International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878, Volume-8 Issue-5, January 2020

Multi-Protocol Gateway System for Heterogeneous Network on Internet of Things

Delphi Hanggoro, Jauzak Hussaini Windiatmaja, Jonathan, Lukman Rosyidi, Riri Fitri Sari

Abstract*: One of the main problems on the Internet of Things is heterogeneity. One of them is diversity in communication protocols. Generally, we will need one gateway for each existing protocol which causes inefficient communication networks. With these problems, we propose to build a hybrid network infrastructure using the Multi-Protocol Gateway system, this allows existing gateways to accept more than one protocol. This system is divided into two parts, hardware and software. For hardware systems, we use a single board processor equipped with the necessary transceiver modules. Our software system uses Python scripts to manage all backend processes, MySQL as data storage, PHP as a web server and Progressive Web Application (PWA) web application so that all devices get the same experience. We also provide a dashboard on the gateway to show and ensure that all data from different protocols can be received simultaneously in real-time. As a result, the gateway we developed can receive all data from four different protocols. We also evaluate the performance of the gateway by testing hardware capabilities by measuring the transmit time and Packet Delivery Ratio (PDR). The result shows that all communication protocols show compliance with existing hardware specifications. In addition, we also test the load time performance of the PWA service and compare it to non-PWA. The load time on PWA is smaller than that of non-PWA, due to the fact that PWA has offline capabilities that can store caches temporarily so that the next load time will be lighter. Using hybrid network infrastructure with Multi-Protocol Gateway is the answer to the problem of the diversity of protocols that we often find in the implementation of IoT such as smart building, smart city and other smart environments, with the existence of this Multi-Protocol Gateway can make communication services in the IoT structure to be more efficient.

***Keywords*: Multi-Protocol Gateway, Internet of Things, PWA, heterogenous, communication protocol.**

Revised Manuscript Received on January 15, 2020
***** Correspondence Author
Delphi Hanggoro*, Department of Electrical Engineering, Universitas Indonesia, Depok, Indonesia.
Jauzak Hussaini W, Department of Electrical Engineering, Universitas Indonesia, Depok, Indonesia.
Jonathan, Department of Electrical Engineering, Universitas Indonesia, Depok, Indonesia.
Lukman Rosyidi, Department of Electrical Engineering, Universitas Indonesia, Depok, Indonesia.
Riri Fitri Sari*, Department of Electrical Engineering, Universitas Indonesia, Depok, Indonesia.

I. INTRODUCTION

The concept of the Internet of Things was introduced by Kevin Ashton in early 1999, which is a new paradigm in the field of wireless networks and communications [1]. Minerva et al. argue that the Internet of Things is a global network of devices such as mobile devices, actuators and sensors that communicate via the internet to exchange data, collect data and process data [2].

Internet of Things is growing rapidly, the concept and service of the Internet of Things have been used in various fields such as military [3], industry [4], agriculture [5], transportation [6], smart home [7], smart building [8] and so forth. As the number of devices that have been installed increased, in 2008 the number of devices connected to the internet had exceeded the world's population at that year [9]. By 2020, it is predicted that there will be approximately 26 million installed IoT devices [10]. This rapid development has triggered many researchers to solve several existing problems.

To be able to communicate with each other, IoT devices must have certain communication modules. Now many communication protocols are used on IoT devices such as WiFi, Bluetooth Low Energy (BLE), ZigBee and Lora. However, until now there is no standard communication protocol created for IoT devices, causing diversity in the communication protocol.

Diversity in the types of communication protocols is a top challenge in IoT at this time. This is because each vendor builds its own network interface and protocol [11, 12]. Thus each communication protocol requires its own gateway which causes the addition of the number of gateways and infrastructure on the network to be inefficient.

Gateway has an important role for a network structure that applies the Internet of Things. Its main task is to convert the non-TCP / IP protocol into a TCP / IP protocol so that it can connect, send and receive data to the Internet, facilitate access to information, improve performance and improve computing performance [13]. However, a gateway has only one communication protocol so that it cannot overcome the problem of diversity in the types of communication protocols.

In this work, we propose a system of Multi-Protocol Gateways that can collect data from four different communication protocols and present data to users using the PWA. We discuss the system embedded in the Multi-Protocol Gateway's hardware and software that we use.

