Journal of Multimedia and Information System VOL. 3, NO. 3, September 2016(pp. 63-68): ISSN2383-7632(Online)
http://dx.doi.org/10.9717/JMIS.2016.3.3.63

Design of Data Center Environmental Monitoring System Based On Lower Hardware Cost

Lionel Nkenyereye[1], Jongwook Jang[1,*]

Abstract

Environmental downtime produces a significant cost to organizations and makes them unable to do business because what happens in the data center affects everyone. In addition, the amount of electrical energy consumed by data centers increases with the amount of computing power installed. Installation of physical Information Technology and facilities related to environmental concerns, such as monitoring temperature, humidity, power, flood, smoke, air flow, and room entry, is the most proactive way to reduce the unnecessary costs of expensive hardware replacement or unplanned downtime and decrease energy consumed by servers. In this paper, we present remote system for monitoring datacenter implementing using open-source hardware platforms; Arduino, Raspberry Pi, and the Gobetwino. The sensed data displayed through Arduino are transferred using Gobetwino to the nearest host server such as temperature, humidity and distance every time an object hitting another object or a person coming in entrance. The raspberry Pi records the sensed data at the remote location. The objective of collecting temperature and humidity data allows monitoring of the server's health and getting alerts if things start to go wrong. When the temperature hits 50ºC, the supervisor at remote headquarters would get a SMS, and then they would take appropriate actions to reduce electrical costs and preserve functionality of servers in data centers.

Key Words: Data Center, Environmental Monitoring System, Arduino, Gobetwino .

I. INTRODUCTION

There is a large number of prior efforts in building data center and equipping an organization with servers that are used to facilitate their employees to carry out their daily activity, store information, and data warehouse for reporting. As auxiliary data centers are located far from the primary datacenter, their computation tasks rely on proper functioning of the organization and monitoring them could prevent unexpected downtime of the primary data center.

Data centers hold scalable Information and Technology (IT) equipment-servers in order to ensure business continuity in many companies which operate in distributed manner for the reason of increasing quality-of-service of the system perceived by end-users. However, the amount of electrical energy consumed by data centers increase with the amount of computing power. In the same line, as the workload of computing grows in size, it seems that day after day a failure situation is close to happening, then increasing the energy waste even more and affecting the business continuity. Servers produce a lot of heat, figures vary but a rough figure for a large server is around 2500 British Thermal Unit (BTU) [1]. Normally, proper ventilation is good enough to maintain a safe and reliable environment, but if that fails then overheating can be a problem. The overheating causes IT heads to think what they should monitor.

Design and implementation of a monitoring and reporting strategy for metering the energy use and temperature of various area of IT equipment-server is essential for efficient maintenance, energy efficiency and planning further change. Therefore, installation of physical IT (Information Technology) and facilities environment monitors based on monitoring temperature, humidity, power, flood, smoke, air flow, room entry and other conditions is the most proactive way to reduce the unnecessary costs of expensive hardware replacement or unplanned downtime.

In this paper, we present datacenter environmental monitoring developed using open-source hardware platforms, Arduino [2], Raspberry Pi [3], and the Gobetwino [4]. Gobetwino is used to store sensor data to the nearest host server in the datacenter. The sensed data record in a file on the host server are temperature, humidity and distance every time an object hitting some object around the datacenter or person walks towards the entrance

Manuscript received August 31, 2016; Revised September 21, 2016; Accepted October 10, 2016 (ID No. JMIS-2016-0014)
Corresponding Author(*): Jong-Wook Jang, Dong-Eui University, 176 Eomgwangro Busanjin-Gu, Busan 614-714 Korea, +82-10-5600-0845, jwjang@deu.ac.kr
[1]Department of Computer Engineering, Dong-Eui University, Busan, Korea, lionelnk82@gmail.com,jwjang@deu.ac.kr

of the data centers. Such a design has the advantages of low cost, easy to build, and easy to maintain. The major obstacle is that Gobetwino runs on Windows only. Raspberry Pi is known for its reliability, flexibility, and scalability. A regular monitoring of temperature and humidity is straightforward because new equipment such as servers can easily cope with energy efficiency by increasing the temperature and decreasing the minimum relative humidity in data centers.

The rest of paper is organized as follows. Section 2 provides related work and essential background information on monitoring Data Center environmental conditions system. We describe hardware and software requirements in section 3. In section 4, we present the system design of the proposed system, followed by the deployment of our system in Section 5. Finally, our conclusion and future research directions are described in Section 6.

