Oblig 1

Gruppe 2 Aleksander Vanberg Eriksen, Gjermund Østensvig og Tobias Hallingstad

Energy saver

We created a new thing called “Energy Saver” (ES), with ports for setting comfort temperature, timer to keep track of day/night and a port for other commands. The ES has a state chart containing 4 states, setup, present, day and night. Setup is entered when the ES first starts and is responsible for transitioning to the correct state.

Present state

The present state is used when the light in the room is turned on. The temperature is then set to the comfort temperature the user has specified. If the light in the room is turned off, we should move to the day state, unless it also is night, then we should move to the night state.

Night state

When entering night state, the temperature is reduced by a set number and is only active if the luminance value does not exceed a given luminance threshold. The night state consists of two transitions, present or day. Day transition will happen if the shift from night to day occurs and is guarded by a timer. Present transition will happen if a luminance value is measured higher than the set threshold. This indicates that there are people in the room. Apart from the transitions, night state has internal events that keep track of the luminance and temperature.

Day state

When we are in the day state, we should reduce the temperature to a small, user set, amount. This state is only active when it is day and the light in the room is turned off. When in this state we can either move to the night state, if it becomes night, or the present state, if the light turns on.

General

During all three states the user can change the comfort temperature, when day/night starts, day/night temperature difference. The user can also turn on/off the heating independent of the rest of the system. When the heating is off, no heating should happen. This results in the system just running, but no changes happen.

Test state

In test state we have different states that test the system for the expected output values from the given input values. The main tests that we implemented are how the system works in day state, how it transitions to the night state as well as the normal functionalities regarding temperature and luminance.

Configuration

There are implemented two configurations for the system. The first is the configuration that was there at the start of the project. The second configuration is one we made to handle testing.

CONFIG.THINGML

Consists of instances simulation, PSM, PIM, energy_saver, human, night_day_timer. The main change to this config is adding in the energy_saver. This is how the energy saver is connected to the system:

- Human is connected to energy_saver so that it can send set_temperature and set_lum_threshold
- PSM is connected to energy_saver so that it can send it luminance
- Energy_saver is connected to PIM to send set_temperature.

TESTCONFIG.THINGML

Consists of the same instances as config.thingml, but simulation and human are merged into the same test instance. The connections are also the same as config.thingml, but there are no GUI generated. The point of the testConfig.thingml is to use the Test thing's state chart to test the "test item" (energy saver). The tests consist of checking changes in luminance, temperature, and the transition from the states.

Additional information

IO.THINGML

IO. THINGML consists of two mocks, Human and Simulation.

Human sends input switch on, switch off, set_lum_threshold, set_temperature, set_delta and fetch_temp with the use of a Gui, and receives temperature.

Simulation is used to send temperature and luminance to the energy saver and PIM.