International Journal of Geo-Information

MDPI

Article

Proximity-Based Asynchronous Messaging Platform for Location-Based Internet of Things Service

Hyeong gon Jo [1], Tae Yong Son [2], Seol Young Jeong [2] and Soon Ju Kang [1,*]

[1] School of Electronics Engineering, College of IT Engineering, Kyungpook National University, 80 Daehakro, Bukgu, Daegu 702-701, Korea; tsana@ee.knu.ac.kr

[2] Center of Self-Organizing Software-Platform, Kyungpook National University, 80 Daehakro, Bukgu, Daegu 702-701, Korea; pipikako@gmail.com (T.Y.S.); snowflower@ee.knu.ac.kr (S.Y.J.)

* Correspondence: sjkang@ee.knu.ac.kr; Tel.: +82-53-950-6604

Academic Editors: Chi-Hua Chen, Kuen-Rong Lo and Wolfgang Kainz
Received: 29 April 2016; Accepted: 8 July 2016; Published: 14 July 2016

Abstract: The Internet of Things (IoT) opens up tremendous opportunities to provide location-based applications. However, despite the services around a user being physically adjacent, common IoT platforms use a centralized structure, like a cloud-computing architecture, which transfers large amounts of data to a central server. This raises problems, such as traffic concentration, long service latency, and high communication cost. In this paper, we propose a physical distance-based asynchronous messaging platform that specializes in processing personalized data and location-based messages. The proposed system disperses traffic using a location-based message-delivery protocol, and has high stability.

Keywords: location-based service; Internet of Things; distributed system architecture

1. Introduction

With the recent development of network and embedded system technologies, interest has grown in the Internet of Things (IoT). For example, a smart home service [1] can check and control appliances in the home by connecting the devices to a network. A smart healthcare service [2] manages the personal medical state by connecting with wearable devices, fitness equipment, and medical equipment. Mobile-asset monitoring [3] provides real-time monitoring and tracking of mobile assets in a factory.

However, despite the devices being physically adjacent, common IoT platforms transfer large amounts of data, generated by multiple devices, to a central server via a global network. This causes problems, such as network congestion due to traffic concentration, service-delay problems caused by multi-hop communication, and a high communication cost because the device is always connected with a centralized server via a global network. In addition, many mobile objects require a complex network protocol, like 6LoWPAN (IPv6 over Low-power Wireless Personal Area Networks) or TCP/IP (Transmission Control Protocol/Internet Protocol), to connect to a network. This is difficult to implement in mobile phones and resource-constrained embedded systems like wearable devices. From the end user's viewpoint, since the personal device information is stored in a central server, privacy violations and security issues [4,5] are possible.

In this paper, to solve the above problems, we propose a physical distance-based asynchronous messaging platform between neighboring nodes. IoT services closely related to people have personalized and localized features, like smart home services and smart healthcare services. Therefore, we define the location-based IoT (LIoT) service, combining IoT services and location information. This specialized platform processes personalized data and location-based messages. The proposed system is composed of (1) a wireless communication proxy that is responsible for direct communication between various mobile nodes; (2) a self-organizing localized IoT messaging hub (SLIM Hub) that

makes up the autonomous overlay network; and (3) an ePost-it messaging platform that provides a common format for the services. The proposed system can disperse traffic by delivering messages based on location, increase the system stability via asynchronous message transmission, and solve the privacy issue by using personal or temporary storage rather than unnecessary centralized storage. Especially, our platform can fundamentally prevent personal information leaks by storing private data to personal or temporary storage and providing a service to a user after identifying the user's location.

The remainder of this paper is structured as follows. In Section 2, we introduce related work. Section 3 provides an overview of the LIoT service and the proposed system concept. Section 4 describes the detailed design of the wireless communication proxy responsible for direct communication with a mobile node, and Section 5 presents a detailed design of the ePost-it middleware to provide the LIoT service. Section 6 introduces the implementation of the proposed system and evaluates its performance. Finally, conclusions are drawn in Section 7.

2. Related Work

Real-world location-based applications aim to detect the location of targets in various service domains, such as medical personnel or equipment in a hospital [6,7], smart home management system [8] or stored inventories in a warehouse [9]. There has been research on IoT devices such as Bluetooth low Energy (BLE) sensor module [10] and power consumption issues on IoT devices [11]. EZ [12] is a gateway that supports an efficient asynchronous protocol in IoT. EZ enables the creation of gateways with either C or Java platforms without requiring developers to have any substantial understanding of either relevant protocols or low-level network programming. These research works well-define the characteristics of location-based services (LBS) and IoT. However, they have the aforementioned problems of centralized architecture.

Apple, Inc. has developed the iBeacon [13] service using BLE. A beacon node installed in a fixed location sends a beacon message with its location ID. Then, a mobile node scans this beacon message and determines the location by comparing the signal strength of each beacon. The iBeacon is a localization system without a central server; however, it must go through a service server for real service. NextMe [14] is a phone-based localization system for providing location-based services in IoT. It uses mobile call patterns, which are strongly correlated with co-location patterns. However, it has disadvantages, such as low precision when providing indoor service.

Many researchers have attempted to address the traffic concentration in IoT platforms. Wang and Ranjan discussed the capabilities and limitations of big-data technologies in the fifth installment of "Blue Skies" [15]. The concept of fog computing [16] was introduced to disperse the mass of traffic, based on location. Fog computing can distribute large-scale data and improve the service response time by storing large amounts of data on terminal devices, such as a network router, rather than a central server. However, since fog computing only disperses the data using the network location, regardless of the actual physical location, it is not suitable for location-based services. In addition, privacy issues are still present because it saves all the information in a distributed database on the router.

3. Proposed System Concept

This section provides an overview of LIoT service and the proposed platform concept. First, we introduce the scenario for LIoT under a real environment, and then itemize the characteristics of the proposed LIoT service. The design consideration and the proposed platform concept are followed in the next subsection. Finally, we introduce some protocols suggested in our previous research to realize the proposed platform.

