IoT in agriculture. An automated watering system based on wireless sensors.

MsC in Applied Informatics
Antonios V. Stoulos

Internet Of Things

- 30 Billion interconnected devices by 2020
- Applied in every possible sector (smart cities, houses, automotive, industry etc.)
- Optimization, ease, exploit possibilities

Internet Of Things

- Big Data applications
  - Energy efficient smart building (Psannis et al 2017)
  - Knowledge extraction in healthcare (Firouzi et al 2018)
- Big Data applications in farming
  - Presentation of current applications digging into socioeconomic pros n' cons (Wolfert 2017)
  - Reduction of noise, enhance of accuracy (Muangprathub et al 2019, Lovas et al 2018)

Internet Of Things

- Cloud applications
  - Presentation of current state with the optimizations that come with Cloud in IoT (Mohammad et al 2018)
  - From monoliths to micro-clouds (Varghese et al 2018)
- Cloud applications in farming
  - Clustered weather prediction (Molthan et al 2015)
  - Glue between WSNs and Big Data
  - Identification is crucial (Botta et al 2015)

Internet Of Things

- Smartphones
  - Part of the ecosystem
  - Gives the opportunity to human to become member of the IoT environment by controlling and monitoring processes
- Smartphones in Agriculture
  - Exploit smartphones' capabilities (camera, GPS etc.) in order to get knowledge (Pongnumkul et al 2015)

IoT in Agriculture

Pros

- ✓ Processes optimization
- ✓ Environmental advantages
- ✓ Rapid economic growth

Cons

- Lack of technical background
- Location limitations
- Privacy and security

IoT in Agriculture

- Attempts to introducing automation have been taking place by 80s already (Precision Agriculture)
- IoT can support the implementation of a decision making system

Proposal

A decision making system based on wireless sensors' network, measuring soil moisture, forwarding data to a cloud application which extracts result through an algorithm and informs back the WSN. An accompanying mobile application will help monitoring and controlling.

