Bulletin of Electrical Engineering and Informatics
Vol. 9, No. 5, October 2020, pp. 1990∼1997
ISSN: 2302-9285, DOI: 10.11591/eei.v9i5.1978

Cloud-based middleware for supporting batch and stream access over smart healthcare wearable device

Adhitya Bhawiyuga[1], Satria Adi Kharisma[2], Bagus Jati Santoso[3], Dany Primanita Kartikasari[4], Annisa Puspa Kirana[5]
[1, 2, 4]Faculty of Computer Science, Brawijaya University, Indonesia
[3]Department of Informatics, Institut Teknologi Sepuluh Nopember, Indonesia
[5]Department of Informatics, State Polytechnic of Malang, Indonesia

Article Info

Article history:

Received Dec 4, 2019
Revised Feb 18, 2020
Accepted Mar 22, 2020

Keywords:

Batch
Internet of things
Middleware
Stream

ABSTRACT

In IoT-based smart healthcare services, the heterogeneity of connected wearable sensing devices open up a wide opportunity to develop various healthcare services. However, it also poses an interoperability challenge since each sensing device and application may have different communication mechanisms. Considering that challenge, web platform can be seen as a promising candidate for providing an interoperability layer as we can abstract various devices as single representation i.e. web resource. In this paper, we propose the design of middleware for enabling efficient web of things access over healthcare wearable devices. The proposed middleware consists of three components: gateway-to-cloud device, messaging service and data access interface. The gateway-to-cloud device has a role to perform low level sensor data collection from various wearable sensing device through bluetooth low energy (BLE) communication protocol. Collected data are then relayed to the cloud IoT platform using a lightweight MQTT messaging protocol. In order to provide device abstraction along with access to the stored data, the system offers two kind of interfaces: the Restful HTTP identified by unique universal resource locator (URL) for batch access and MQTT websocket interface identified by unique topic to accommodate access on sensing data in near real time stream manner.

Corresponding Author:

Adhitya Bhawiyuga,
Faculty of Computer Science,
Brawijaya University,
Jalan Veteran No. 8, Malang, East Java, Indonesia.
Email: bhawiyuga@ub.ac.id

1. INTRODUCTION

In recent years, the internet of things (IoT) has been one of the key enabling technology of the Industrial Revolution 4.0. In general, an IoT based system consists of several pervasive and ubiquitous computing devices equipped with sensing and communication capabilities for performing a continuous environmental data acquisition [1]. On top of those acquired data, we can possibly develop any smart services ranging from convenience to life-critical applications.[2, 3].

An IoT based smart healthcare is one of the promising services to be developed due to its important impact on human life [4, 5]. In this kind of service, a number of wearable sensing devices as mentioned in [6] including electrocardiograph (ECG) [7], photoplethysmogram (PPG) [8] or pedometer [9] are attached to human body for performing a periodic collective biosignal data collection through communication protocols such as BLE [10], LoRa [11] or Wifi [12]. This mechanism can be then combined with various data analytic methods to provide either personal health monitoring and assistance [13]. Furthermore, the collected data can be utilized by doctor to precisely diagnose a disease and decide its correct medication [14].

Despite its promising utilization, an IoT-based smart healthcare poses a challenge regarding to how the biosignal data are efficiently collected and accessed [15]. On one side, as the wearable sensing devices becoming more heterogeneous, it may opens a possibility to develop more interesting and useful smart healthcare services. However, on the other side, its heterogeneity may poses additional challenge to the developer for developing an application since he/she must deal with different device communication mechanism [16]. Furthermore, there is a possibility that the stored data collections are accessed as batch or stream depending on the application requirement. For instance, the heart rate variability detection service requires that the ECG measurement data is accessed in a near real-time stream fashion [17]. However, the daily fitness assistance service may access the user data collectively each day using batch mode. Therefore, in this case, a middleware is required to provide an interoperability layer between various wearable sensing devices and applications taking into account several requirements such as data management and device abstraction [18, 19].

