International Journal of Grid and Distributed Computing
Vol. 11, No. 2 (2018), pp.55-68
http://dx.doi.org/10.14257/ijgdc.2018.11.2.06

Management of Mobility of CoAP-based IoT Device for Continuous Movement Management

Joo Ho Choi[1], Eun-Surk Yi[2], Ji-Youn Kim[3] and Byung Mun Lee[4*]

[1]Dept. of IT Convergence Engineering, Gachon University, Seongnam-si, Korea
[2,3]Dept. of Exercise Rehabilitation & Welfare, Gachon University, Incheon, Korea
[4]Dept. of Computer Engineering, Gachon University, Seongnam-si, Korea
[1]onlysm0707@nate.com, [2]yies@gachon.ac.kr, [3]eve14jiyoun@naver.com, [4]bmlee@gachon.ac.kr

Abstract

IoT devices such as smart watches and smart bands are used in an IoT environment to provide healthcare services. These IoT devices measure users' continuous movement and transmit the results through the network. However, if the IoT device is out of the network range, continuous services are interrupted. In addition, it is difficult to monitor and control a large number of IoT devices connected to the network with only a single integrated management server. Therefore, in this study, we propose a smart model for mobility management of IoT devices by defining a smart gateway to manage IoT device connection to the network. In this model, we also designed and implemented a protocol for mobility management between Smart gateway and IoT device. Finally, comparison of handover delay time with existing mobility management protocols confirmed that the delay time was approximately halved. We additionally measured the request/response time according to the distance between the smart gateway and the mobile IoT device. This confirmed the effectiveness of the IoT device mobility management method using the proposed protocol, which is about 80 ms faster than the existing protocol.

Keywords: *IoT, CoAP, Mobility Management, Clustering*

1. Introduction

IoT (Internet of Things) is a technology that enables provision of various application services through various IoT devices connected to the network, and is widely used in various fields such as health care and smart homes. IoT devices transmit data that has been acquired and processed by a device that can be uniquely identified on the network. The transmitted data is provided as a service to the user through the IoT application [1]. For example, smart watches and smart bands are used in healthcare services. These wearable devices attached to the user's wrist measure health information such as heart rate and amount of exercise and analyze exercise and sleep data based on the results of measurement [2,3]. These devices can measure the movements and exercise posture of hospitalized patients to provide appropriate exercise for patients and thus accelerate recovery. However, if these IoT devices are not connected to the network or if the connection is interrupted, IoT service cannot be provided. Mobile IoT devices in particular can easily leave the network range, which may lead to problems such as frequent service interruption.

Many studies have been conducted on the management of mobile IoT devices for the purpose of providing continuous IoT service. [4]. IETF has standardized various mobility management protocols, such as Mobile IPv4/v6, FMIPv4/v6 and PMIPv4/v6 [5, 6, 7, 8,

Received (October 15, 2017), Review Result (January 15, 2018), Accepted (January 24, 2018)
[*] Corresponding Author

9,10]. However, these mobility management protocols require high signaling overhead, high-level CPU processing, and significant power consumption. IoT devices use different CPU, memory, battery capacity, and communication bandwidth. The IETF's Light-Weight Implementation Guidance (LWIG) classifies according to IoT device limits [11]. The criterion for the limitation is the memory used by the device and the loadable code size, from the most limited class of 0 with less than 10 KB of memory and less than 100 KB of loadable code to class 2 with more than 50 KB of memory and loadable code capacity of 250 KB or more. These limited devices use a standard such as IEEE 802.15.4 appropriate for low-speed and low-power applications. In addition, some studies have been conducted on the application of CoAP (constrained application protocol), appropriate for limited devices, to IEEE 802.15.4 [12,13,14]. However, the above-described mobility management protocols are difficult to use in limited devices because they require high performance. A mobility management system based on CoAP has been proposed to solve this problem [15]. This system manages the mobility of each node by managing the IP addresses of CoAP nodes through WMMS (web of things mobility management system) in an integrative manner. For example, CoAP node A and CoAP node B are connected to the network. CoAP nodes A and B receive each other's IP address from the WMMS for communication. When CoAP node A moves out of the network range, it notifies the WMMS that it is moving. WMMS determines that CoAP node A is no longer connected to the network and does not transmit the address of CoAP node A to other CoAP nodes. After that, CoAP node A notifies that it is connected to the network and is reconnected to WMMS and its current IP address. The WMMS updates the IP address of CoAP node A and transmits the IP address in the same way as before to support communication between CoAP nodes. However, if tens or hundreds of thousands of IoT devices exist at the same time, WMMS using a single server is difficult to monitor and control the mobility management of all devices. In addition, if WMMS does not operate appropriately, the mobility management of IoT devices is interrupted because it cannot detect the movement of CoAP nodes or receive updated IP address.

To solve this problem, a method is required to identify IoT devices within the network range and to manage mobility by distributing and arranging smart gateways operating as a single management node. Therefore, in this paper, we propose an IoT device mobility management model. We also designed a mobility management protocol between smart gateway and IoT devices based on CoAP to be used with this model. In this model, smart gateways that connect neighboring IoT devices to the network are distributed. The distributed smart gateways are connected in a single cluster and each cluster has a cluster manager that manages the smart gateway. If there are multiple clusters, the server managing the cluster manager is the mobility management server. The smart gateways connected by clustering identify and manage the information on neighboring IoT devices. Information about the IoT devices connected to the smart gateway is also transmitted to the cluster manager. As the distributed smart gateways are clustered, mobility management for IoT devices in the cluster based on the cluster manager becomes possible even if the mobility management server does not operate appropriately. In this paper, we implemented and tested the protocol and evaluated its efficiency by comparing its performance with that of the mobility management protocol proposed previously to validate the proposed and designed protocol.

Following the introduction in Chapter 1, we address the requirements for mobility management of IoT devices and the existing CoAP-based mobility management protocol in Chapter 2, and describe the IoT device mobility management model and protocol that reflects the requirements that we have designed in Chapter 3. Chapter 4 addresses and evaluates the experiment results to confirm the effectiveness of the proposed protocol. Finally, in Chapter 5, we discuss the results of this study and suggest directions for future studies.

2. Related Researches

2.1. Issues Related to IoT Device Management

Many IoT devices must be managed efficiently because they provide services. IoT devices with limited battery capacity and mobility are especially difficult to manage. For example, if a battery-powered device runs out of battery, the network signal becomes weak, resulting in the connection disruption. In addition, mobile IoT devices may go out of network range or suddenly enter a new network range and become available. To solve these problems, studies have been conducted on issues related to the management of IoT devices and requirements and solutions [18].