Previous approaches

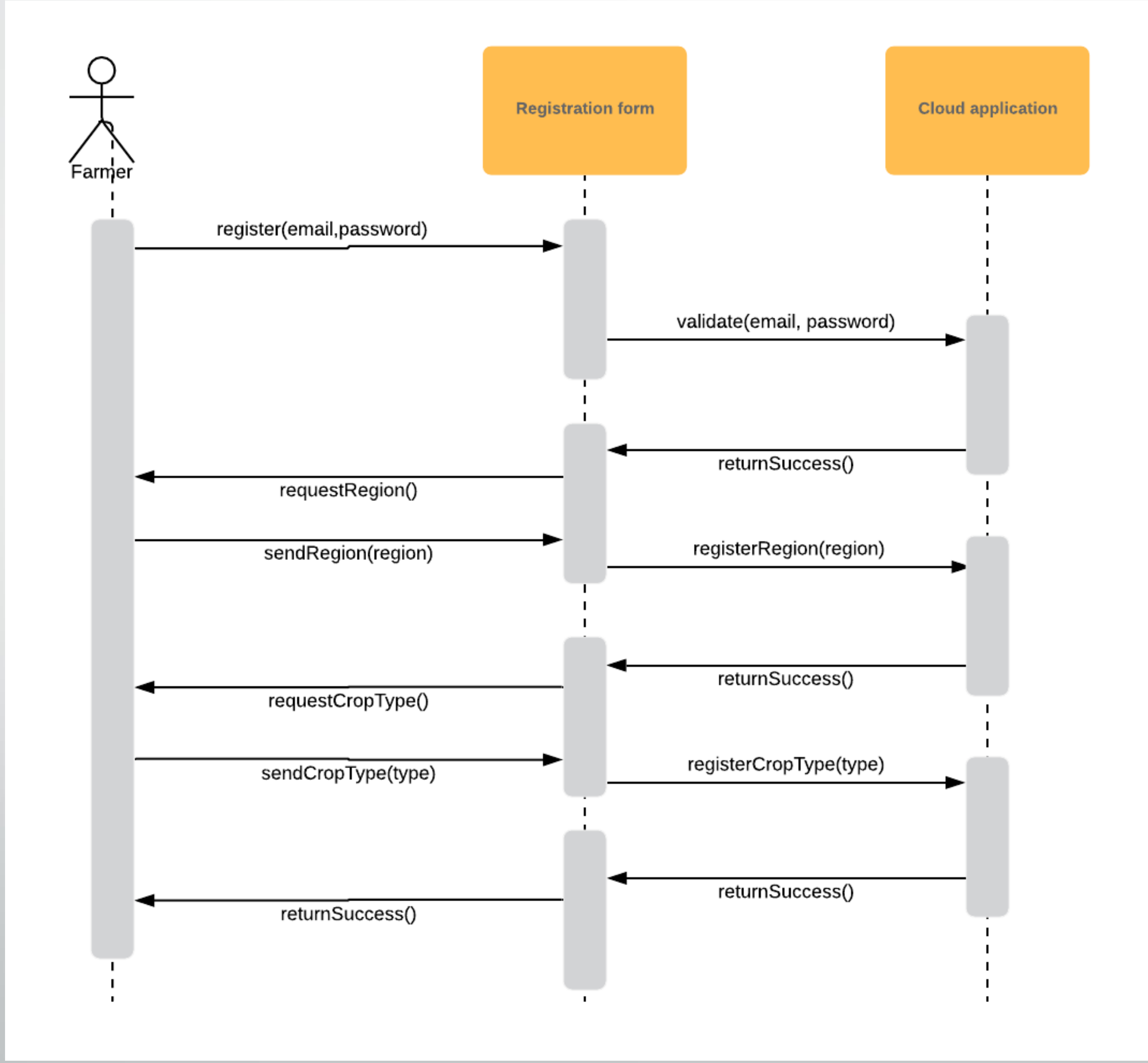
- Fiware: A testbed application to evaluate existing approaches (Careras et al 2016)
- Rainwater harvesting and pest repellent (Sukhdave et al 2016)
- Drip irrigation strategy based on Bluetooth (Hong et al 2016)
- Monitoring using Android application (Parameswaran et al 2016)