The goal is to build a hybrid network infrastructure by combining all WiFi, BLE, ZigBee and Lora communication protocols. This will make communication services on the IoT structure become more efficient. Our contributions are summarized as follows:

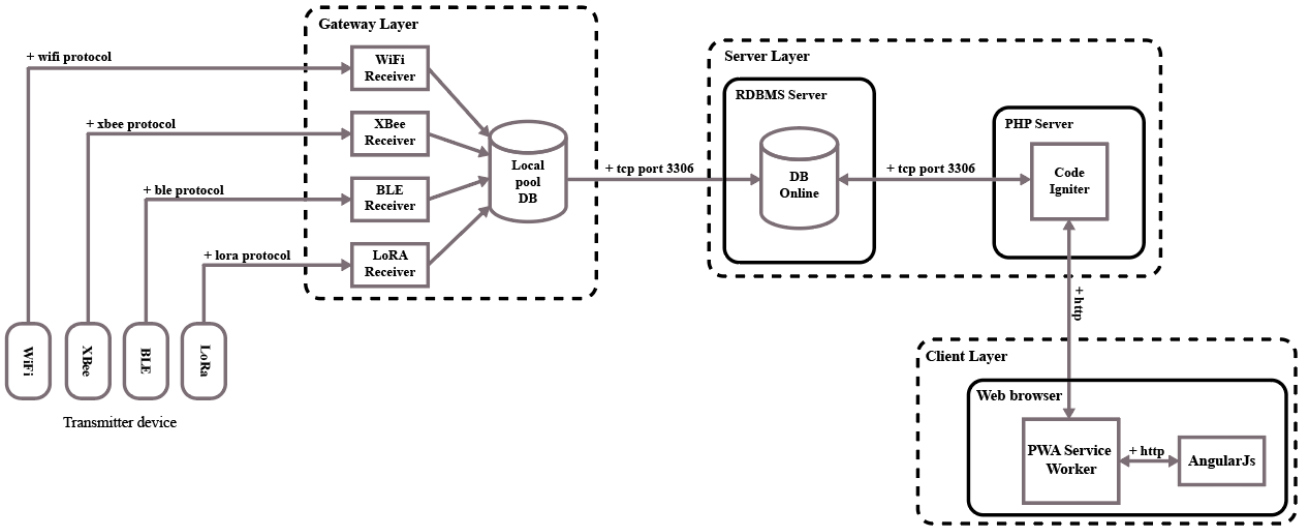

Fig. 1.Proposed Multi-Protocol Gateway system scheme.

- Combine the Wifi, BLE, ZigBee and Lora protocols into one board and use data parsing to group data according to protocol. This allows all of these protocols to be accepted by a single gateway.
- Integrate the system with the server so that it can be accessed by users.
- Service users who have a variety of devices such as smartphones, tablets, laptops and other devices. We use the PWA framework to allow all devices to have an equivalent user experience and have better performance and advantages over ordinary web apps.

The structure of this paper is organized as follows. We will explain and discuss the literature review in Section II. Next, we will explain the hardware and software of the Multi-Protocol Gateway System in Section III. Subsequently, we present the results and discussion in Section IV. Finally, we present the conclusions in Section V.

II. LITERATURE REVIEW

In this section, we review the gateways that can solve the problem of diversity in existing communication protocols.

Oniga et al. in [14] developed a Multi-Protocol Gateway that is capable of receiving two-way signals from two different communication protocols, i.e LoRa and Bluetooth Low Energy so that both of these protocols can be received by one gateway device. The gateway will also act as a bridge between the device and the network server when operating through these two protocols.

Yin et al. in [15] have also developed an embedded model of a multi-protocol gateway. The proposed model uses the S3C6410 board which has an ARM 11 processor. They built a system capable of receiving communication protocols from the Wireless Sensor Network (WSN), CAN bus, 3G network, WLAN, and Ethernet. Data buffer settings make it possible to develop and access a modular subnet communication interface, which is convenient for expanding different field equipment and thus improves gateway compatibility.

Another study shows a smart protocol based gateway utilizing ZigBee [16]. The multi-protocol gateway framework is structured with the goal that the gateway is partitioned into two separate parts. The initial segment turns into the CPU board and the baseboard to gather data on streetlight control. The subsequent part is utilized to screen and confirm nearby conditions and perform continuous controls utilizing wired and remote systems. This application produces intelligent interchanges innovation at the multi-protocol gateway for vitality sparing and control of hubs utilizing ZigBee with half and half system and remote system design with different network topology. In view of the consequences of tests on quality, voltage strength and the effect of nonstop activity utilizing the unwavering quality test got achievement estimation of over than 95%.

