Modelling Cyber Physical Systems

Obligatory Excercise 1

Table of Contents

[Specification Sequence Digrams 2](#_Toc508312938)

[Architectural Composite Diagram 7](#_Toc508312939)

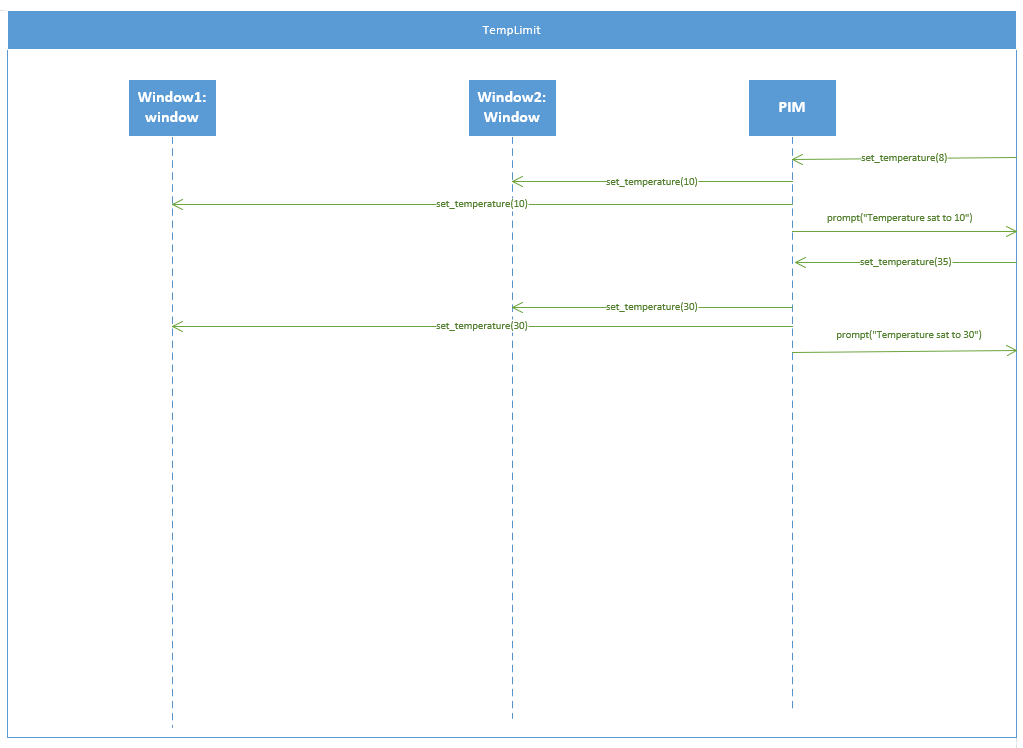
[State Machine Diagrams - PlantUML 8](#_Toc508312940)

