Electricity measuring system

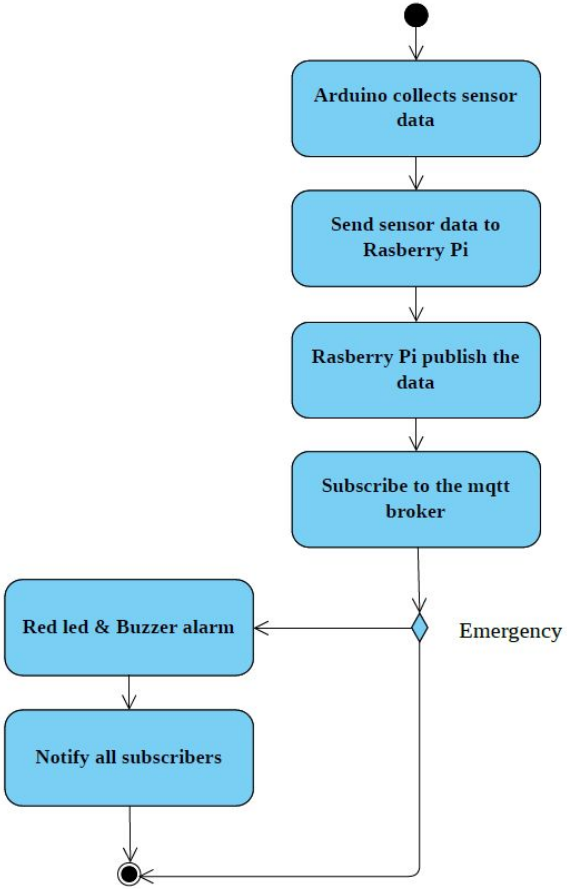
Team Pi-sense

Lochana Abhayawardana, Arsany Girgis and Emirkan Sali

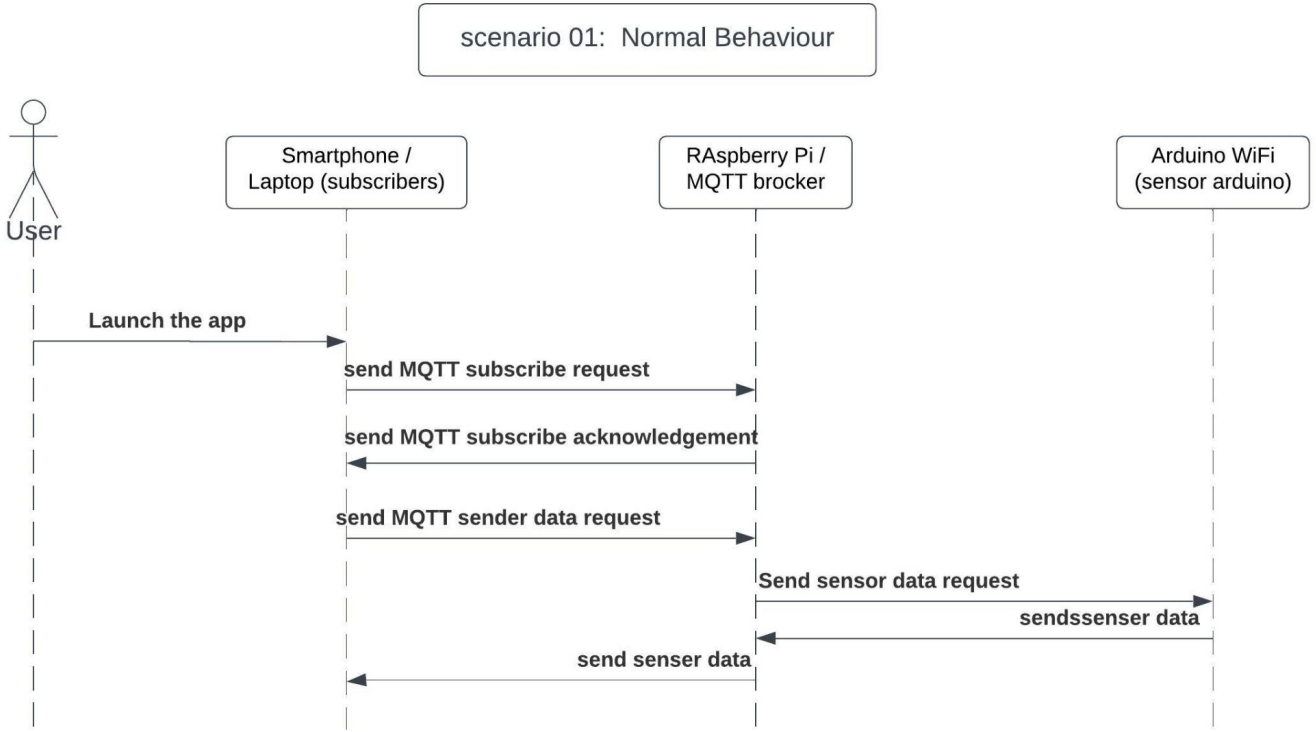
Concept


Current Sensor
Door lock/Motor
Green LED
Red LED
Buzzer
Voltage Sensor
Arduino/Publisher
Arduino/Subscriber
Temperature Sensor
Laptop/Computer (Subscriber)
Current Sensor
Raspberry Pi/ Broker
Voltage Sensor
Arduino/Publisher
Phone (Subscriber)
Temperature Sensor
Cloud Service