Considering the stated issues and requirements, the web platform can be seen as a promising candidate to provide an interoperability layer between various sensing devices and applications thanks to its massive adoption in the current Internet era [20]. The integration between web platform and IoT technology leads to a conceptual change from the internet to the Web of things (WoT). In WoT concept, every device, regardless of its data format or underlying communication protocol is abstracted as web resource [21]. With this kind of programming abstraction, a developer can have a broaden possibility to make use of available health sensing data to develop various attractive applications running on almost any kind of platform including web and native applications [22]. In a broader perspective, the WoT concept also offers a further integration between the existing medical record data owned by hospital and government agency with a more personal smart healthcare service supported by various wearable sensing device.

In this paper, we propose the design of cloud-based middleware for enabling the efficient web of things access over healthcare wearable devices with both batch and stream data access support. The proposed middleware consists of three components: gateway-to-cloud device, messaging service and data access interface. The gateway-to-cloud device has an important role to perform low level sensor data collection from various wearable sensing device through BLE communication protocol. The collected data are then relayed to the cloud IoT platform using a more lightweight MQTT messaging protocol instead of HTTP [23]. In order to provide device abstraction along with access to the stored data, the system offers two kind of interfaces : the Restful HTTP identified by unique universal resource locator (URL) for batch access mode and MQTT websocket interface identified by unique topic to accommodate access on sensing data in near real time stream manner.

2. RELATED WORK

In literature, there exist several works that deal with the development of WoT based middleware. In [24] authors proposed the Restful middleware for enabling web of medical things. The proposed middleware consists of several components including the communication manager, device manager, and Restful web server. The communication manager handles sensing data transfer from various devices using BLE and Wifi protocol which is then synced to the cloud data center using standard HTTP protocol. The synced data can be then accessed by user or other apps through the Restful web server. In previous study, we proposed the design of IoT-cloud platform for integrating both IoT devices and cloud entity [25]. The proposed platform utilized the Restful HTTP protocol as device-to-cloud data endpoint interface between cloud systems and IoT devices.

While the previous works offer the implementation of WoT middleware, however, the proposed middleware do not consider the real-time sensor streaming data as it only provides the Restful HTTP interface in which the application is required to send a periodic message to the cloud for obtaining the latest sensing data. In some cases, an application requires a real-time data analytic provided by time-critical wearable sensors such as electrocardiogram (ECG) or electroencephalogram (EEG). Furthermore, the synchronization between device manager and the cloud is performed using standard HTTP which causes an additional overhead since every sync activity requires a TCP handshaking for initializing a connection. Therefore, in this case, we require a WoT middleware that supports both real-time data access in one side and lightweight data synchronization

3. PROPOSED MIDDLEWARE

This section explains the design of the middleware system. Figure 1 shows the general architecture of middleware with three main actors:

a. Sensor devices e.x. ECG, EEG, blood pressure as a data producer
b. Client application as data consumer
c. Proposed middleware as a communication bridge between data producer and consumer.

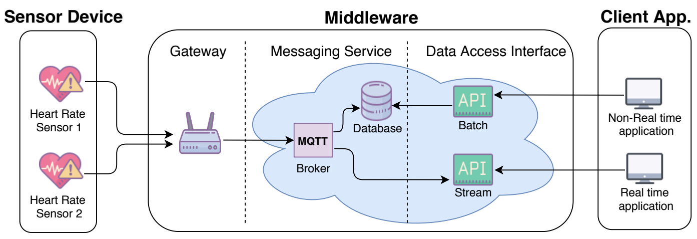

Figure 1. General system archutecture

3.1. System architecture

The proposed middleware itself consists of three components: gateway-to-cloud device, messaging service and data access interface.

3.1.1. Device-to-cloud gateway

The gateway device runs on a Raspberry Pi, connected to sensors device and cloud. As a device manager of the middleware, gateway has a role to manage connection and data transmission across different sensor device to middleware using BLE protocol. To achieve this purpose, the gateway first scan a nearby sensor devices and make a connection. Once the connention established, the gateway acting as the BLE client and retrieve sensing data from sensor device. Upon the reception, the gateway send those data as a message to the cloud using MQTT messaging protocol.

