Implementing and Evaluating Priority Control Mechanism for Heterogeneous Remote Monitoring IoT System

Implementing and Evaluating Priority Control Mechanism for Heterogeneous Remote Monitoring IoT System

Takuma Tachibana
Graduate School of Integrated Science and Technology, Shizuoka University
tachibana@minelab.jp

Tetsuo Furuichi
Graduate School of Science and Technology, Shizuoka University
furuichi.tetsuo.15@shizuoka.ac.jp

Hiroshi Mineno
College of Informatics, Shizuoka University /JST PRESTO
mineno@inf.shizuoka.ac.jp

ABSTRACT
All people expect the Internet of Things (IoT) to grow. Accordingly, there are various IoT devices. There is an extremely wide variety of data from IoT devices such as images, and sounds. It will become possible to construct a heterogeneous remote monitoring IoT system using various valuations of IoT devices. However, a heterogeneous remote monitoring IoT system cannot send data in completely because almost all mobile network services for the IoT system do not guarantee bandwidth. In this paper, we propose a priority control mechanism for a heterogeneous remote monitoring IoT system. The proposed mechanism enables a best-effort IoT system that guarantees only the minimum necessary data telecommunications. The proposed mechanism controls the priority by using application requirements. We implemented the proposed mechanism and evaluated its effectiveness with a sending ratio and average delay time of each priority. The sending ratio results of each priority increased to 29.2% in daytime. The average delay time results of each priority decreased by up to 96.0% in nighttime.

CCS Concepts
• Computer systems organization→ Embedded and cyber-physical systems • Networks→ Application layer protocols; Cloud computing; Cyber-physical networks; Mobile networks

Keywords
Internet of Things; priority control; mobile networks; data collection

1. INTRODUCTION
All people expect the Internet of Things (IoT) to grow due to downsized sensors and the appearance of various mobile networks [1, 2, and 3]. There is an extremely wide variety of data from IoT devices such as texts, images, and sounds. Therefore, after 5 years' growth of the number of devices connected to the IoT, the IoT has grown by 7 times, and network traffic per device has grown by 5 times [4]. Similarly, mobile network service providers have handle services for the IoT system in recent years. However, mobile network services for the IoT cannot sustain the required traffic from a heterogeneous remote monitoring IoT system completely [3] because almost all mobile network services for the IoT system do not guarantee bandwidth.

In this paper, we propose a priority control mechanism for a heterogeneous remote monitoring IoT system. The proposed mechanism focuses on data that is sent from sensors, cameras, and so on. The data has different real-time characteristics and importance. The proposed mechanism controls the amount of data, sending rate, and sending timing of IoT devices, which is decided in accordance with the quality of service (QoS) (we called this QoS priority) required from the application. The proposed mechanism uses a broker server as a priority control server. The broker server manages the transition of data from the IoT device and application, which is decided in accordance with the QoS as a priority.

The remainder of this paper is organized as follows. Section 2 shows related work in term of methods and protocols for IoT telecommunication. The basic idea of the proposed system is described in Section 3. The prototype implementation and experimental results are shown in Section 4. Section 5 is the conclusion.

2. RELATED WORK
Recently, various methods and protocols for IoT telecommunication have been proposed [5-9]. One protocol for IoT, MQ Telemetry Transport (MQTT) [10], targets IoT traffic that has a small data amount, high frequency, and many devices. The protocol header size of MQTT is 2 bytes minimum. MQTT is a very effective protocol for IoT devices for sending lightweight data. In contrast, header size reduction does not affect IoT devices, for example, cameras, when they send a large amount of data. In addition, MQTT has a QoS control mechanism for unstable mobile networks. However, this mechanism guarantees arrival data by using retransmission, that is, it does not consider data characteristics and does not select sending data.

The Mobile Data Offloading Protocol (MDOP) [11] is a load-balancing protocol in mobile networks that smooths the localization of time and place. The MDOP focuses on different delay tolerance, which exists in various data. The MDOP delays high delay tolerant data to control the sending rate and debt at base stations. By contrast, the MDOP does not consider the characteristics of multiple data from heterogeneous IoT devices. We consider that adding a priority control method is more effective.