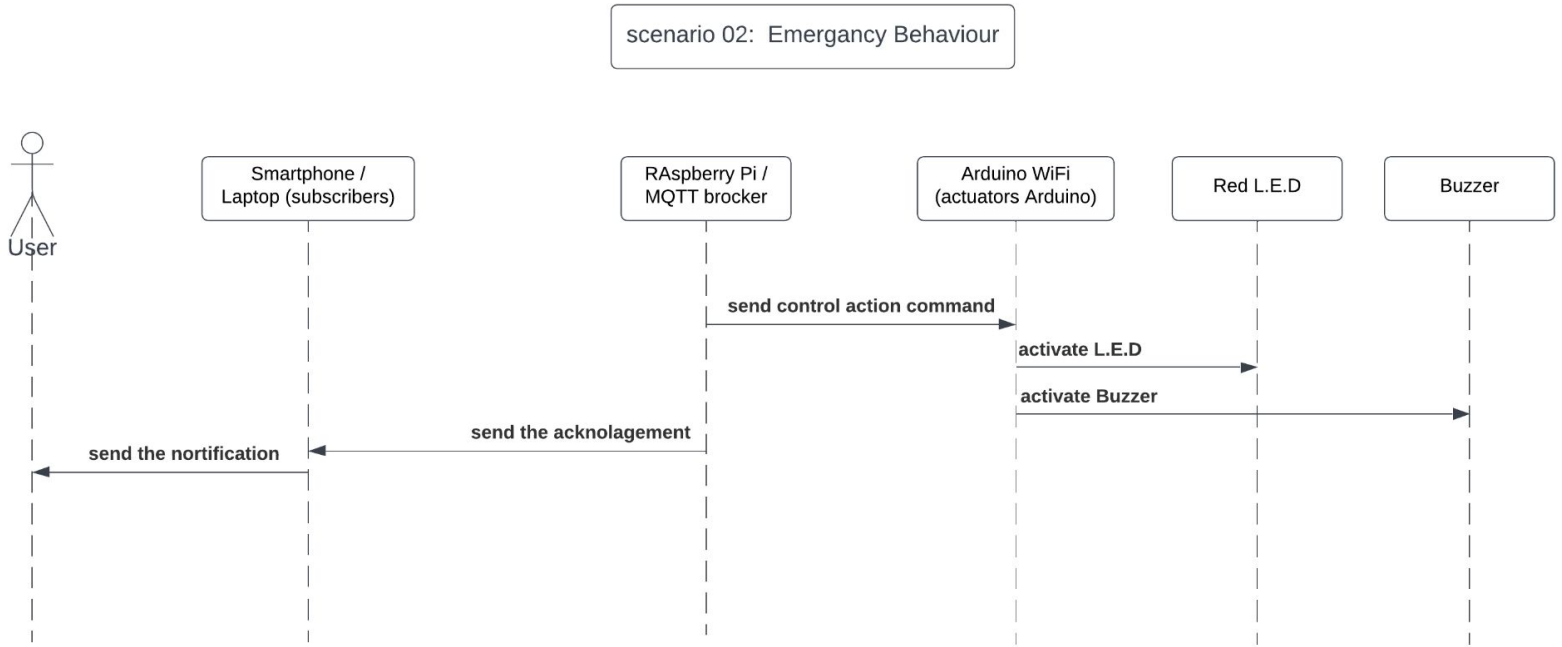
Concept


Arduino collects sensor data
Send sensor data to Rasberry Pi
Rasberry Pi publish the data
Subscribe to the mqtt broker
Emergency
Red led & Buzzer alarm
Notify all subscribers