Figure 1 shows issues related to IoT devices with mobility. Activity1 shows a case in which A1, which is a new IoT device not identified in the existing network range, enters. Activity2 shows the process of determining whether that device is available by monitoring multiple IoT devices within the network range. Activity2-1 shows the process of selecting and connecting an available IoT device, while Activity2-2 shows the process of replacing the existing IoT device with alternative IoT device in the event of insufficient battery or connection interruption. Finally, Activity3 shows a case in which an IoT device that should be within network range disappears due to battery depletion or leaving the network range. Studies have been conducted on IoT device management frameworks to solve problems arising from mobile IoT device issues, and the following functional requirements have been defined.

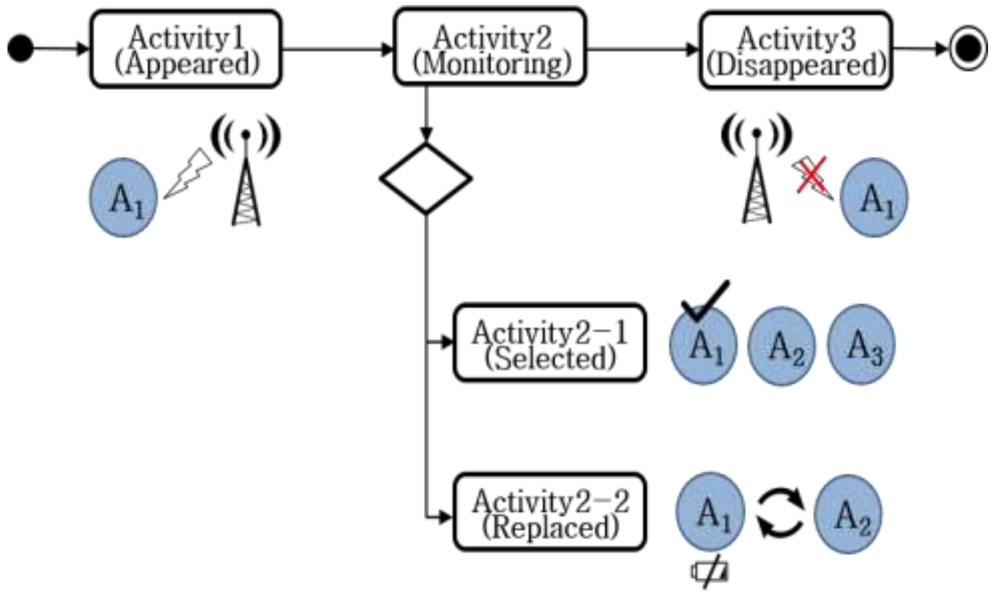


Figure 1. Mobile IoT Device Issues

First, it is necessary to discover a device that enters and operates within the network. The discovered device is connected to the network and is used to provide services through IoT applications. Second, it is necessary to monitor the status of the IoT device operating within the network range to determine whether it is available or is out of the network range. Third, it is necessary to provide a connection so that the IoT device sought by IoT applications can be used. Finally, if the device is not operating normally due to network instability, battery consumption, or if a higher quality device is identified, the device must be replaceable. These requirements must be met to manage IoT device mobility. However, performance- related issues such as overhead, handover delay, and packet loss rate should be taken into consideration.

2.2. CoAP-based Mobility Management Protocol

Various existing mobility management protocols are used to avoid service interruption caused by the mobility of IoT devices. For example, Mobile IPv4/v6, FMIPv4/v6, and PMIPv4/v6 have been standardized for the management of network layer mobility, and Migrate TCP and mSCTP (mobile stream control transmission protocol) have been standardized for the management of transport layer mobility [19,20]. However, the standardized mobility management protocols are very complicated and require high overhead and high-level CPU processing, so they are not appropriate for a limited network environment.

Therefore, the MPCS (mobility protocol of CoAP sensor) and the mobility management system have been proposed as a mobility management protocol appropriate for a limited network environment, and Figure 2 shows the procedure of managing the mobility of CoAP nodes.

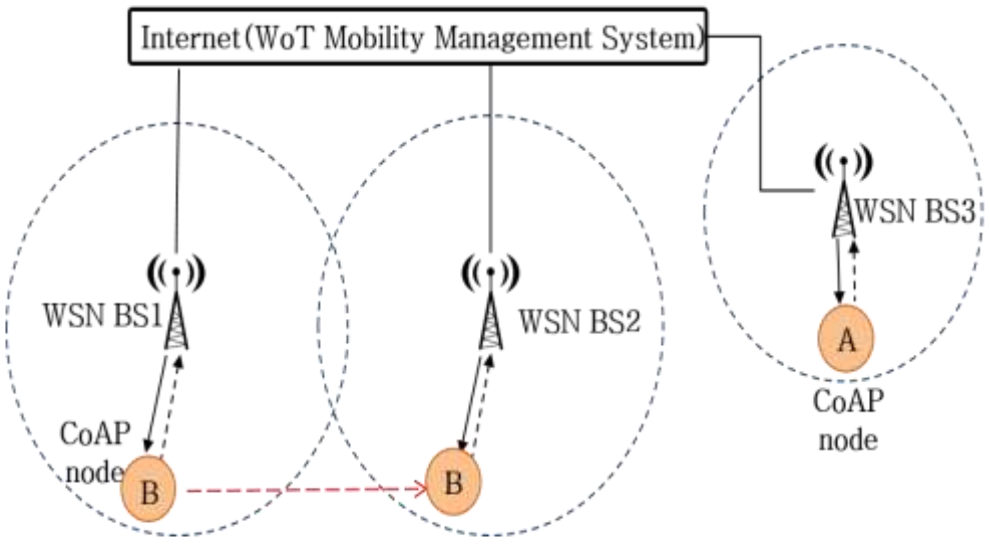


Figure 2. CoAP Sensor Mobility Management Procedure

The mobility management system comprises a WMMS (web of things mobility management system) that manages CoAP nodes and nodes. Each CoAP node functions as server and client and transmits the data between nodes. WMMS includes a MMT (mobility management table) to manage the mobility of the CoAP node and the IP addresses of each CoAP node through it. CoAP node A can therefore exchange data after receiving the IP address through the WMMS for communication with CoAP node B. Next, it is assumed that CoAP node B moves from WSN1 to WSN2. CoAP node B detects the RSS (radio signal strength) signal of the physical layer when it deviates from the network range of WSN BS1. When this signal value drops, CoAP Node B notifies the WMMS that it is preparing for handover. Upon receiving this signal, the WMMS notifies CoAP node A that CoAP node B is switched to the handover state. After that, CoAP node B moves into the new network WSN BS2 range and notifies the WMMS that it is connected to the network. The WMMS updates the new IP address of node B and notifies CoAP node A that the handover state of CoAP node B has ended. After that, communication between CoAP node A is reestablished.