II. RELATED WORK AND BACKGROUND

A significant amount of research in the area of monitoring data centers from environmental conditions focuses on the implementation of such monitoring system based on the Wireless Sensor Networks (WSN) [5-9]. The architecture of deploying WSN in data center consists of a sensor in front and behind every server [9]. With the help of TelosB mote running on TinyOS 2.1, the collection of temperature data is made possible. However, one of the challenges of using that kind of TelosB is power consumption because it utilizes a battery. The power consumption is not the only challenge; the lower performance of wireless sensor network stands as a challenge because WSN topology changes dynamically most of the time.

Recently, the trends towards design made a transition from 8 bit to 32 bit (ARM Cortex M3) presages the next release of Arduino in the form of cheap practical computer (like Raspberry Pi) or a 64-bit processing [4]. The idea for considering Arduino as a single board for practical industrial applications leads us to implement the remote monitoring of data center environmental conditions using lower hardware cost with Arduino. Using Arduino, the challenges of energy consumption of sensor implemented in WSN for collecting data such as heat, humidity, airflow, smoke are solved

III. REQUIREMENTS

These requirements are required in order to implement the data center environmental monitoring system

3.1. Hardware Requirements

Hardware requirements are listed as follows:

- Arduino Uno: microcontroller board based on the ATmega 328P with 14 digital input/output pins, 6 analog inputs. It has a 16 MHZ quartz crystal, a USB connection, a power jack, an iCSP header and a reset button
- LED, along with a 470 ohms resistor
- Temperature sensor (at the Adafruit shop), the Analog Devices TMP36(-40 to 150C): this sensor has a very wide range and does not require a negative voltage to read sub-zero temperatures
- Digital output temperature and humidity sensor DHT11 basic temperature-humidity sensor: it uses a capacitive sensor and a thermistor to measure the surroundings air, and spits out a digital signal. Good for 20-80% humidity readings with 5% accuracy
- Sharp GP2YoA21YKOF IR Range sensor-10 cm to 80 cm: interfacing to most microcontrollers is straightforward: the single analog output can be connected to an analog-to-digital converter for taking distance measurements
- Mini Photocell Light sensor: photocell is basically little light sensor, which behave like resistors but the value of its resistor varies with the light level. To integrate it in our system, we will also need another resistor of 10K ohms
- Raspberry Pi 3: it includes an ARMv8 900MHz processor, videoCore IV GPU, and 1.2 GHz of RAM

3.2. Software Requirements

Software requirements for this projects are listed as follows:

- The Arduino Integrated Development Environment (or Arduino Software (IDE): contains a text editor for writing code. It connects to the Arduino hardware to upload programs and communicate with it.
- Raspbian: It is the official supported Raspberry Pi Operating System based on Debian
- LAMP (Linux Apache MySQL PHP): components (Apache 2, php 5, mysql-client, mysql-server, tomcat 6) required to run a dynamic HTML webpage and these are the first steps to build a LAMP webserver on a Raspberry Pi
- Gobetwino Arduino: a kind of a “generic proxy” for Arduino. It is a program running on a personal computer (windows only), that will act on the behalf of Arduino [4]
- Secure copy: securely transferring computer files

between a local host and a remote host. It is based on the secure shell.

IV. SYSTEM DESIGN

The system design is the process of transforming the user requirements and behavior into a purely programmatic oriented system by defining the architecture, components, modules, and data for a system in order to satisfy specified requirements [14]. The target system under implementation is settled into subsystems based on the analysis structure and the logical architecture.

4.1. System Architecture

Figure 1. represents the functional architecture of the system. Here temperature-TMP 36, humidity sensor DHT11, photocell light sensor and infrared range sensor are the inputs being used to generate the parameters which are the sensor data for the environmental monitoring system in the data center.

Gobetwino is considered as a “generic proxy” for Arduino. It’s a program running on a PC (Windows only), that will act on behalf of Arduino and do some of the tasks that Arduino can’t do on its own. So it works like a go between, hence the name [4]. Using defined command types you can create commands in Gobetwino that Arduino can request Gobetwino to execute. These commands are used to start a program on the PC, run a program until it finishes, and send data to any windows program from Arduino,

The implementation of the proposed environmental monitoring costs a lot less money. A simple Arduino can offer more data for lot less money. For the budget conscious you can get a cheap Arduino for less $ 5. To monitor the humidity, we use a DHT11 or DS18b20 [10] and a 4.7Ω resistor for $ 9.95. To read data from the sensor, we use an Arduino.

