Automatic Watering Plant Application Based on Android and Web Using REST Protocol

Restu Isjaka Purwandana, Fairuz Azmi, Agung Nugroho Jati
Robotic and Embedded System Technology Research Group, School of Electrical Engineering, Telkom University, Bandung, Indonesia.
worldliner@telkomuniversity.ac.id

***Abstract*—Gardening in private lawn has become a common activity among the metropolitan citizen, and they have diverse goals of planting, ranging from home decoration, income generation or personal consumption. However, this activity may lead to a waste effort if they do not treat the plants consistently, such as forgetting to water the plant. This project provides the solution to control and monitor the activity of watering the plants and water circulation in an aquarium on a system called APA (AQUAPONIC AUTOMATIC). Utilizing the REST protocol, based on android and web services with case studies on the private lawn, this system performs a remote automatic watering by making use of the weather information and provides a decision as well as notification to the controller when it is time to water the plant. The target of this project is to create a system that monitors and controls the watering plant system. Using android and web services, it run automatically by considering weather forecast information and the implementation of the system ensures the plants to grow.**

***Index Terms*—Cloud Service, Weather Service, REST Protocol, Cultivation of Crops.**

I. INTRODUCTION

Plant conservation and optimization are critical issues in large countries [1]. Indonesia is one of the large countries that has agriculture potential with more than 40% of the country's agricultural land are under arid or semi-arid climatic (Gamo, 1999) [2]. The environment condition has become a constraint on agrarian production [2]. However, demand on the metropolitan citizen's lifestyle to gardening and cropping on personal lawn for a individual or commercial purpose is very substantial [3], leading to the important role of handling lawn's biota [3].

Given the high trend of planting that prevails on the metropolitan citizen, it is becoming important to encourage them to conserve and optimize the planting activity as optimum plant handling would emerge as an economic value [3]. Therefore, a cloud service system that controls, monitors, and gives information of the plant condition is required to achieve a favorable result with a minimal resource [4].

Considering that plant needs water and nutrition, an automatic watering system that uses humidity and moisture calculation is needed [5]. A specific time is needed to water the plant. RTC (Real-Time Clock Time Clock) could be used to watering [6], but it is inefficient and dangerous for climate change. Alternatively, a water pump and moisture sensor can be used to compensate the watering problem [7], although it is still uncontrollable. Thus, a mobile controlling system is one of the solutions to monitor and address the efficiency issues [8]. Furthermore, a system that provides control-ability [9] can produce a system that runs automatically. However, it tends to be more flexible, versatile for climatic change and relay on everyday devices.

Motivated by the problem and previous researches, APA system has been designed. Runningr on the web and mobile with cloud integration, the system functions to monitor the watering of plants. Adopting IoT (Internet of Things) concept, the APA's devices can understand commands that are transmitted via APA's gateway by utilizing REST (Representational State Transfer) protocol [10]. APA also understands today's weather and makes an automatic watering calculation based on given information from APA's weather service. The weather service on APA uses forecasting variables from the Open Weather Map and values from weatherboard, which are the temperature (Celsius), relative humidity (%), and pressures (hectoPascal).

Temperature, humidity, and pressure are climatic variables, which are affecting each other (Givoni B. 1976) [11]. The air temperature variation affects water saturation and evaporation, which leads to changes in air humidity. The differences between temperatures on different locations can cause pressure alteration. This can be explained on Figure 1. APA uses the concept of climatic variable relationship to trigger automatic watering, based on weather forecast.

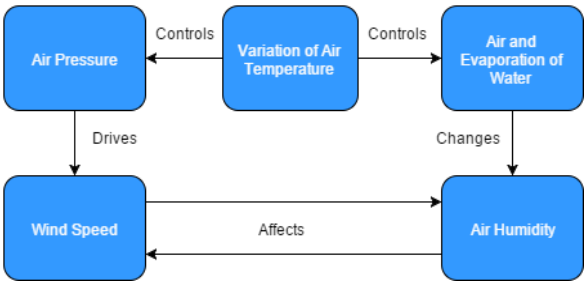

Figure 1: Flow diagram of climatic variables relationship

Further, APA system is implemented on massive agricultural issues. The self-powered device helps cultivators to cultivate their lands, conserve water and foresee climatic issues like drought climate. Hence, the major objectives of these present work are:

1. APA system supports automatic watering by calculating and parsing climatic variables.
2. APA Client (software) brings responsive web and friendly android application to manually water plant and/or monitors plant.