Previously we have developed a gateway that is capable of receiving three different communication protocols, i.e WiFi, BLE to ZigBee simultaneously [17]. They use microcomputer boards that are equipped with additional modules needed. A local dashboard that is used to ensure that data has been received by the system has been guaranteed.

In this research we extend our previous work on the embedded gateway system [17]. We expanded it by adding the LoRa communication protocol and connecting the database with the online databases and presenting data to users through the PWA. Therefore the gateway, we developed can accept all four different communication protocols and can be accessed by various devices.

III. SYSTEM DESIGN

In this section, we explain the flow of the proposed system as illustrated in Figure 1. In this system, we propose a multi-protocol gateway system to be able to receive four communication protocols namely WiFi, BLE, Zigbee and LoRa. Fig 1 shows that the system we made is divided into 3 parts, namely the Gateway Layer, Server Layer and Client Layer.

Gateway Layer is all activities of the gateway to receive all incoming transmissions from various protocols, parsing them and changing all these protocols into TCP / IP that allows forwarding to the Internet. Server layer as a place to store data from the gateway and provide data to be served to users through the PHP server. The client layer is the part that is responsible for displaying data using the PWA web application.

Data coming from the transmitter will be received by the Multi-Protocol Gateway. Subsequently, the data will be parsed using a python script and stored into the MariaDB local pool. When the gateway does not have an Internet network then the data will be stored in the local pool. If the gateway has an Internet connection, the data in the local pool will be permanently moved to the database server via port TCP 3306.

After being stored in the server the data will be presented to the web. Before entering the web browser, the data will be processed first using CodeIgniter (CI) on the PHP server via port TCP 3306. After that, the data will be served into the web browser via port HTTP.

The system design consists of two parts, software and hardware. In the hardware section, we explain the types of devices and communication standards that we use. In the software section, we explain the flow and the program that we propose.

A. Hardware

The gateway device we use consists of a modified microcomputer board with additional communication protocol modules as needed. Illustration of an embedded gateway system can be seen in Figure 2.

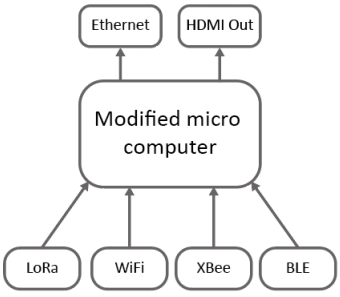

Fig. 2. Block diagram of Hardware System

This system supports four communication protocol standards, including:

- WiFi protocol with IEEE 802.11 b/g/n standard (standard for Local Area Network (LAN)).
- Bluetooth Low Energy protocol with IEEE 802.15.1 standard (standard for Wireless Personal Area Network (WPAN))
- ZigBee protocol with 802.15.4 standard (standard for Wireless Personal Area Network (WPAN))
- LoRa protocol with 802.15.4 standard (standard for Wireless Personal Area Network (WPAN))

In order to receive data from all the different protocols, the gateway board must be equipped with the required protocol receiver modules. Data entering the gateway will be parsed and grouped according to the respective communication protocols that are completed by the python script. To ascertain whether the gateway has received data from the transmitter, we added a dashboard to be able to display a simple network topology and data table that has entered the gateway through the HDMI out port. The Ethernet port is used to forward data from the gateway to the database server.

B. Software

On this software design Section, we describe the design of the gateway and the server.

1) Gateway

The gateway Section will discuss the software that is on the gateway hardware. We use the Python programming language to handle backend problems. MySQL for local databases and Lazarus to display data on the local dashboard.

```
Algorithm 1. Transmit-Receive Data
1    procedure DATA-TRANSMIT
2      while true
3        temperature ← analogRead(DATA_PIN);
4        lcd.print ← temperature
5        send(temperature)
6      end while
7    end procedure
8
9    procedure DATA-RECEIVE
10     while true
11
12       data ← receive(temperature)
13       connect local_pool
14       insert data into local_pool
15     end while
16   end procedure
17
18   procedure LOCAL-DASHBOARD
19     while true
20       connect local_pool
21       rows ← select all from local_pool sort
22       descending timestamp
23       output ← rows
24     end while
     end procedure
```

The data parsing process begins with the node reading the temperature data (line 3, algorithm 1) and then printing the data to each node's LCD. Then each transmitter will send data to the gateway together (line 5, algorithm 1). The gateway will continue to standby and receive data from the node if there is data sent from the nodes (line 11, algorithm 1). If data is received, the gateway will enter the data into the local pool for temporary storage (line 13, algorithm 1). Simultaneously with receiving data, we can see the data received in realtime through the local dashboard (lines 17-24, algorithm 1).