3.1.2. Messaging service

Messaging service run on the cloud. It has a role to receive messages that previously sent from the gateway, store those messages in a database and provide the stream access to the client using MQTT protocol. There are two main module inside this component that is MQTT broker and storage subscriber. The MQTT broker receives the message containing sensing data from gateway device and relay those messages to both storage subscriber and stream interface client app.

3.1.3. Data access interface

Data access interface runs on the cloud. It has a role to provide an access to the stored data for client applications. Data access interface has two kind access that is batch access and stream access. Batch interface is designed as a webservice with Restful HTTP architecture to provide data access in a non-real time manner. Stream interfaces is designed with the websocket to provide data access in a real-time manner using MQTT over websocket.

3.2. Data pipeline

Figures 2 and 3 illustrate the sequence of data starting from sensing devices to the non-stream real time client app using batch and stream data access mechanism respectively. At first, the sensor periodically emits its measurement to a gateway device which then relays those data to cloud messaging service using MQTT protocol. Upon reception, the broker component routes the data to subscriber entities depending on the data access mechanism.

For batch case, sensing data are relayed to a cloud based storage subscriber which then store those data to a database service. The stored data can be then accessed by non-real time client apps through HTTP request-response methodology. On the other hand, for stream case, sensing data can be directly routed to real time client apps through an MQTT topic subscription mechanism without passing a database service.

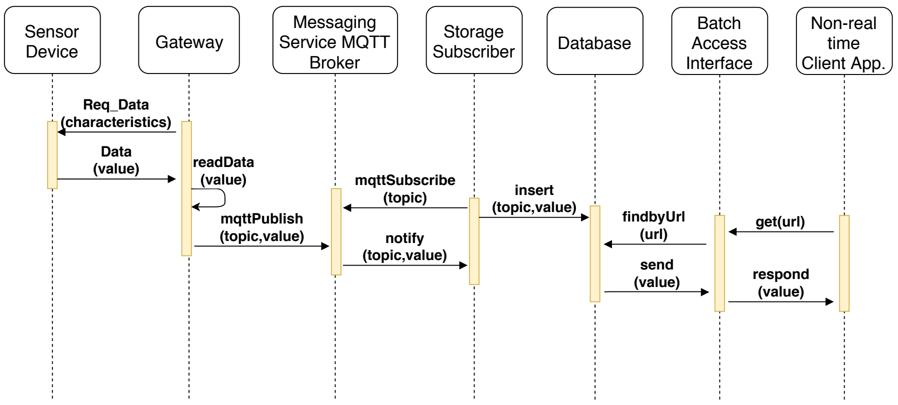

Figure 2. Batch data access pipeline

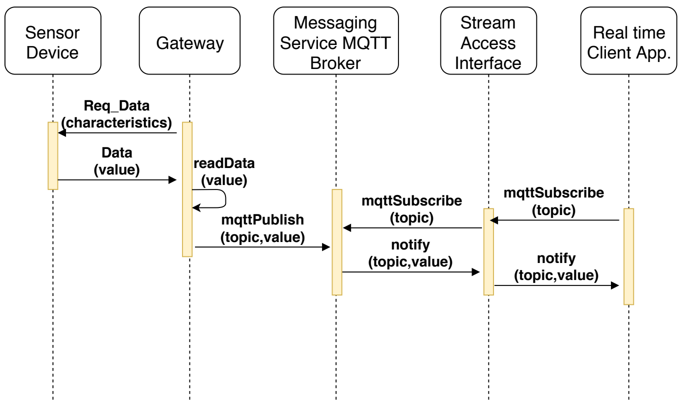

Figure 3. Stream data access interface

3.3. Device abstraction and data structure

In order to provide an unified abstraction of various devices, we represent a gateway, device together with its attached sensor in an hierarchical manner as presented in Figure 4. For batch case, each device is represented as unique HTTP URL while for stream case, every single device is represented as a distinct MQTT topic. Notice that, if a sensor is directly attached to a single device or gateway, it can be represented using the "root" name for both URL and topic. For storing various sensing data value, we utilize a key-value pair data structure represented in JSON format. The "key" part represents the sensor unique name, while the "value" part represents the sensing data measurement result.