However, when tens or hundreds of thousands of CoAP nodes are present simultaneously, WMMS management of all nodes will incur overhead, which may result in handover time delay between CoAP nodes or increased packet loss. When WMMS does not function normally, mobility management of CoAP nodes is completely stopped. Therefore, we used a method of distributing and arranging smart gateways to manage IoT devices with mobility for this study. We also proposed a mobility management model based on CoAP, which is a protocol appropriate for a limited IoT environment, and an application protocol appropriate for this model.

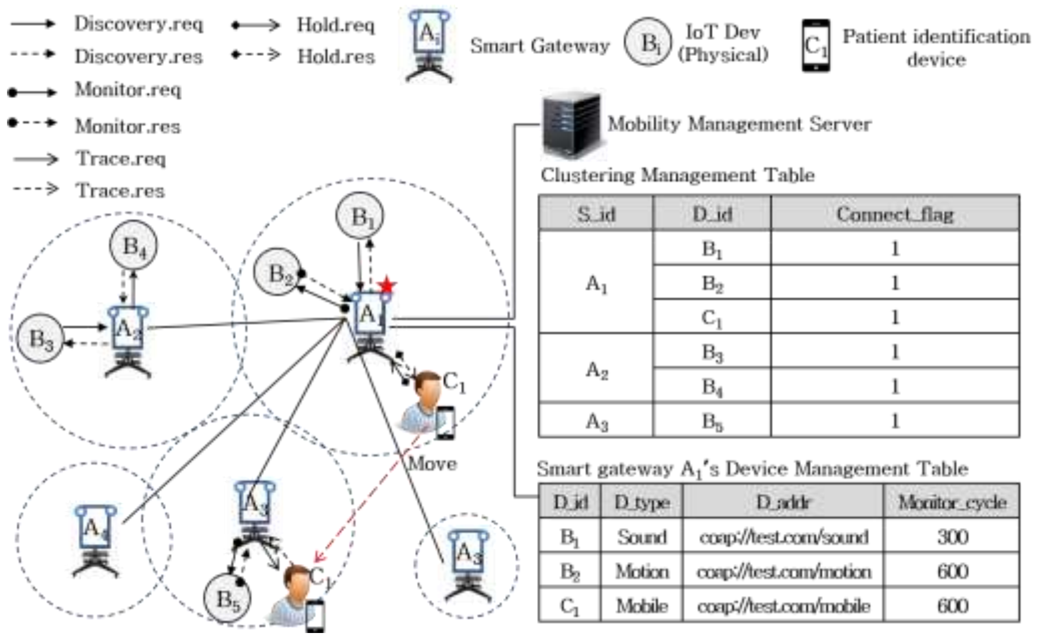
3. Continuous Mobility Management of IoT Devices

3.1. Smart Model for Continuous Mobility Management using IoT Devices

As a large number of IoT devices operate in a large-scale IoT environment, they must be efficiently managed by reducing overhead. For example, in rehabilitation equipment for hospitalized patients, patient identification devices are required to manage the continuous movement of each patient. In addition, the movement information measured by the identification device must be transmitted to the network. To this end, we used a clustering technique that groups IoT devices distributed over a network based on a certain criterion. The smart poster devices in the hospital are grouped by clustering to receive measurement information from neighboring patient identification devices. At this time, the smart poster device functions as a smart gateway that connects the patient identification device to the network. The smart gateway is an interface device that provides the function to manage each IoT device to the gateway through which IoT devices are connected to the network. In addition, it manages the information of each IoT device connected through the cluster manager by clustering distributed gateways. In this study, we proposed a smart model for the integrative management of the mobility of neighboring IoT devices using distributed smart gateways.

As shown in Figure 3, distributed gateways are grouped to form a cluster. The smart gateways A1-A5 are grouped and are connected based on the cluster manager A1 for operation. In addition, there are IoT devices B1, B2, B3, B4, B5, and C1 operating around the smart gateways A1-A5. IoT devices B1, B2, and C1 operate around gateway A1. Since they must pass through the smart gateway A1 to transmit measured information to the network through the mounted physical sensors, they can operate appropriately only when they are connected to the smart gateway A1. In this way, the smart gateway manages IoT devices with mobility through connection to neighboring IoT devices. Each smart gateway contains information about its neighboring IoT devices in its own DMT (device management table). In addition, each smart gateway transmits information about the IoT devices it manages to the cluster manager A1. A1 manages the smart gateways connected to the cluster through the information stored in the clustering management table (CMT). The cluster manager has information about smart gateways and IoT devices in the cluster, and the MMS (mobility management server) manages cluster managers in an integrative manner. In this environment, the smart gateway manages the mobility of neighboring IoT devices with three main functions.

The first function is to discover IoT devices operating within the smart gateway range. In this model, when the IoT device is booted, it tries to connect with the smart gateway through a discovery message. When the IoT device B1 starts to operate, B1 broadcasts a Discovery.req message indicating the start of operation. Since IoT device B1 operates within range of the smart gateway A1 network, the smart gateway A1 receives the Discovery.req message broadcasted by B1. The smart gateway A1, which normally receives the message, registers information about the IoT device B1 included in the message in the DMT and responds with a Discovery.res message to complete the process of discovering the IoT device. Information about the IoT device B1, including identification ID, device address, and monitoring period is used when the smart gateway A1 monitors the IoT device.



Clustering Management Table

S_id	D_id	Connect_flag
A_1	B_1	1
	B_2	1
	C_1	1
A_2	B_3	1
	B_4	1
A_3	B_5	1

Smart gateway A_1's Device Management Table

D_id	D_type	D_addr	Monitor_cycle
B_1	Sound	coap://test.com/sound	300
B_2	Motion	coap://test.com/motion	600
C_1	Mobile	coap://test.com/mobile	600

Figure 3. Mobility Management Process using Smart Gateways and IoT Devices

The second function is to monitor the status of the IoT devices connected to the network. The smart gateway regularly verifies that detected IoT devices correctly maintain connection. A monitor message is used at this time. As shown in Figure 6, the smart gateway A1 transmits a Monitor.req message to the IoT device B2 within its network range. This message is created based on the information stored in the DMT managed by the smart gateway A1. D_id in DMT identifies each IoT device and D_addr represents the address value of IoT devices. Monitor_cycle is the period of the Monitor.req message transmitted by the smart gateway and the IoT device B2 transmits a Monitor.req message every 600 seconds.