Concept

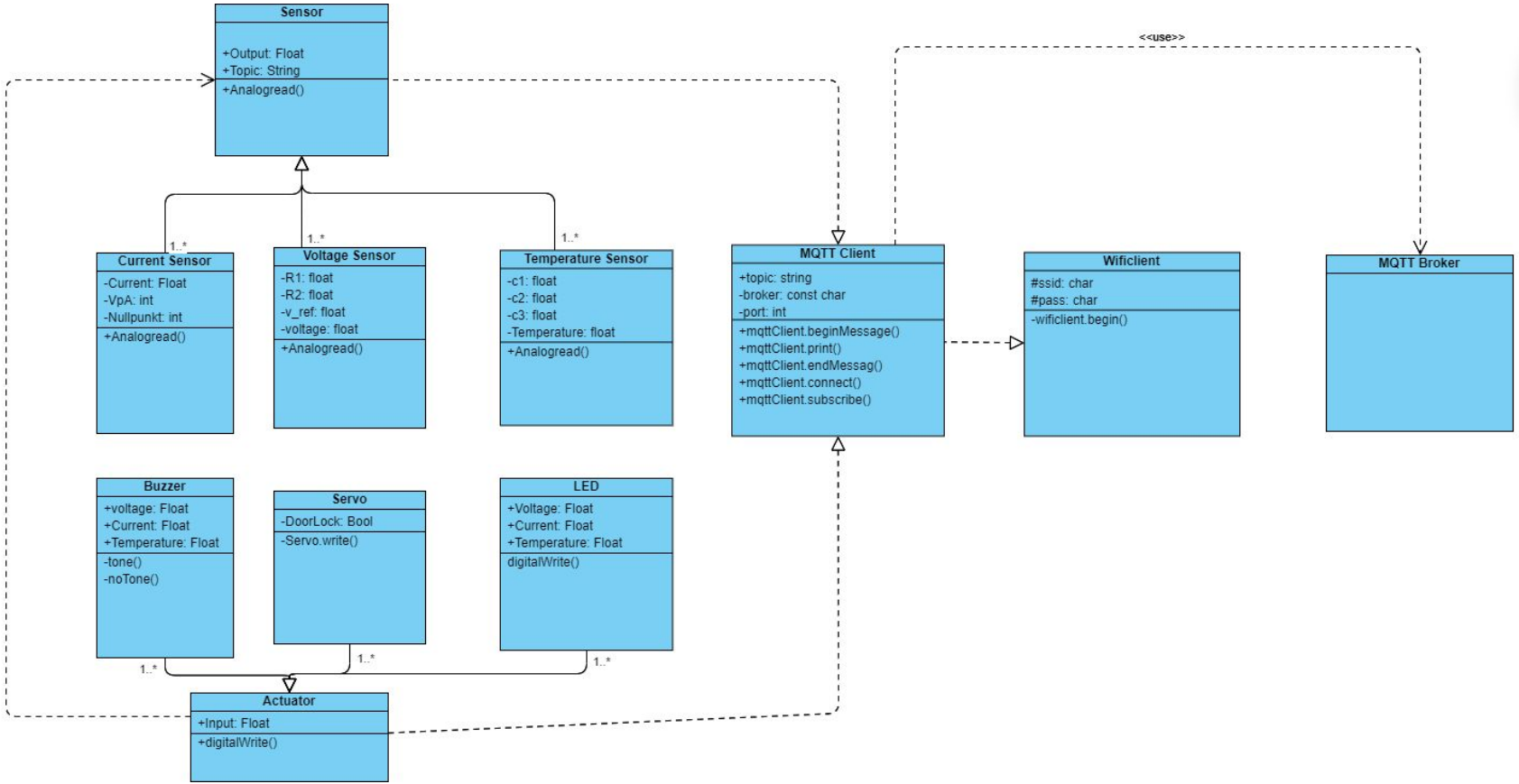

scenario 01: Normal Behaviour
User
Smartphone / Laptop (subscribers)
RAspberry Pi / MQTT brocker
Arduino WiFi (sensor arduino)
Launch the app
send MQTT subscribe request
send MQTT subscribe acknowledgement
send MQTT sender data request
Send sensor data request
sendssenser data
send senser data

Concept


scenario 02: Emergancy Behaviour
User
Smartphone / Laptop (subscribers)
RAspberry Pi / MQTT brocker
Arduino WiFi (actuators Arduino)
Red L.E.D
Buzzer
send control action command
activate L.E.D
activate Buzzer
send the acknolagement
send the nortification

Implementation of sensors and actuators

- Setting up variables for Voltage, Current, temperature and Servo
- Choosing the right resistors for our voltage sensor
- Sensors connected to analog pins
- Creating the right functions for the sensors
- Setting up code for triggering an alarm and moving the Servo



Class diagram of our system

```
void VOLT1()
{
  // Measuring up to 17.5 Volts with this Resistor Set-up

  float R1 = 10000;
  float R2 = 4000;
  float v_ref = 5;
  float resistor_ratio = 0;
  float adc_value = 0;
  float voltage = 0;
  resistor_ratio = (R2/(R1+R2));

  for (int i = 0; i<20; i++)
  {
    adc_value = adc_value + analogRead(voltmeasure);