The Arduino can be connected over a USB serial cable for the current value to be extracted and reported back. Gobetwino is installed on a computer with Windows as operating system to collected sensor data from Arduino connected to it.

The objective of collecting temperature and humidity contributes to the business continuity by not only preserving functionality of servers in data centers but also monitoring server’s health and sending alerts if things start to go wrong. For instance, if the temperature recorded is about 50^{o} C., the supervisor wherever he (she) is would get a SMS, then take appropriate actions to reduce electrical costs and then heading off failover of active application, server, system, or hardware components.

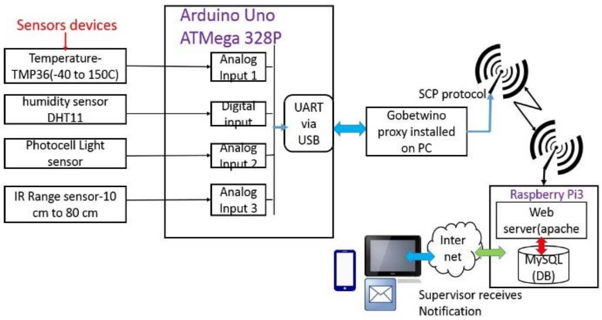


Fig. 1. Block Diagram of the system

4.2. System Flow Diagram

The readings of the sensors connected to the Arduino are programmed using the Arduino IDE. All Arduino sketches must have two functions: setup and loop. The setup function is executed only once when the Arduino board is first turned on, or when the reset button is pressed. The loop function is executed repeatedly (and indefinitely) after the setup function is finished. The process starts with initialization of program’s variables (inputs/outputs) of Arduino sketch and configuration of header and footer part that the Gobetwino command uses to log readings data from sensors in csv form on the personal computer via Universal Asynchronous Receiver Transmitter (UART) serial communication. Global variables defined in the header are shared by all functions in the sketch. After initialization of the setup function; inputs and outputs of analog and digital corresponds to the sensors devices listed early in hardware requirements are applied to the loop function (repeated continuously or body). The loop function executes the main tasks, waits one seconds and wait one more seconds.

Gobetwino is listening on the serial port [4], for “commands” coming from Arduino, and in response it performs something for Arduino and possibly sends command to Arduino. WinSCP is Secure Shell File Transfer Protocol (SFTP) client for Windows. Legacy Secure Copy (SCP) protocol is also supported [11]. Its main function is the safe copying of files between a local and a remote computer. Fig.2. and Fig.3 shows the flow chart for implementing the environmental monitoring system

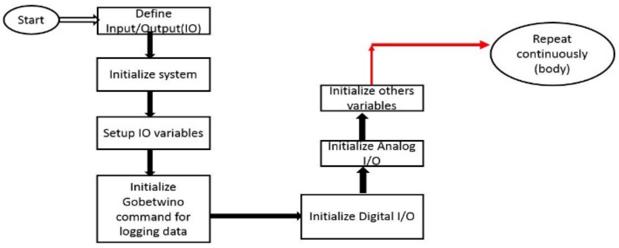


Fig. 2. Flow chart of the system

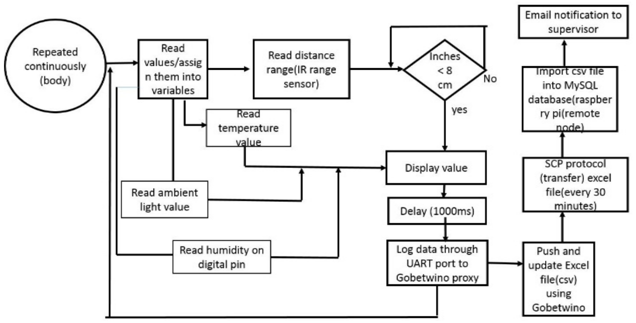

Repeated continuously (body)
Read values/assign them into variables
Read distance range(IR range sensor)
Inches < 8 cm
No
yes
Email notification to supervisor
Import csv file into MySQL database(raspberry pi(remote node)
Read temperature value
Display value
SCP protocol (transfer) excel file(every 30 minutes)
Read ambient light value
Delay (1000ms)
Read humidity on digital pin
Log data through UART port to Gobetwino proxy
Push and update Excel file(csv) using Gobetwino

Fig. 3. Flow chart of the system (cont...)