The final function is to detect and trace the movement of IoT devices to provide appropriate service. As shown in Figure 6, Mobile C1, a typical mobile device, operates within the range of smart gateway A1. Since mobile C1 is connected to the network through smart gateway A1, A1 includes information about C1 in its DMT and transmits Monitor messages to C1 at intervals of 600 seconds, which is the period of Monitor_cycle. The mobile C1 performs a handover operation when it moves out of the range of the network of smart gateway A1. To do this, the mobile C1 continuously detects the RSS value from the connected smart gateway A1, and transmits a Hold.req message to the smart gateway A1 when the detected value falls below a predetermined standard. Upon receiving this request, the smart gateway A1 deletes the items for the mobile C1 belonging to its own DMT and notifies the smart gateway acting as the cluster manager that the C1 managed by the smart gateway A1 is in a Hold state. After changing the Connect_flag value of the corresponding mobile C1 in its own CMT to 0 and then asking the smart gateways clustered with it to search the mobile C1, smart gateways A1-A5 broadcast a Trace message to track the mobile C1. At this time, the mobile C1 which has moved into the network range of the smart gateway A3 transmits Trace.res in response to the Trace.req message broadcasted by the smart gateway A3. Upon receiving this message, the smart gateway A3 registers a new mobile C1 in its DMT and notifies the cluster manager of this fact. The cluster manager confirms that the mobile C1 is appropriately connected to the other smart gateway and updates the CTM accordingly.

In the smart model for managing IoT mobility, a network protocol is needed to support transmission between a smart tracker and IoT device. CoAP, which is the basis of the mobility management protocol proposed in the previous studies, will enable it to transmit the IoT device data in a limited environment or efficiently with limited resources. Therefore, the application of CoAP-based MMPI (mobility management protocol of IoT device) to the smart model enables it to manage the mobility of IoT devices through a smart tracker.

3.2. Design of IoT Device Mobility Management Protocol

The mobility management protocol for IoT device (MMPI) to be used in the IoT device mobility management model is broadly classified into two types: a protocol for the transmission between IoT devices and smart gateways and a protocol for the transmission between the smart gateways that are responsible for cluster managers. Figure 4 shows the structure of the data transmitted between IoT devices and smart gateways.

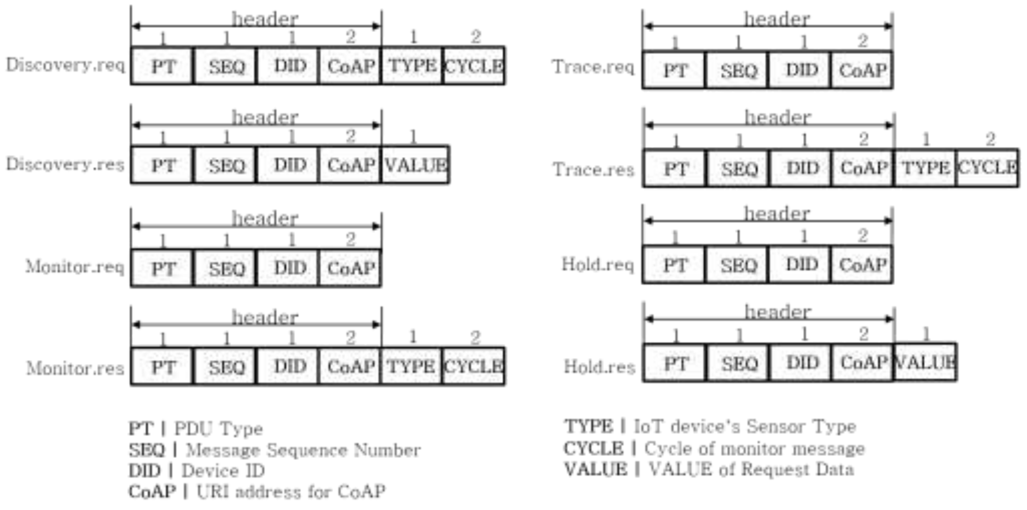

header
Discovery.req | PT 1 | SEQ 1 | DID 1 | CoAP 2 | TYPE 1 | CYCLE 2
Discovery.res | PT 1 | SEQ 1 | DID 1 | CoAP 2 | VALUE 1
Monitor.req | PT 1 | SEQ 1 | DID 1 | CoAP 2
Monitor.res | PT 1 | SEQ 1 | DID 1 | CoAP 2 | TYPE 1 | CYCLE 2
Trace.req | PT 1 | SEQ 1 | DID 1 | CoAP 2
Trace.res | PT 1 | SEQ 1 | DID 1 | CoAP 2 | TYPE 1 | CYCLE 2
Hold.req | PT 1 | SEQ 1 | DID 1 | CoAP 2
Hold.res | PT 1 | SEQ 1 | DID 1 | CoAP 2 | VALUE 1
PT | PDU Type
SEQ | Message Sequence Number
DID | Device ID
CoAP | URI address for CoAP
TYPE | IoT device's Sensor Type
CYCLE | Cycle of monitor message
VALUE | VALUE of Request Data

Figure 4. Structure of the Data Transmitted between IoT Device and Smart Gateway

The message header consists of the first to fourth fields. The first field, PT, used to identify each message, consists of 1 byte. The second field, SEQ, representing the message serial number, consists of 2 bytes. Monitor messages are transmitted regularly according to the defined CYCLE. If transmission is performed frequently at intervals of 1 sec or less, about 3600 messages are transmitted per hour. Therefore, a message size of 2 bytes is required to use up to 65535 messages. The third field represents the ID of the IoT device or the smart gateway transmitting the message. It consists of 1 byte because it stores the value for distinguishing the unique ID of the IoT device and smart gateway. The fourth field represents the address for utilizing the URI in the CoAP. This field can express a URI of up to 50 bytes, and its size is determined variably according to the URI length. Excluding the four fields in the message header, items are used according to the type of message. The TYPE field represents the type of IoT device and consists of 1 byte. The CYCLE field represents the monitoring period of the IoT device. The value of the CYCLE field is stored in the DMT of the smart gateway, and the smart gateway regularly transmits monitor messages to the IoT device according to the stored value. The VALUE field represents the result value for the request.

Figure 5 shows the structure of the data transmitted between the smart gateway and the smart gateway responsible for the cluster manager. Like the structure of the data transmitted between the IoT device and the smart gateway, it includes the message headers with four fields and has the same configuration. Since Register messages include information about the IoT device detected by the smart gateway, the ID value of the IoT device detected using the DID field is transmitted to the cluster manager. Since Inform messages are used to transmit information about the IoT device detected by the smart gateway, the DID field is included. Finally, since the Explore message is used to trace when the cluster manager transmits to neighboring smart gateways to trace the IoT device, the value of the IoT device to be tracked is included in the DID field.