3.1. Overview of the Location-Based Internet of Things Service

Figure 1 is a representation of a location-based IoT service scenario in a hospital environment. A hospital has numerous mobile assets, such as wheelchairs, chemicals, medicine, and hazardous waste materials. The administrator will request typical IoT services, such as (a) monitoring the location

and status of equipment; and (b) checking the path of hazardous materials to confirm their safety (shown as green line). In addition, a nurse can request location-based services, such as (c) finding or reserving an available wheelchair near her current position; (d) broadcasting only to the position of the patient who called rather than broadcasting to the entire hospital; or (e) checking the proper medicine by proximity-based direct Device to Device (D2D) communication with a patient's wearable device. Lastly, (f) a user uses the private data such as daily activity or vital signal stored in personal storage.

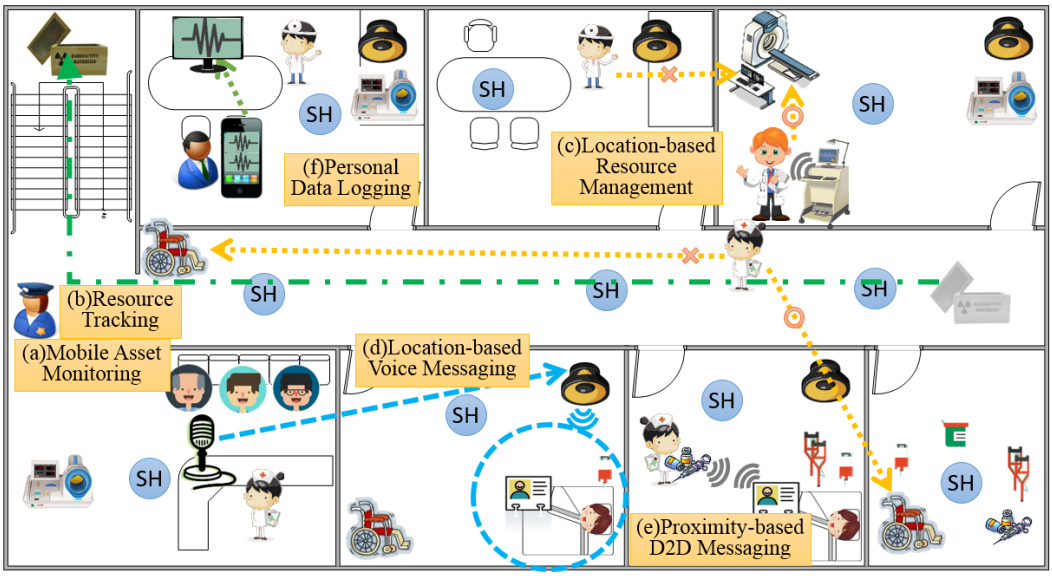

Figure 1. Scenario for Location-based IoT service.

The proposed LIoT service is composed of two types of hardware devices: an end device and a SLIM Hub. The end device is any service device that can identify the user or provide a service. These devices are classified as mobile nodes since they can be moved, even though they may have been installed in a fixed position, e.g., a TV or printer. SLIM Hubs are installed in a unit space such as a room or corridor, representing the installed area and providing a network-access point to the end devices. Thus, an end device can determine its location according to the proximity of a SLIM Hub. In addition, a SLIM Hub is a management device installed at a fixed location to collect assorted information from the end devices in the service area, and provide location-based services based on the proposed messaging middleware.

In this paper, we proposed the LIoT service platform, which is optimized for collecting and delivering localized data by combining the location information and an IoT messaging platform. Devices in our system provide services through proximity-based communication between neighbors, and store personal information to the user's personal device storage. Therefore, the proposed system solves the above-mentioned issues by storing collected personal data to the individual's distributed storage, not the central server, and performing location-based neighbor searching.

3.2. Characteristics of Location-Based IoT Data

LIoT service data has different features than other existing services. Briefly, it has characteristics of both location-based services (LBS) and IoT services. The following are the characteristics of LIoT:

- **Locality**

  Most IoT services have a centralized structure, e.g., cloud-computing architecture. However, LIoT services have location-based characteristics, such as data collection from the required area and service matching based on the current location. Therefore, it is important to combine the location information and service-messaging path.

- **Real-time service**

  In a user-centric service, response time is an important part of the service quality [16]. Particularly, if the service occurs near a user, the response time should be limited to a few seconds to avoid inconvenience for the user. However, in existing centralized structures, the service delay time increases because of the inevitable communication delay when connecting with the server. To solve this problem when using a distributed infrastructure, it is necessary to make the communication distance between the service elements as short as possible.

- **Data sharing and privacy**

  The LIoT service has a variety of service-connection methods between devices: 1:1 connections for text messaging, 1:N for notification messaging, and N:1 for monitoring an area. Hence, the service platform supports a flexible connection method for sharing data. However, the common centralized storage and messaging architecture have a possibility of personal data leak. To prevent such an occurrence, it needs a distributed architecture equipped with a personal storage and a location-based messaging system.

- **Asynchronicity**

  LIoT has steps such as data acquisition, processing, and delivery. In this process, each device connection is asynchronous. Since mobile devices also participate in the service, the LIoT service requires asynchronous transfer capability to improve the communication reliability.

- **Heterogeneity**

  Each service device has many variations, e.g., the type of sensors used for data acquisition, the communication method to be delivered to the service platform, and the format of the data to be distributed. The final stage of the IoT services needs to provide services through generalizing the data obtained from the disparate devices.

- **Small message size, huge data volume [17]**

  The data generated by devices, e.g., sensor values and status, are small. However, the total amount of data becomes huge, since numerous devices participate in the service. To handle these data efficiently, LIoT must have a communication structure suitable for a large number of small messages.

3.3. Design Considerations

To provide a service that matches the characteristics of the LIoT data in Section 3.2, we organized the following requirements and proposed a location-based asynchronous messaging platform.

- **Message-oriented platform**

  To asynchronously transfer a large number of small data messages, message-oriented middleware (MOM) is a suitable messaging platform, since it is designed to rapidly convey large numbers of messages [17]. MOM's publish/subscribe structure has a specialized feature for data sharing between devices. In addition, MOM creates a weak coupling between the mobile nodes and service applications by communicating asynchronously, thereby removing the communication dependency. Thus, it is possible to perform highly reliable communication in an unstable environment, even if traffic congestion occurs.

