIM\_PERIOD***, sceonario);

HeaterTankControllerComponent tankController = **new** HeaterTankControllerComponent(plant, ***SIM\_PERIOD***);

RoomTemperatureControllerComponent termostat = **new** RoomTemperatureControllerComponent(plant, ***SIM\_PERIOD***);

OutsideReferenceCalculatorComponent outTermo = **new** OutsideReferenceCalculatorComponent(plant, tankController, ***SIM\_PERIOD***);

termostat.start();

tankController.start();

plant.start();

outTermo.start();

// double waterRefTemp = 62.0;

**double** roomTemperature = 24.0;

**for** (**int** i = 0; i < sceonario.getScenarioLength(); i++) {

outTermo.setOutsideTemp(sceonario.getOutSideTemepratue(i));

tankController.setTankWaterTemp(plant.getTankWaterTemperature());

termostat.setInput(roomTemperature, plant.getRoomTemperature());

**try** { Thread.*sleep*(10);

} **catch** (InterruptedException e) {

// **TODO** Auto-generated catch block

e.printStackTrace(); }

}

tankController.stop();

termostat.stop();

outTermo.stop();

MainView windowTankController = FuzzyPVizualzer.*visualize*(tankController.getNet(),

tankController.getRecorder());

MainView windowTermostat = FuzzyPVizualzer.*visualize*(termostat.getNet(), termostat.getRecorder());

MainView windowOutTermo = FuzzyPVizualzer.*visualize*(outTermo.getNet(), outTermo.getRecorder());

Plotter plotterTemperatureLog = **new** Plotter(plant.getTemeartureLogs());

Plotter plotterCommandLog = **new** Plotter(plant.getCommandLogs());

windowTankController.addInteractivePanel("TempLogs", plotterTemperatureLog.makeInteractivePlot());

windowTermostat.addInteractivePanel("TempLogs", plotterTemperatureLog.makeInteractivePlot());

windowTankController.addInteractivePanel("ComandLogs", plotterCommandLog.makeInteractivePlot());

windowTermostat.addInteractivePanel("ComandLogs", plotterCommandLog.makeInteractivePlot());

windowOutTermo.addInteractivePanel("ComandLogs", plotterCommandLog.makeInteractivePlot());

windowOutTermo.addInteractivePanel("ComandLogs", plotterCommandLog.makeInteractivePlot());

**double**[] tankTempStats = SimpelMain.*calcStatistics*(plant.getTemeartureLogs().get("tankTemp"));

**double**[] rommTempStsats =SimpelMain.*calcStatistics*(plant.getTemeartureLogs().get("roomTemp"));

System.***out***.println("max tank temp :" + tankTempStats[0]);

System.***out***.println("min tank temp :" + tankTempStats[1]);

System.***out***.println("avg tank temp :" + tankTempStats[2]);

System.***out***.println("max room temp :" + rommTempStsats[0]);

System.***out***.println("min room temp :" + rommTempStsats[1]);

System.***out***.println("avg room temp :" + rommTempStsats[2]);

System.***out***.println("heater on ratio:" + plant.heatingOnRatio());

System.***out***.println("max continous heating on:" + plant.maxContiniousHeaterOn());

System.***out***.println("all consunption ::" + plant.gasConsumption());

System.***out***.println("avg consunption in min ::" + plant.gasConsumption() / sceonario.getScenarioLength()); }

}