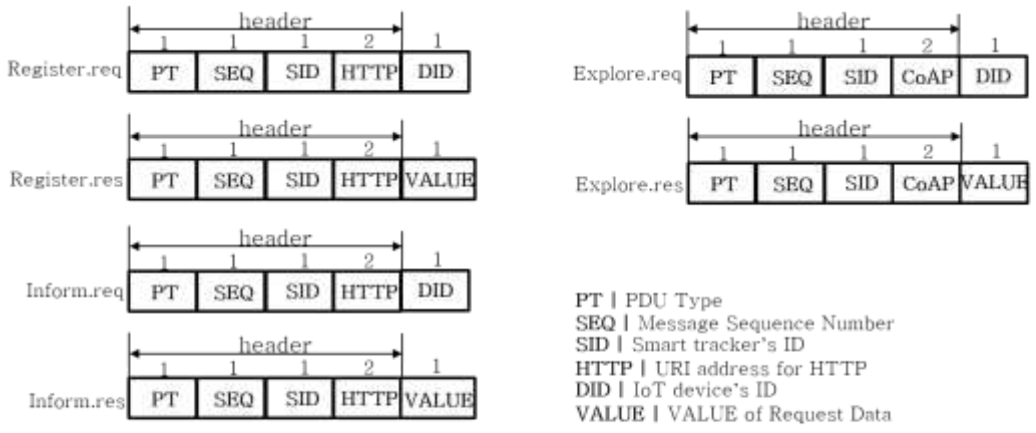
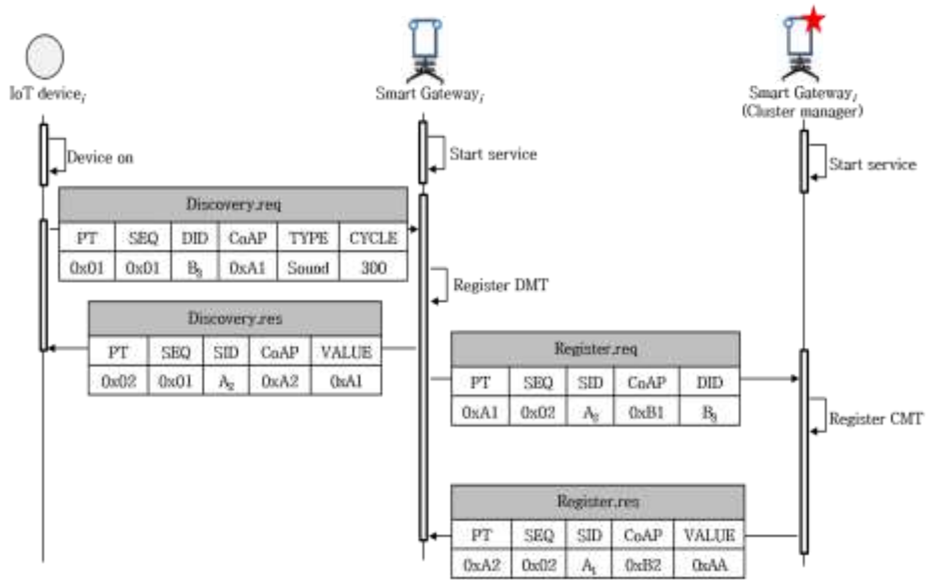


Figure 5. Structure of the Data Transmitted between Smart Gateway and Cluster Manager

Figure 6-8 is a sequence diagram that shows how the smart gateway and cluster manager manage the mobility of IoT devices using the defined protocols. As shown in Figure 7, when the IoT device operates, it broadcasts a Discovery.req message. The Discovery.req message includes the type of PDU, message sequence number, DID representing the ID of the IoT device, and CoAP address value. It also has a TYPE representing the type of IoT device and CYCLE representing the monitoring cycle. The smart gateway receives this broadcast message and registers the information of the IoT device included in the message in its DMT. When the DMT registration process is complete, the smart gateway responds to the IoT device with a Discovery.res message. After that, the smart gateway transmits a Register.req message to the cluster manager, and this message includes the SID, which is the ID of the smart gateway, along with the DID, which is the ID of the IoT device ID registered in the corresponding smart gateway. This information is registered in the CMT of the cluster manager. After that, the cluster manager responds by including the registration results in VALUE as a Register.res message.

Figure 7 shows how IoT devices move out of the smart gateway range. When the IoT device moves out of the network range of the smart gateway, the IoT device detects the RSS value from the smart gateway. If the detected RSS value falls below a predetermined reference value, a Hold.req message is requested. The smart gateway responds with a Hold.res message and notifies the cluster manager that the IoT device has moved out the network range with an Inform message. As before, the cluster manager changes the Connect_flag value of the CMT and transmits a response message to the smart gateway. The smart gateway that received the response message deletes the corresponding IoT device information in the DMT.