- **Localization system**

  A localization system is needed to collect data with regional properties and provide location-based services. A localization system recognizes the position of the mobile node and records the location

to provide location-based services. In addition, it has a location-based message transmission architecture, rather than a centralized architecture.

- **Protocol gateway**

  The gateway is essential for communicating with mobile nodes that have various protocol types. The protocol gateway provides connection transparency for a mobile node using any protocol by abstracting various communication protocols.

- **Worst-case performance evaluation**

  As described above, an important question in LIoT services is "How long will it take to respond?" The service response time is associated with "How much traffic can a service platform handle simultaneously?" Therefore, in an environment with a large number of messages, the environment should evaluate whether it can respond to the user without the service running out of control.

3.4. Asynchronous Messaging Platform Concept

Figure 2 shows the proposed system's asynchronous-messaging structure. The proposed platform is composed of mobile devices, protocol gateway, and Self-Organized Software platform (SoSp) middleware. Mobile devices are connected to SoSp middleware using various protocols; a mobile device using TCP/IP directly connects to the messaging middleware, and the other protocol devices can communicate to the messaging middleware through the protocol gateway. By just adding a protocol driver into the protocol gateway, the proposed platform can support any communication protocol for mobile devices. A SLIM Hub includes protocol gateway, local storage, and messaging middleware.

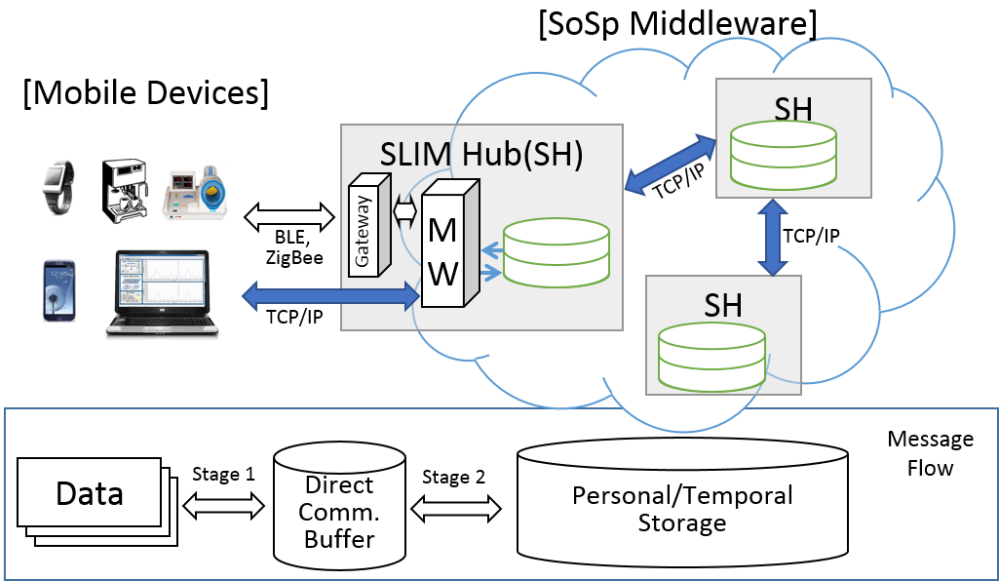

Figure 2. Concept diagram of asynchronous messaging platform.

The messaging flow is composed of the following two communication stages.

- Stage 1: Mobile nodes↔ Protocol gateway

  First, a message generated by a mobile node is transferred to the protocol gateway through direct wireless communication, and stored in the direct communication buffer inside the gateway. In this process, various communication protocols are abstracted by the protocol gateway, so that the message delivery is entrusted to it. For example, for a Bluetooth Low Energy communication, which is usually mounted on a mobile device and has a short maximum transmission unit

(MTU) size of about 20 bytes, we can provide enhanced services without the MTU limitation by abstracting the communication. We named this the 'wireless communication proxy'.

- Stage 2: Message oriented middleware↔ SoSp messaging middleware

  After that, the message is transferred to the SLIM Hub (SH), and stored in local storage inside the SH. The stored message will be delivered to personal storage via a preset path, or stored in temporary storage until the SH finds the destination using a location-based neighbor-searching protocol.

3.5. Previous Research for Proposed Platform

Protocols have been suggested in previous research to support the distributed location-based services. By adding the concepts of Section 3.4, based on the localization and discovery protocols, we designed a proximity-based asynchronous-messaging platform.

Location-ID exchange and asynchronous message delivery (LIDx & AMD) [18,19] can provide real-time localization for numerous mobile nodes in a complex and dynamic indoor environment, such as a hospital, warehouse, or museum. Each stationary node installed in a unit space, like a room, periodically sends a beacon message with its location ID. A mobile node can determine its location by selecting the nearest stationary node, which is done by comparing the signal strengths of the beacon messages. This localization protocol uses a simple bidirectional communication between the stationary node and the mobile node, so it guarantees efficient movement of mobile devices.

The location-based service discovery protocol (LSDP) [20] is a resource-discovery protocol. In this protocol, stationary nodes make an overlay network based on the physical-neighbor relationship. A stationary node uses only information about resources within its management range and those of its neighboring stationary nodes. Then, the protocol searches for target resources, using an algorithm similar to graph traversal. This discovery protocol uses only information about the local area and the neighboring stationary nodes, so we can freely look up a resource using distributed-service discovery without a centralized server.

4. Wireless Communication Proxy

A wireless communication proxy is responsible for direct communication with the mobile nodes, using localization and asynchronous message delivery. The proxy is composed of the frontend module (FE) for abstracting the wireless communication, and RFProxy for generalizing the messages from the mobile nodes.

4.1. Structure of Wireless Communication Proxy

Figure 3 shows the elements of the frontend module and the RFProxy agent. The FE supports a variety of wireless-communication protocols using a communication manager. The address book converts an address from a communication address (like a MAC address) to an identification address (like a unique user ID), or reversely. It helps that we can communicate with a mobile device using a unique ID without concerns about the real communication protocol. The network manager and connection manager manage the specific protocol depending on the characteristics of the communication protocol. Particularly, the connection manager manages connect-based protocols, such as Bluetooth, allowing them to communicate with a number of mobile nodes by repeatedly connecting and disconnecting as necessary. The message controller divides a message into a size suitable for the communication protocol and combines the divided message into a single service message once it arrives.