    delay(1);
  }

  adc_value = adc_value/20;
  voltage = ((adc_value* v_ref) / 1024);

  // Serial.print("\nADC Voltage CH1VOLT =");
  // Serial.print(voltage);
  // Serial.print("\n\n");

  V1 = voltage/ resistor_ratio;

}
```

```
void CurrentSense()
{
  sensorwert = analogRead(Sensor);
  SensorSpannung = (sensorwert / 1024.0) * 5000;
  Ampere = ((SensorSpannung - Nullpunkt) / VpA);
```

Measuring current and voltage

MQTT implementation

- Initialisation

```
WiFiClient wifiClient;
MqttClient mqttClient(wifiClient);
const char broker[] = "10.42.0.1";
int        port     = 1883;
const char topic[]  = "Voltage";
const char topic2[]  = "Temperature";
const char topic3[]  = "Current";
//set interval for sending messages (milliseconds)
const long interval = 4000;
```

1. Publisher

```
// send message, the Print interface can be used to set the message contents
mqttClient.beginMessage(topic);
mqttClient.print(V1);
mqttClient.endMessage();

mqttClient.beginMessage(topic2);
mqttClient.print(T);
mqttClient.endMessage();

mqttClient.beginMessage(topic3);
mqttClient.print(Ampere);
mqttClient.endMessage();
```

2. Subscriber

```
// subscribe to a topic
mqttClient.subscribe(topic);
mqttClient.subscribe(topic2);
mqttClient.subscribe(topic3);
```

