Multi Agent Systems

Smart Home System

Arshavee Das

CPS2 (M1)

INDEX

1. Project Overview

1.1 Context:

This project is based on Smart Home System. In this system each and every appliance in home should be connected with each other through internet. And each and every member of the family can use them according to them necessarily from anywhere. Firstly, we need to ensure that the person who wants to use a gadget (Home Appliance) is a member of the family to ensure that unauthorized person is not getting access to this system. For this user need to login their account by their credentials.

If a person wants to use an appliance and it is currently not used by other client then it will be assigned by this person and the person will get access to this appliance on first come first serve basic.

But here home appliances will work as the agents and they will be able to take some decision on its own by the help of some sensors, for example if the temperature outside is below to the threshold temperature set, the windows will shut and room heater gets turned on. If the luminance of the room gets low the lights will on etc.

If we depend on a centralized system there could be some issues

a) For better computation we need to use a powerful central machine.
b) Cost will be higher.
c) If more signals come at the same time there is a chance of error for the same times
d) This system would use more electric power.

To overcome that we have proposed a decentralized solution.

1.2 Multiagent Solution:

In Smart Home System, where all the home appliances are connected with internet and each and every member of the home is able to use any home appliances from any places in the world by the help of an application installed in their smart phone / PC. If the client is authentic and requests for a gadget which is free, then the client will get the access of it directly. Otherwise, access will be given on first come first serve basic.

Other than that, the smart appliances will be able to make changes on its own. There will be sensors which will check if the smart device is in its threshold value, if it is not then the state of the appliances will be changed.

After changing the state of the agent, it will inform other devices of the same group about its new state. After getting an update, they will decide if a changing of state is required for them or not.

Agents: -

- The home appliances-
  They are connected with each other also connected with internet.
- The members of the family are the Object (obedient agent).

Environment: -

- Home and the surrounding environments of it. The environmental external stimuli of the house.
  - The environmental stimuli here are surrounding temperature, luminance, humidity, general habits of the member of the family (To watch a Television at a particular time of the day, to use heater at a particular time, etc.) etc.

Interaction: -

- Interaction is the connection between the appliances.
  - When an agent changes it state, it notifies other agents by sending message to other agents about that change so that other agents can decide if a change of state is required for them also or not.
- Giving access to the client to desired Home Appliance.

Organization: - Agents can exchange information with other agents and able to take decision of its state change.

1.3 Report Modification: -

29/09/2021- Report
6/10/2021- Report [modification] + Reactive model implementation
20/10/2021 – Report [modification] + Reactive Model [modification]

2. <u>Reactive Model:-</u>

Interaction: -

- Client wants to use a HOME APPLIANCE.
- If HOME APPLIANCE is currently not in use, user can get the access.
- HOME APPLIANCE will work on its own as the requirements.

Goal: - To use HOME APPLIANCE smartly and securely and to save energy.

- Smartly: -Home appliances can change of their own their states as required.
- Securely: - Though it is connected with internet it is not possible to get access of it without authentic client.
- Save energy: - Home application will work more efficient way, for example if heater will be turned on then window will be closed. And unused appliances will be turned off.

Motivation: -

- Quality of Service – The service will be better; the Home Appliance will change their state by themselves when required. And human interaction will not be necessary for that.
- Conservation of energy – A un-used home appliance will be off and a Home Appliance will work only it is required. So it will help in conservation of energy.

Cooperate: - HOME APPLIANCE should work efficiently and only when it is required.

Coordinate: - - HOME APPLIANCES will change their state and share this Information with others, so that others can able to check if any change of state is required or not for them.

- Client will choose HOME APPLIANCE to use
- State of one HOME APPLIANCE will be sheared with other HOME APPLIANCES of the same group.

2.1- Reactive Agents (a): - Reactive agent can be defined by following model.

a: =<Pr,Ar,Dr>

Perceptions (Pr): {change of state of a HOME APPLIANCE, Threshold values of physical entities (for example temperature for smart windows and heater, luminance for light), State change request}

Action (Ar): {Check the value of physical entity with threshold value, Change the state of a HOME APPLIANCE, giving access}

Decision (Dr): {If the Physical entity values is under threshold value, then do not change the state}

2.2 Environment: -

Environment E: =<O, A, R>

Observable information (O):

1. Threshold values
2. State of HOME APPLIANCE
3. Physical Entities (Weather, Temperature, Luminance)
4. Client request

Accessible information (A):

1. Current physical entity value
2. Request for a HOME APPLIANCE

Regulation Rules (R)

1. One HOME APPLIANCE can be used by one client at the same time
2. If a client does not respond for a particular time, then the access will be withdrawn
3. If a modification is occurred in a HOME APPLIANCE. Then a modification request will come to all the HOME APPLIANCE of same group.

2.3 Interaction: Each HOME APPLIANCE will interact with other HOME APPLIANCE in same group. Mainly there will be intra interaction between HOME APPLIANCES.

2.4 Organization: Every application can able to exchange their state with the help of client or own selves. They share their new state with other HOME APPLIANCE in same group.