The RFProxy agent abstracts the mobile node information and messages. It communicates with the FE over serial communication, combines the user ID and data from the mobile node, and forwards the message to the appropriate service agent. The message monitor checks the delivery result of the messages transferred asynchronously, and notifies the result to the sender. The message converter abstracts a message by converting a compressed binary to the common format.

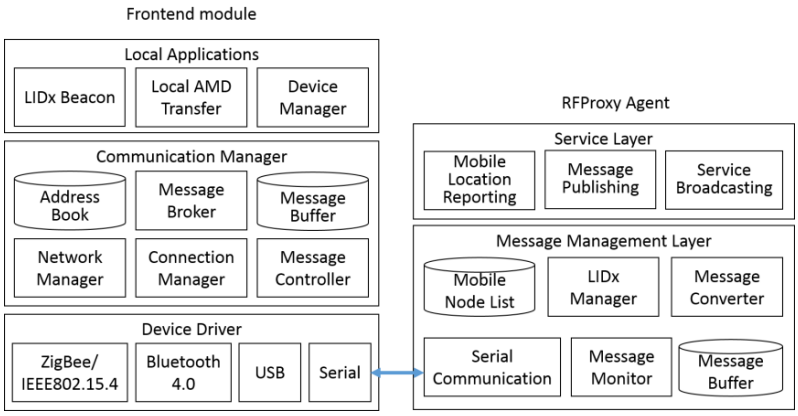

Figure 3. Component diagram of frontend module and RFProxy agent.

4.2. Process of Wireless Communication Proxy

4.2.1. Communication between the Mobile Node and SH

Asynchronous message delivery via direct communication between the mobile node and the frontend module can be classified into two types, depending on the transmission direction, i.e., to or from the mobile node. Moreover, a connection-less protocol, such as IEEE 802.15.4, can easily communicate with a mobile node; however, a connection-based protocol, such as Bluetooth, requires a connection-management mechanism that creates and deletes the communication connection with the mobile node. Following is the communication process for a mobile node and a frontend module using the Bluetooth protocol.

Figure 4 shows how the frontend module sends a message to the mobile node. If the FE receives a message request (1, 2), it switches the Bluetooth mode to a central mode, which can create a connection. When it is connected to the mobile node (3), the FE transmits the message and saves the results (4, 5). Next, the FE forwards the result to RFProxy (7, 8). If the transmission fails because the mobile node has moved, the message is re-transmitted by a location-based protocol, so the message is transferred regardless of the mobile node's movement.

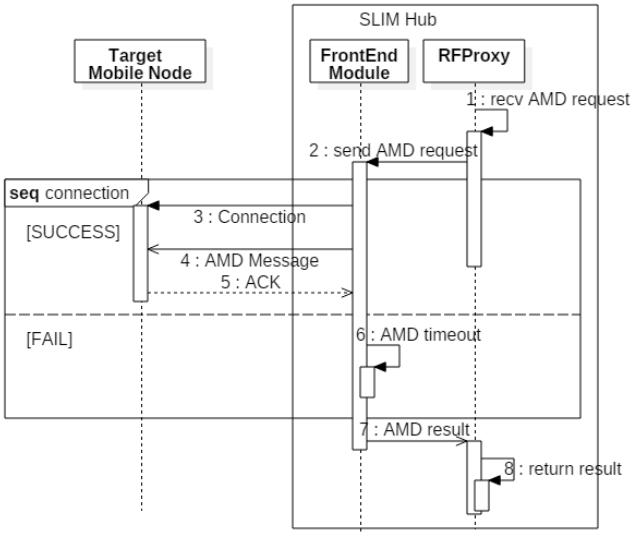

Figure 4. Messaging sequence from SLIM Hub (SH) to target mobile node.

Figure 5 shows the messaging procedure from the mobile node to the SH. When the outgoing message is prepared (1), the transmission starts by advertising that there is a transmission message (3).

If the mobile node knows its location, it advertises with the MAC address of the FE responsible for its position; thus, the SH knows that there is a request from the mobile node (4). Then, the FE initiates a connection (6), communicates with the mobile node (7), and forwards the message to the RFProxy (8). However, the mobile node may not know its current location; therefore, it must request help from the SH to determine which of several FEs it should connect to. When a FE receives an advertising message without a target FE address (9), the RFProxy uses the mobile-node list to determine who is in charge of the mobile node (11). After that, the SH that manages the mobile node communicates with it (12–16).