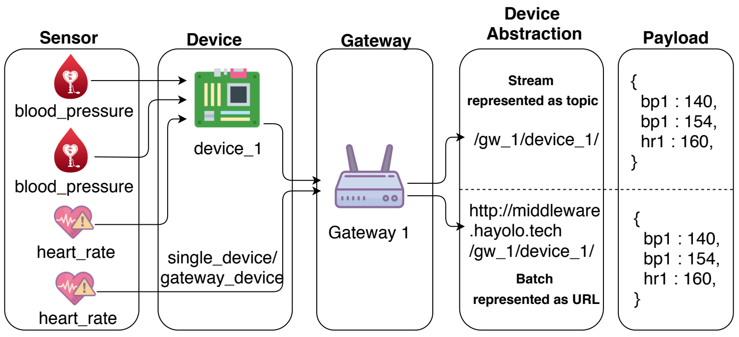
Sensor
Device
Gateway
Device Abstraction
Payload
blood_pressure
blood_pressure
heart_rate
heart_rate
device_1
single_device/ gateway_device
Gateway 1
Stream represented as topic
/gw_1/device_1/
http://middleware .hayolo.tech /gw_1/device_1/
Batch represented as URL
{ bp1 : 140, bp1 : 154, hr1 : 160, }
{ bp1 : 140, bp1 : 154, hr1 : 160, }

Figure 4. Batch and stream device abstraction and data structure

4. RESULT AND ANALYSIS

In this section we present the result and analysis of proposed system in term performance testing.

4.1. Testing environment

System design and implementation are tested in an environment depicted by Figure 5. Based on Figure 5, testing environment consists of three sections, they are sensing devices, middleware, and client application. There are two sensing devices called ESP32 which have a Bluetooth interface with a physical address as 3C:71:BF:9C:FF:1A and 30:AE:A4:42:2C:A2, and play a role as a BLE server. Then as for the middleware section, the section is divided into two parts which are called gateway and cloud. The gateway itself is implemented in Raspberry Pi which consists of a Bluetooth interface with the address as B8:27:EB:2A:A2:54. Along with that, the gateway also consists of a wireless network interface with a dynamic public IP address. While the testing is conducted, the gateway obtains a public IP address as 140.213.56.50. Beside of the gateway part, the cloud part of middleware is implemented in the virtual instance in Google Cloud Platform, addressed by a domain name as middleware.hayolo.tech and the IP address 35.188.201.80. The virtual instance is located in us-central1-a zone in Council Bluffs city, US. Finally the last section is client app which is used to conduct some tests. It has the same IP address as the gateway.

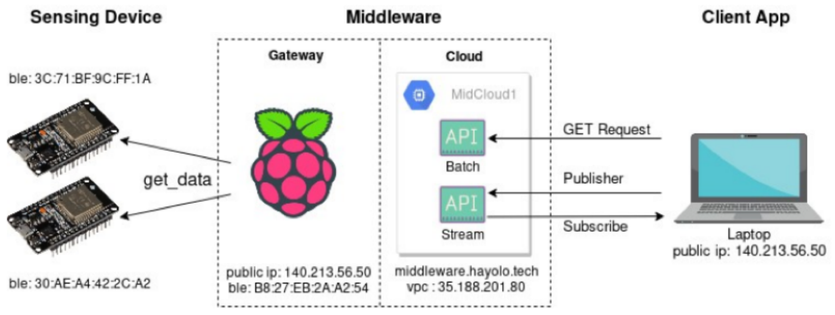

Figure 5. Testing environment

4.2. Performance testing

We test the middleware in three performance metrics including batch data access latency, concurrent connection throughput, and end-to-end stream data delay.