3. Cognitive: -

3.1 Agent:

Agent a:=<P,K,Ka,A,D>

D = {Changing the state of a Home Appliance if required,

According to the recent modification of a same grouped Home Appliance, if other modification is required to others,

If a client wants to use a HOME APPLIANCE which is not in use, he will be given the access,

If a HOME APPLIANCE is not in use, then it will be switched off to save power}

A ={Ar, give access to the HOME APPLIANCE to the client, Communication with other HOME APPLIANCE}

Ka – {Client's request, HOME APPLIANCE availability, Authentication of the client in their applications, message broadcasting protocol, Threshold value}

K –{external factors, changing its state according to the external factor }

P – {Pr, External factors, network definition, information for other appliances in same group}

3.2 Environment: -

Environment E:=<O,A,R>

Observable information (O):

5. Threshold values
6. State of HOME APPLIANCE
7. External factors (weather)
8. Client request

Accessible information (A):

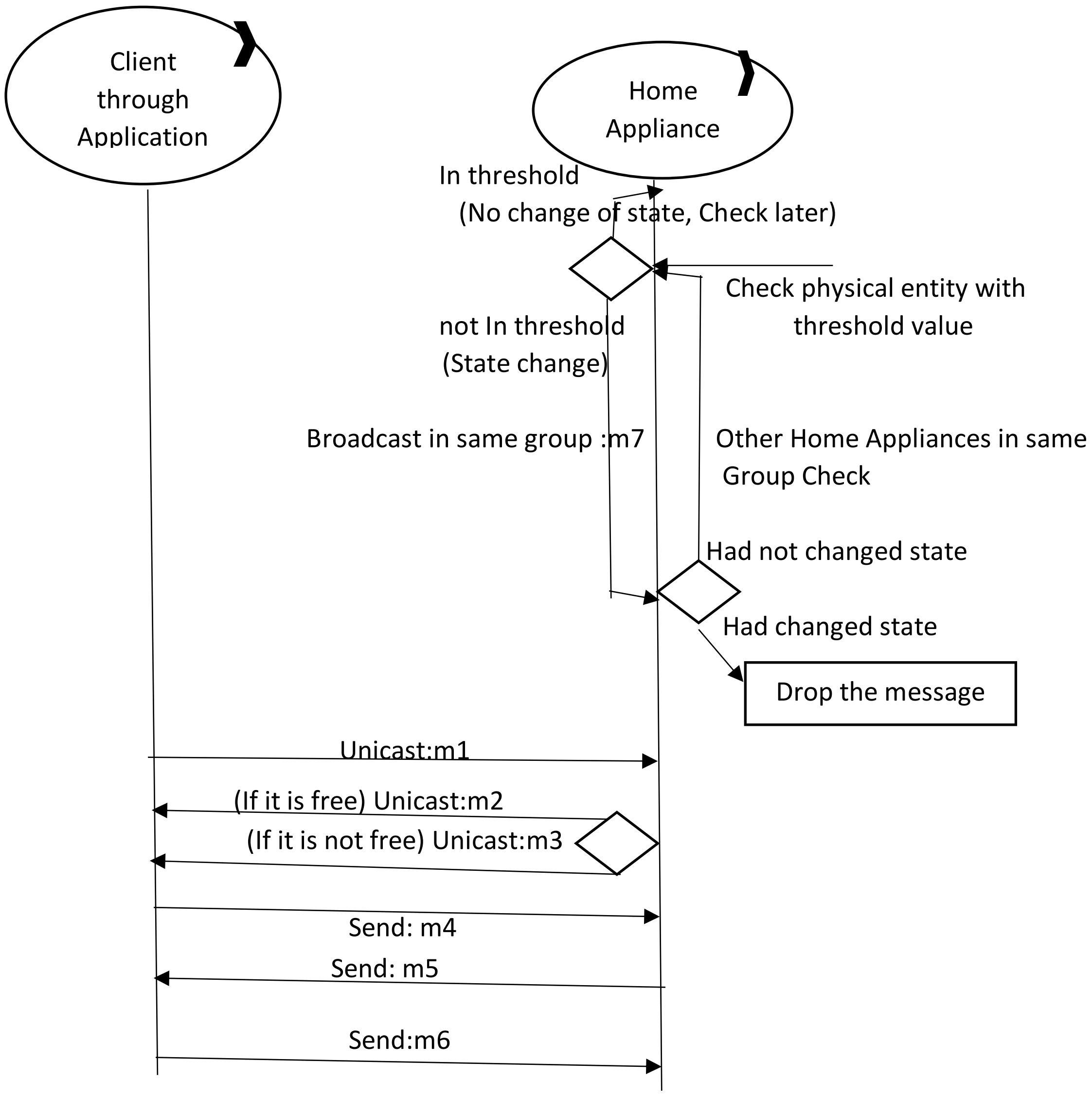
3. Current physical entity value
4. Request for a HOME APPLIANCE

Regulation Rules (R)

4. One HOME APPLIANCE can be used by one client at the same time
5. If a client does not respond for a particular time, then the access will be withdrawn

6. If a modification is occurred in a HOME APPLIANCE. Then a modification request will come to all the HOME APPLIANCE of same group.

3.3 Interaction:


Client through Application
Home Appliance
In threshold
(No change of state, Check later)
Check physical entity with threshold value
not In threshold
(State change)
Broadcast in same group :m7
Other Home Appliances in same Group Check
Had not changed state
Had changed state
Drop the message
Unicast:m1
(If it is free) Unicast:m2
(If it is not free) Unicast:m3
Send: m4
Send: m5
Send:m6

M1:: < client, Home appliance ,Request for use, time>
M2::=< Home appliance, client, requested client, Approval, time>
M3::=< Home appliance, client, requested client, NOT Approval, time>
M4::=< Client, Home appliance, Connection request, null, null>
M5::=< Home Appliance, client, Connection request, accept, null>
M6::=< Home Appliance, client, Connection request, accept, accept>
M7::=< Home Appliance, All Home Appliance of same group , Changing of state>

If M7 comes to a Home Appliance then it will calculate if it is in threshold or not. If it is in threshold then there will not be any change otherwise the state of that

Home Appliance will be changed. And this changing report will be shared with all other Home Appliances of the same group. Then they will check if any changing of state is required or not. If a Home Appliance had already changed the state, then it will drop the message.

