Sensor Network Playground Monitoring System

WSN Lab (WS 13/14)
Group number 1
Group Members :
Arjun Maryankandy Rajendran
Konstantin März
Mohammed Refat Chowdhury
Pratyush Agnihotri
Rick Nitsche

Project Guide: Eugen Berlin

TECHNISCHE UNIVERSITÄT DARMSTADT

1. Introduction

Plants are the backbone of all life on Earth and an essential resource for human well-being. We depend on them not only for food but also most importantantly the primary life-support. „oxygen“.
Since 1750, concentration of greenhouse gases has quickly increased. The coined Global climate change have had enormous impacts on plant diversity patterns in the past and are seen as having significant current impacts. **Wireless sensor networks** provides a good platform to keep a check on enviromental behavior. Using a system of sensor nodes one can capture, store, analyze, manage, and presents data which can be used in turn for monitoring the system.

1.1. Problem Definition

Goal of the project is to study the living conditions of two known plants, *Peperomia (Radiator Plant)* and *Kanlanchoe* and build a sensing system to monitor their requirements, as of to give them a sustainable environment. The monitoring requirements of the plants included air humidity, temperature, CO2 concentration, soil moisture and light exposure. For building them a nourished environment we took help of actuators such as fan, light and heater, as of to supply them fresh air, provide light at correct times and adjust the ambient temperature, respectively.

1.2. Plant living Condition

We indulged ourselves into a lot of research before we started the actual work on the project "Save the Plant!". One of which was to study growing conditions[1] for the provided plants *Peperomia* and *Kalanchoe*:-

Peperomia(Radiator Plant)	Kalanchoe
• **Air humidity:** Love warm humid conditions.	• **Air humidity:** Plenty of air flow around plant material.
• **CO2 concentration:** Between 40 ppm to 950 ppm.	• **CO2 concentration:** Between 40 ppm to 950 ppm.
• **Temperature:** Optimal temperature should be 24-28°C (75-82°F) and minimal temperature should be 13-15°C (55-59°F).	• **Temperature:** It should not be fall below 55ºF (12.8 ºC). It should be 10- 21°C (50-70 °F) in day and 7.2-18.3ºC(45-65 °F) in night.
• **Soil moisture:** Soil range should be sandy Loam to Loam. Water range should be normal and avoiding wetting the crown of the plant.	• **Soil moisture:** It can be damaged by over watering. Allow the soil to dry slightly between watering.
• **Light exposure:** There should be bright and mid-shade. Best cultivated in a light.	• **Light exposure:** There should be full sun light.

Table 1 : Plant living condition

1.3. Challenges

While building the monitoring system for plants, we faced several challenges which are quite common in every wireless sensor networks environment. We would like to state some of them, and also state our resolution we opted for in order to overcome these issues:-

Challenges faced	Description	Resolution opted
Energy-requirements	Lifetime of sensor nodes, energy consumption and	Since the monitoring system had to be of a longer lifetime, we decided to keep our sensor nodes active throughout, by connecting them to our centralized university test bed

	battery usage	TUDμNet.
Robustness and reliability	Node and communication failures	For the node failures, we decided to simply replace the sensor node (for eg. We replaced soldered soil mositure sensor node when it wasn't working fine). To handle communication failures, such as packet loss, we opted for *runicast communication protocol* and retransmit packets in case of communication failure.
Limited Resources	Memory and computation (handling floating point operations)	We used *unsigned integer*(data type) for most of our operations, however where required, rather than transmitting the values(*float* or *double*), we preferred to perform computation locally. (For instance we compared the measured sensor values to the threshold values - ideal conditions of plants, locally at the sensor node itself).
Calibration	Handling sensors actual output (raw values) wrt the living conditions of plants.	We calibrated the raw values received from sensors using the formulas in the datasheets(refer appendix), to adhere them to the living conditions of plants.