- Total control and monitor automatically irrigated system. Up to 90% savings. (Guttierrez et al 2014)
- Fuzzy logic algorithm. 22% savings in energy, 33% in water (Azaza et al 2015).

Communication Protocols

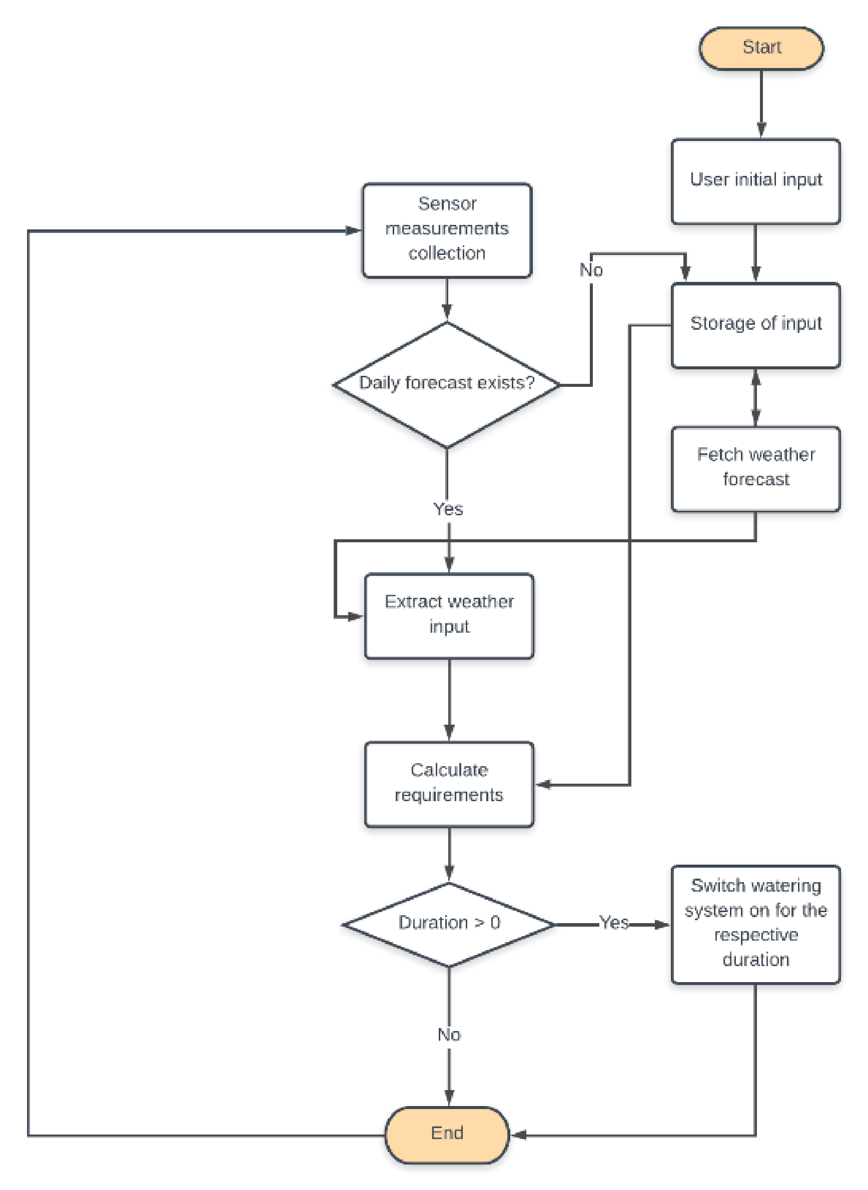
- Zigbee
- 6LowPAN
- Bluetooth
- WiFi
- Cellular networks