Displaying data to the local dashboard is done at the application layer by creating a table view that displays id, node name, time, and temperature data. Each data from a different protocol will be entered into the MySQL database. Subsequently, the data is called and displayed on the dashboard in realtime.

```
Algorithm 2. Storage Migration Service
1   procedure MIGRATION-DB
2     while true
3       connect local_pool
4       connect db_online
5       rows ← select all from local_pool
6       if rows length > 0
7         insert rows into local_pool
8         delete rows into db_online
9       end if
10    end while
11  end procedure
```

Storage Migration Service is performed to move the data of each node in the database gateway (local pool) to the cloud database (online buffer) through the MySQL protocol. This storage migration mechanism uses the python language which is executed on the gateway. Storage migration starts by making a database gateway and cloud database connection as shown in Algorithm 2. When the connection is successful, the query to retrieve data from the gateway database sorted by time descending will be executed (line 5). After data retrieval is successful, and the data is not empty, the data is sent to the cloud database via the MySQL protocol (lines 6-7). When the query data execution is successful, data that has been entered into the cloud database, it would be deleted from the local database to avoid data redundancy and save storage space from the gateway (line 8).

2) Server

The server section will discuss the flow and software used on the server and client. We use MySQL for databases on servers, PHP servers as web servers and PWA as web apps frameworks in browsers.

```
Algorithm 3. Display dashboard on the web
 1  procedure FETCH-DATA
 2    while true
 3      request rest_get_node
 4      data_node ← response data
 5
 6      node-latest-data ← fetch
 7      data_node[length(data_node)-1]
 8
 9      node-chart ← initiate chart[data ←
10      data_node]
11    end while
12  end procedure
```

The web dashboard display mechanism is performed at the application layer via HTTPS and MySQL protocols. This mechanism serves to present web-based id, node name, time, temperature data and graph. We chose PWA as a web apps framework, with the aim that dashboard can be seen from various types of devices such as desktops, android-based smartphones, iOS-based smartphones with equivalent user experience. Dashboards are made with PHP, JavaScript and use the CodeIgniter and AngularJS frameworks. This mechanism starts when the user opens the dashboard menu on the web page. After the page has finished loading, the script from AngularJS will communicate with the REST API created to retrieve data from the cloud database and provide a response in the form of JavaScript Object Notation (JSON) data as shown in lines 3-4, Algorithm 3. When the data collection process is successful, the AngularJS script performs Document Object Model (DOM) manipulation to display data in the form of one log table and WiFi, BLE, ZigBee and LoRa widgets as shown in lines 6-10. This process continues until the user decides to close the dashboard page.

IV. RESULT AND DISCUSSION

To evaluate this system, we divide it into the hardware and software.

A. Hardware

To test and validate the performance of prototypes an evaluation scenario is needed. All existing transmitters will be activated to send data packets containing temperature measurement information. We only use one transmitter for each communication protocol. Subsequently the gateway will gather information from different corresponding protocols simultaneously. The information will be stored in a MySQL database, e.g the sensor data and timestamp. Due the hardware test, transmit time and Packet Delivery Ratio will be analyzed from the the gateway's data.

The transmitter specifications we use for WiFi, BLE, ZigBee and LoRa are as follows:

- Wifi - Arduino Uno with the mini WeMos D1mini 1 module
- BLE - Arduino Uno with Bluetooth HC-05 Macho module
- ZigBee - Arduino Uno with XBee ZNet 2.5 module
- LoRa - Raspberry Pi 3 with iFrogLab Lora module

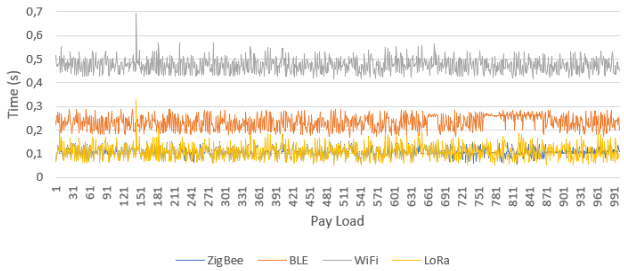

Fig. 3. Transmit time for every communication protocol

Figure 3 shows the transmit time of each protocol. It shows that the LoRa protocol is stable at 0.9 ms - 0.19 ms, the ZigBee protocol also shows an equivalent result from the LoRa protocol. Meanwhile Bluetooth has a higher transmit time in the range 0.19 ms - 0.29 ms. Finally, WiFi's transmit time is in the range 0.44 ms - 0.54 ms. The experiment are in accordance with standard technical specifications that exist on the equipment. This happens because the transmitter we use is only one for each communication protocol, and there are no signal interference.