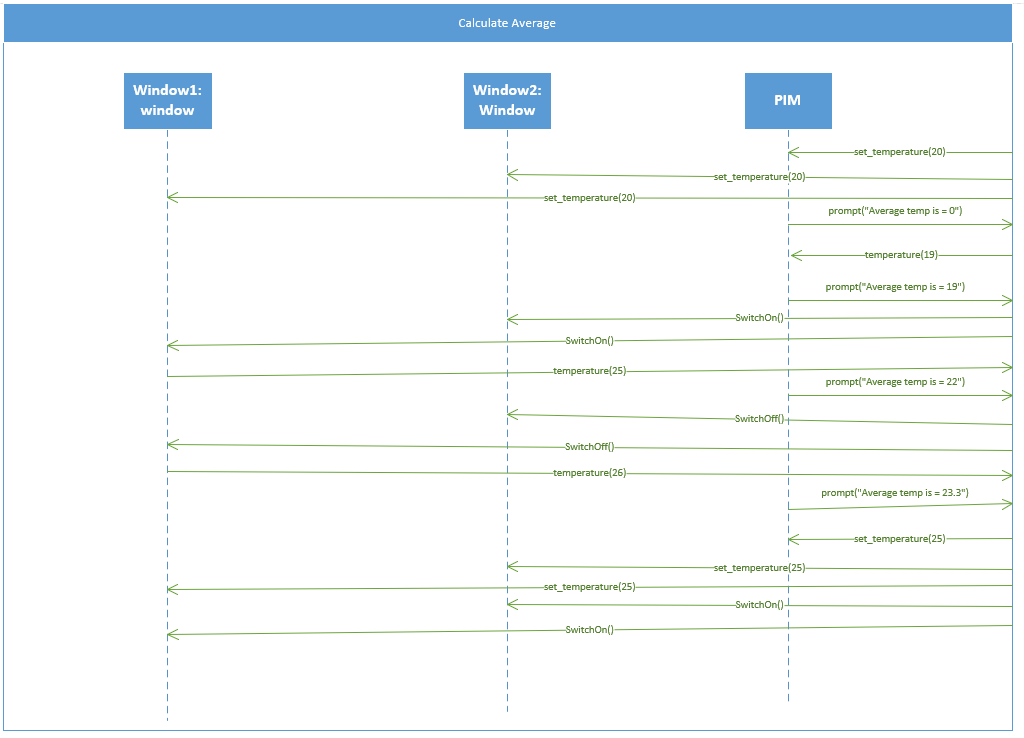
[Assessment of Our Solution 10](#_Toc508312941)

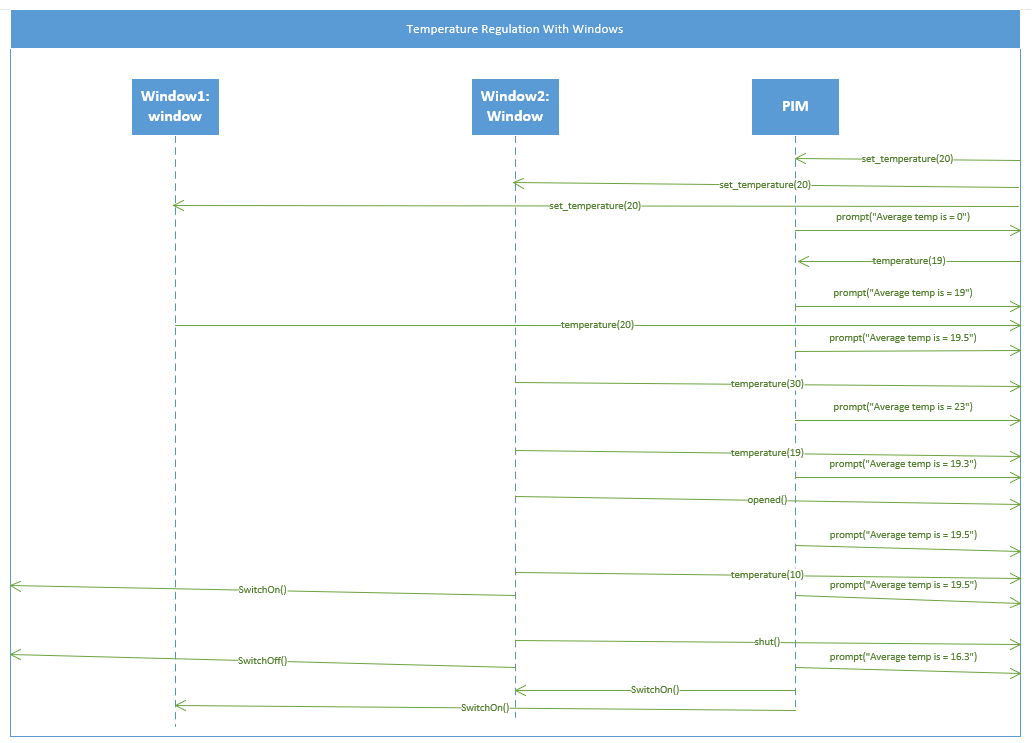
[Execution Instructions 11](#_Toc508312942)

