ReadMe

Overview:
I have used Java & Jess together to construct the system. Shadow facts are created to reason about java objects – the sensors in this case, and produce the resultant environment based on various rules.

What's inside:
The three sensors are written as Java Beans. The corresponding files are:

1. LuminositySensor.java
2. MotionSensor.java
3. TemperatureSensor.java

The concept of "time" has been implemented as a direct Jess template with slot values "current time" & "timeOfday". The main Jess program is written in "main.clp"

How the knowledge base is structured?
The knowledge base is the sensors and the information provided to them. You can specify a range of values to each sensor field and combined with all the other values, the rules are applied and output produced.

The sensors are defined as templates and then added to the working memory as "shadow facts". These shadow facts are the bridge back to java objects & I have used them to set sensor values and also actuate changes in the environment.

The various inputs you will need to specify are:

1. time
2. timeOfday
3. lightsOn
4. daylightFactor
5. acOn
6. heatOn
7. environmentTemp
8. windowsOpen
9. doorsOpen

Installation:

Since I have used Java and Jess together, you will need to use “Eclipse 3.3.0” installed in your system. And once you have that installed, you will need “Jess DE” – Jess Developer Environment with Eclipse to run the program.
The steps to install Jess DE can be found here:

http://www.jessrules.com/doc/70/eclipse.html

Build & run instructions:

Once you have that up and running,

1. Import the project archive file “SmartHomeJess” into an empty project in Eclipse (Select the “Import” from “File” menu > “General” > “Archive file” option).
2. Build the project and all its files
3. Open “main.clp” file
4. Now “Run” this file
5. You should see the facts - before & after application of rules and then the output in the console

3 facts input:

The program comes with some values for facts already in it. But here are three cases you can test the program on:

Case 1:

1. time – 8
2. timeOfday – “daytime”
3. environmentTemp – 90
4. acOn – TRUE
5. heatOn – FALSE
6. daylightFactor – 3.6
7. lightsOn – TRUE
8. doorsOpen – TRUE
9. windowsOpen – FALSE

Case 2:

1. time – 3
2. timeOfday – “evening”
3. environmentTemp – 50
4. acOn – FALSE

5. heatOn – FALSE
6. daylightFactor – 1.6
7. lightsOn – FALSE
8. doorsOpen – TRUE
9. windowsOpen – FALSE

Case 3:

1. time – 4
2. timeOfday – “night”
3. environmentTemp – 80
4. acOn – FALSE
5. heatOn – FALSE
6. daylightFactor – 1.0
7. lightsOn – FALSE
8. doorsOpen – FALSE
9. windowsOpen – FALSE

Range of values:

For time values,
6 AM – 5 PM – is considered “daytime”
above 5PM – 11 PM – “evening”
After 11PM – 6 AM – “night”
Note: Kindly keep these values in mind while specifying “time” & “timeOfday”. They should pair well w.r.t these values.

For temperature,
Below 58 degrees Farenheit – cold
Above 95 degrees Farenheit – hot,
Between 58 & 75 – normal
Between 75 & 95 - warm
Note: These values are merely for reference so that you can check the result of ac or heat being on or off as appropriate and nothing more.

For light,
daylightFactor of 2.5% & below (i.e. <=2.5) – dark so light turned on
daylightFactor of 4% & above (i.e. >4) – bright so light turned off

Note: These values are merely for reference so that you can check the result of lights being on or off as appropriate and nothing more.

For exits,
When either AC or heat is on, all doors and windows are closed.