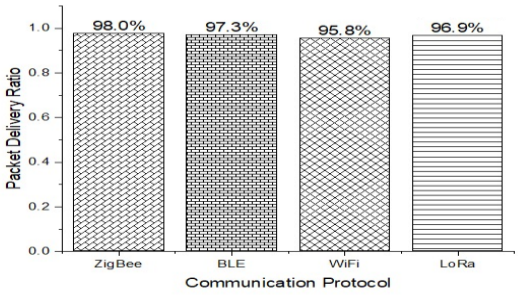

Fig. 4. Packet delivery ratio for every communication protocol

Figure 4 shows the Packet Delivery Ratio (PDR) for each type of communication protocol. The rate of packet delivery ratio from Bluetooth Low Energy is 97.25%, WiFi is 95.5%, Zigbee is 98% and LoRa is 96.85%. it can be stated that this result conform with the standart technical spesification of the equipment used. The impedance signal is viewed as low because we just utilize one transmitter for every communication protocol.

Retrieval Number: E5894018520/2020©BEIESP
DOI:10.35940/ijrte.E5894.018520
Journal Website: www.ijrte.org

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

This evaluation test was conducted to test the usefulness of the gateway, and not the solidness of the gadget. In the event that there are multiple gateways in the territory, every packet will be gotten by the goal gateway. In this case, parcel crashes will be limited by the CSMA/CA component, which is actualized by the WiFi, Bluetooth, ZigBee and LoRa protocols.

B. Software

To test and validate the performance of the server, a web server evaluation scenarios are needed. We test the PWA using Google Chrome Lighthouse 5.7.0. Lighthouse is an open-source, mechanized instrument for improving the presentation, quality, and accuracy of web apps. We test the load time of each page in 100 iterations between the PWA and non-PWA page.

PWA test was conducted using in the test is Asus A455LB-i5 5200U CPU with Nexus 5X emulated and 4x CPU throttling. Host user agent used is Mozilla. Network user agent used is Mozilla. Network throttling was conducted at 150 ms TCP Round Trip Time (RTT), with 1,638.4 Kbps throughput.

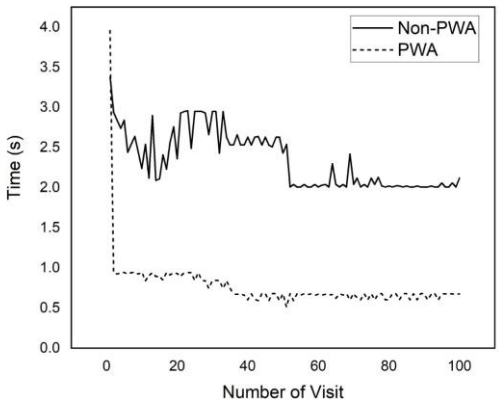

Fig. 5. Load time comparison for PWA and Non-PWA

Figure 5 shows that using PWA web apps affects the load time which is very different from those of the non-PWA. The repetitive visit was performed on this application when PWA is registered to proof that PWA reduce load time. The result then plotted in Figure 5. In 100 iteration visits, non-PWA page averaging 2.34 seconds while PWA averaging only 0.76 seconds. This happened because of all the static content which remains the same to an extent all over the application like the basic HTML template, CSS and JavaScript without versioning are stored in local storage by the service worker. This content can easily be cached and thus loaded instantly from the cache when the user repeatedly visits them. We can see that there is a drastic drop between the first and the second PWA visit. This happened because the service worker is installed after the first load. Due to network inconsistency, the load time for non-PWA application might be fluctuating. PWA load time fluctuated due to the local device performance and usage.For usability test, we use survey methodology developed by Brooke [18]. This questionnaire consists of ten different statements, with the comparison between positive and negative statements is 5:5. The form of questions is multiple choice by using a variable measurement scale that refers to a Likert scale. Each has a scale of 1-5, i.e Strongly Disagree, Disagree, Neither Agree nor Disagree, Agree, Strongly Agree.