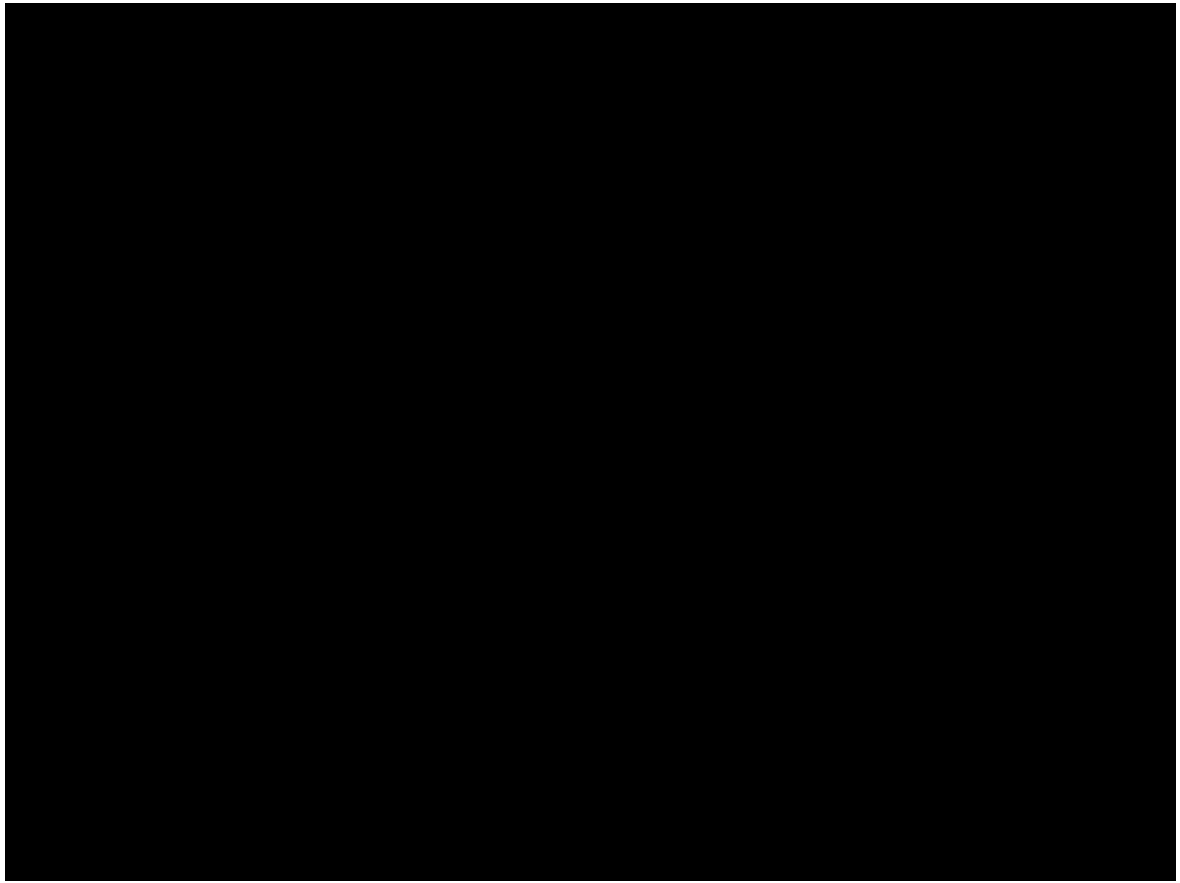
```
void onMqttMessage(int messageSize) {
  // we received a message, print out the topic and contents
  Serial.println("Received a message with topic '");
  currentTopic = mqttClient.messageTopic();
  Serial.println(currentTopic);
  Serial.print("', length ");
  Serial.print(messageSize);
  Serial.println(" bytes:");
  mqttString = "";

  // use the Stream interface to print the contents
  while (mqttClient.available()) {
    // Serial.print((char)mqttClient.read());
    mqttMessage = (char)mqttClient.read();
    mqttString += (mqttMessage);

  }
  finalvalue = mqttString.toFloat();
  Serial.println(finalvalue);

  if (currentTopic == "Voltage")
  {
    voltage = finalvalue;
  }
  else if (currentTopic == "Current")
  {
    current = finalvalue;
  }
  else if (currentTopic == "Temperature")
  {
    temperature = finalvalue;
  }

  Serial.println();
  Serial.println();
}
```

MQTT in Arduino

- Using library “ArduinoMqttClient”
- Setting important variables for connecting to broker
- Connecting to the Raspberry Pi Wi-Fi and to the broker
- Sending or receiving messages

Conclusion

- Concept of an reliable IoT electricity measuring system applicable in industrial environments
- Using Actuator and Sensor Device for the prototype and a Raspberry Pi
- Using MQTT, establishing a network with actuators, sensors and a broker
- Implementing MQTT on the Arduinos and Raspberry Pi
- Final design possibility would include more sensitive Current sensor and industrial screw NTC Thermistor



Demo for our system