3.4 Organization: Every application can able to exchange their state with the help of client or own selves. They share their new state with other HOME APPLIANCE in same group.

4. Implementation: -

4.1. Reactive Solution Implementation: -

4.1.1. Description:

Reactive Implementation is done in NetLogo. In reactive implementation There are two different groups of Home Appliances, Home Appliances in same groups are connected with each-other (fig:1).

Here Yellow Squares are in a same group of Home Appliances and Magenta Triangles are in another group (fig:1).

There are some blue human, represents the family members of the family, who can use a Home Appliance after authorization. Number of men can be changed by the slider. It has a range of 0 to 10 (fig:1).

There is another turtle in leaf shape this represents the External Factors (fig:1).

There are Two buttons, Setup button is used to setup the environment.

And Start button is used to start the implementation.

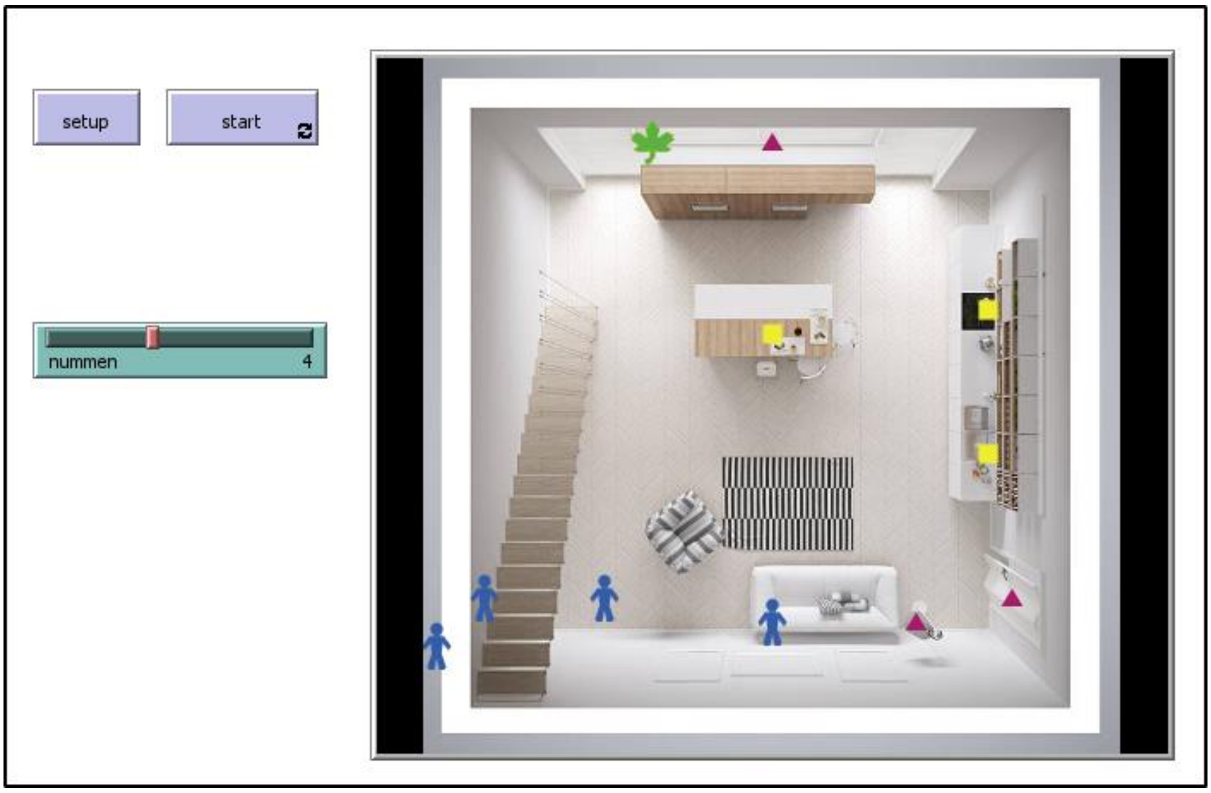


Fig:1

4.1.2. Implementation: Set the speed in the NetLogo in very less otherwise it will be very hard to understand.

- Home Appliance will Change their state if an external factor comes and it will inform other Home Appliances in same group (Fig :2).

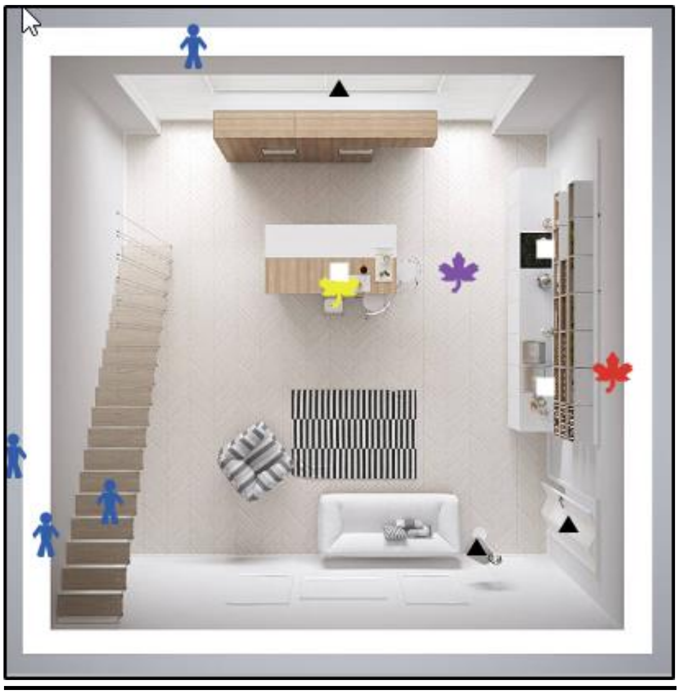


Fig:2

- A Client can use home Appliances if these are free or currently not used by another client. (fig:3)