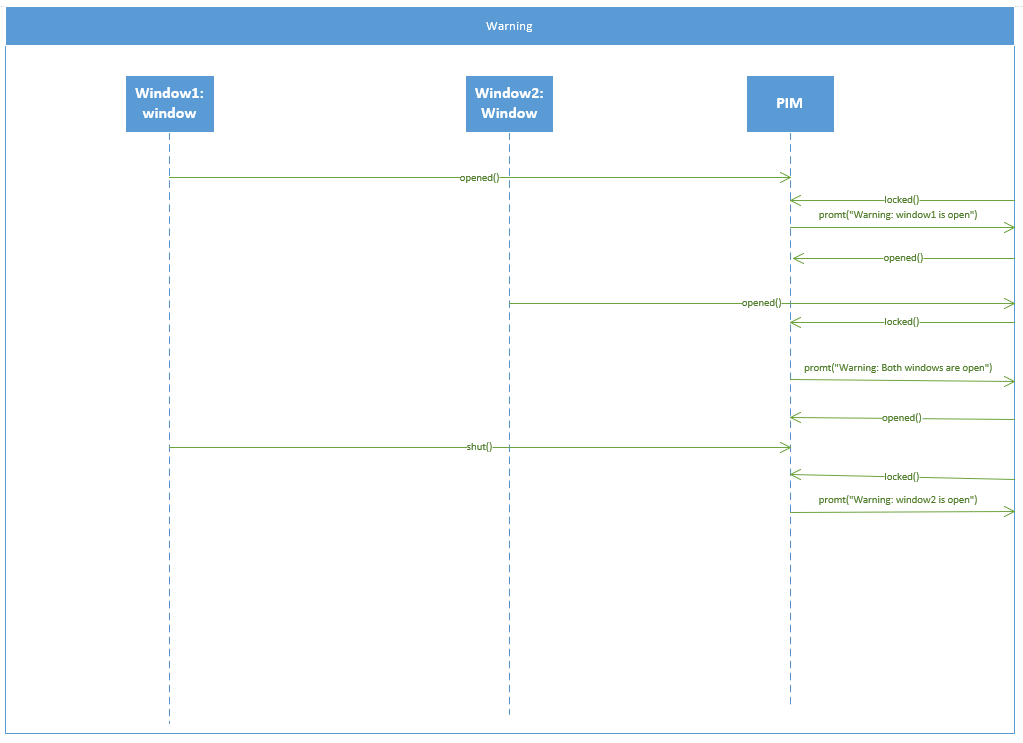
[Improvements 15](#_Toc508312943)