System Flows

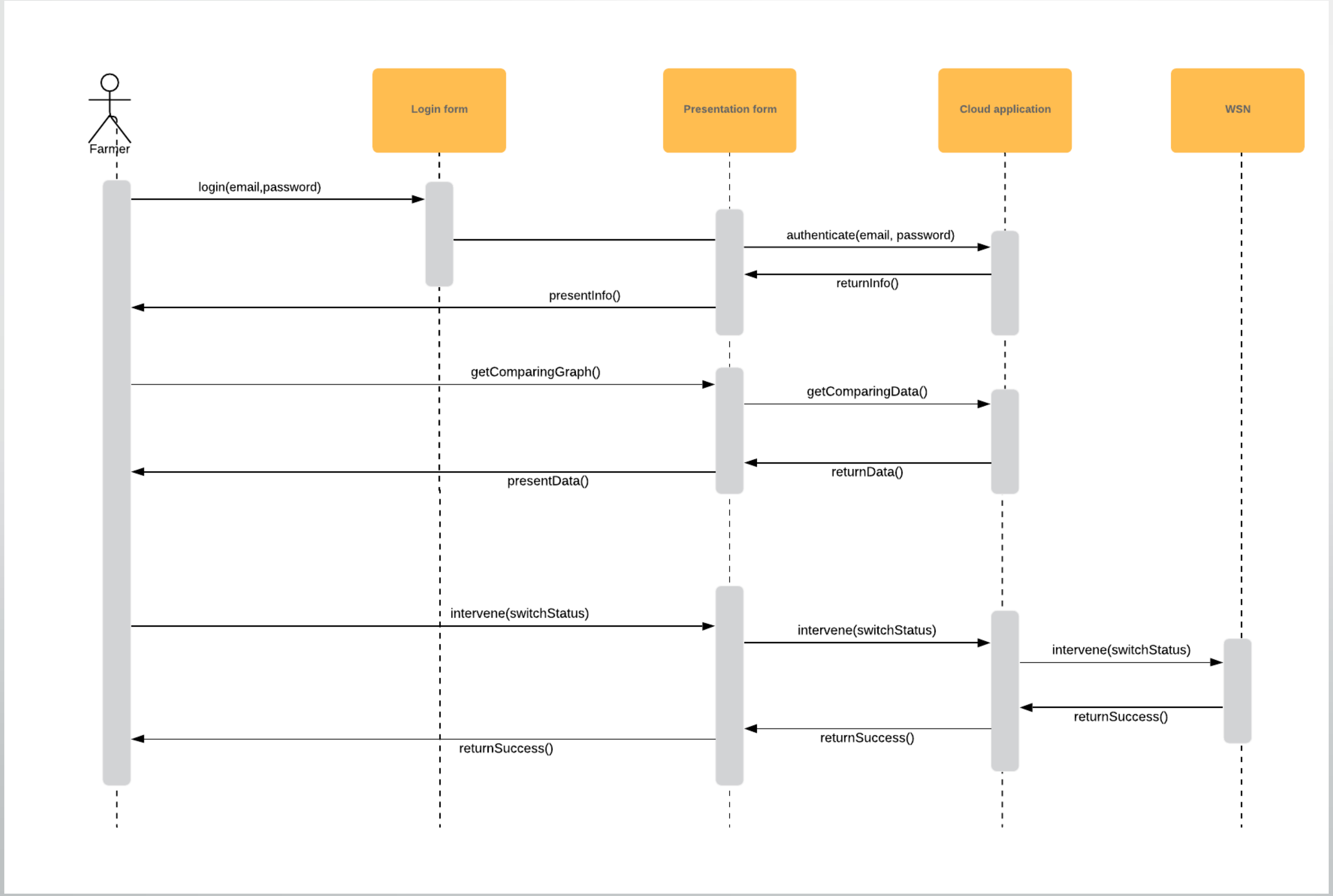
Registration flow



Automated flow



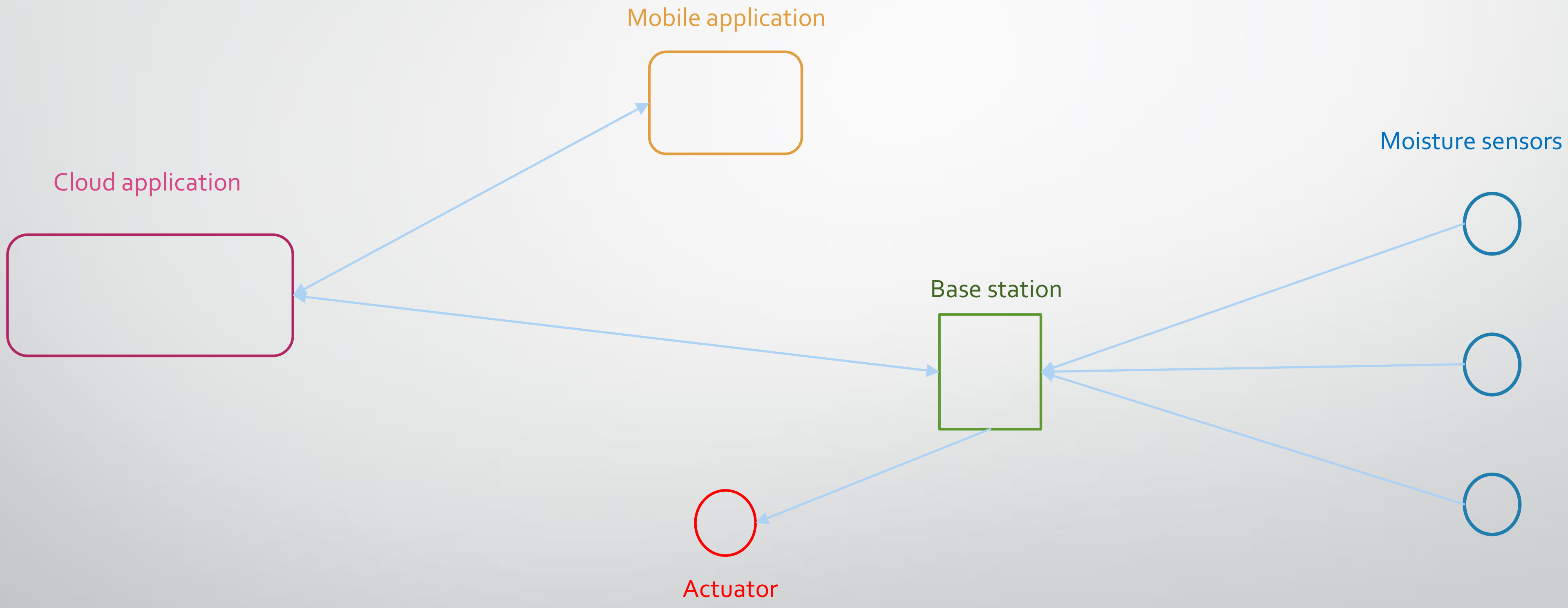
User flows


Farmer
Login form
Presentation form
Cloud application
WSN
login(email,password)
authenticate(email, password)
returnInfo()
presentInfo()
getComparingGraph()