3. APA system adopts IoT concept, using Intel Galileo Gen 2 as gateway and devices which are connected to the internet via local area network.
4. APA creates a scalable, yet flexible system. Hence, it can be implemented on a large-scaled field.

II. System Design

This system consists of four parts, including the device, gateway, cloud, and client as shown in Figure 2. The device and gateway use IoT device, Intel Galileo Gen 2, while the cloud acts as data and command provider and the client as a command and monitor instrument. The device is integrated with DC 12V solenoid valve to control the water gate and the IP camera is used to compensate the monitor's parameter. Many of devices are connected to a gateway, which provides the connection to the Internet. Gateway always listens to the cloud, and forward the information to to the device as well as provide information to clients through the cloud. Clients give commands through a cloud and receive information from the gateways. Thus, one client can have many gateways and one gateway can provide connection to many devices. REST protocol compensates communications between parts. The device and gateway are programmed to run based on Node.js, developed through Intel XDK.

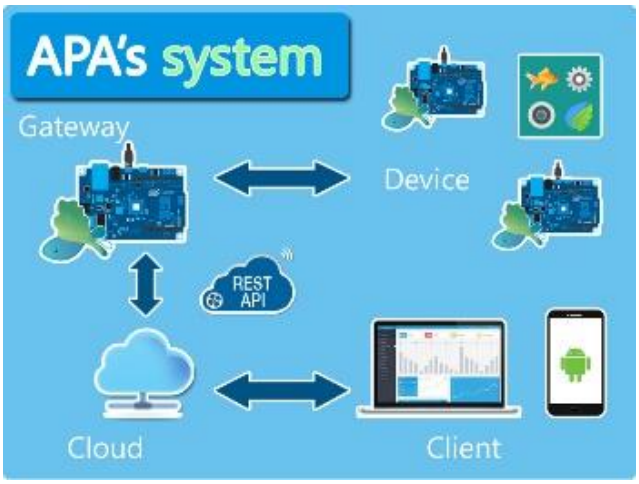

Figure 2: Architecture design of APA System

A. Cloud Architecture

Cloud provides weather information of the selected gateway. Basically acts as a server, it provides information to the client and the gateway itself. REST protocol utilizes data to be encoded on JSON (Javascript Object Notation) [12] which is more versatile than the others [13] as a communication medium through HTTP connection. HTTP connection has four general methods: GET, POST, PUT, and DELETE. In this case, GET and POST are used to communicate. Thus, data can be passed between client and gateway through the cloud.


```
{"coord":{"lon":107.62,"lat":-6.9},"weather":
[{"id":803,"main":"Clouds","description":"broken
clouds","icon":"04n"}],"base":"stations","main":
{"temp":290.954,"pressure":924,"humidity":94,"temp_m
in":290.954,"temp_max":290.954,"sea_level":1023.24,"
grnd_level":924},"wind":
{"speed":1.03,"deg":107.002},"clouds":
{"all":56},"dt":1447613818,"sys":
{"message":0.0059,"country":"ID","sunrise":144753967
3,"sunset":1447584424},"id":1650357,"name":"Bandung"
,"cod":200}
```


APA Cloud service makes use of JSON response and seamlessly updates weather data on each gateway every 30 minutes, by using CronJob. Hence, it re-provides the data on APA Cloud service to be simpler. Figure 4 shows the weather data of the three gateways on two different cities, Bandung and Sorong, which are provided by APA Cloud service.


```
{"status":true,"gateway":
[{"id_gateway":"1","latitude":"-6.91746","longitude"
:"107.61912","status":"online","weather":
{"local":"Bandung,ID","id_weather":"803","dt":"14476
04748","sunrise":"1447539673","sunset":"1447584422",
"tempK":"296.793","tempC":23.643,"description":"brok
en clouds","humidity":"94","pressure":"983.16"}},
{"id_gateway":"2","latitude":"-0.91203","longitude":
"131.88406","status":"online","weather":
{"local":"Sorong,ID","id_weather":"803","dt":"144760
4750","sunrise":"1447534490","sunset":"1447578258","
tempK":"300.918","tempC":27.768,"description":"broke
n
clouds","humidity":"100","pressure":"1019.48"}}],"co
unt":2}
```


Figure 4: JSON response from APA cloud service

Based on the cloud's weather data, the gateway calculates it with values given by weatherboard, using the fuzzy logic algorithm on the same variables. This results in the probability of watering the plant.