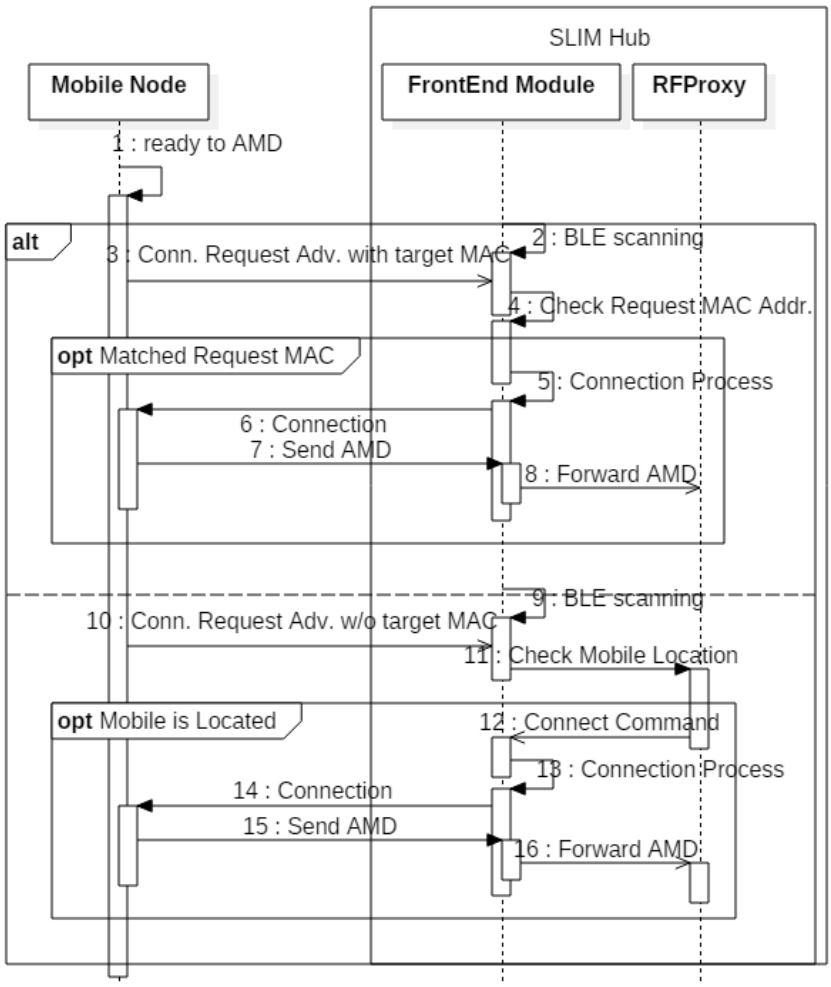

Figure 5. Messaging sequence from source mobile node to SH.

4.2.2. Communication between Mobile Node and Service Agent

Figure 6 is a sequence diagram showing how the service agent and mobile node reconnect when the mobile node changes location. First, the mobile node binds with various service agents in its previous location. Here, "bind" refers to the path for asynchronous message delivery, not the connection binding for synchronous messaging. When the mobile node moves to the new location, the location leader informs the new RFProxy of the movement of the mobile node, with information on the previous RFProxy (2). The new RFProxy receives the service information associated with the mobile node from the previous RFProxy (3, 4), and requests a re-binding to the service agents (6, 8). Now, the service agents and the mobile node can communicate through the RFProxy (10). By running this process as a mobile node changes its location, the proposed system allows the service agents and mobile node to communicate without a central server to relay the connection.

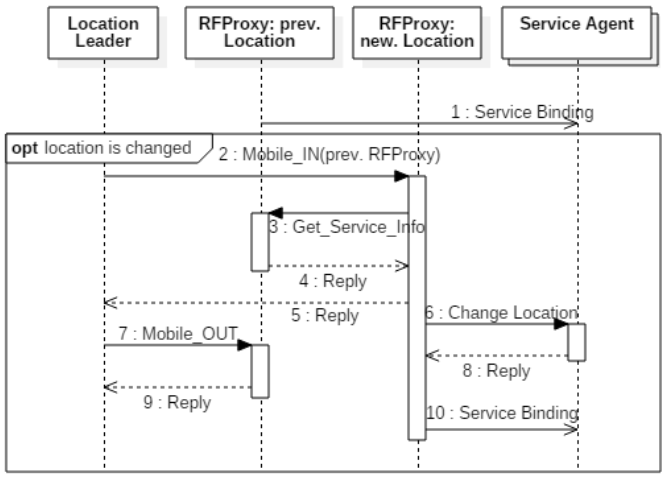

Figure 6. Sequence diagram of service rebinding when a mobile node is moved.

5. ePost-it Middleware

5.1. ePost-it Structure

5.1.1. ePost-it Concept

To implement asynchronous messaging among various devices and services, we propose the ePost-it concept to provide location-based asynchronous messaging. This concept comes from Post-it in the real world, which can be easily separated or attached to the target.

Following are the features of the ePost-it concept, as shown in Figure 7:

- It has a complete message-block type containing a string, encoded binary data, etc.
- It will last as long as the survival time configured in the message.
- It can be sent to all of the users in the region by targeting the area as the destination.
- It can easily be temporarily stored and moved anywhere along the target.

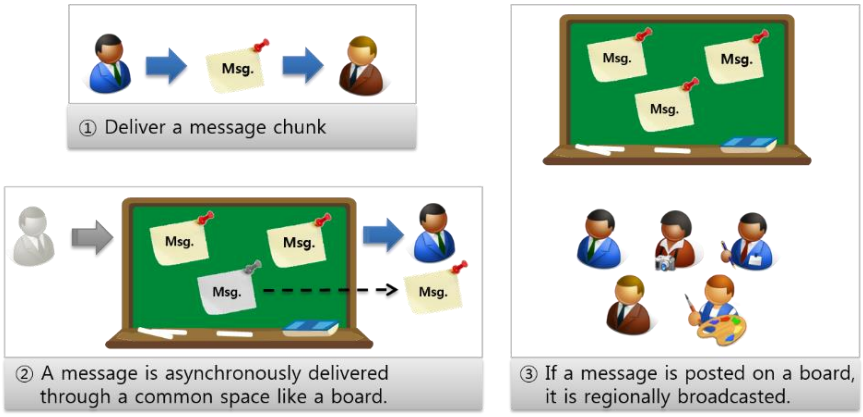

Figure 7. Features of the ePost-it concept.

We defined the common ePost-it data structure in XML format so it can be easily recognized by a variety of devices and services. Figure 8 shows the XML structure of the ePost-it. It has a variety of attributes to process the ePost-it: a callback attribute to obtain the messaging result, and source and destination fields to represent the sender and receiver, respectively. It can select various devices via RFProxy, or select an SH for targeting a region. The contents field can be defined as required for the service. For example, it is possible to include a simple text-based message or an encoded audio file.