Table 2 : Challenges

2. Device/ driver information and plant living condtion

To use sensor in better way, it was required to study them both theoratically and pratically. We studied about the drivers and programmed them to know the behavior of them. We used following sensor and nodes [1] do develop a monitoring system :

Basic Node	:	XM1000 (aka TelosB) as basic node
On Board Sensors	:	Light sensors (Hamamatsu)
		Temperature and humidity sensors (SHT11)
Co2 Sensor	:	DS1000 as gas sensor (CO, CO2)
Soil Moisture Sensor	:	VH400 as soil moisture sensor
Base Station	:	Raspberry Pi as base station

To utilize the above sensors (like CO2 and soil moisture), we made use of the off the shelf available sensor nodes (XM1000), so that we could successfully be able to reprogram the drivers and make use of the sensors, as explained in the following Section[3].

3. Soldering of Sensors

3.1. XM1000 pin structure

The XM1000 includes an expansion connector that allows the access to a number of pins in the microcontroller[1].

3.2. Soldering Soil Moisture sensor

Soil moisture sensor (VH400) has three wire which has specific identity as Bare : Ground , Red : Power, Black output. We used xm1000 pin structure to solder moisture sensor[1]. We soldered PIN 1(DVCC) : Red wire, PIN 3(ADC0) : Black wire and PIN 9(GND) : Bare wire.

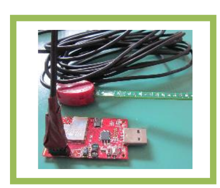

Figure 1 : Soldered moisture sensor

3.3. Soldering CO2 sensor

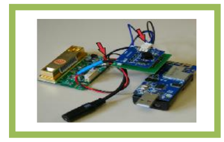

Figure 2 : Soldered CO2 sensor

For soldering Co2 sensor required xm1000 nodes, CO2 sensor (DS1000), PCB board and SH-300-DG. We soldered all these compontent together and we used DC 9V adapter jack to connect DS1000 Board and SH-300-DC for power supply[1].

4. Test Bed Set up

4.1. Design Architecture

To create better monitoring system, it is necessary to manage and place the sensor nodes and sensors, in such way that monitoring can be done in efficiently. Therefore we proposed our test bed setup[2].However, a common test bed setup used by us was also somewhat same (instead of two sinks) as defined in Figure 3. Here there are total 6 nodes. Two nodes are placed at left and right sides of the plants which are used as sinks. One CO2 sensor is placed between two plants and one on board sensor is placed at left side of plant. Two dedicated soil moisture sensors are used for left and right plants.

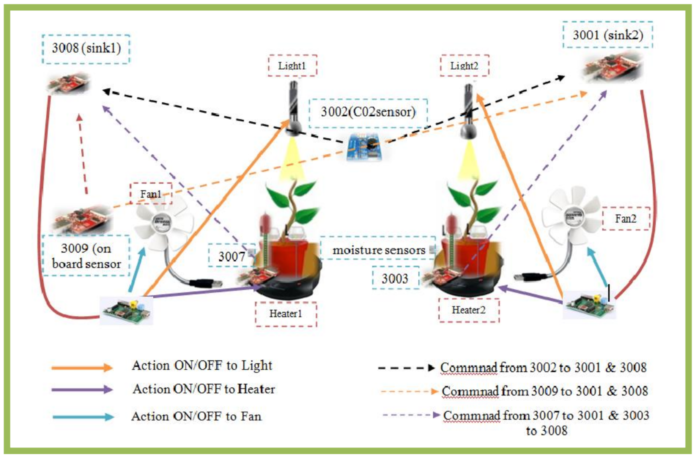

Figure 3 : Test Bed Setup

On board Sensor (light, humidity and temperature), Co2 and soil moisture sensors collect sensor values, caliberate them, compare them to the threshold, check the global state of actuators and send commands to sink for handling actuators. For minimizing the number of packets (commands) sent to the sink, we have saved and maintained the global state of actuator as described in following Section[6.2]. If the caliberated value is below /Ok/ higher than threshold value and check to the global actuator state passes then command is sent to sinks. After that, the commands are forwarded to Raspberry Pi base station by the sink node, correspondly to handle to state of the actuators.