Actions based on the command from the client rely from other APIs (Application Program Interfaces) on the cloud service. These APIs also utilize REST concept to do the communication.

B. Client Architecture

APA Client was designed on two platforms (recently), responsive web application and versatile android application while the responsive web was designed using HTML, PHP, CSS, Javascript and bootstrap as the design foundation. The responsive used to grant feasible access to a small-screen clients, like iOS devices as well to a big one like smart television. As for android apps, they tend to give simpler credential access by integrating it to google plus account. Thus, android apps provides one taps access to control and monitor APA devices.

Both the web and application have the same function, thus having the same work-flow. Figure 5 presents the pseudo code of web and android apps work-flow. After the credential checked process, the client automatically requests selected gateway and standby to give or receive information, such as watering, monitor (taking a picture), and review at watering history. Client grants access to the gateway to set up which mode should be active, automatic or manual. When the automatic is selected, the gateway will harness

weather data from APA cloud service to do a calculation of the watering process.

III. Implementation and Results

The system architecture design (Figure 1) can be described as follows:

A. Full Hardware Design

APA system's hardware consists of device and gateway. Ip camera (figure 6) and DC 12 V Solenoid valve are integrated on APA device (figure 7), whilst weatherboard shield are installed on gateway, which is also connected to the internet. Figure 8 shows how the device and the gateway are connected. Figure 9 illustrates the entire implementation of the hardware system.

```
Initialize credential
If the credential is valid then
    If user's gateway more than or equals 1 then
        set first gateway as monitored gateway
        initialize gateway's weather information
        while monitored gateway selected
            if selected gateway is changed
                set monitored gateway
            else
                if action given by user then
                    initialize rest protocol communication
                    send request to cloud service (api)
                    if request sent then
                        update monitored gateway and weather information
                        update device information on monitored gateway
                        listen cloud response
                    else
                        print failure command message
                else
                    listen monitored gateway and weather update
                    listen device information on monitored gateway update
    else
        print must have registered gateway message
else
    print failure credential message
```

Figure 5: Pseudocode of APA client

Figure 6: IP Camera

Figure 7: Solenoid valve

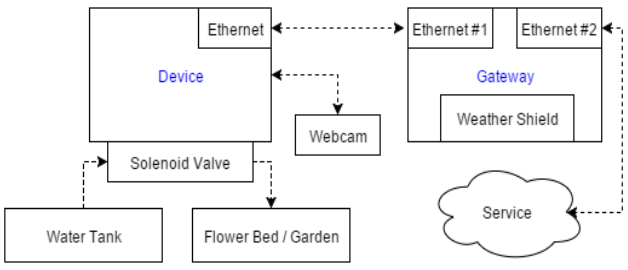

Figure 8: Hardware design

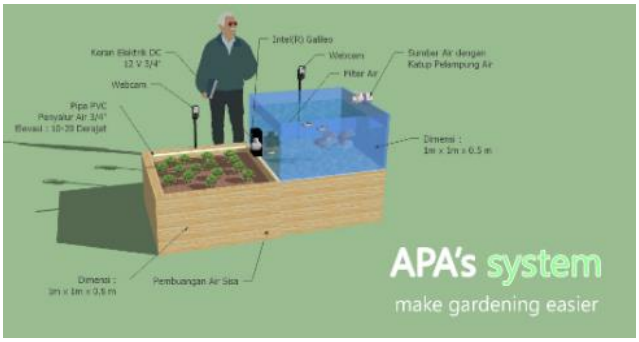

Figure 9: Illustration of APA device

B. Hardware and Connectivity Setup

APA system is connected to each other parts as shown in Figure 2. Device(s) connected to the gateway via a direct connection on ethernet port are as shown on Figure 8. They can also be connected by using switch and harness star network topology. The gateway must be connected to the Internet to be able to receive information from the cloud.

