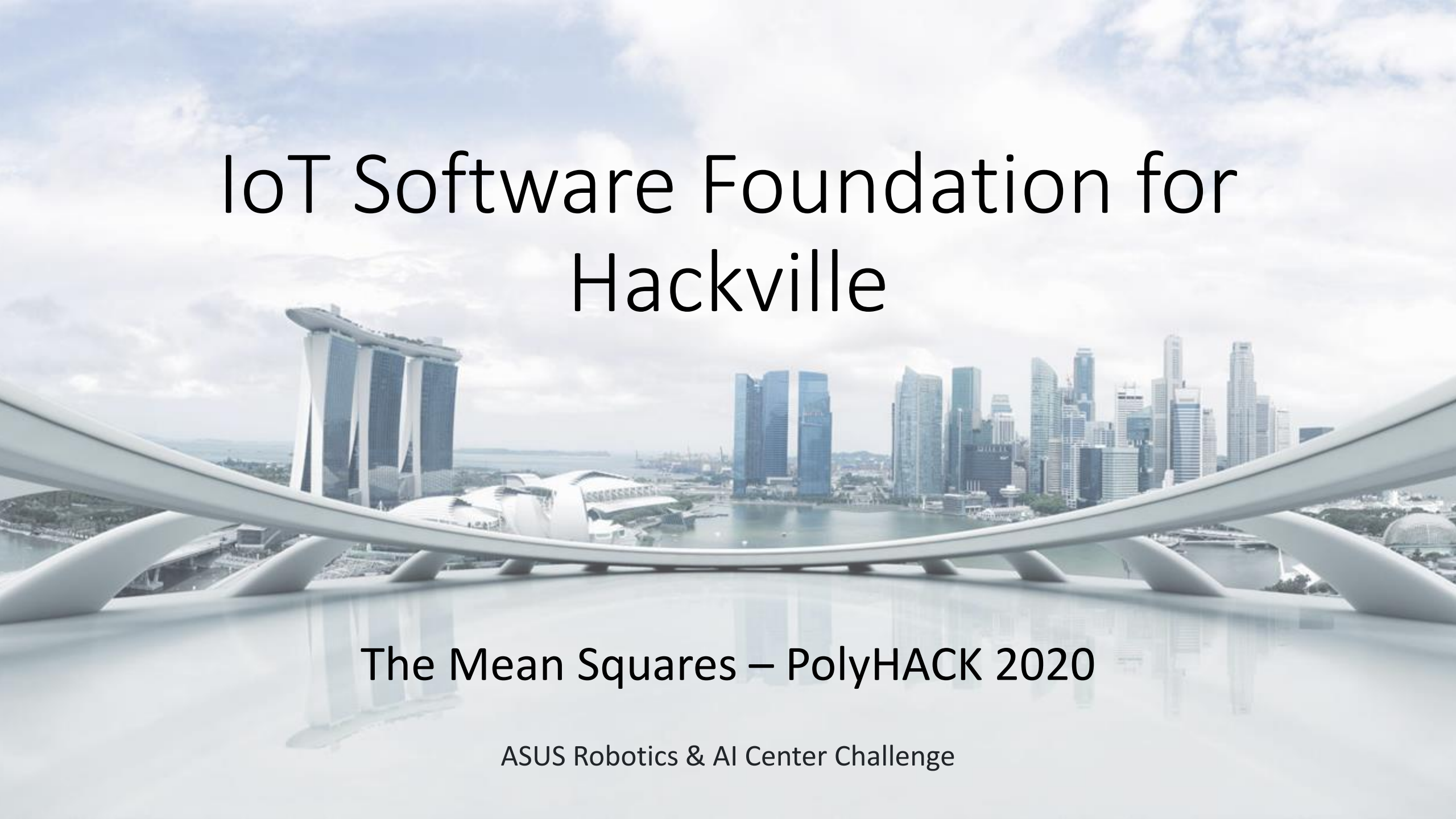


IoT Software Foundation for Hackville

The Mean Squares – PolyHACK 2020

ASUS Robotics & AI Center Challenge

Our mission

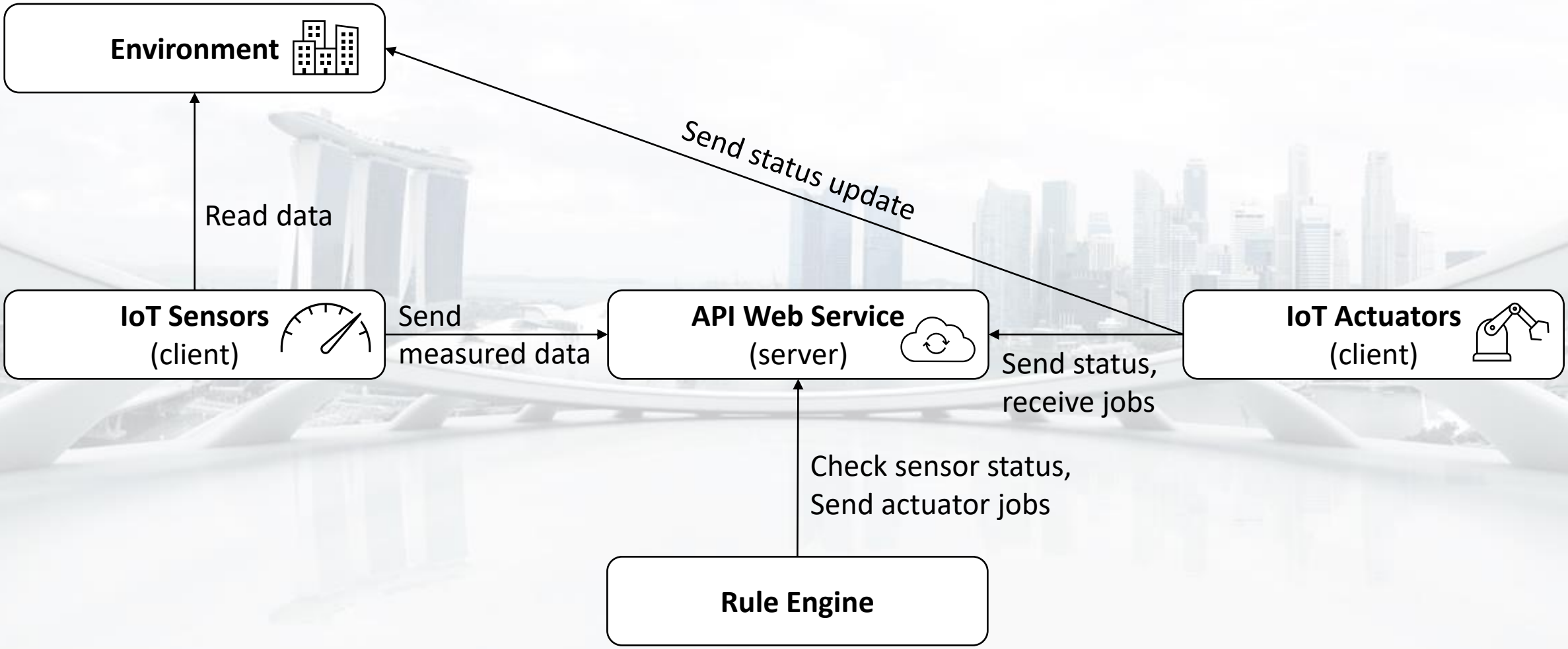
We, *The Mean Squares*, are representatives of the company Polycraft.

Our goal is to provide the software to automate IoT devices throughout Hackville.

To proof our technical competence we have build an MVP with various types of sensors and actuators communicating via a server that act in a simulated environment.