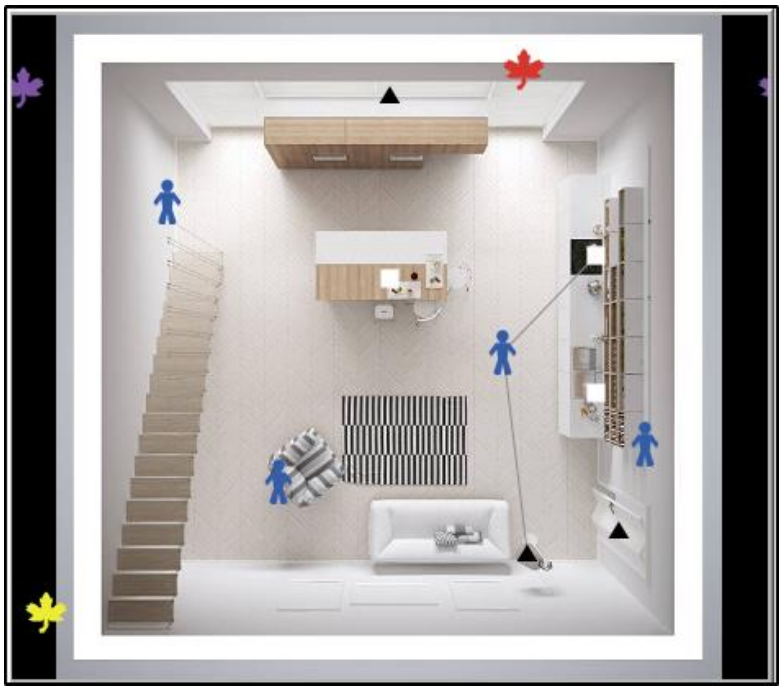
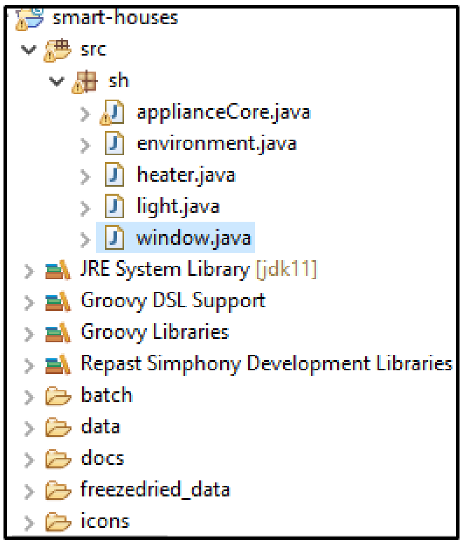


Fig:3

- A Home Appliance will change its state by the external factor or by the member of the family.

4.2. Cognitive Solution Implementation:

Cognitive model representation should be done in Repast Symphony. Here under source (SRC) there are 5 .java files (fig 2.1).

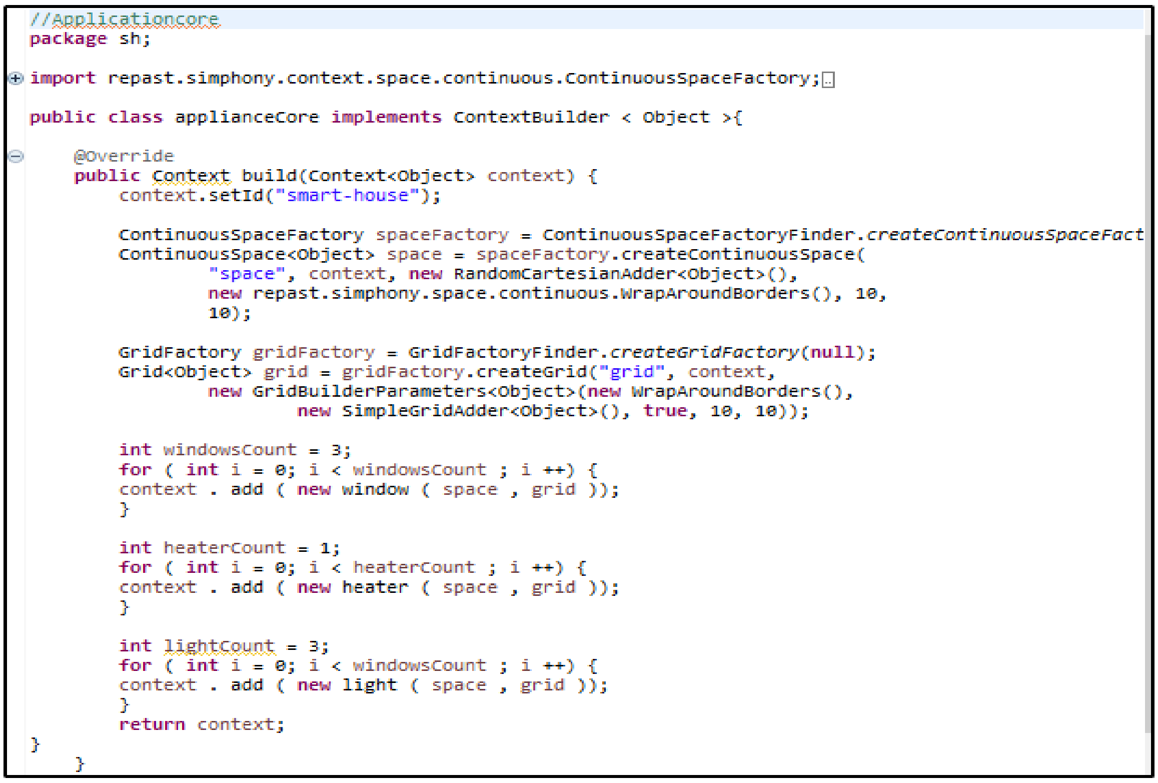


```
smart-houses
  src
    sh
      applianceCore.java
      environment.java
      heater.java
      light.java
      window.java
  JRE System Library [jdk11]
  Groovy DSL Support
  Groovy Libraries
  Repast Simphony Development Libraries
  batch
  data
  docs
  freezedried_data
  icons
```