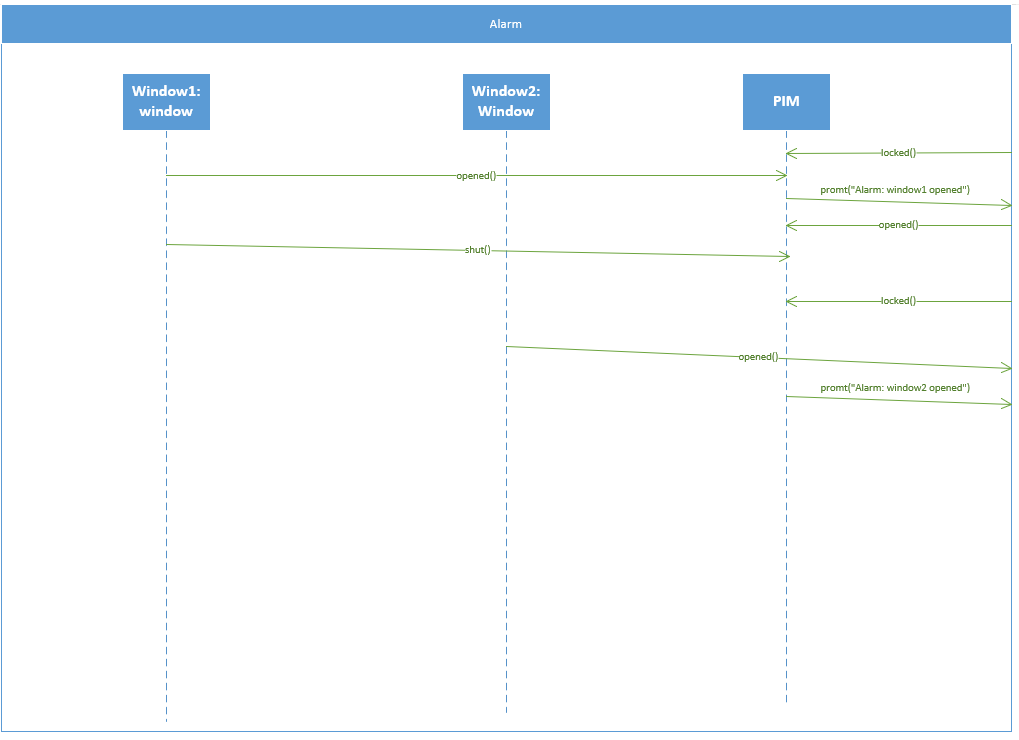
# Specification Sequence Digrams



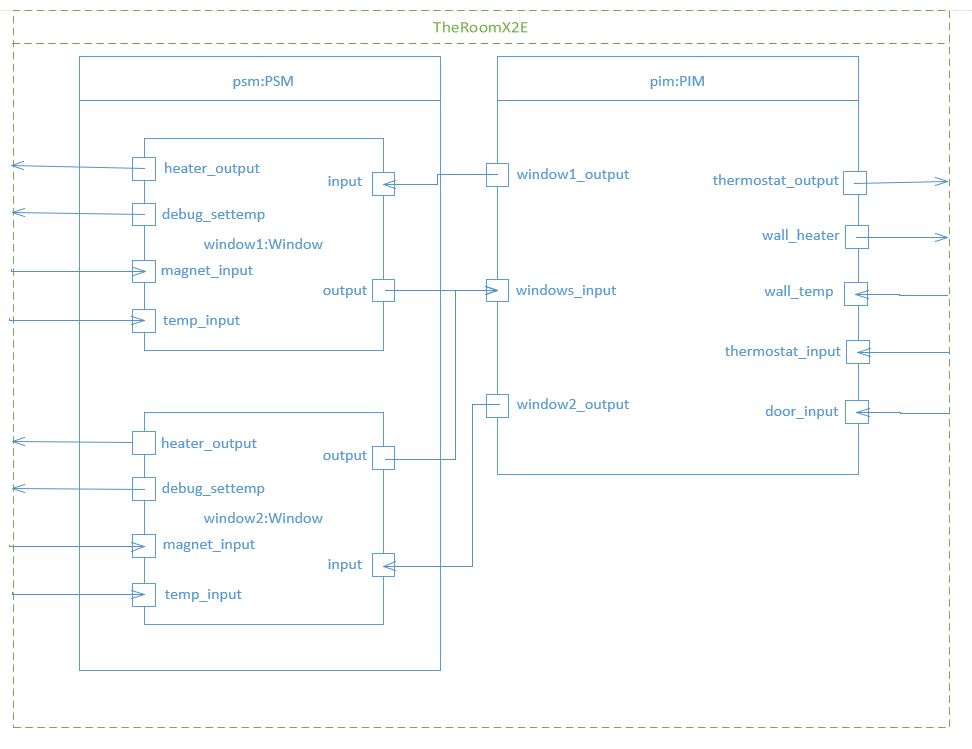




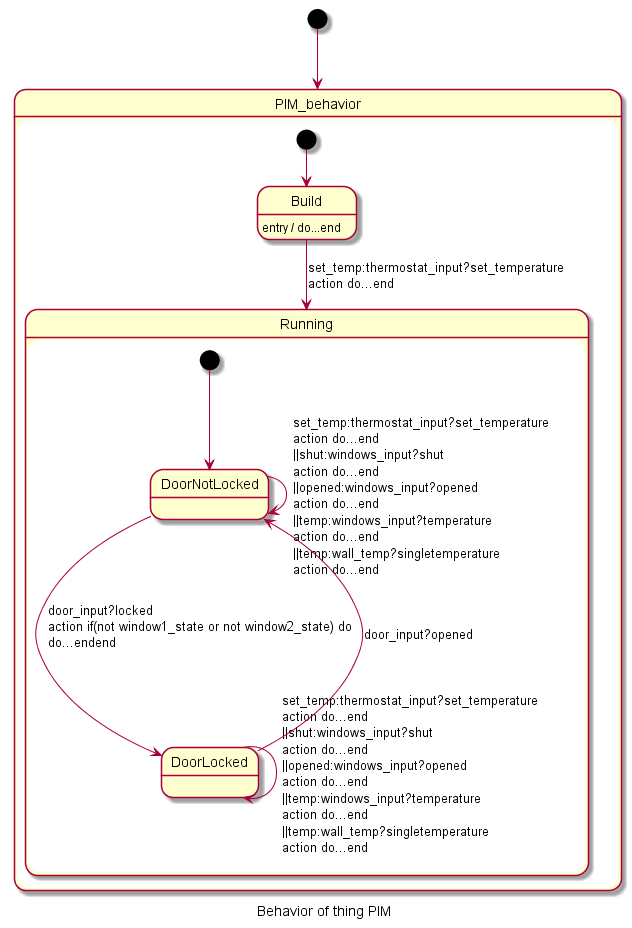


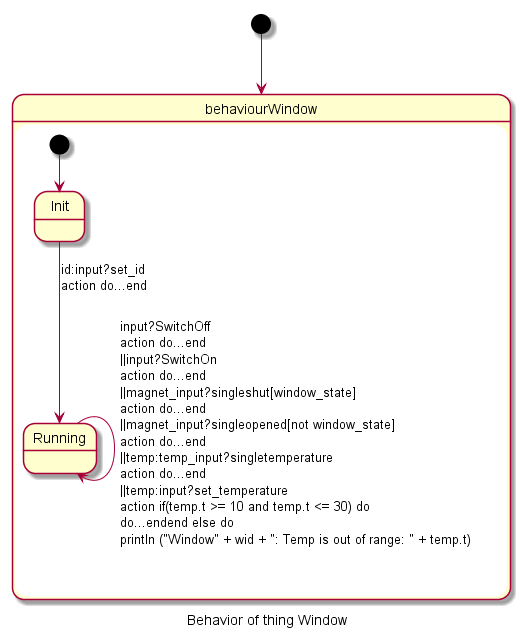


# Architectural Composite Diagram



# State Machine Diagrams - PlantUML





# Assessment of Our Solution

We decided to go for a solution that relies on keeping track of states of the door, windows and the heaters in variables, as well as what temperatures has been received from the sensors. We could possibly have chosen a solution that exploits the nature of state machines more heavily but opted for this approach as it seemed natural based on our previous programming experiences.