getComparingData()
returnData()
presentData()
intervene(switchStatus)
intervene(switchStatus)
intervene(switchStatus)
returnSuccess()
returnSuccess()
returnSuccess()

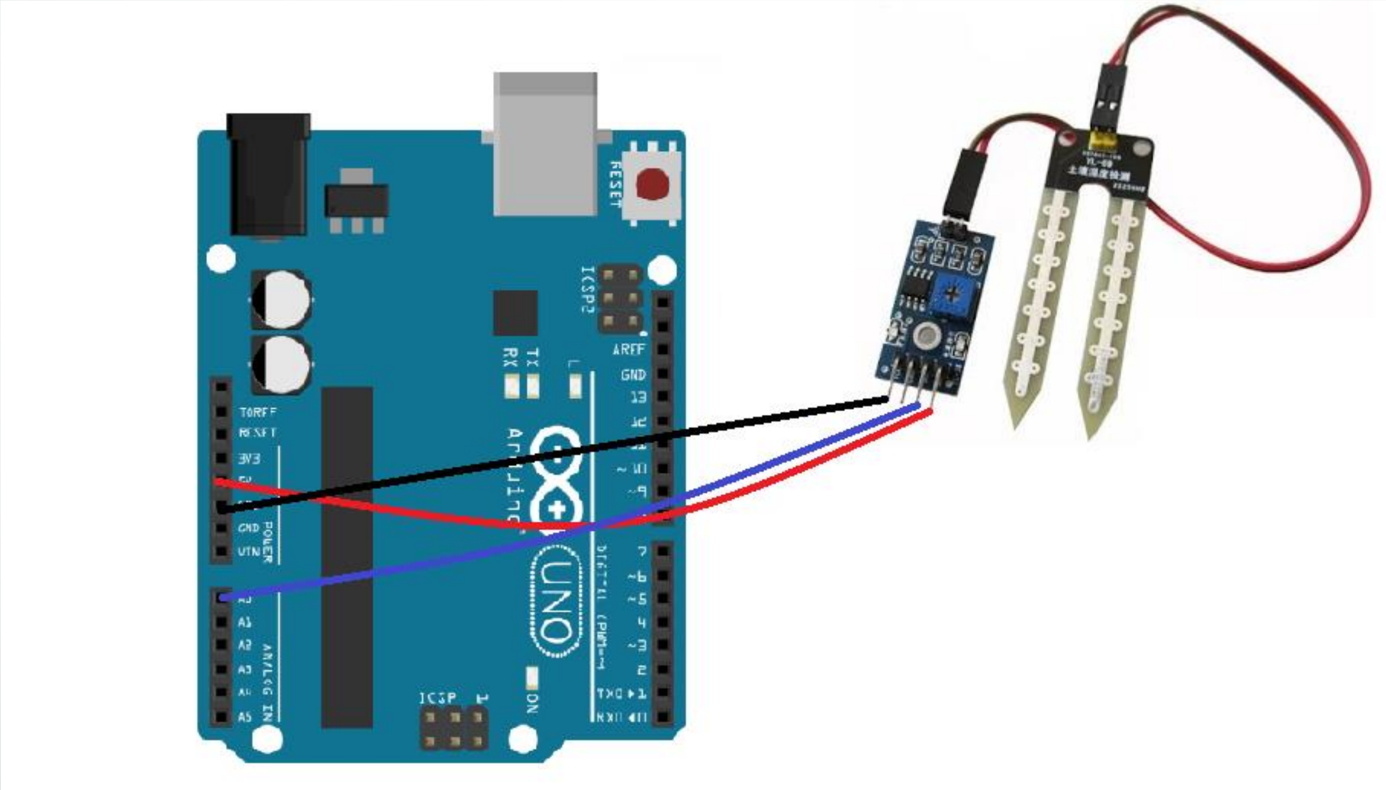
Components

- Moisture sensors
- Actuators
- Base station
- Cloud application
- Mobile application