Building an environmental monitoring requires development and integration of many hardware and software components. In this paper, we develop a script that performs a transfer by copying the log sensor file to the remote server on Raspberry Pi. The supervisor team should receive a notification email directly at the primary working place. We create one LGFIL type command with the name SENSORLOG that logs sensor readings from Arduino to a file.

V. SYSTEM IMPLEMENTATION

Implementation of the data centers environmental monitoring system succeeds because of the design of the system architecture described in section 4. This section deals with the detailed description of how functional requirements of the system are performed.

The whole Implementation part of this project is divided into three different levels:

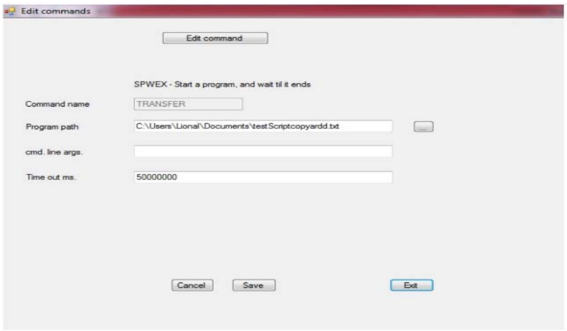
- Design schematic circuit
- Configuration of Gobetwino
- Application development

5.1. Design schematic circuit

With the system definition in place and different sensors required, it is now time to start designing the schematic circuit using the Fritzing [12]. Fritzing is a free open-source design suite that works on Windows, Mac, and Linux. It has a simple and realistic interface that makes designing circuits intuitive. The idea is to allow us to just copy the breadboard design, using images of the used parts and then easily move to a Printed Circuit Board (PCB) design. Fritzing highly supports prototyping using the Arduino, as the open hardware board is one of the first parts implemented.

5.2. Configuration of Gobetwino

There are essential commands that should be programmed on Gobetwino in order to save the log monitoring on the host server. Three commands have been developed with Gobetwino. As stated earlier Gobetwino defines a set of command types that can be used as templates to create commands. Each command type has a name that is an abbreviation of its function. The difference between a command type and a command can best be explained with an example. The Fig .4. shows the log to file command type in the project (TRANSFER) defines a command type that has parameters, program path, Timestamp. We create one LGFIL type command with the name SENSORLOG that logs sensor readings from Arduino to a file called "testlog.cvs." So the command type defines an action and a set of parameters that each has a data type.



SPWEX - Start a program, and wait til it ends

Command name: TRANSFER

Program path: C:\Users\Lional\Documents\testScriptcopyardd.txt

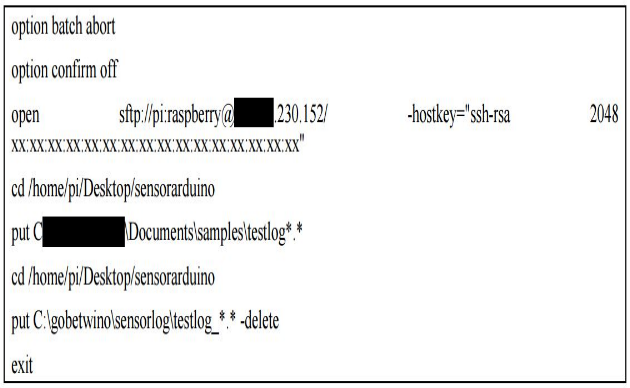
cmd. line args.:

Time out ms.: 50000000

Fig. 4. Example of Gobetwino command allows transferring data from Arduino to host server

5.3. Application Development

The application code would allow to implement the functional requirements specified in the system flow diagram. The code must co-exist with log sensors data (temperature, humidity, ambient light value and distance range of an objects) transferred to the host computer through Gobetwino. These sensed data are firstly recorded in a specific path as configured on Gobetwino. After that, they are copied safely to a Raspberry Pi where a various tasks are executed in order to get an accurate status of data centers environmental monitoring system. In order to copy the sensed data from the personal computer to the Raspberry Pi, we will need to understand how WinSCP operates and SSH (secure shell) protocol for operating data transfer over an unsecured network.



```
option batch abort
option confirm off
open        sftp://pi:raspberry@[redacted].230.152/        -hostkey="ssh-rsa        2048
xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx"
cd /home/pi/Desktop/sensorarduino
put C[redacted]\Documents\samples\testlog*.*
cd /home/pi/Desktop/sensorarduino
put C:\gobetwino\sensorlog\testlog_*.* -delete
exit
```