The approximate connection type to support full features of APA is the ADSL or broadband connection, which gives 512kbps to 1 Mbps upstream and downstream. Dial-up connection is acceptable, but it tends to give a problem when monitoring and it has slow response on giving commands because of 56 kbps speed. Average request from point to point sized on 1-2 kB (8-16 kb) (without other header and connection establishment), while the response sized approximately 5-8 kB (40-64 kb) except to monitor response, which is giving image streams of 100-200 kB (800-1600 kb), VGA (640x480) camera quality. With respect to mobile, 3G connection or higher is recommended. In this case, EDGE is fine except for receiving an image result from the monitor.

C. Client-Side Application

The second part of the implementation of the system makes use of client application for giving a command or receiving information to the gateway, which manages the device(s). Figure 10 shows the client dashboard via responsive web application tested on a small-screen device.

Figure 10: Web apps (small)

Figure 11: Android apps

Lumia 920 (4.5" with HD resolution) presented in Figure 12 shows a medium screen device like a personal computer, tested on Asus notebook x550dp (15.6" with 1366x768 resolution). Figure 11 shows the majority of versatile android applications. Three of them can do the same thing,

namely watering, monitoring and observing watering history. A unique id is provided to give access for a client. The client must add gateway(s) via the managed gateway (Figure 13). Added gateway will be identified as the user's gateway.

After selecting the gateway, users can choose the types of action can be performed by the system. Figure 14 shows a result of the request of monitors (taking a picture) from an android application, using a stable WiFi connection. Approximately 5-7 sec until the request is handled. The image will be downloaded depend on the gateway and client internet connection.

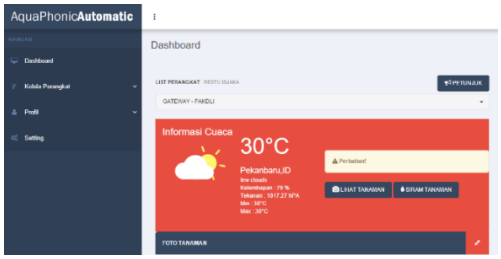

Figure 12: Web apps (medium)

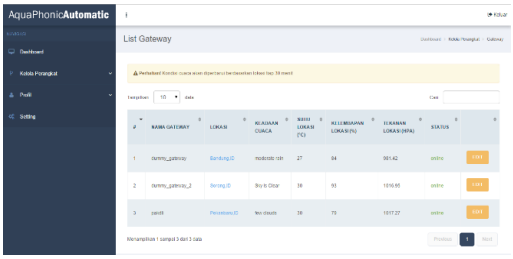

Figure 13: Gateway management (medium)

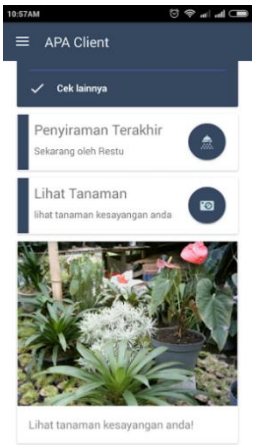

Figure 14: Plants monitoring

The ip camera on galileo has matrix 640*480. Broadband or ADSL gateway could process the image and upload it around 5-6 sec. A client with WiFi connection can process the image within 4-5 sec. The whole process can take up to one minute, depending on the connection.

IV. CONCLUSION

Weather climatic variables affected by the temperature and humidity led to a relative humidity, which is identified as the relative amount of water in the air in relation to the amount of air saturation of the current temperature. During a long day, the temperature fluctuates according to other climatic variables. If the relative humidity on that day rises up to 100% and the temperature goes down, rainfall or snow might happen. Hence, this phenomena relates to the concept of APA cloud service.

Weather updates run every 30 minutes, which gives an enhancement on the current decision to water. Histories of weather updates saved up to the previous 12 hours, specifically for each gateway. REST protocol adopted in this system resulted in the versatile way of communicating by parsing data in JSON format. It is highly suggested to adopt from REST protocol because of its simple format using HTTP methods (easy to integrate to software side).