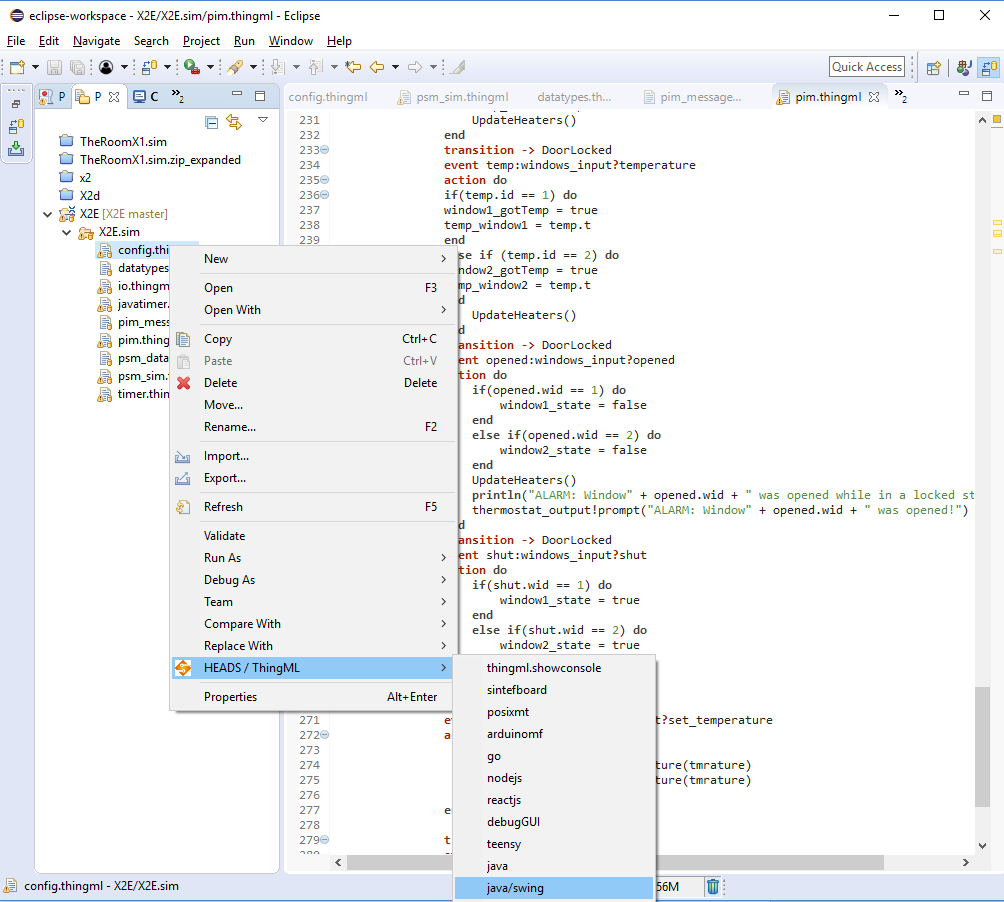
We changed the system dramatically from the way it worked in the example “X2D”, assigning the heaters and thermometers on startup and also removing the ability to add more at runtime. We could also have used the “Sets” for keeping track of the different thermometers and heaters, letting our window thing and the “PIM” keep track of what sensor or heater belonged where using the id’s.

We combined a lot of messages into the mock things so that the testing would be more manageable, we went from an initial 12 individual mock windows to a single one.

Where our solution falls short of what we would like is handling of exceptions. Making the solution robust against unexpected messages or poor ordering of received messages.