Fig. 5. Script code transferring log data from host server

5.3.1. Secure transferring log data from host server to remote server Raspberry Pi

WinSCP is an open source free SFTP client and FTP client for Windows. Legacy SCP protocol is also supported [11]. Its main function is safe copying of files between a local and a remote computer. In this project, we develop a script that can set up as a task scheduled to run on a regular interval. The script performs a transfer by copying the log sensor file to the remote server on raspberry Pi. Fig .5. shows in detail the script code for transferring log data from host server to raspberry Pi where the additional controls tasks are performed. An example of control task consists of controlling the limit temperature on where the supervisor of the data must be informed via an email notification.

5.3.2. Application development performed on Raspberry Pi as a remote server

5.3.2.1. Shell file to load data from local directory to MySQL Database

We design a database that will contain our static and dynamic data as follows: one table named “sensorupload “where we store the temperature logged .It is classified by date in timestamp format. We write out a shell script that will be executed as a job every 5 minutes. The script consists to read the text file from local path and loads the data into the table “sensorupload” after deleting the previously data inserted.

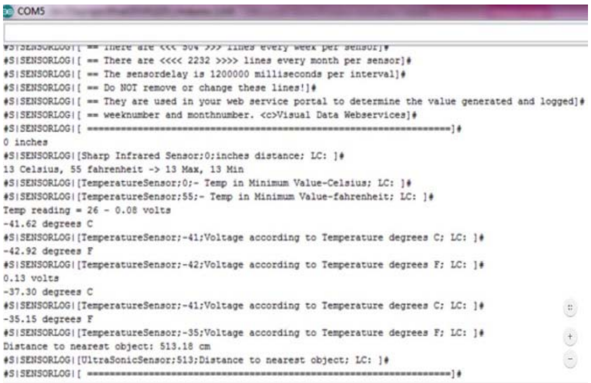
5.3.2.2. Design email notification of environmental monitoring using Python

Python is a powerful programming language that's easy to use together with Raspberry Pi lets you connect your project to the real world. The easiest introduction to Python is through IDLE. The detail of the script code consists of sending email within attached the log sensor data. This log data sensor data are transferred from the data center where the environmental monitoring system is deployed to the remote server using secure tool WinSCP. Beside python script, we write out a shell script to execute automatically the script as a predefined function every 5 minutes using “crontab” [13].

5.3.2. Results

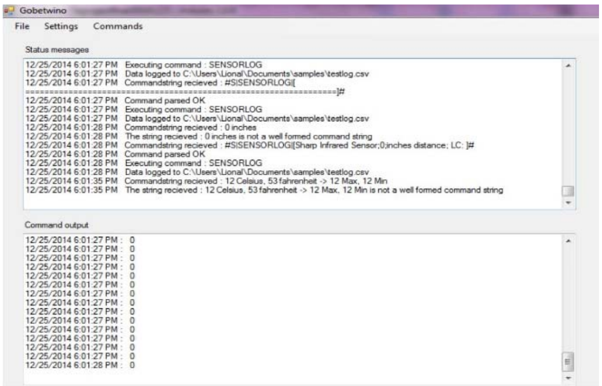
We deployed the project at a data center located in the Department of Computer Engineering for development and testing purposes. The deployment includes an Arduino board within temperature sensor, infrared, light sensor, and a host sever on which we have installed Gobetwino software. The Arduino is connected to the host server and the port COM 5 is used to ensure communication between Arduino and host server through Gobetwino that acts as a proxy. Fig.6. shows a real-time monitoring on the Arduino board. The log data is saved on Host server in csv format. The overview of the file after being saved on the specified path on the Gobetwino interface. On the host server, WinSCP command scheduled as a task is running. Fig.7. shows the log of Gobetwino commands executed on the Arduino when it is logging reading sensors data in csv format.