Discovery.req

PT	SEQ	DID	CoAP	TYPE	CYCLE
0x01	0x01	B_3	0xA1	Sound	300

Discovery.res

PT	SEQ	SID	CoAP	VALUE
0x02	0x01	A_2	0xA2	0xA1

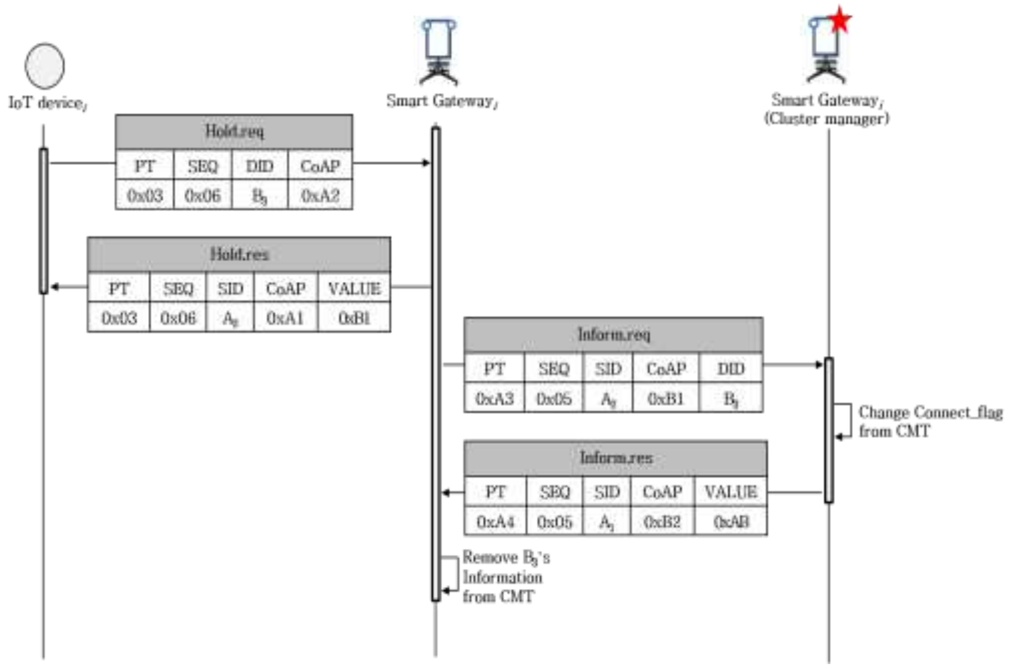
Register.req

PT	SEQ	SID	CoAP	DID
0xA1	0x02	A_2	0xB1	B_3

Register.res

PT	SEQ	SID	CoAP	VALUE
0xA2	0x02	A_1	0xB2	0xAA

Figure 6. Sequence Diagram of IoT Device Registration Process



Hold.req

PT	SEQ	DID	CoAP
0x03	0x06	B_3	0xA2

Hold.res

PT	SEQ	SID	CoAP	VALUE
0x03	0x06	A_2	0xA1	0xB1

Inform.req

PT	SEQ	SID	CoAP	DID
0xA3	0x05	A_2	0xB1	B_3

Inform.res

PT	SEQ	SID	CoAP	VALUE
0xA4	0x05	A_1	0xB2	0xAB

Figure 7. Sequence Diagram of IoT Device Movement Process

Figure 8 shows the process of searching IoT devices within the cluster range of the smart gateway when the value of Connect_flag is changed to 0. The IoT device whose Connect_flag value from the cluster manager CMT has changed to 0 via an Inform message requested by the smart gateway is likely to move and operate within the network range of other smart gateways in the cluster. Therefore, the cluster manager transmits an Explore.req message to smart gateways to search for IoT devices whose Connect_flag value is changed to 0. The smart gateways connected to the cluster manager that received the request message broadcast each Trace.req message.

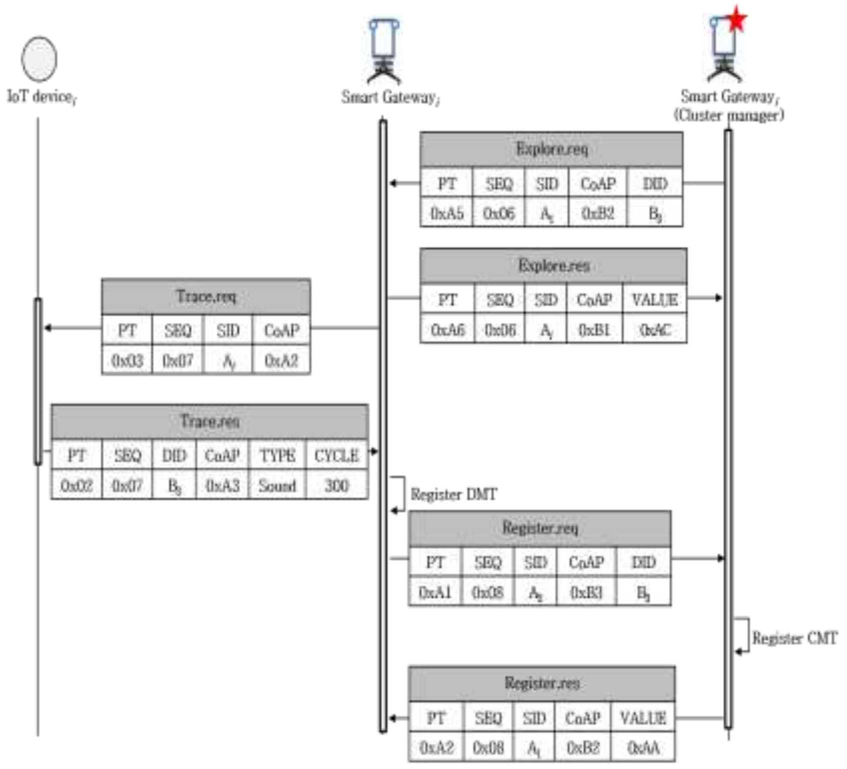


Figure 8. Sequence Diagram of IoT Device Tracing Process

4. Experiment and Evaluation

4.1. Experimental Overview and Environment

To evaluate the performance of the proposed MMPI in this study, we built and experimented with a test bed using the IoT device mobility management model. Two experiments were conducted like Figure 9.

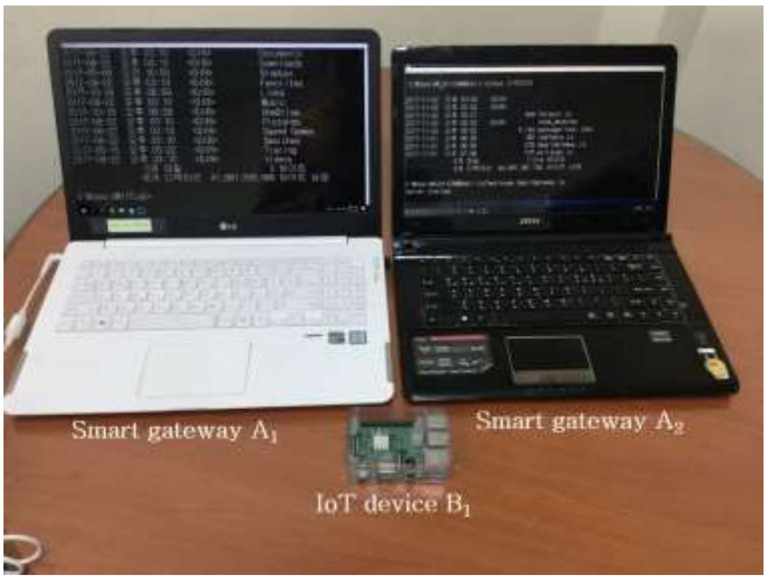
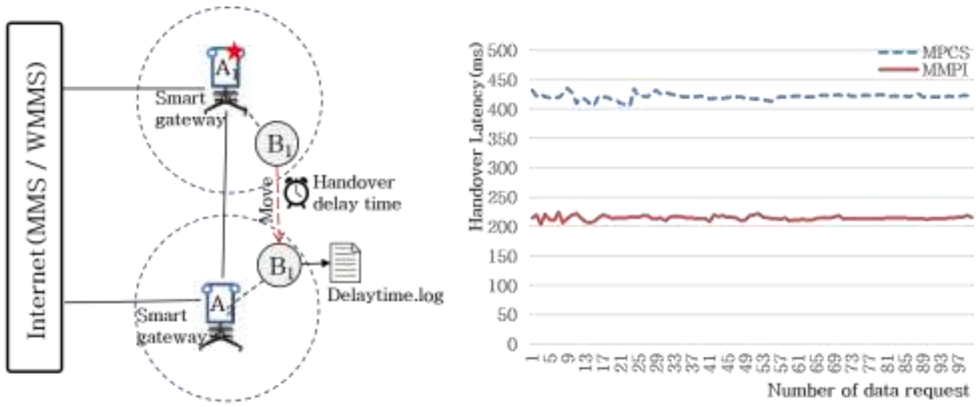


