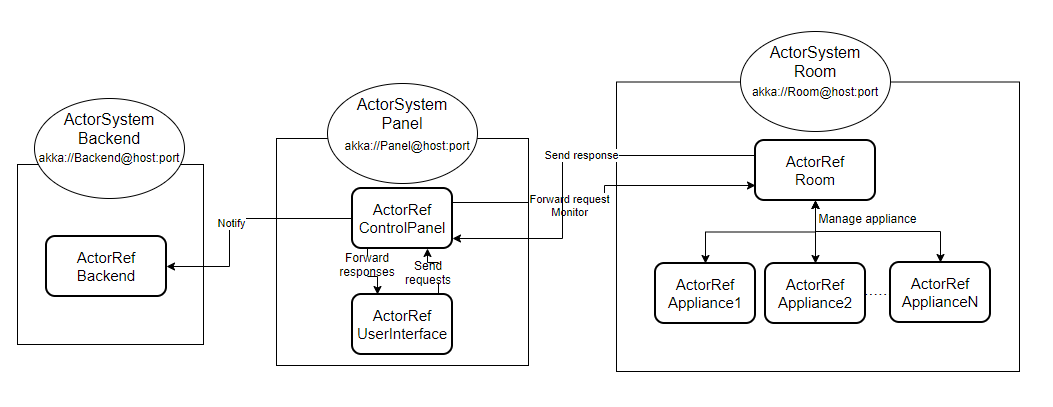
Project 3 Documentation

**Introduction**

The goal of the project was to create a software layer for a distributed smart homing system. This system must be able to manage appliances throughout the house, even from different rooms, allowing the user to enter preferences and achieve changes in the behavior of those appliances. In addition, the system must be covered against any process crashes.

The system uses akka's actor model and its key components are described below.

**Implementation**



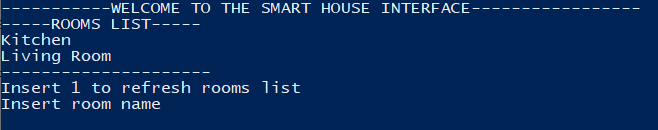
The diagram represents the basic structure of the system. It is defined by three main sections, the **Backend** section, the **Panel** section and the **Rooms** section. In a real environment Panel and Rooms are deployed in the same network, which is the home network, while Backend is deployed on a remote node.

**Panel**

The term Panel is defined as the interface that allows a user to interact with the home system. The **Control Panel Actor** is connected to each Room and can control the Appliances inside them. It is responsible for the elaboration of the requests coming from the User Interface Actor and the monitoring of the Rooms Actors. When a Room wants to connect to the system it sends a message to the Control Panel which adds the room to the list of available rooms. The Control Panel then Monitors the functioning of the Room with the watch() function. As soon as the Room process is interrupted it receives a Terminated message and removes the disconnected room from the list.

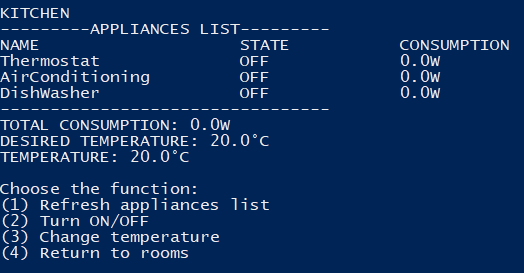
The **User Interface Actor** embodies the interface that allows the user to input the preferences for a specific room. This is done with the help of two interfaces.

The first interface makes the user decide which Room to check by inserting its name.



The second interface shows the situation of the specified room with all the information that a user might need and the functionalities:

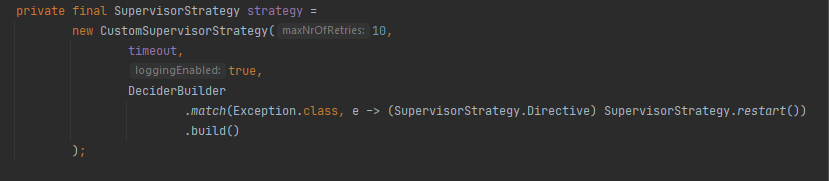
* Observe and refresh the list of appliances connected to the system and their current status, i.e. whether they are on or off and the resulting power consumption.
* Turn on or off any of the equipment connected to the system.
* Change the desired temperature.
* Return to the previous interface.



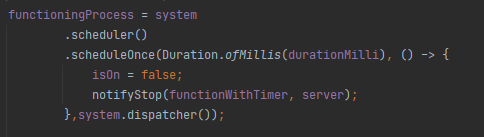
**Room**

The Room section as seen in the graph represents a generic Room the Control Panel might be connected to. In the given code there are two Rooms connected to the Panel (Living Room and Kitchen). Each of them is composed by two types of Actors:

* The **Room Actor** is the one responsible for supervising the work of all the underlying actors. So, its main purpose is to redirect the requests coming from the Control Panel to the corresponding Appliance. But another important job is to supervise the Appliances process and restarting them in case of crash using its Supervisor Strategy.



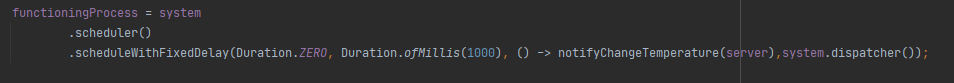
* The **Appliance** actor is a generic term for any such actor that is connected to the Room. They have very different functions but can be summarized in 3 different types:
  + Appliances that are turned on and off only by an explicit request from the user. They are the simplest type of appliance that only has a sensor that verifies the current consumption.
  + Appliances that have a timed operation. Once they are turned on, they stay on for a preset time and then turn off by themselves, notifying the control panel of the event. This is done using the akka scheduler class.



* + Devices that affect temperature. Temperature is a data that is collected by a special device, the **Thermostat**. This is the only one that has a sensor that detects the temperature of the room and through this device the control panel receives the temperature whenever it needs it.

In addition, this device, once turned on, takes care of heating the environment by increasing the temperature. The **Air conditioning** appliancedoes the opposite thing. Every time the user communicates that he wants to change the temperature, the room controller turns on the reference device to heat or cool the room until the desired temperature is reached. Once this is done it turns off automatically.

Also in this case the akka scheduler is used, but unlike the previous case, before shutting down, it performs the action of adjusting the temperature of one degree at a time until it reaches the desired one.



**Backend**

The main purpose of the Backend Actor is to act as a remote server which receives updates about the situation of the SmartHome from the control Panel. In our case the Panel informs it about the overall energy consumption of the Appliances in the House.

Each time the Control Panel is updated about the situation of a specific Room It sends an update message to the Backend which then calculates the total consumption.