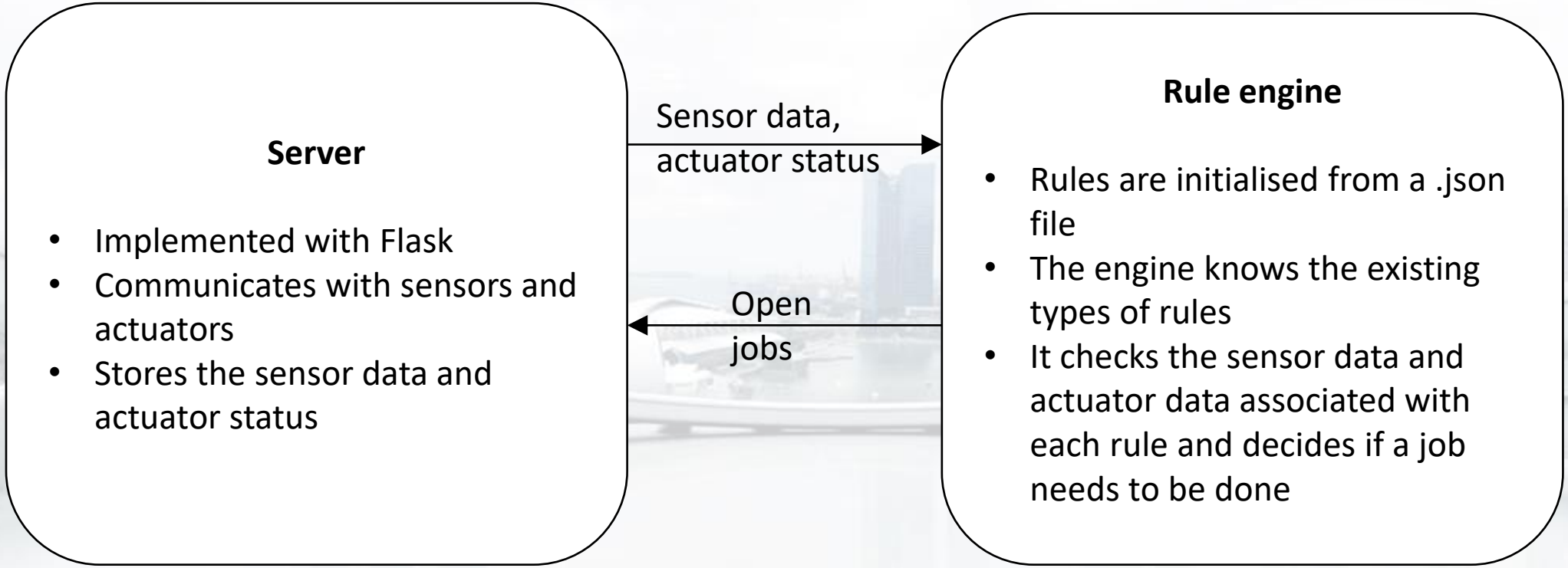
System Overview

Fat server with thin clients implemented using Flask



API Web Service & Rule engine

Fat server containing most of the application‘s logic



Client-Server Communication Protocols

Client side communication

All clients communication must contain the following fields:

- 'id': a unique identifier for the device
- 'ancestors': a list of inheritance of the device in question, in order from youngest to oldest (e.g. ['TemperatureSensor', 'Sensor', 'IOT_Device'])

Clients are Sensor or Actuator.:

- A Sensor has a data entry, which contains a dictionary of measured values
- An actuator instance has a status entry, containing a dictionary describing the state of the device.

Server side communication

The server receives messages from the client, and stores the received data locally.

Sensor:

- The sensor only sends data

Actuator:

- If an actuator needs actuating, the server will wait until the next time the actuator contacts the server, and will return the job to do be executed by the actuator.
- The response from the server is a list of jobs from the rules engine as a dictonary .
- If there is nothing to do, the dictonary is empty. If there is something to do, the dictionary will list what values to change. E.g. if a door should be unlocked, the response will be: {'DoorLocked': False}.

IoT Sensors

Sensors are Thin clients that read from the environment and send their data to the server

Implemented sensor types:

- Temperature sensor
- Humidity sensor
- Brightness sensor
- Proximity sensor
- Noise sensor
- Motion sensor
- Distance sensor
- Airquality sensor

Initialisation:

- Each sensor has an ID, a type and a position
- The number and type of sensors in Hackville can by configured in a .json file

IoT Actuators

Actuators are thin clients that sent their status to the server and receive jobs to execute

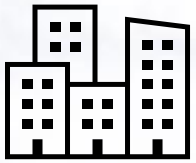
Implemented actuator types:

- Smart lamp
- Smart door lock
- Motor position
- Smart heating
- Smart sprinkler

Initialisation:

- Each actuator has an ID, a type, a position and a status
- The number and type of sensors in Hackville can by configured in a .json file

Simulated Environment



- The environment is a class simulating the input data for each sensor
- Each sensor can read a feed of random variables that are scaled according to its type
- The actuators can influence the environmet (e.g. a heater inreases the temperature in its surrounding)
- Due to the modular implementation, the environment can easily be replaced by real input data

Try our MVP

Our sourcecode: https://github.com/Karko93/Polyhack2020-Project.git

Run the code according to the readme.md

Go to the webpage http://127.0.0.1:5000/ to see the simulations of Hackville

The Mean Squares

Dominik Windey
Eric Bonvin
Marco Ketzel
Andrei Militaru
Rebecca Westphal