4.2. Usage of actuator

In order to make use of actuators we corresponded to API provided by our course supervisors. The sinks send "printf"-commands[4] to the Raspberry Pi which in turn, controls the actuators. The API permits the control over three types of actuators via ON/OFF commands, *the fan*, *the lamp* and *the heating plate* placed under the plant. When the monitored value differs from the ideal values, a specific control command is sent to the base station (Raspberry Pi), e.g. PG:LIGHT ON to turn the light on [detailed information can be found in Actuators condition] [5].

5. Formulas for calculation

Monitoring requires converting raw data into human readable forms which we result in efficient sampling and visualization. We used various formulas to convert raw data into correspondind sensor values as raw data of Light into LUX[3], Humidity into percentage[7], Temperature into °C[7], Co2 into ppm[7], and soil moisture into VWC (Volumetric Water Content) [8]. We converter these value for monitoring and visualization purpose while we used raw data for comparison with threshold value and sending command correspondingly.

6. Principal of application

6.1. Introduction to solution

To support modularity each sensor type has its own definition (c-file). It leaves a possibility to add additional sensor types in the future. A separate header file (settings.h) is used to define constant values such as node ID's, thresholds, measuring periods etc. After waiting 180 seconds for initialization of the sinks and Raspberry Pi boards, the nodes begin to operate. After every 5 seconds, a measurement procedure is invoked depending upon the respective node-ID (sensor types). Each procedure reads the sensor value, calibrates it and compares it with the corresponding threshold. The respective value is then printed in readable units (e.g. LUX for the light). These print statements ("printf"s) are later used for the visualization of the measured values. To reduce energy consumption (minimize the number of packets sent), calculations are done on the sensor nodes, rather than sending measured values each time to the sink node.

6.2. Global actuator state

The commands to manipulate the actuators are only sent to the sink node if a measured value crosses a certain threshold and only when the *global state* of actuator changes. This state variable prevents sending unnecessary actuator commands when the measured value stays constant. This also provides a co-ordination mechanism between the various sensor types. For instance, since Fan was required for both monitoring temperature as well as CO2, thus, each time before turning ON/OFF the actuator, the sensor checks the global state of the actuator and then send command to the sink. This provides high efficiency in our code by significantly minimizing the number of packets sent to sink as well as co-ordination between the different sensors. Three state types are used: valueLOW, valueOK and valueHIGH.

6.3. Routing protocol used

Reliable single-hop unicast (runicast) primitive is used to prevent packet losses. The runicast primitive uses acknowledgments and retransmissions to ensure that packets are successfully received. The maximum number of retransmissions used in the project was 10. Different but short delays before sending were implemented for each sensor type in order to prevent occasional message loses due to simultaneous transmissions on the carrier.

7. Sampling and visualization

7.1. Sampling

After rigorous testing of our application in the mid-term evaluation, we finalized the following threshold values for our system as per the living conditions of plants[Table 1] :-

	Air humidity(%)	CO2 (ppm)	Temperature(ºC)	Soil Moisture (VWC)	Light Exposure (lux)
Peperomia	20 to 40	40 to 950	24 to 28	14.75 to 38.08	220
Kalanchoe	20 to 40	40 to 950	12.7 to 29	10 to19.8	250

Table 3 : Thresold value used

Now, after collecting raw sensor values, we compare the above threshold to these values, and send across the command (sample) to the sink. In order to reduce the sampling rate, we use the global actuator state mechanism as discussed above in Section [6.2].

7.2. Visualization

From the final evaluation of our code, we extracted the following visualization graphs of our sampling. Figure 4, 5, 6, 7, 8 and 9 depicts the visualization data of temperature, humidity, light, CO2, Soil Moisture Peperomia and Kalanchoe, respectively.