4.2.1. Batch data access latency

The purpose of this test is to measure the performance of proposed middleware in term of batch historical data access latency for various requested data size i.e. 0.5MB, 1MB, 5MB and 10MB. In order to perform this test, we generate 50 concurrent HTTP connections using JMeter tool. From Figure 6, we obtain that as the response time rises significantly as size of requested data increases. However, the result still shows a decent performance since with the 10MB data size, the middleware can serve the request with latency around 100ms which is still satisfy most of historical data processing requirement.

4.2.2. End-to-end stream data delay

This test is performed to compare the delay of stream data delivery and access on two different approach : using MQTT protocol as proposed in this paper as well as using HTTP as mentioned in [24]. In order to perform this test, we develop an application that retrieves the sensing data using both MQTT and HTTP protocols with various data retrieval periods i.e. 1s, 1.25s, 1.5s, 1.75s and 2s. From result presented in Figure 7, we obtain that proposed MQTT implementation can significantly reduce the retrieval delay up to 52 percent on average of all scenarios.

4.2.3. Concurrent connection throughput

The goal of this test is to measure the number of concurrent requests than can be handled by the proposed middleware for each second. In order to perform this test, we generate various amount of concurrent connections i.e. 500, 1000, 1500 and 2000 for both batch and stream functionality. From Figure 8 we obtain that the proposed stream data access mechanism can still handle up to 172 requests/second which outperforms the existing batch only mechanism in 2000 concurrent connections scenario.