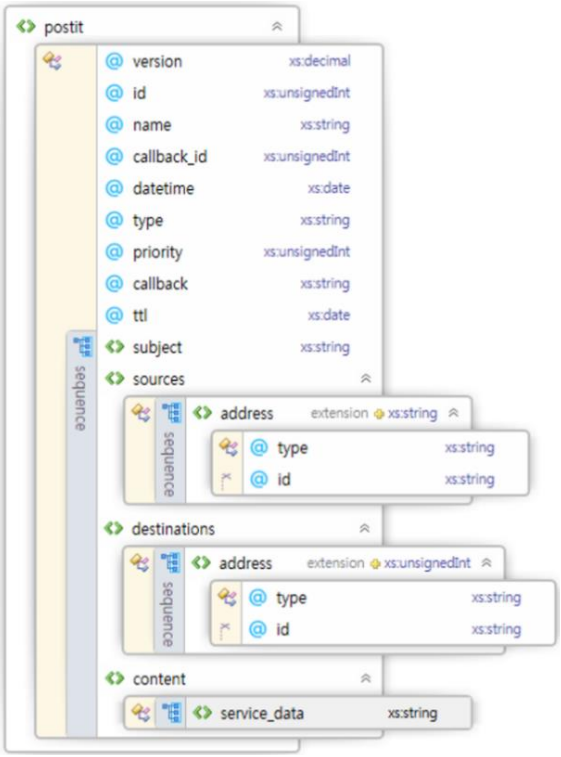

Figure 8. XML schema for ePost-it.

5.1.2. Structure of ePost-it Middleware

In this paper, we propose the ePost-it middleware by adding an ePost-it agent, based on previously developed SoSp middleware [20]. Figure 9 shows the structure of the ePost-it middleware. All SoSp middleware service messages are delivered in ePost-it format through RFProxy or WiFi, and processed by the ePost-it agent. The ePost-it agent analyzes the destination described in the XML document, searches the destination SH to find where the message should be delivered—using the location-based search algorithm [20]—and transfers the message to the messaging agent of the target SH. The messaging agent is responsible for processing and transmitting. First, it stores the message into the local buffer. Then, the message is conveyed directly to the service agents or pushed through a push agent. The push agent transfers the messages to the target according to the device type: it uses RFProxy for small embedded devices and the local push protocol for WiFi devices. If the ePost-it destination is specified as the region, it sends the message to the entire mobile node, to visit the area during the specified time to live (TTL).

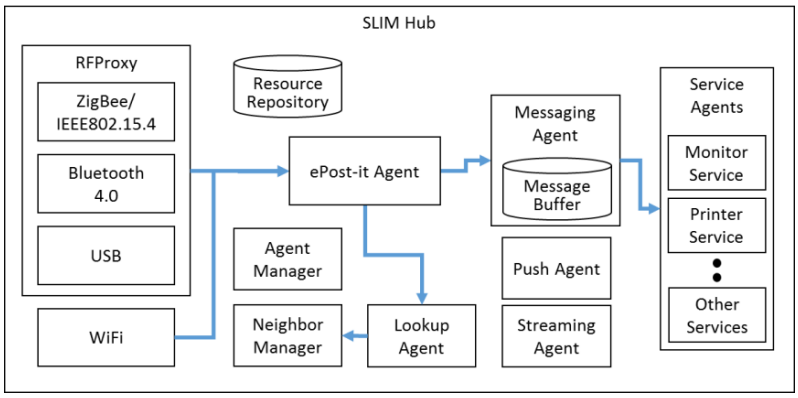

Figure 9. Software structure of ePost-it middleware.

5.2. ePost-it Middleware Process

5.2.1. Message Delivery Process

Figure 10 describes the process of delivering ePost-its. A message generated in the mobile node is transmitted to the ePost-it agent of the source SH (1, 2). Then, the ePost-it agent obtains the target SH information using a lookup engine (3, 4), and transmits the message to the messaging agent in the target-SH managing destination (5). The message stored in the message buffer (6) is transferred through the push agent (7–10). Finally, the ePost-it message is removed, as the message was successfully delivered or its TTL expired.

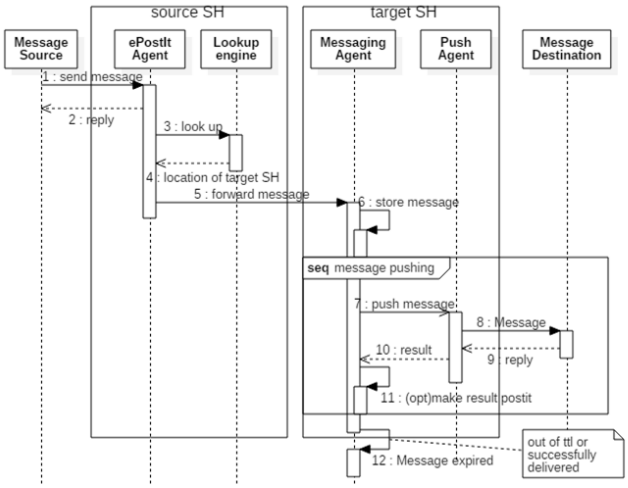

Figure 10. Delivery sequence of an ePost-it message.

5.2.2. Push flow of ePost-it

ePost-it middleware use four types of push methods to deliver a message to a mobile node using various communication protocols. Figure 11 illustrates the process of transmitting an ePost-it to mobile nodes of various kinds. The messaging agent forwards the message to the destinations, depending on the destination type. A message targeted to a service agent is transmitted directly. A message targeted to a mobile node is passed to a push agent, and the push agent transfers the message according to the mobile node's communication type. A global pushing system, such as Google Cloud Messaging (GCM) or Apple Push Notification Service (APNS), transfers the message to a mobile node outside of the SoSp infrastructure. A local pushing protocol transfers it to a mobile node inside the SoSp infrastructure. RFProxy is used for small embedded devices.

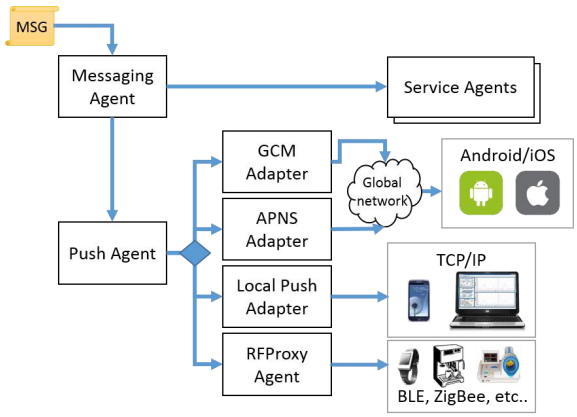

Figure 11. Messaging flow from messaging agent to target agents or devices.