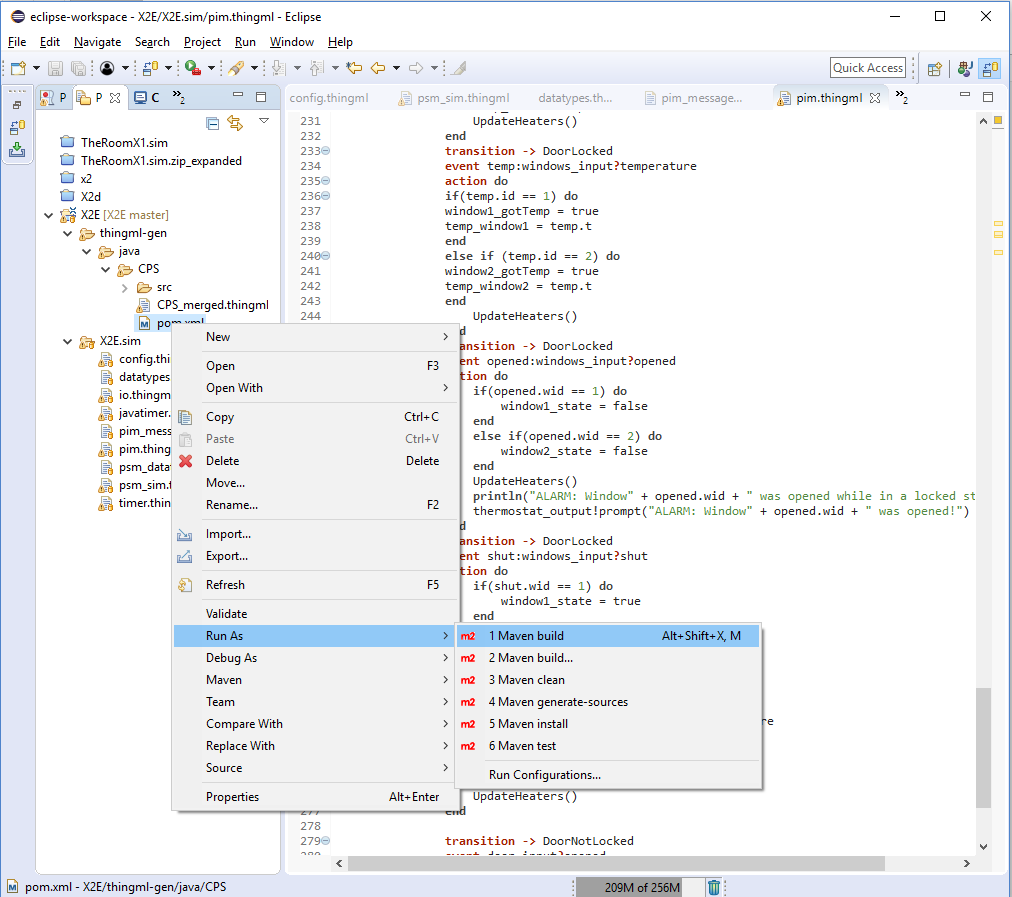
# Execution Instructions

To execute the ThingML model, first build it using java/swing by right clicking the config.thingml file and selecting HEADS/ThingML -> java/swing

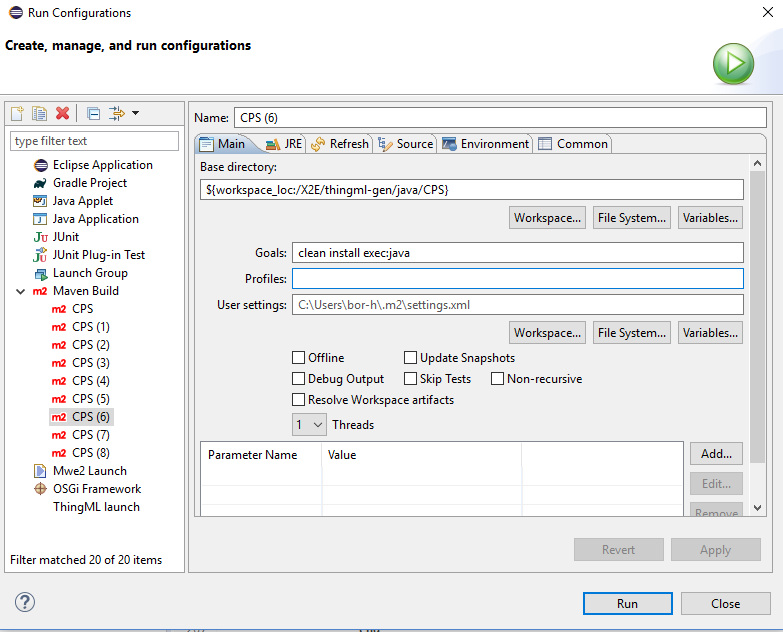


This will compile the model into a runnable java program with swing ui.