On the remote server which in this report is implemented on raspberry board, the LAMP (Linux Apache MySQL PHP) web service model. After uploading the csv file that holds monitoring data, the log data file is sent to the e-mail of the supervisor team as shown in Figure 8.



```
#S|SENSORLOG|[ == There are <<< 504 >>> lines every week per sensor]#
#S|SENSORLOG|[ == There are <<<< 2232 >>>> lines every month per sensor]#
#S|SENSORLOG|[ == The sensordelay is 1200000 milliseconds per interval]#
#S|SENSORLOG|[ == Do NOT remove or change these lines!]#
#S|SENSORLOG|[ == They are used in your web service portal to determine the value generated and logged]#
#S|SENSORLOG|[ == weeknumber and monthnumber. <c>Visual Data Webservices]#
#S|SENSORLOG|[ =================================================================]#
0 inches
#S|SENSORLOG|[Sharp Infrared Sensor:0;inches distance: LC: ]#
13 Celsius, 55 fahrenheit -> 13 Max, 13 Min
#S|SENSORLOG|[TemperatureSensor;0;- Temp in Minimum Value-Celsius; LC: ]#
#S|SENSORLOG|[TemperatureSensor;55;- Temp in Minimum Value-fahrenheit; LC: ]#
Temp reading = 26 - 0.08 volts
-41.62 degrees C
#S|SENSORLOG|[TemperatureSensor;-41;Voltage according to Temperature degrees C; LC: ]#
-42.92 degrees F
#S|SENSORLOG|[TemperatureSensor;-42;Voltage according to Temperature degrees F; LC: ]#
0.13 volts
-37.30 degrees C
#S|SENSORLOG|[TemperatureSensor;-41;Voltage according to Temperature degrees C; LC: ]#
-35.15 degrees F
#S|SENSORLOG|[TemperatureSensor;-35;Voltage according to Temperature degrees F; LC: ]#
Distance to nearest object: 513.18 cm
#S|SENSORLOG|[UltraSonicSensor;513;Distance to nearest object; LC: ]#
#S|SENSORLOG|[ =================================================================]#
```

Fig.6. Sensor’s temperature read from Arduino and logged on the computer which hosts Gobetwino application



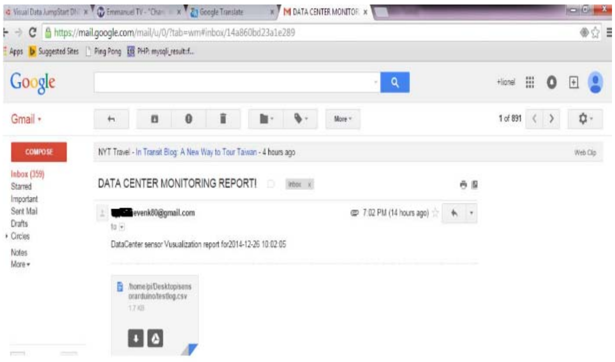
Status messages

```
12/25/2014 6:01:27 PM   Executing command : SENSORLOG
12/25/2014 6:01:27 PM   Data logged to C:\Users\Lional\Documents\samples\testlog.csv
12/25/2014 6:01:27 PM   Commandstring recieved : #S|SENSORLOG|[
=================================================================]#
12/25/2014 6:01:27 PM   Command parsed OK
12/25/2014 6:01:27 PM   Executing command : SENSORLOG
12/25/2014 6:01:27 PM   Data logged to C:\Users\Lional\Documents\samples\testlog.csv
12/25/2014 6:01:28 PM   Commandstring recieved : 0 inches
12/25/2014 6:01:28 PM   The string recieved : 0 inches is not a well formed command string
12/25/2014 6:01:28 PM   Commandstring recieved : #S|SENSORLOG|[Sharp Infrared Sensor;0;inches distance; LC: ]#
12/25/2014 6:01:28 PM   Command parsed OK
12/25/2014 6:01:28 PM   Executing command : SENSORLOG
12/25/2014 6:01:28 PM   Data logged to C:\Users\Lional\Documents\samples\testlog.csv
12/25/2014 6:01:35 PM   Commandstring recieved : 12 Celsius, 53 fahrenheit -> 12 Max, 12 Min
12/25/2014 6:01:35 PM   The string recieved : 12 Celsius, 53 fahrenheit -> 12 Max, 12 Min is not a well formed command string
```