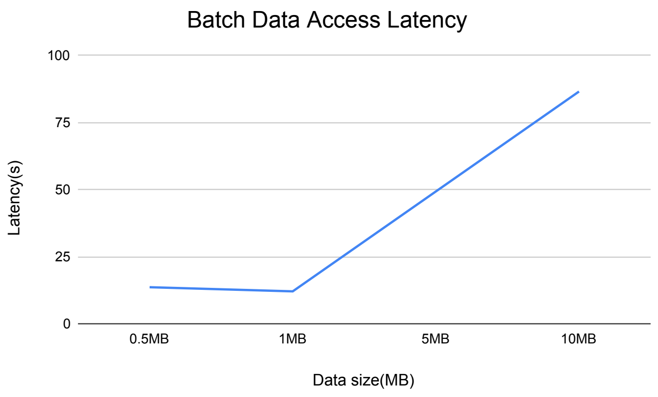

Figure 6. Batch data access response time for various requested data size

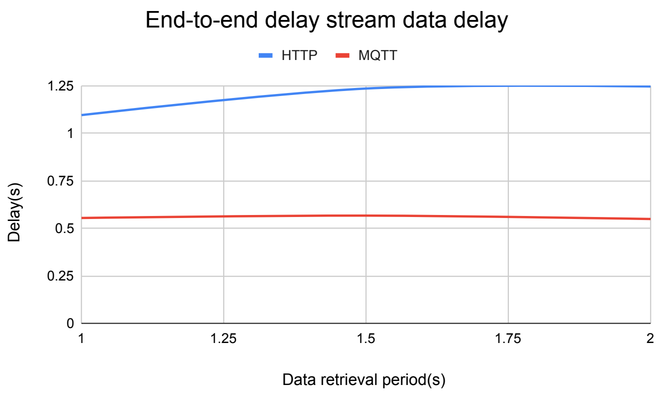

Figure 7. End-to-end stream data delay for various data retrieval periods

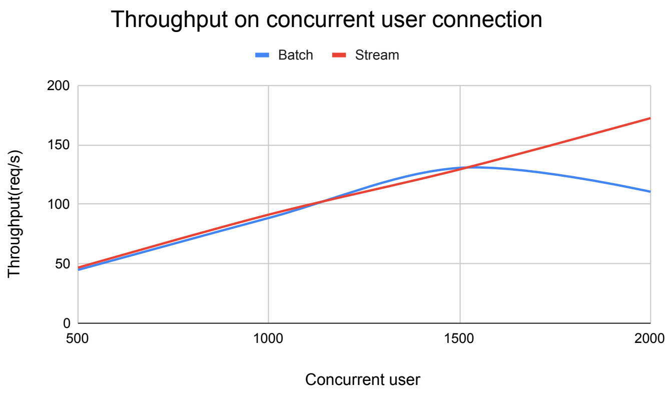

Figure 8. Throughput for various concurrent connections

5. CONCLUSION

In this paper we proposed the design of cloud based middleware for enabling the efficient web of things access over healthcare wearable devices with both batch and stream data access support. The proposed middleware consists of three components: gateway-to-cloud device, messaging service, and data access interface. The gateway-to-cloud device has an important role to perform low level sensor data collection from various wearable sensing device through BLE communication protocol. The collected data are then relayed to the cloud IoT platform using a more lightweight MQTT messaging protocol instead of HTTP. In order to provide device abstraction along with access to the stored data, the system offers two kind of interfaces : the

Restful HTTP identified by a unique URL for batch access mode and MQTT websocket interface identified by unique topic to accommodate access on sensing data in a near real time stream manner. From the testing result, we can conclude that the proposed middleware is able to provide an interoperability layer between devices and application in both stream and batch manners with reasonable performance.

ACKNOWLEDGEMENT

The acknowledgment section is optional. The funding source of the research can be put here.

REFERENCES

[1] Atzori, L., Iera, A., and Morabito, G., "The Internet of Things: A survey," *Computer Networks*, vol. 54, no. 15, pp. 2787-2805, 2010.

[2] Rahman, L.F., Ozcelebi, T., and Lukkien, J., "Understanding IoT Systems: A Life Cycle Approch," *Procedia Computing Science*, vol. 130, pp. 1057-1062, 2018.

[3] Nguyen Gia, T., Sarker, V.K., Tcarenko, I., Rahmani, A.M., Westerlund, T., Liljeberg, I., and Tenhunen, H., "Energy efficient wearable sensor node for IoT-based fall detection systems," *Microprocessor Microsystem*, vol. 56, pp. 34-46, 2018.

[4] Baker, S. B., Xiang, W., and Atkinson, I., "Internet of Things for Smart Healthcare: Technologies, Challenges, and Opportunities," *IEEE Access*, vol. 5, pp. 26521-26544, 2017.

[5] Pasha, M., and Shah, S. M. W., "Framework for E-health Systems in IoT-Based Environment," *Wireless Communications and Mobile Computing*, pp. 1-11, 2018.

[6] Dias, D., and Paulo Silva Cunha, J., "Wearable Health Devices-Vital Sign Monitoring, Systems and Technolgogies," *Sensors*, vol. 18, no. 8, 2018.

[7] Chi, Y.M., and Cauwenberghs, G., "Wireless Non-contact EEG/ECG Electrodes for Body Sensor Networks," *International Conference on Body Sensor networks*, 2010.

[8] Mouradian, V., Poghosyan, A., and Hovhannisyan, L., "PPG Sensor and Device for Continuous Mobile Human Vital Signs Monitoring," *11th Int., Conference on Mobile Ad Hoc and Sensor Systems*, 2014.

[9] Sheng Zhong, Li Wang, Bernardos, A.M., and Mei Song., "An accurate and adaptive pedometer integrated in mobile health application," *IET International Conference on Wireless Sensor Network*, 2010.

[10] Hasan, M. K., et al., "Real-Time Healthcare Data Transmission for Remote Patient Monitoring in Patch-Based Hybrid OCC/BLE Networks," *Sensors*, vol. 19, no. 5, 2019.

[11] Buyukakkaslar, M. T., et al., "LoRaWAN as an e-Health Communication Technology," *IEEE 41st Annual Computer Software and Application Conference*, vol. 2017, no. 2, 2017.

[12] Petrelis, N., Birbas, M., and Gioulekas, F., "On the Design of Low-Cost IoT Sensor Node for e-Health Environments," *Electronics*, vol. 8, no. 2, 2019.