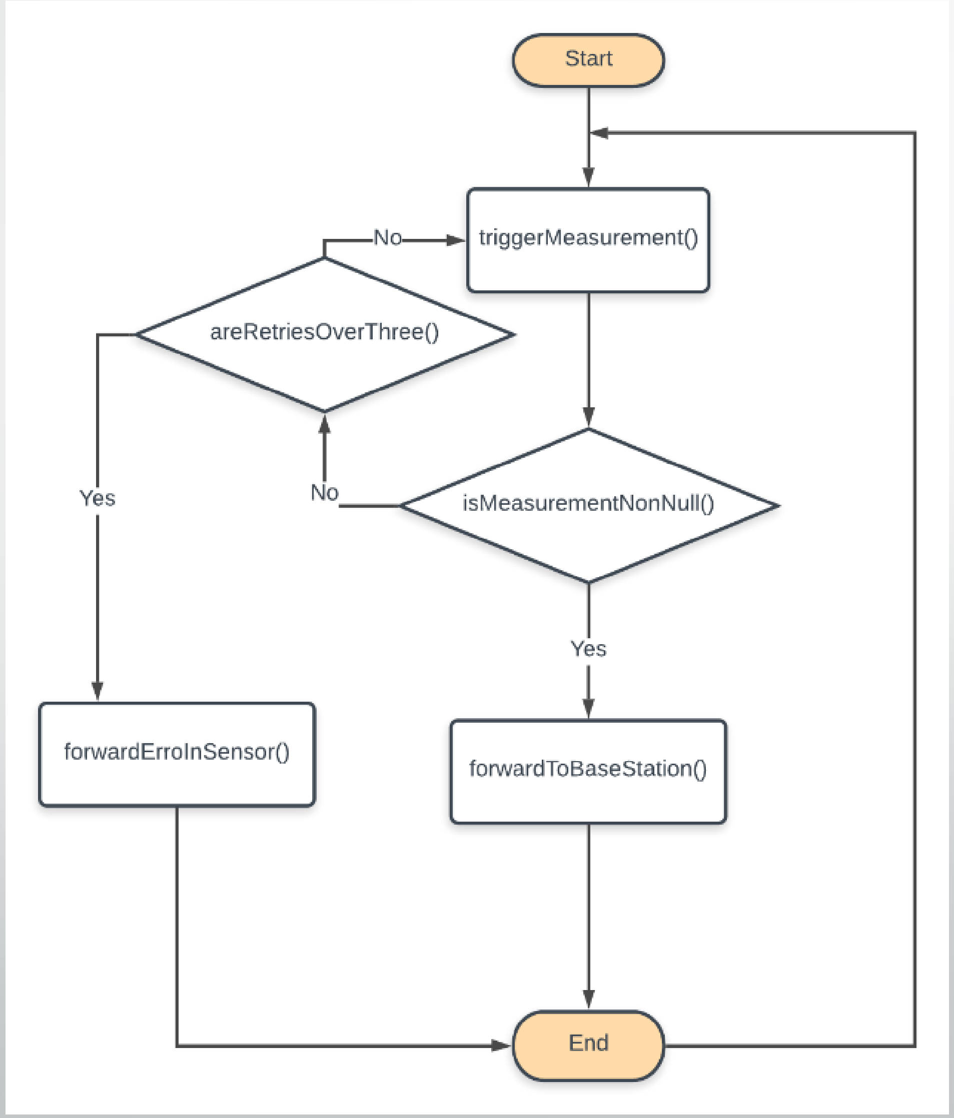
Components setup



Moisture sensor (1)



Moisture sensor (2)



Actuator

- Microcontroller
- Responsible for switching the functional state of the water pump
- Taking instructions from the base station

Base station (1)

- Orchestrator and delegator
- RMQ template
- Microcontroller vs Microprocessor

Base station (2)