Figure 9. Experimental Environment for the Evaluation of MMPI Performance

The first experiment was to evaluate availability by measuring the delay time according to the movement of the IoT device, and the second experiment was to evaluate the efficiency of the MMPI at providing continuous IoT service. Test bed is built to evaluate two experiments with the following specifications. Smart gateways A1 and A2 were implemented using the Nodejs framework v6.9.1 in Windows OS, and IoT device B1 was implemented using Raspberry Pi 3, which supports wireless networks, to ensure mobility. Based on Raspbian OS v4.4.49, the program was implemented using JavaScript on the Nodejs Framework v4.4.0

4.2. Experiment to Evaluate Availability According to the Movement of IoT Devices

In this experiment, we measured the handover delay time due to the movement of IoT devices to evaluate the availability of mobile IoT devices. Figure 10(a) shows the process by which the IoT device B1 moves out of range of the smart gate A1 and is connected to the smart gateway A2. We compared this process by applying both the WMMS and MPCS used in the proposed CoAP-based mobility management system. In the conventional method, A1 and A2 only function as internet connection gateways. We measured the delay time for the IoT device B1 to be connected to the internet again through gateway A2 when it was out of the range of gateway A1. We also compared the delay time of each method by creating a Delaytime.log file storing the delay time.



(a) Measuring handover delay time in mobility management model

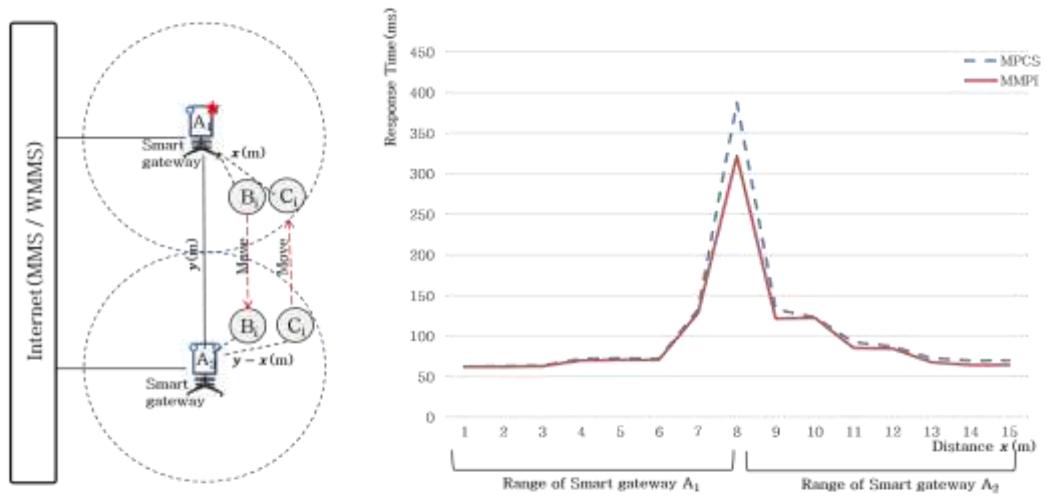
(b) Handover delay time measurement result in mobility management model

Figure 10. Measurement Results of Handover Delay Time in Mobility Management Model

Figure 10(b) shows the results of comparing the handover delay time when the IoT device moves between the MPCS of the existing studies and the MMPI proposed in this study. MPCS handover delay time has a range of about 400 to 450 ms. Since MMPI has a range of about 200 to 230 ms, the delay time was approximately halved. This is because the smart gateway detected and processed the IoT movement rather than the time it took to reach the WMMS through the gateway, so it was judged that the delay time was decreased. Therefore, since the IoT mobility management model using MMPI decreases handover delay time more than the existing model, the IoT device was judged to have higher availability when it moves because it can provide continuous IoT service through a faster connection.

4.3. Experiment to Evaluate IoT Service Efficiency

In the second experiment, we measured the data transmission time according to the distance between the smart gateway and the IoT device. The mobile IoT device has different network speeds depending on the distance from the gateway. Therefore, it is necessary to confirm how efficiently IoT service can be provided. To measure this, we measured the time taken to send a request to an IoT device and to receive a response according to network range. As shown in Figure 11(a), the distance between smart gateway A1 and A2 was defined as y (m), and the distance between IoT device B1 and smart gateway A1 was defined as x (m). Accordingly, the distance between the smart gateway A2 and the IoT device B1 is y-x (m). In the experiment, the value of y was defined as 15 m, and it was measured by moving it by 1 m increments. IoT device C1 was defined as moving from gateway A2 to A1 and had the same specifications and performance as B1. We measured by moving the IoT device B1 from the smart gateway A1 to A2 and C1 from A2 to A1. Each IoT device B1 and C1 responded to 100 requests through the smart gateway, and the average of their response times was measured. The time taken for the request from the server to be sent to the IoT device through the gateway and responded to was also measured and compared with the existing system.



(a) Measuring the response time between IoT devices and smart gateway

(b) Response time measurement result between IoT device and smart gateway

Figure 11. Response Time between IoT Device and Smart Gateway

Figure 11(b) shows the results of measuring the response time by distance between IoT device and smart gateway. This experiment also shows the comparison with the system proposed previously. It can be seen that the response time increases as the distance between IoT device and smart gateway increases. At a distance of 7-8 m from the smart gateway A1, the device is connected to the smart gateway A2 and receives a response to the request. When the distance from the smart gateway is not very large, there is almost no difference in response time between the model using MMPI and the existing system. On the other hand, when IOT device movement is monitored and requests and responses are made through connection with new gateway, the mobility management model using MMPI has a fast speed of approximately 80 ms. This is because IoT devices can be detected quickly and connected to other gateways. Therefore, there is not much difference in response time between IoT devices and smart gateways when compared with the existing system. In addition, the IoT device can provide more effective service than was previously possible when it moves as it can respond more quickly.

5. Conclusion

In this study, we proposed an IoT device mobility management model for managing mobile IoT devices and designed an MMPI to be used in the model. The IoT device mobility management model groups the distributed smart gateways into clusters and manages the mobility of the neighboring IoT devices. This is not mobility management through a single server. Since it uses the gateway through which the IoT device is connected to the network, it can manage efficiently through distributed processing. We also defined and designed a CoAP-based MMPI appropriate for the proposed model.