6. Implementation and Evaluation

6.1. Test Environment

Figure 12 shows the hardware used to evaluate the proposed platform. The SLIM Hub acts as a gateway and provides the location-based services (LBS). It includes speakers, LCD, and ethernet access, and is connected to adjacent SHs using TCP/IP. The frontend module is a protocol gateway connected to an SH to abstract the communication with mobile nodes, such as BLE and ZigBee. The mobile tag is a type of mobile node. The tags are used to evaluate the abstraction and management of a connection-based protocol. Finally, to verify the ePost-it middleware performance, a mobile node connected by TCP/IP is simulated on a PC.

Figure 12. Hardware module used to evaluate the proposed platform.

6.2. Performance of the Frontend Module

To evaluate the FE, we tested the traffic performance. Figure 13 shows the average total transmission time while increasing the number of mobile nodes, each of which sends only one message to the FE. First, a mobile node sends a variable size message to the FE. For a given number of mobile nodes, we measured a total transmission time until all queued requests are processed and calculated an average of 30 runs. As shown in the figure, despite a number of nodes requesting a connection to send a message, the FE can process all the requests sequentially using connection management.

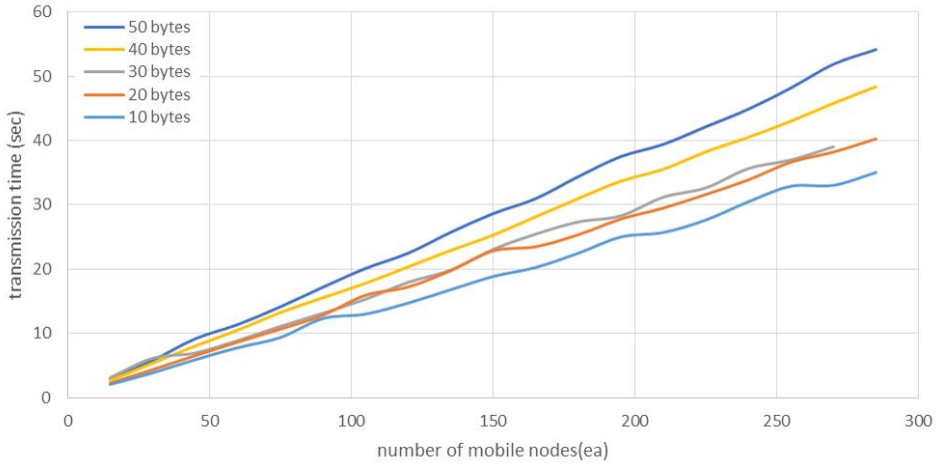

Figure 13. Total transmission time according to the number of mobile nodes.

To estimate the performance with more devices, we measure the time used for messaging during a connection. Figure 14 shows the transmission time for sending a message to the FE according to the

message size. The transmission time is composed of the connection time and the data-transmission time, as described in Equation (1). By regression analysis of Figure 14 (95% confidence interval), we determined the constants as $T_{connection} = 110$ ms and $T_{byte} = 1.56$ ms.

$$T_{transmission} = T_{connection} + T_{data\ transmssion} = T_{connection} + T_{byte} \times (\text{number of bytes}) \quad (1)$$

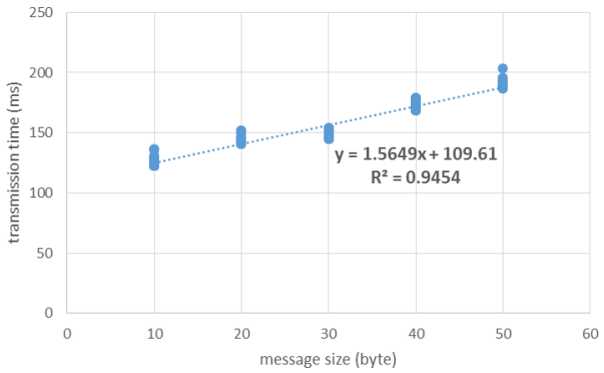

Figure 14. Transmission time per connection according to the message size.

6.3. Performance of the ePost-it Middleware

We created a simple service scenario to verify the performance of the ePost-it. In a conference with a large number of users participating, a presenter wants to distribute the presentation materials to the people in attendance. SHs are installed in each presentation room, and an SH sends the file to the mobile devices by detecting the presence of a user. Figure 15 shows the response time from user's entrance to receiving the message, according to the number of mobile nodes. As shown in the figure, the response time increases slowly as the number of mobile nodes increases. We determined that an SH responds within a few seconds, in spite of numerous mobile nodes.

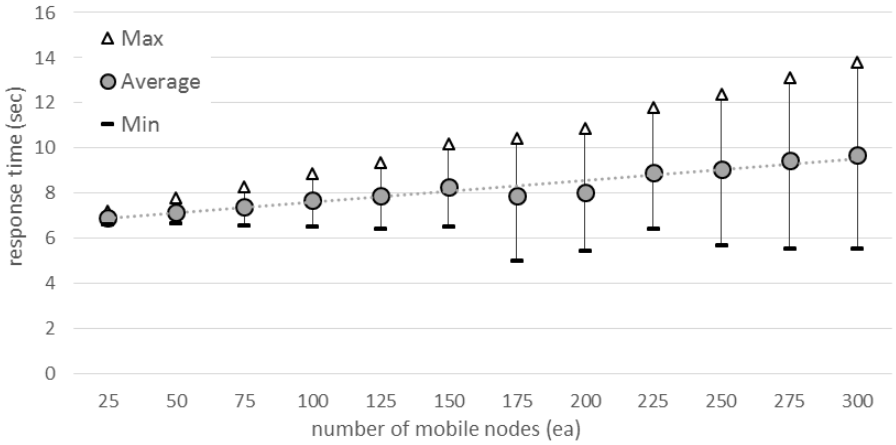

Figure 15. Response time of ePost-it according to the number of mobile nodes.