Fig 2.1

APPLICATIONCORE- This file is responsible for making the base of the output environment. The size of the grid and the places of the agents should be mentioned here.

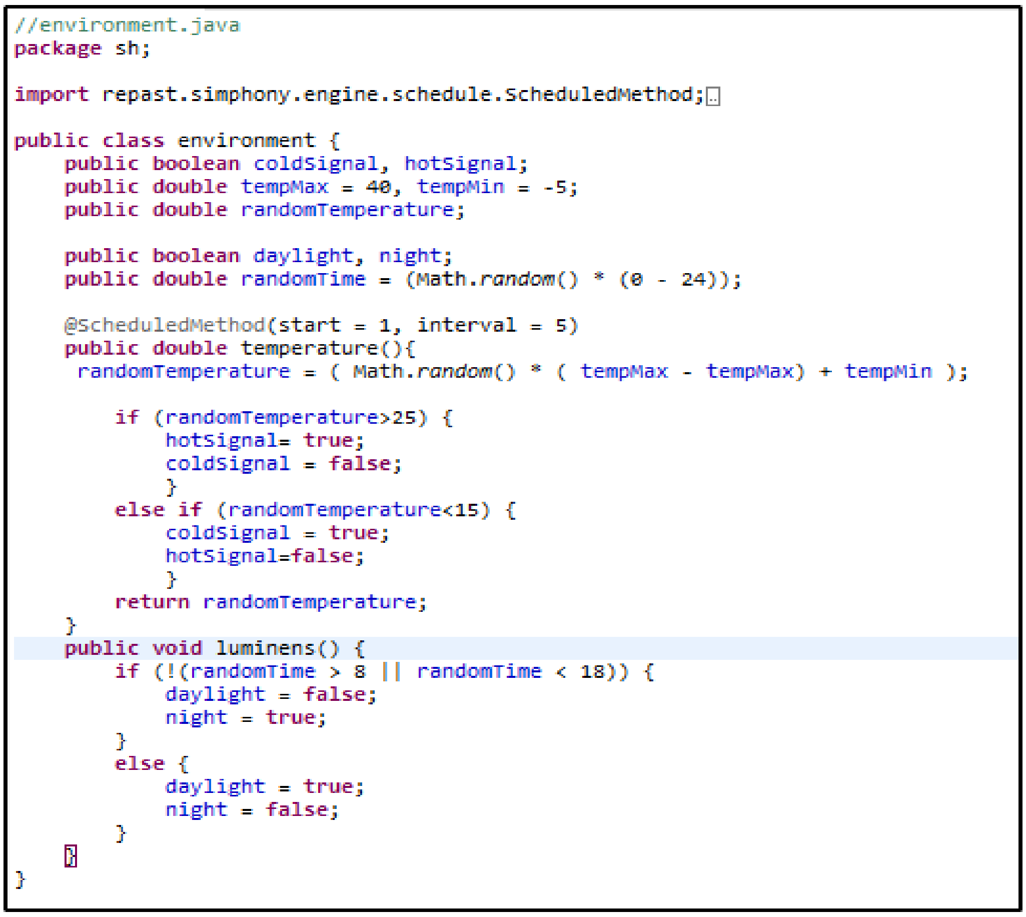
In this project this file contains the grid information it is a 10X10 grid. And it contains 3 windows, 3 Lights and 1 heater. They will come to the grid at random places.



```
//Applicationcore
package sh;

import repast.simphony.context.space.continuous.ContinuousSpaceFactory;

public class applianceCore implements ContextBuilder < Object >{

    @Override
    public Context build(Context<Object> context) {
        context.setId("smart-house");

        ContinuousSpaceFactory spaceFactory = ContinuousSpaceFactoryFinder.createContinuousSpaceFact
        ContinuousSpace<Object> space = spaceFactory.createContinuousSpace(
                "space", context, new RandomCartesianAdder<Object>(),
                new repast.simphony.space.continuous.WrapAroundBorders(), 10,
                10);

        GridFactory gridFactory = GridFactoryFinder.createGridFactory(null);
        Grid<Object> grid = gridFactory.createGrid("grid", context,
                new GridBuilderParameters<Object>(new WrapAroundBorders(),
                        new SimpleGridAdder<Object>(), true, 10, 10));

        int windowsCount = 3;
        for ( int i = 0; i < windowsCount ; i ++) {
        context . add ( new window ( space , grid ));
        }

        int heaterCount = 1;
        for ( int i = 0; i < heaterCount ; i ++) {
        context . add ( new heater ( space , grid ));
        }

        int lightCount = 3;
        for ( int i = 0; i < windowsCount ; i ++) {
        context . add ( new light ( space , grid ));
        }
        return context;
}
    }
```