The test subject is a randomly picked users. Thirty-five respondents conducted the usability test. The Thirty-five respondents run the application and then fill out the questionnaire provided. The question regarding the usability test is described in Table 1 [19].

Tabel 1. Usability test questionnaire [19].

No	Question
1	I think that I would like to use this system frequently
2	I found the system unnecessarily complex.
3	I thought the system was easy to use.
4	I think that I would need the support of a technical person to be able to use this system.
5	I found the various functions in this system were well integrated.
6	I thought there was too much inconsistency in this system.
7	I would imagine that most people would learn to use this system very quickly.
8	I found the system very cumbersome to use.
9	I felt very confident using the system.
10	I needed to learn a lot of things before I could get going with this system.

Questionnaire results and calculations are explained in Figure 6. The results are then calculated using the usability score calculation according to the usability test plan.

Figure 6 shows that the average score of our systems is 81.25. This value is considered acceptable in Brooke's calculation. This shows that the application is well accepted by end-users in terms of usability aspects.

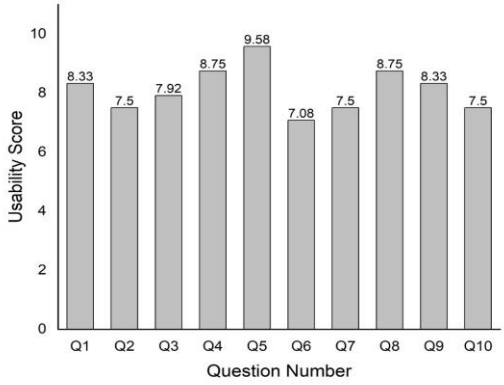

Fig. 6. Calculation result for usability score

V. CONCLUSION

We have presented a Multi-Protocol Gateway System for the Internet of Things network. The goal is to form a hybrid infrastructure network by receiving many communication protocols at once in one gateway and can be accessed from various devices. Compared to using one gateway for each communication protocol, the system we propose is more efficient in the number of gateways.

The evaluation results from the gateway also show that one microcomputer board can receive all the communication protocols properly. The use of PWA web applications is more intuitive and has a good load time compared to non-PWA.

PWA has a higher load time with 4s compared to non-PWA with 3.4s. The load time on PWA will drop dramatically to 0.7-1s after the first visit because of PWA has an offline capability feature, which stores a temporary cache so that subsequent loads will be lighter, while the non-PWA load time will remain the same because they have to load everything from the start.

This shows that the system we propose will form a hybrid infrastructure network and make communication services on the IoT structure more efficient.

ACKNOWLEDGEMENT

This work is supported by the Indonesian Government Scholarship of PMDSU from the Ministry of Research, Technology, and Higher Education (Kemristekdikti).