Two experiments were conducted to evaluate the performance of the designed MMPI with the proposed IoT device mobility management model. The first experiment was to evaluate the availability of mobile IoT devices. In this experiment, we measured the delay time occurring when the IoT device moves out of the smart gateway range and is connected to the network through other smart gateways. Results confirmed that the proposed system had less delay time than the existing system. The second experiment concerned efficiency in IoT service provision according to the distance between IoT device and smart gateway. In this experiment, we measured the response time to requests by distance. We did not find any significant difference from the existing system at near distances, but found a faster response time when the IoT device was re-connected to the network because the distance was too large, so we judged that it can provide more efficient service. To improve the performance of the proposed mobility management model and protocol, it is necessary to further study smart gateway clustering techniques. It is also necessary to study efficient distributed processing by applying the proposed clustering method as one of various clustering methods studied to date. It is also necessary to supplement the proposed protocol to make it suitable for various clustering methods.

Acknowledgments

This work was supported by the Ministry of Education of the Republic of Korea and the National Research Foundation of Korea (NRF-2016S1A5B6913737).

This paper is a revised and expanded version of a paper entitled “Continuous Mobility Management of CoAP-based IoT Devices” presented at 2017 1st International Workshop on Cultural and Technological Exchange and Mutual Development of the Pacific Rim Countries, Cheonan-si, Korea, (2017), December 16-17.

This paper is reconstructed in part on an M.S. thesis by one of the authors (Joo Ho Choi), submitted to the Gachon University Graduate School in 2018.

References

[1] Korea IoT Association, http://www.kiot.or.kr/webzine/2015/12/IoTF_standard_000_16.pdf, **(2015)**.
[2] Fitbit, https://www.fitbit.com/kr/home, **(2017)**.
[3] Samsung gear-s3, http://www.samsung.com/sec/wearables/gear-s3-frontier-r760, **(2017)**.
[4] S. K. Park, “Proposal of a mobility management scheme for sensor nodes in IoT”, Convergence Society for SMB, vol. 6, no. 4, **(2016)**, pp. 59-64.
[5] C. Perkins, “IP Mobility Support for IPv4”, Nokia Research Center, California, **(2002)**.
[6] D. Johnson, C. Perkins, “Mobility Support in IPv6”, Nokia Research Center, California, **(2004)**.
[7] C. Perkins, “Mobile IPv4 Fast Handovers”, Nokia Siemens Networks, Espoo, **(2007)**.
[8] R. Koodlim “Mobile IPv6 Fast Handovers”, Starent Networks, Massachusetts, **(2008)**.
[9] K. Leung, G. Dommety, P. Yegani and K. Chowdhury, “WiMAX Fourm / 3GPP2 Proxy Mobile IPv4”, Cisco Systems, California, **(2010)**.
[10] R. Wakikawa and S. Gundavelli, “IPv4 Support for Proxy Mobile IPv6”, Toyota ITC, Minato-ku, **(2010)**.
[11] C. Bormann, M. Ersue and A. Keranen, “Terminology for Constrained Node Network”, Universitaet Bremen TZI, Bremen, **(2013)**.
[12] J. Mitsugi, S. Yonemura, H. Hada and T. Inaba, “Bridging UPnP and Zigbee with CoAP”, Proceedings of the workshop on Internet of Things and Service Platforms, Tokyo, Japan, **(2011)** December 6-9.
[13] C. Bormann, A. Castellani and Z. Shelby, “CoAP: An Application Protocol for Bilions of Tiny Internet Nodes”, IEEE Computer Society, vol. 16, no. 4, **(2012)**, pp. 62-67.
[14] Z. Shelby, K. Hartke and C. Bormann, “The Constrained Application Protocol (CoAP)”, ARM, California, **(2014)**.

[15] S. M. Chun, S. Y. Ge and J. T. Park, "Mobility Management Method for Constrained Sensor Nodes in WoT Environment", Journal of the institute of Electronics and Information Engineers, vol. 51, no. 9, **(2014)**, pp. 11-20.
[16] J. A. Jun, N. S. Kim, J. G. Park, H. Y. Kang and C. S. Pyo, "IoT Device product and Technology Trend", The Journal of the Korean Institute of Communication Sciences, vol. 31, no. 4, **(2014)**, pp. 44-52.
[17] M. Darianian and M. P. Michael, "Smart Home Mobile RFID-based Internet of Things Systems and Services", Advanced Computer Theory and Engineering International Conference on, Phuket, Thailand, **(2008)** December 20-22.
[18] H. J. La, C. W. Park and S. D. Kim, "A Framework for Effectively Managing Dynamism of IoT Devices", Journal of KISS: Software and Applications, vol. 41, no. 8, **(2014)**, pp. 545-556.
[19] A. C. Snoern and H. Balakrishnan, "TCP Connection Migration", MIT LCS, Massachusetts, **(2001)**.
[20] M. Riegel and M. Tuexen, "Mobile SCTP", Siemens AG, Berlin, **(2005)**.

Authors

Joo Ho Choi, he received a B.S degree in 2016 from Gachon University, South Korea. He is currently studying in the department of IT Convergence Engineering, Gachon University Graduate School, South Korea. His research interests include IoT for healthcare, wireless sensor network and Embedded System.

Ji-Youn Kim, she received a B.S. degree in 1998 from Kyusyu woman's College, Fukuoka, Japan and a M.S. degree from Fukuoka Education University and a Ph.D. degree from Sungkyunkwan University, in 2004 and 2008. He is currently a lecturer in the department of exercise Rehabilitation & Welfare, Gachon University, South Korea. Her research interests are Exercise physiology, Sports nutrition, and health related exercise rehabilitation.

Eun-Surk Yi, he received a B.S., M.S., Ph.D., degree in 1993,1998, 2003 from Korea National Sport, Seoul, Korea. He had worked for professor in the department of Gerokinesiology and Sports Industry, Daegu Haany University for 7 years Gyeongsangbuk-Do, Republic Korea. He is currently a professor in the department of Exercise Rehabilitation & Welfare, Gachon University, Republic Korea. His research interests are sociology of sports and leisure, activity program protocol for elderly, leisure patterns, convergence Exercise Rehabilitation etc.

Byung Mun Lee, he received a B.S. degree in 1988 from Dongguk University, Seoul, Korea and a M.S. degree from Sogang University and a Ph.D. degree from University of Incheon Korea, in 1990 and 2007. He had worked for LG Electronics for 7 years. He is currently a professor in the department of Computer Engineering, Gachon University, South Korea. He had been at California State University Sacramento, USA from 2013 to 2014 as a visiting scholar. His research interests are Edge Computing, Wearable IoT Device, IoT for healthcare, wireless sensor networks, operating system, etc.