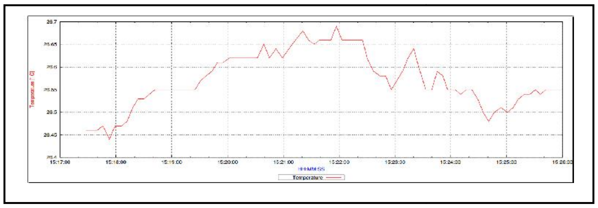

Figure 4 : Temperature

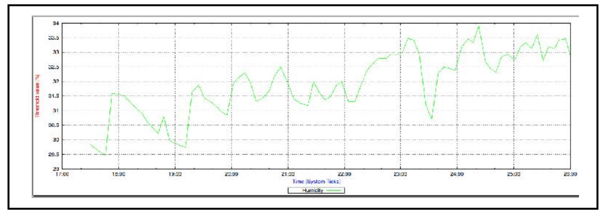

Figure 5 : Humidity

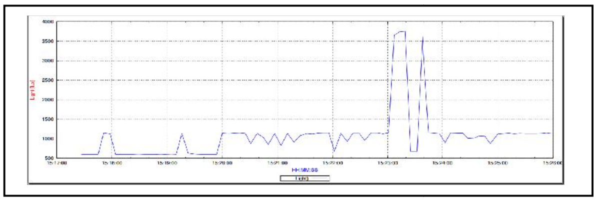

Figure 6 : Light

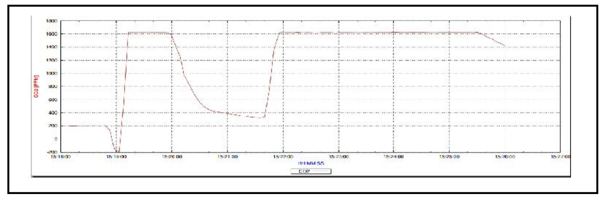

Figure 7 : CO2

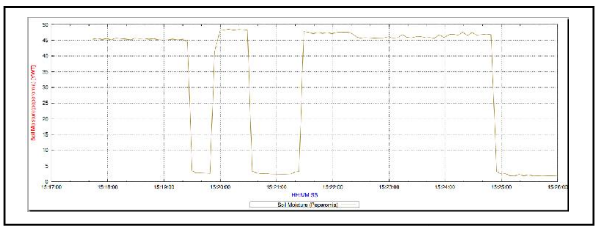

Figure 9 : Soil Moisture Peperomia

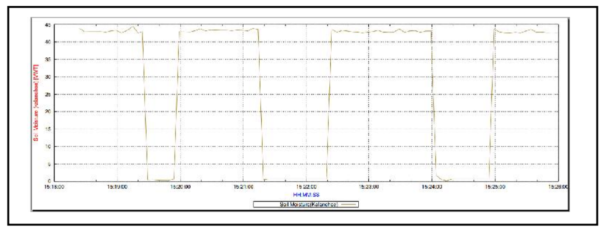

Figure 8 : Soil Moisture Kalanchoe

8. Testing of application

To make system more reliable and efficient, it is required to do testing of application. Testing does not only help in solving bugs but also make code more optimized and provide efficiency to application. We tested our application in various scenarios by applying different test cases[13] to check how actuators behave in these scenarios. Testing included checking the actuator functionality, run-time monitoring and system responsiveness by increasing/decreaseing soil moisture, lowering/highering light exposure, increasing/decreasing CO2 level w.r.t threshold values given in table and rigorous testing.

References

1. ..\Project_Documents\Plant_condition\
2. .. \Project_Documents\Test bed Step\
3. http://www.advanticsys.com/wiki/index.php?title=Hamamatsu%C2%AE_S1087_Series
4. ..\Project_Documents\Actuator_API\
5. ..\Project_Documents\Actuators_condition\
6. http://www.advanticsys.com/wiki/index.php?title=Sensirion%C2%AE_SHT11
7. ..\Project_Documents\Sensors_Datasheet\
8. http://www.vegetronix.com/Products/VH400/VH400-Piecewise-Curve.phtml
9. ..\Project_Documents\Test_cases\