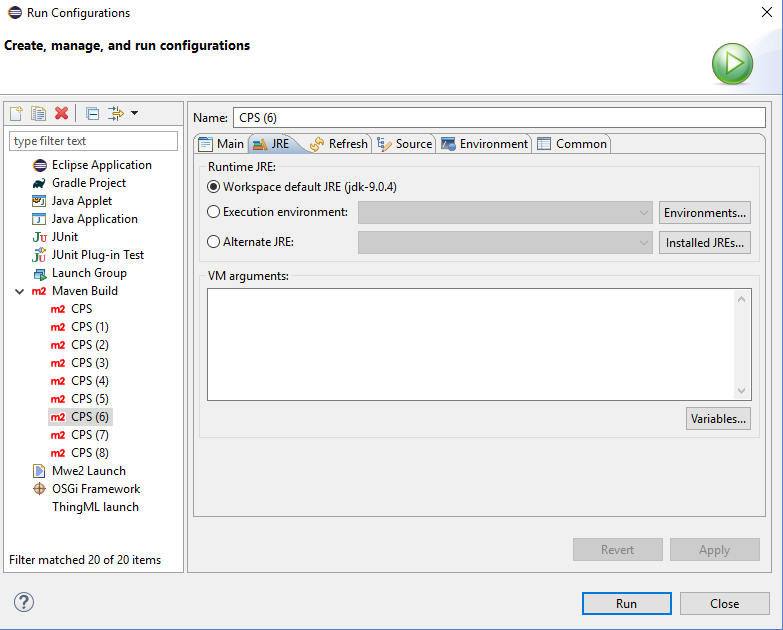
To start the solution, right click the pom.xml file in the generated folder thingml-gen/java/CPS/



This should open a configuration window.

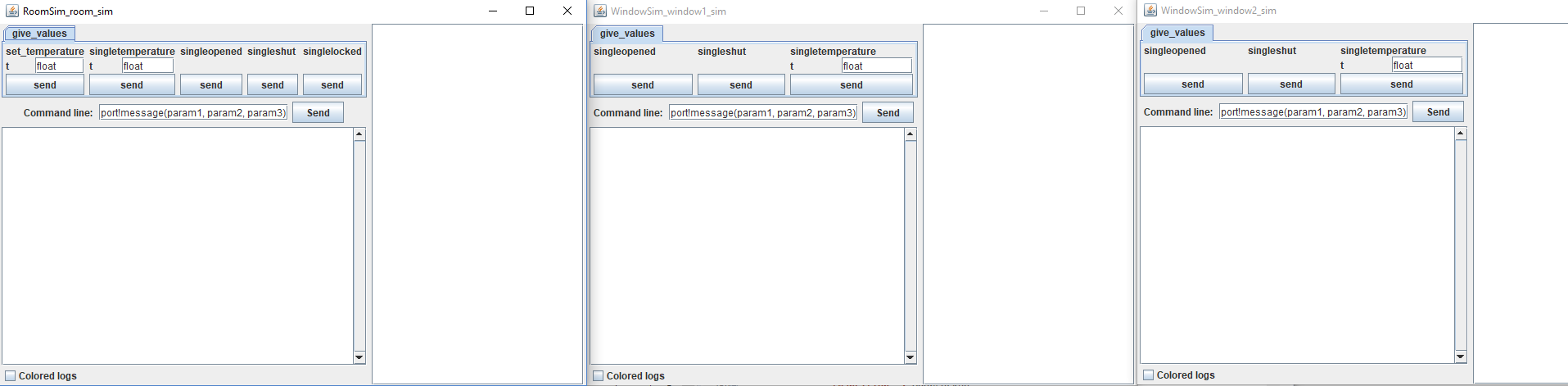
Chose the folder X2E/thingml-gen/java-CPS as the base directory, then enter “clean install exec:java” as the goals. (As shown in the figure)

Make sure jdk (not jre) is chosen as the runtime jre. We ran the model using jdk-9.0.4.



After correctly configuring the model, click the Run button, the model should compile and run.

The program will open three windows on launch, (room\_sim, window1\_sim and window2\_sim).



Room\_sim represents the thermostat, the door sensor and the independent thermostat and heater.

The window\_sim windows represents each window (window sensor, thermometer and heater).

# Improvements

Should we be able to change comfort temperature when the door is locked (People could still be inside when the door is locked). Animals can disrupt the different parts, like the heater and thermostat (notify on changes during locked door). Notify on drastic temperature increases (fire).