- Observable pattern
- Energy efficiency
- Different communication protocol regarding the entity

Cloud application (1)

- RESTful/Websockets implementation
- Error handling
- Fuzzy vs Boolean

Cloud application (2)

- Current moisture level
- Upcoming weather condition
- Crop/soil type
- Upper/lower thressholds

Cloud application (3)

```java
public class WateringAlgorithm{

    public static int main(String []args){
        float currentMoistureLevel = args[0];
        float rainfallPercentage = args[1];
        float[] thresholds = args[2];
        private static final int CROP_TYPE = 1;
        private static final int SOIL_TYPE = 2;

        if (currentMoistureLevel >= thresholds[1])
            return 0;

        int rainfallCase = getRainfallCase(rainfallPercentage);

        float currentDiff = currentMoistureLevel - thresholds[0];
        float generalDiff = thresholds[1] - thresholds[0];

        float waterCoveragePercentage = (currentDiff/generaDiff)*100;

        long duration = getWateringDuration(waterCoveragePercentage);

        switch(rainfallCase) {
            case 1:
                return calibrateDuration(duration);
            case 3:
                return calibrateDuration((long)(duration * 0.8));
            case 2:
                return calibrateDuration((long)(duration * 0.3));
            default:
                return 0;
        }
    }

    private int getRainfallCase(float percentage) {
        if(percentage <= 0.33)
            return 0;

        if(percentage <= 0.5)
            return 1;

        if(percentage <= 0.8)
            return 2;

        return 3;
    }

    private long getWateringDuration(int waterCoveragePercentage) {
        //TODO usage of the two constants. for sake of ease will return a default number
        return 7200;
    }

    private long calibrateDuration(long duration) {
        if(duration <= 1800)
            return 0;

        return duration;
    }
}
```

Cloud application (4)

- Scheduled moisture request
  - Weather conditions affect the regularity
  - Smart handling leads to less required resources

```java
public class Scheduler{

    public static int main(String []args){
        int temperature = args[0];

        if(temperature <= 10)
            return 4;

        if(temperature <= 17)
            return 6;

        if(temperature <= 25)
            return 8;

        return 12;
    }
}
```

Cloud application (5)

- Communication with the mobile application
  - Keeping historical data
  - Presenting current state of the system
  - Register/Login
  - Actions over system’s functionality

Mobile application (1)

- User-oriented
- Monitoring and controlling
- Connects user to field on a one-to-one pattern

Mobile application (2)

- MVP pattern
  - Cleaner code
  - Extensible
  - Testable
  - Supports the Android MVVM Jetpack Component

Mobile application (3)

```
public Inteface MainView {
        void presentCurrentStatus(FieldConditionDto fieldCondition);

    void presentHistoricalData(ArrayList<HistoricalDto> historicalsIotDto
                        , ArrayList<HistoricalDto> historicalsTraditionalsDto);

    void wateringCommandSuccess();

    void wateringCommandFailed(int errorCode);
}
```

Mobile application (3)

```
public class MainPresenter {
    private MainView view;

    public MainPresenter(MainView view) {
        this.view = view;
    }

    public void getCurrentFieldCondition() {
        //make api call
        //if success
        view.presentCurrentStatus(someFieldCondition);
        //if error
        view.presentCurrentStatus(newFieldCondition);
    }

    public void getHistoricalData(int offset, int pageSize) {
        //make api call
        //if success
        view.presentHistoricalData(someHistoricalIotListDto,
                                   someHistoricalTraditionalListDto);
        //if error
        view.presentHistoricalData(newHistoricalIotListDto,
                                   newHistoricalTraditionalListDto);
    }

    public void waterTheField(long duration) {
        //make api call
        //if success
        view.wateringCommandSuccess();
        //if error
        view.wateringCommandFailed(someErrorCode);
    }
}
```

Conclusions

- 3 entities cooperation
- Main idea is typical
- Algorithm enhancement through moisture and weather prediction
- Diverse protocol selection
- Extensible separation of concerns

Future works

- System implementation
- Algorithm enhancement
- Correction and calibration (machine learning introduction)
- Big data introduction
- Delivery of data to other applications

Thank you for your attention