REFERENCES

1. I. Bose and R. Pal, "Auto-ID: managing anything, anywhere, anytime in the supply chain," Commun. ACM, vol. 48, no. 8, pp. 100-106, 2005.
2. Minerva, "Towards a definition of the Internet of Things (IoT)," Rev.1. IEEE IoT Initiative, 2015.
3. L. Yushi, J. Fei, and Y. Hui, "Study on application modes of military Internet of Things (MIOT)," in 2012 IEEE International Conference on Computer Science and Automation Engineering (CSAE), 2012, vol. 3, pp. 630-634.
4. F. Shrouf, J. Ordieres, and G. Miragliotta, "Smart factories in Industry 4.0: A review of the concept and of energy management approached in production based on the Internet of Things paradigm," in 2014 IEEE International Conference on Industrial Engineering and Engineering Management, 2014, pp. 697-701.
5. J. Wu, S. Han, and J. Liu, "Application Progress of Agricultural Internet of Things in Major Countries," Journal of Physics: Conference Series, vol. 1087, p. 032013, 09/01 2018.
6. O. Jo, Y. Kim, and J. Kim, "Internet of Things for Smart Railway: Feasibility and Applications," IEEE Internet of Things Journal, vol. 5, no. 2, pp. 482-490, 2018.
7. S. u. Rehman and V. Gruhn, "An approach to secure smart homes in cyber-physical systems/Internet-of-Things," in 2018 Fifth International Conference on Software Defined Systems (SDS), 2018, pp. 126-129.
8. N. Havard, S. McGrath, C. Flanagan, and C. MacNamee, "Smart Building Based on Internet of Things Technology," in 2018 12th International Conference on Sensing Technology (ICST), 2018, pp. 278-281.
9. S. E. Collier, "The Emerging Enernet: Convergence of the Smart Grid with the Internet of Things," IEEE Industry Applications Magazine, vol. 23, no. 2, pp. 12-16, 2017.
10. J. T. Peter Middleton, Peter Kjeldsen, "Forecast: The Internet of Things, Worldwide, 2013," Gartner Inc., Connecticut, CT. G00259115, 2013.J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73. , 2013.
11. A. Muthanna, A. Prokopiev, A. Paramonov, and A. Koucheryavy, "Comparison of protocols for Ubiquitous wireless sensor network," in 2014 6th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2014, pp. 334-337.
12. V. Kulik and R. Kirichek, "The Heterogeneous Gateways in the Industrial Internet of Things," in 2018 10th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2018, pp. 1-5.
13. R. Madduri et al., "The Globus Galaxies platform: delivering science gateways as a service," Concurrency and Computation: Practice and Experience, vol. 27, no. 16, pp. 4344-4360, 2015.
14. B. Oniga, A. Munteanu, and V. Dadarlat, "Open-source multi-protocol gateway for Internet of Things," in 2018 17th RoEduNet Conference: Networking in Education and Research (RoEduNet), 2018, pp. 1-6.
15. R. Yin, F. Zhang, M. Liu, F. Gui, F. Li, and W. Shen, "Communication model of embedded multi-protocol gateway for MRO online monitoring system," in 2015 IEEE 19th International Conference on Computer Supported Cooperative Work in Design (CSCWD), 2015, pp. 97-102.
16. S. Hong, S. Song, and C. Lin, "An efficient multi-protocol gateway system design on the zigbee," in 2015 17th International Conference on Advanced Communication Technology (ICACT), 2015, pp. 525-528.
17. D. Hanggoro, L. Rosyidi, and R. F. Sari, "Design and Implementation of Multi-Protocol Gateway for Internet of Things," in 2019 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT), 2019, pp. 64-69.
18. Jeff Sauro , James R. Lewis, When designing usability questionnaires, does it hurt to be positive?, Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, May 07-12, 2011, Vancouver, BC, Canada [doi>10.1145/1978942.1979266]
19. Brooke, J. (1996). SUS: A "quick and dirty" usability scale. In P. W. Jordan, B. Thomas, B. A. Weerdmeester, & A. L. McClelland (Eds.), Usability Evaluation in Industry. London: Taylor and Francis.

AUTHORS PROFILE

Delphi Hanggoro has completed a bachelor program in Computer Engineering at Diponegoro University and is currently running a master's program at the Department of Electrical Engineering, University of Indonesia.

Jauzak Hussaini W has completed a bachelor program in Computer Science at Diponegoro University and is currently running a master's program at the Department of Electrical Engineering, University of Indonesia.

Jonathan currently completing his bachelor program at the University of Indonesia. The research is in the field of embedded systems.

Lukman Rosyidi He completed all his education from Bachelor (2001), Masters (2015) to Doctoral (2018) at the Department of Electrical, University of Indonesia. He is currently active as a lecturer and research in the field of information networks and embedded systems at the Nurul Fikri Integrated Technology College. He is also the Chair of the Nurul Fikri Integrated Technology College.

Riri Fitri Sari* She earned a Bachelor of Electrical Engineering from UI in 1994. In 1997, she received an MSc in Software Systems and Parallel Processing from the Department of Computer Science, University of Sheffield, England with the Chevening Award from the British Council. She successfully completed her doctoral dissertation with research in the field of Active Network Congestion-Based Congestion Management and obtained her PhD from the School of Computing, University of Leeds, England. She has been approved as a full Professor in Computer Engineering since May 1, 2009, in the Department of Electrical Engineering, Faculty of Engineering, University of Indonesia. Currently, she is actively teaching and researching in the field of Computer Network, and Information and Communication Technology implementation. She has written more than 200 peer reviewed Journals and International Conference Papers, and has been a Senior Member of the Institute of Electronics and Electrical Engineers (IEEE) since 2011. She has been a chairperson of UI GreenMetric World University Rankings and ranked university worldwide based on sustainability effort since 2010.