[13] Ristevski, B., and Chen, M., "Big Data Analytics in Medicine and Healthcare," *Journal Integr. Biofarma*, vol. 15, no3, 2018.

[14] Hansen, B., et al., "Use of Electronic Health and Its Impact on Doctor-Visiting Decisions Among People With Diabetes: Cross-Sectional Study," *Journal Med.Internet Res.*, vol. 21, no. 4, 2018.

[15] Sarker, V. K., et al., "Portable multipurpose bio-signal acquisition and wireless streaming device for wearables," *IEEE Sensors Applications Symposium*, 2017.

[16] Islam, S. M. R., Kwak, D., Kabir, M. H., Hossain, M., and Kwak, K. S., "The internet of things for health care: A comprehensive survey," *IEEE Access*, vol. 3, pp. 678-708, 2015.

[17] Wang, Y., Liu, Z., and Dong, B., "Heart rate monitoring from wrist-type PPG based on singular spectrum analysis with motion decision," *38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 3511-3514, 2016.

[18] Al-fuqaha, A., Member, S., Guizani, M., Mohammadi, M., and Member, S., "Internet of Things : A Survey on Enabling," *IEEE communications surveys tutorials*, vol. 17, no. 4, pp. 2347-2376, 2015.

[19] Preuveneers, D., Berbers, Y., and Joosen, W, "SAMURAI: A batch and streaming context architecture for large-scale intelligent applications and environments," *Journal Ambient Intelligence Smart Environment*, vol. 8, no. 1, pp. 63-78, 2016.

[20] Heuer, J., Hund, J., and Pfaff, O., "Toward the web of things: Applying web technologies to the physical world," *Computer*, vol. 48, no. 5, pp. 34-42, 2015.

[21] Ragget, D., "The Web of Things: Challenges and Opportunities," *Comp.*, vol. 48, no. 5, pp. 34-42, 2015.

[22] Bhawiyuga, A., Kartikasari, D. P., and Pramukantoro, E. S., "A publish subscribe based middleware for enabling real time web access on constrained device," *9th International Conference on Information Technology and Electrical Engineering*, pp. 1-5, 2017.
[23] Banks, A., and Rahul G., "MQTT Version 3.1. 1," *OASIS standard*, vol. 29, 2014.
[24] Philip, N., et al. "Design of a RESTful middleware to enable a web of medical things," *4th International Conference on. Wireless Mobile Communication and Healthcare (Mobihealth)*, 2014.
[25] Bhawiyuga, Adhitya, et al. "Architectural design of IoT-cloud computing integration platform," *Telkomnika Telecommunication, Computing, Electronics and Control*, vol. 17, no. 3, pp. 1399-1408, 2019.

BIOGRAPHIES OF AUTHORS

Adhitya Bhawiyuga received the M. Sc. degree in computer science from Pusan National University, Republic of Korea (2013). He is now affiliated with Brawijaya University as lecturer and researcher. His current research interests include internet of things, distributed systems, cloud computing, and information security.

Satria Adi Kharisma received the B.S. degree in computer science from Faculty of Computer Science, Brawijaya University. His current research interests include internet of things and open source software.

Bagus Jati Santoso received the PhD degree in computer science from National Taiwan University Of Science And Technology (2016). He is now affiliated with Sepuluh Nopember Institute of Technology (ITS) as lecturer and researcher. His current research interests include data engineering, knowledge discovery and mobile computing.

Dany Primanita Kartikasari received the M.Sc. degree in computer science from Sepuluh Nopember Institute of Technology (ITS). She is now affiliated with Brawijaya University as lecturer and researcher. Her current research interests include internet of things and cloud computing.

Annisa Puspa Kirana received the M.Sc. degree in computer science from Bogor Agricultural University, Indonesia. She is now affiliated with Department of Informatics, State Polytechnic of Malang as lecturer and researcher. Her current research interests include geographical information system (GIS) and database.