Command output

```
12/25/2014 6:01:27 PM :  0
12/25/2014 6:01:27 PM :  0
12/25/2014 6:01:27 PM :  0
12/25/2014 6:01:27 PM :  0
12/25/2014 6:01:27 PM :  0
12/25/2014 6:01:27 PM :  0
12/25/2014 6:01:27 PM :  0
12/25/2014 6:01:27 PM :  0
12/25/2014 6:01:27 PM :  0
12/25/2014 6:01:27 PM :  0
12/25/2014 6:01:27 PM :  0
12/25/2014 6:01:27 PM :  0
12/25/2014 6:01:27 PM :  0
12/25/2014 6:01:27 PM :  0
12/25/2014 6:01:28 PM :  0
```

Fig. 7. The log of Gobetwino commands executed on the Arduino when it is logging reading sensors data in csv format



NYT Travel - In Transit Blog: A New Way to Tour Taiwan - 4 hours ago

DATA CENTER MONITORING REPORT!

[redacted]evenk80@gmail.com 7:02 PM (14 hours ago)
to

DataCenter sensor Vusualization report for2014-12-26 10:02:05

/home/pi/Desktop/sensorarduino/testlog.csv
1.7 KB

Fig. 8. Detail email within attached log file sent to supervisor

VI. CONCLUSION

The Data Center environmental monitoring represents a step forward towards addressing awareness monitoring of remotely located data centers located by receiving a notification by email. With the proposed monitoring data centers environmental system, organizations can effortlessly install the system with no additional wiring installation or maintenance cost. Additionally, our proposed system is capable of viewing the system logs and status via Internet in real time. Simply monitoring this data is not enough. As future work, the system design presented in this project can be expanded in a number of different aspects. For instance, additional user friend web monitoring portal allowing supervisors to monitor sensor data in real time through charts which can be designed using HTML5

Acknowledgement

This paper was supported by Dong-Eui University research in 2016, National Research Foundation-BTP-2015DB0160001), Training business area of leading new business (201601700001) in National Research Foundation (National Research Foundation of Korea) in 2016.

References

[1] N. Rasmussen, "Calculating Total Cooling Requirements for Data Centers," White Paper 25 Rev3, Schneider Electric, pp. 1-7, 2011

[2] Arduino Uno, ww.arduino.cc/en/Main/ArduinoBoardUno, 2016.

[3] Raspberry Pi, www.raspberrypi.org , 2016

[4] Gobetwino, http://mikmo.dk/gobetwino.html, 2016

[5] T. Evans, "Humidification Strategies for Data Centers and Network Rooms," White Paper no. 58 rev. 1, American Power Conversion by Schneider Electric, pp. 1-14, 2008.

[6] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, "Wireless sensor Networks for Habitat Monitoring," *in Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, pp. 88-97, 2002.

[7] C. Gui, and P. Mohapatra, "Power Conservation and Quality of Surveillance in Target Tracking Sensor Networks," *in Proceedings of the 10th Annual International Conference on Mobile Computing and Networking*, pp. 129-143, 2004.

[8] T. He, S. Krishnamurthy, J.A. Stankovic, T. Abdelzaher, L. Luo, T. Yan, R. Stoleru, G. GU, J. Hui and B. Krogh, "Energy-efficient surveillance System using Wireless Sensor Networks," *in Proceedings of the 2nd of the 2nd International Conference on Mobile systems*, pp. 270-283, 2004.

[9] S. Choochaisri, V. Niennattrakul, C. Intanagonwiwat and A.C. Ratanamahatana, "SENVM: Server Environment Monitoring and Controlling system for a small Data Center Using Wireless Sensor network," *presented at the 1st International Computer Science and Engineering Conference*, pp. 23-28, 2010.

[10] Waterproof DS18B20 Digital temperature sensor + extras, https://www.adafruit.com/product/381, 2016

[11] WinSCP. Free SFTP, SCP, and FTP client for Windows, https://winscp.net/eng/docs/lang:fr, 2016

[12] Fritzing, http://fritzing.org/home/, 2016

[13] Crontab-Quick Reference, http://www.adminschoice. com/crontab-quick-reference, 2016

[14] J. Waldo, *On System Design*, In an Essay Series Published by Sun Labs, Perspectives 2006-6, pp. 1-40, 2006.

Authors

Lionel Nkenyereye is currently a PhD student in the Department of Computer Engineering, Dong-Eui University, Korea. His research interests lie in embedded system/design and Internet of Things.

JongWook Jang is a professor in th e Department of Computer Engineerin g, Dong-Eui University, Korea, He did his PhD in Computer Engineering at Pusan National University, Korea (199 5)