Fig:2.2

ENVIRONMENT- In this file all the external parameters are assigned. Comfort temperature is between 15 to 25 degrees Celsius. Luminance is between 8 to 18 (Fig 2.3).



```
//environment.java
package sh;

import repast.simphony.engine.schedule.ScheduledMethod;

public class environment {
    public boolean coldSignal, hotSignal;
    public double tempMax = 40, tempMin = -5;
    public double randomTemperature;

    public boolean daylight, night;
    public double randomTime = (Math.random() * (0 - 24));

    @ScheduledMethod(start = 1, interval = 5)
    public double temperature(){
     randomTemperature = ( Math.random() * ( tempMax - tempMax) + tempMin );

        if (randomTemperature>25) {
            hotSignal= true;
            coldSignal = false;
            }
        else if (randomTemperature<15) {
            coldSignal = true;
            hotSignal=false;
            }
        return randomTemperature;
    }
    public void luminens() {
        if (!(randomTime > 8 || randomTime < 18)) {
            daylight = false;
            night = true;
        }
        else {
            daylight = true;
            night = false;
        }
    }
}
```

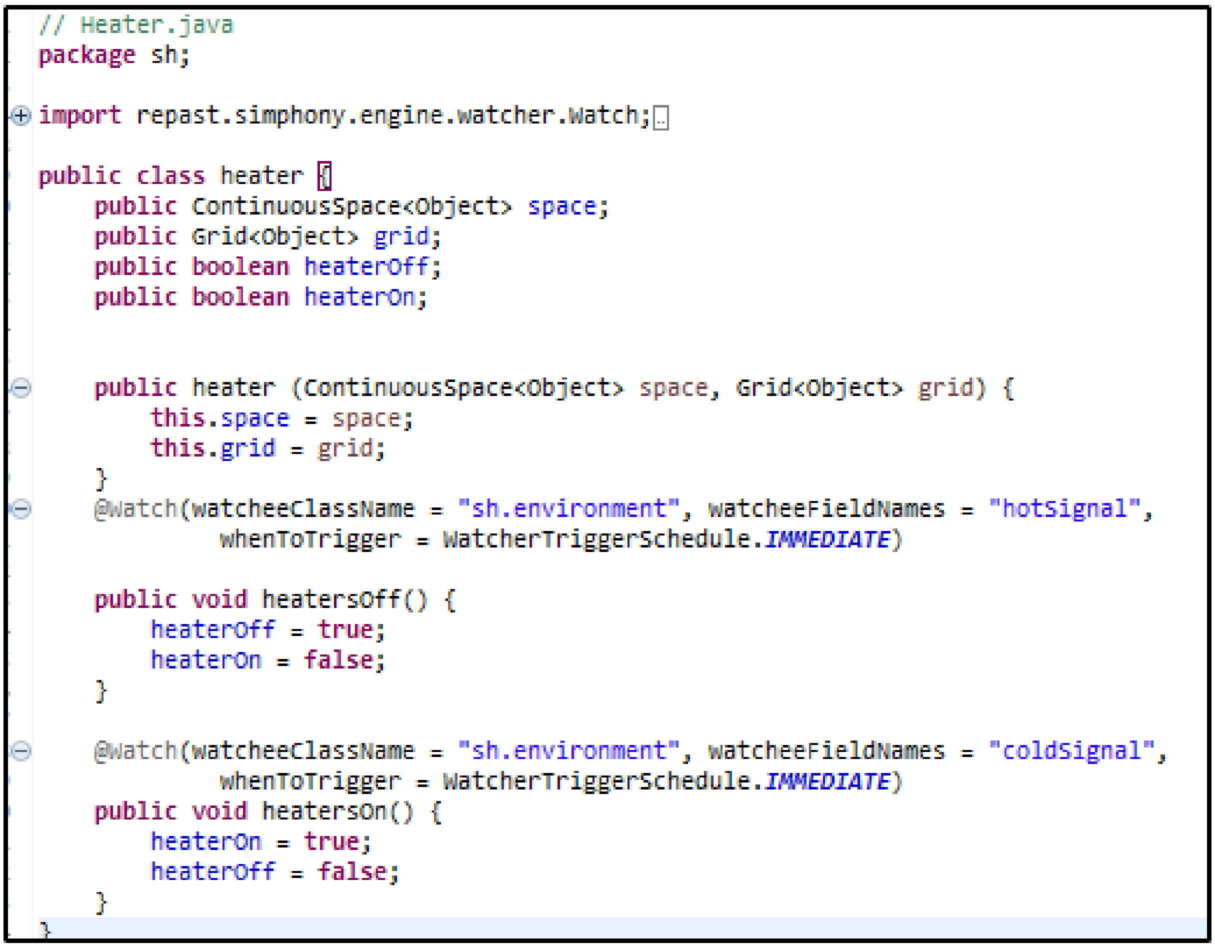
Fig 2.3

Heater: When Hot signal will come the Heater will turned on. (fig: 2.4)

Light: When the luminance goes below or above the threshold the lights will turned on and off. (fig: 2.5)