7. Conclusions

In this paper, we introduced the LIoT service and investigated a proximity-based asynchronous messaging platform specialized in processing personalized data and location-based message. To address traffic congestion and privacy issues, the problems of centralized architectures, the proposed system disperses traffic using a location-based message-delivery protocol, and stores collected personal data to the individual's storage, not the central server. To implement the asynchronous mechanism

in IoT, we designed the following components. The direct communication buffer in a FE abstracted the communication protocol and provided a loosely coupled connection between a mobile node and IoT services. The temporary storage held personal data until the data was transferred by the asynchronous-messaging mechanism. The ePost-it middleware supported location-based messaging services. We experimentally evaluated the performance. The results showed that the FE could sequentially process numerous requests, spending a few hundred milliseconds in a connection, and the SH could rapidly provide location-based messaging.

In future work, we will focus on extending the messaging platform to support continuous data streaming, regardless of the device movement; e.g., healthcare applications for a vital-signal streaming system, which requires the transmission and sharing of large waveform data.

Acknowledgments: This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIP). (No. 10041145, Self-Organized Software platform (SoSp) for Welfare Devices).

Author Contributions: Hyeong gon Jo designed and performed the experiments, analyzed the data, and wrote the paper; Tae Yong Son designed and performed the experiments; Seol Young Jeong supported the design and analysis; Soon Ju Kang supervised the analysis and edited the manuscripts.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Wang, D.; Lo, D.; Bhimani, J.; Sugiura, K. AnyControl—IoT based home appliances monitoring and controlling. In Proceedings of the 2015 IEEE 39th Annual Computer Software and Applications Conference, Taichung, Taiwan, 1–5 July 2015; pp. 487–492.
2. Cecílio, J.; Furtado, P. Middleware solution for healthcare IoT applications. In *Wireless Internet*; Springer International Publishing: Cham, Switzerland, 2015; pp. 53–59.
3. Jeong, S.Y.; Jo, H.G.; Kang, S.J. Fully distributed monitoring architecture supporting multiple trackees and trackers in indoor mobile asset management application. *Sensors* **2014**, *14*, 5702–5724. [CrossRef] [PubMed]
4. Abomhara, M.; Koien, G.M. Security and privacy in the Internet of Things: Current status and open issues. In Proceedings of the 2014 International Conference on Privacy and Security in Mobile Systems (PRISMS), Aalborg, Denmark, 11–14 May 2014; pp. 1–8.
5. Yoon, S.; Park, H.; Yoo, H.S. Security issues on smarthome in IoT environment. In *Computer Science and its Applications*; Springer: Heidelberg, Germany, 2015; pp. 691–696.
6. Kamel Boulos, M.N.; Berry, G. Real-time locating systems (RTLS) in healthcare: A condensed primer. *Int. J. Health Geogr.* **2012**, *11*. [CrossRef] [PubMed]
7. Mun, I.; Kantrowitz, A.; Carmel, P.; Mason, K.; Engels, D. Active RFID system augmented with 2D barcode for asset management in a hospital setting. In Proceedings of the 2007 IEEE International Conference on RFID, Grapevine, TX, USA, 26–28 March 2007; pp. 205–211.
8. Lee, Y.; Hsiao, W.; Huang, C.; Chou, S.T. An integrated cloud-based smart home management system with community hierarchy. *IEEE Trans. Consum. Electron.* **2016**, *62*, 1–9. [CrossRef]
9. Kim, J.Y.; Jeon, B.-W.; Hong, D.G.; Suh, S.-H. A proposition for smart warehouse management system (SWMS) through IoT. *J. Korea Soc. Syst. Eng.* **2015**, *11*, 85–93. [CrossRef]
10. Ryu, D.-H. Development of BLE sensor module based on open source for IoT applications. *J. Korea Inst. Electron. Commun. Sci.* **2015**, *10*, 419–424. [CrossRef]
11. Trappe, W.; Howard, R.; Moore, R.S. Low-Energy security: Limits and opportunities in the internet of things. *IEEE Secur. Priv.* **2015**, *13*, 14–21. [CrossRef]
12. Bromberg, Y.-D.; Morandat, F.; Réveillère, L.; Thomas, G. EZ: Towards efficient asynchronous protocol gateway construction. In *Distributed Applications and Interoperable Systems*; Springer: Heidelberg, Germany, 2013; pp. 169–174.
13. Apple Inc. iBeacon for Developers. Available online: https://developer.apple.com/ibeacon/ (accessed on 15 February 2016).
14. Zhang, D.; Zhao, S.; Yang, L.T.; Chen, M.; Wang, Y.; Liu, H. NextMe: Localization using cellular traces in internet of things. *IEEE Trans. Ind. Inform.* **2015**, *11*, 302–312. [CrossRef]

15. Wang, L.; Ranjan, R. Processing distributed internet of things data in clouds. *IEEE Cloud Comput.* **2015**, *2*, 76–80. [CrossRef]
16. Bonomi, F.; Milito, R.; Natarajan, P.; Zhu, J. Fog computing: A platform for internet of things and analytics. In *Big Data and Internet of Things: A Roadmap for Smart Environments*; Bessis, N., Dobre, C., Eds.; Springer International Publishing: Cham, Switzerland, 2014; pp. 169–186.
17. Dworak, A.; Ehm, F.; Charrue, P.; Sliwinski, W. The new CERN controls middleware. *J. Phys. Conf. Ser.* **2012**, *396*, 012017. [CrossRef]
18. Lee, D.K.; Kim, T.H.; Jeong, S.Y.; Kang, S.J. A three-tier middleware architecture supporting bidirectional location tracking of numerous mobile nodes under legacy WSN environment. *J. Syst. Archit.* **2011**, *57*, 735–748. [CrossRef]
19. Kim, T.H.; Jo, H.G.; Lee, J.S.; Kang, S.J. A Mobile asset tracking system architecture under mobile-stationary co-existing WSNs. *Sensors* **2012**, *12*, 17446–17462. [CrossRef] [PubMed]
20. Jeong, S.Y.; Jo, H.G.; Kang, S.J. Remote service discovery and binding architecture for soft real-time QoS in indoor location-based service. *J. Syst. Archit.* **2014**, *60*, 741–756. [CrossRef]