Node.js which is set up on Intel Galileo Gen 2 brings many possibilities as cross-platform programming. Other than handling upstream and downstream data, hardware can be controlled using mraa library on the node.js foundation. Hence, it is highly recommended concept, especially the IoT concept.

Enhancement on the monitor features is highly suggested. Further, the alternate process that compensates the flow of data focused on image stream is urgently needed.

Plant's needs of water in the device should be measured using a standard nutrition requirements of each plant. The collaboration of the experts is highly recommended.

Overall, future works of this present project could be leading to IoT-based agriculture technology. Thus, giving the enrichment and nourishment effect on the agricultural section. Optimizing the results, whilst using minimal resources. Future discussion and collaboration are welcome.

ACKNOWLEDGEMENT

I wish to thank Mr. Fairuz Azmi and Mr. Agung Nugroho Jati, for their valuable advice and support on this collaboration project. Special thanks should be given to the researchers of the Robotics and Embedded System (RnEST) research group of the school of the electrical engineering for their help in offering me the resources in this project. Support provided by Nursyifa was greatly appreciated. Finally, I wish to thank my parents for their support and encouragement throughout my study.

REFERENCES

[1] Jiangqiong, P. and Mengnan. S. 2014. Water-Saving Campus Park and Garden Construction of Tsinghua University Based on GIS. *Intelligent Systems Design and Engineering Applications (ISDEA), 2014 Fifth International Conference*. Hunan, China. 15-16 June 2014. 1117-1121.

[2] Boutraa, T., Akhkha, A., Alshuaibi, A. and Atta. R. 2011. Evaluation of the effectiveness of an automated irrigation system using wheat crops. *Agriculture and Biology Journal of North America.* 2(1): 80-88.

[3] Nugroho, R. A., Pambudi, L. T., Chilmawati, D. and Condro Haditomo, A. H.. 2012. Aplikasi Teknologi Aquaponic pada budidaya ikan air tawar untuk optimalisasi kapasitas produksi. *Jurnal Saintek Perikanan*. 8(1): 46-51.

[4] Khalyankar, M. A. and Alaspukar, P. S. J. 2013. Data Mining Technique to Analyse the Metrological Data. *International Journal of Advanced Research in Computer Science and Software Engineering*. 3(2): 114-118.

[5] Rohit Gunturi, V. N. 2013. Micro Controller Based Automatic Plant Irrigation System, *International Journal of Advancements in Research & Technology*. 2(4): 194-198.

[6] Ganesh, S. C. S. 2014. Efficient Automatic Plant Irrigation System using ATMEGA Microcontroller. *International Journal of Emerging Trends in Electrical and Electronics*. 7(1): 49-52.

[7] Devika, S. V., Khamuruddeen, S., Khamurunnisa, S., Thota, J. and Shaik, K. 2014. Arduino Based Automatic Plant Watering System. *International Journal of Advanced Research in Computer Science and Software Engineering*. 4(10) :449-456.

[8] P. D. S and Srinath, S. 2014. GSM based Automatic Irrigation Control System for Efficient Use of Resources and Crop Planning by Using an Android Mobile. *IOSR Journal of Mechanical n Civil Engineering (IOSR-JMCE)*. 11(4): 49-55.

[9] Aswani, H. N, R. and Malik, M. 2012. Plant Watering Autonomous Mobile Robot. *International Journal of Robotics and Automation*. 1(3): 152-162.
[10] Kontogiannis, K. and Athanasopoulos, M. 2015. Extracting REST resource models from procedure-oriented service interfaces. *The Journal of Systems and* Software. 100: 149-166.
[11] Valsson, S. and Bharrat, A. Impact of Air Temperature on Relative Humidity – A study. [Online]. From : http://www.coa.gov.in/mag/Feb11Pdf%20file%20of%20website/Sheeba%20Valsson-pg38-41.pdf. [Accessed on 2 August 2015].
[12] Peng, D., Cao, L. and Xu, W. 2011. Using JSON for Data Exchanging in Web Service Applications. *Journal of Computational Information Systems*. 7(16): 5883-5890.
[13] Hamad, H., Saad, M. and Abed, R. 2010. Performance Evaluation of RESTful Web Services for Mobile Devices. *International Arab Journal of e-Technology*. 1(3): 72-78