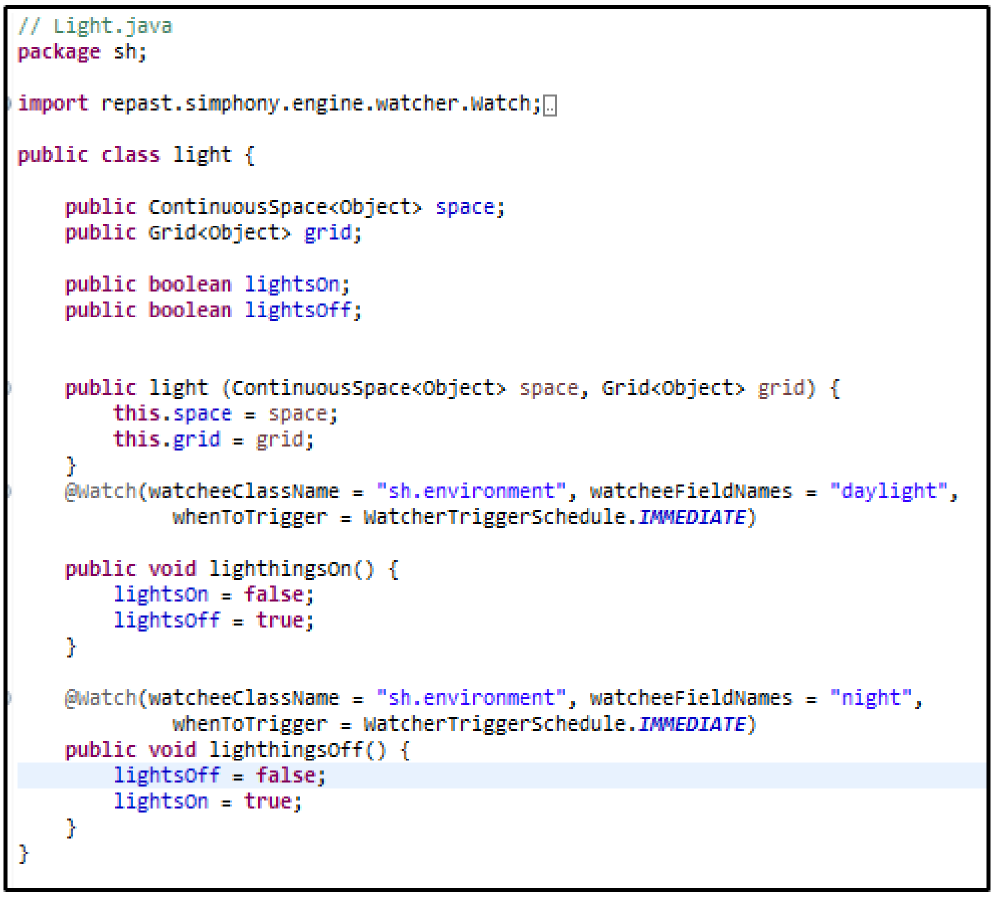
Windows: When the hot signal comes then the window opens and inform others too. And when cold signal comes windows closes. (fig: 2.6).

To make the implementation workable, A heat signal is made in every 5 intervals and agents react with these signals.



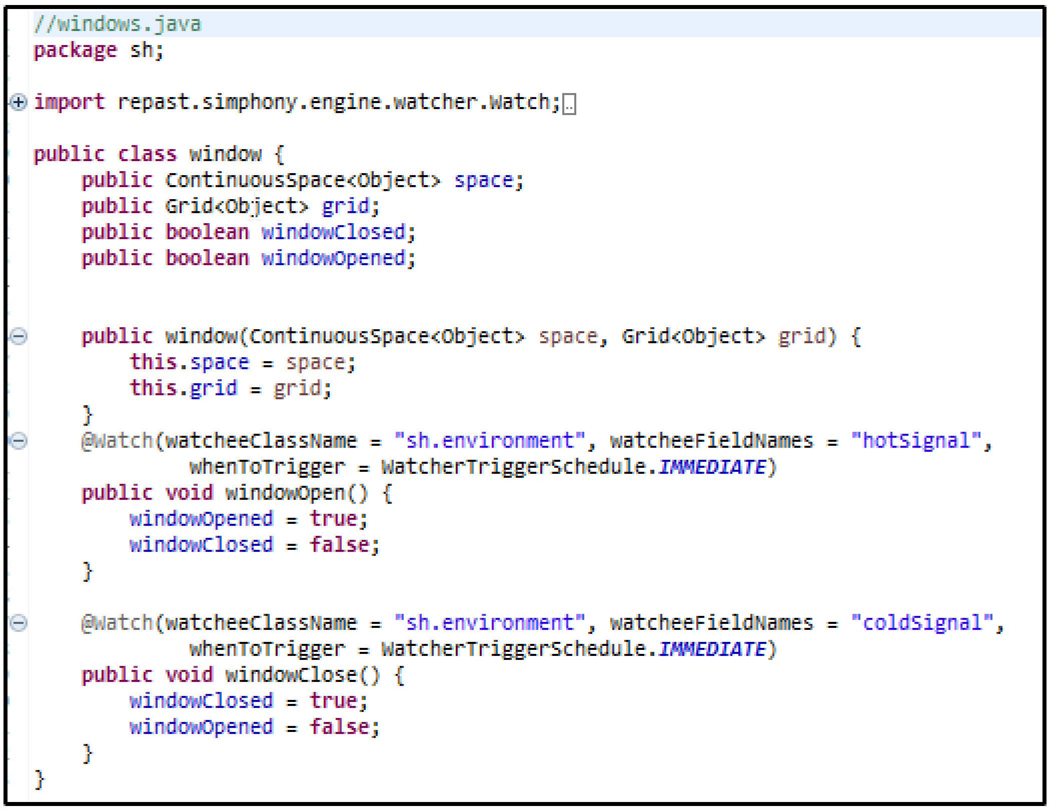
```
// Heater.java
package sh;

import repast.simphony.engine.watcher.Watch;

public class heater {
    public ContinuousSpace<Object> space;
    public Grid<Object> grid;
    public boolean heaterOff;
    public boolean heaterOn;


    public heater (ContinuousSpace<Object> space, Grid<Object> grid) {
        this.space = space;
        this.grid = grid;
    }
    @Watch(watcheeClassName = "sh.environment", watcheeFieldNames = "hotSignal",
            whenToTrigger = WatcherTriggerSchedule.IMMEDIATE)

    public void heatersOff() {
        heaterOff = true;
        heaterOn = false;
    }

    @Watch(watcheeClassName = "sh.environment", watcheeFieldNames = "coldSignal",
            whenToTrigger = WatcherTriggerSchedule.IMMEDIATE)
    public void heatersOn() {
        heaterOn = true;
        heaterOff = false;
    }
}
```

Fig:2.4



```
// Light.java
package sh;

import repast.simphony.engine.watcher.Watch;

public class light {

    public ContinuousSpace<Object> space;
    public Grid<Object> grid;

    public boolean lightsOn;
    public boolean lightsOff;


    public light (ContinuousSpace<Object> space, Grid<Object> grid) {
        this.space = space;
        this.grid = grid;
    }
    @Watch(watcheeClassName = "sh.environment", watcheeFieldNames = "daylight",
            whenToTrigger = WatcherTriggerSchedule.IMMEDIATE)

    public void lighthingsOn() {
        lightsOn = false;
        lightsOff = true;
    }

    @Watch(watcheeClassName = "sh.environment", watcheeFieldNames = "night",
            whenToTrigger = WatcherTriggerSchedule.IMMEDIATE)
    public void lighthingsOff() {
        lightsOff = false;
        lightsOn = true;
    }
}
```

Fig:2.5



```java
//windows.java
package sh;

import repast.simphony.engine.watcher.Watch;

public class window {
    public ContinuousSpace<Object> space;
    public Grid<Object> grid;
    public boolean windowClosed;
    public boolean windowOpened;


    public window(ContinuousSpace<Object> space, Grid<Object> grid) {
        this.space = space;
        this.grid = grid;
    }
    @Watch(watcheeClassName = "sh.environment", watcheeFieldNames = "hotSignal",
            whenToTrigger = WatcherTriggerSchedule.IMMEDIATE)
    public void windowOpen() {
        windowOpened = true;
        windowClosed = false;
    }

    @Watch(watcheeClassName = "sh.environment", watcheeFieldNames = "coldSignal",
            whenToTrigger = WatcherTriggerSchedule.IMMEDIATE)
    public void windowClose() {
        windowClosed = true;
        windowOpened = false;
    }
}
```

Fig: 2.6

4.2.1. Implementation: In the grid there are 3 types of agents. red is windows, blue are the lights and the green one is heater. After 5 seconds there is one signal comes after every five interval and change the state of the appliance. (Fig: 2.7)

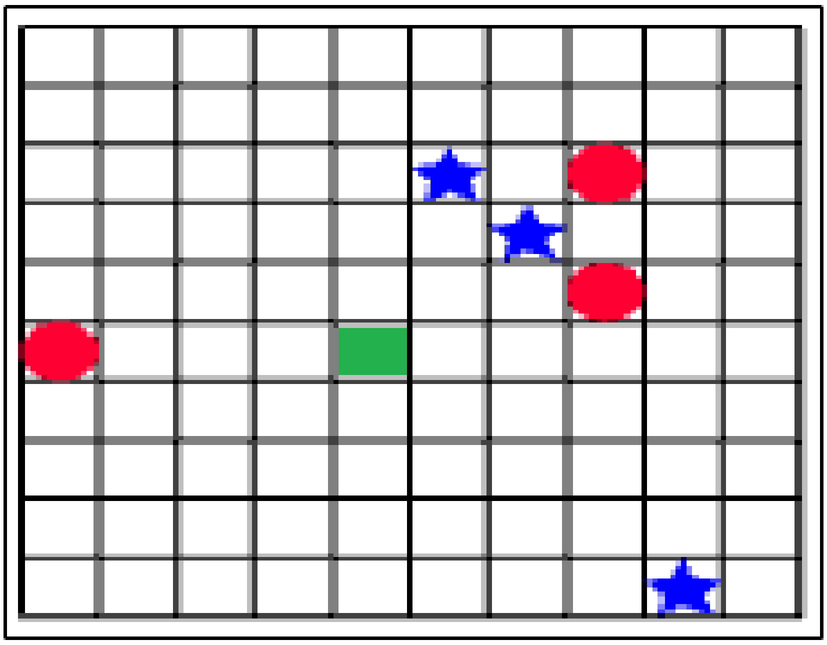


Fig: 2.7

5. Evolution And Discussion:

In this project there are two situations one is when a user accesses a HOME APPLIANCE and another is when a HOME APPLIANCE changes its state according the factors of the environments.
For the first case a multiagent solution is not required, but for the second case a multi agent solution in needed. As when a user wants to change a Home Appliance it will not broadcast the change state signal to the others but if the status is changed by the environmental factors, it will send the change state signal to the others HOME APLIANCES of the same group.

After future modifications, the user should give the parameters of their choice so that the system can work more efficiently and properly with the users.