The Constrained Application Protocol (CoAP) [12] is an IoT protocol based on a simplified form of the Hypertext Transport Protocol (HTTP). CoAP cuts overheads using the UDP in the transport layer and transmits binary in the protocol header. On the other hand, using UDP makes good communication difficult.

MOBIQUITOUS '16 Adjunct Proceedings, November 28 - December 01, 2016, Hiroshima, Japan
ACM 978-1-4503-4759-4/16/11.
http://dx.doi.org/10.1145/3004010.3004040

Therefore, the user must implement function retransmission, QoS, and so on to ensure good communication.

The Advanced Message Queueing Protocol (AMQP) [13] is a message queuing protocol for exchanging business messages. The AMQP has high reliability and advanced message delivering functions including priority telecommunication. However, the AMQP's priority is assigned by the publisher. Therefore, the AMQP does not consider the QoS required by the application.

The QoS methods for IoT systems need to control these systems minutely and flexibly [3]. Heterogeneous IoT systems change the required QoS for the same data depending on the situation. Therefore, conventional QoS methods in internet architecture are not good enough.

As explained above, it is assumed that these methods are not enough to solve the problems in IoT telecommunication, such as low data amount, high frequency, and many devices [14]. Additionally, in IoT telecommunication we must consider the QoS required from applications. Therefore, we presume that it is important to implement a telecommunication mechanism for IoT that considers multiple data characteristics, mobile networks, and QoS required from applications.

3. PRIORITY CONTROL MECHANISM

3.1 Overview

We propose a priority control mechanism for a heterogeneous remote monitoring IoT system. The proposed mechanism focuses on the priority and characteristics of data to control the data sending order and data amount by having application configuration as a requirement. Therefore, the proposed mechanism enables control that satisfies the requirement from the application's QoS.

Figure 1 shows an overview of the proposed mechanism. The proposed mechanism consists of three elements: IoT devices, a broker server, and an application server. The IoT devices are endpoint devices, that are connected to mobile networks. The application server is an IoT system server that processes data from IoT devices. The broker server is a priority control server in the proposed mechanism. The application server and broker server are split by function; therefore, they can run on the same physical server. Because of this split, the elements' relationships are clear. Moreover, the proposed mechanism can extend architecture examples to multiple application servers.

Figure 2 shows the flow of the proposed mechanism. The proposed mechanism consists of three phases; register phase, priority telecommunication phase, and release phase. In the register phase, IoT devices register their own profiles. The application server registers the QoS requirement of its own application. The broker server creates a "priority decision table" and tells the IoT devices the application server address. After that, the IoT devices, application server, and broker server establish TCP connection with each other. In the priority telecommunication phase, the broker server gives priority to the data, which is generated by IoT devices (we call it content data), and manages their data. Additionally, the broker server confirms the sending rate while assigning part of the bandwidth or the delay sending timing to priority control. The release phase closes the connection and initializes the broker server.

3.2 Target IoT Applications

The proposed mechanism targets a remote monitoring IoT system, which collects information from a remote location using IoT devices and mobile networks. For example, agriculture

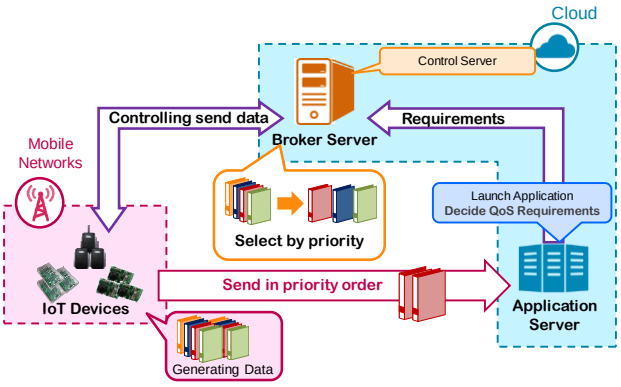

Figure 1. Overview of Priority Control Mechanism

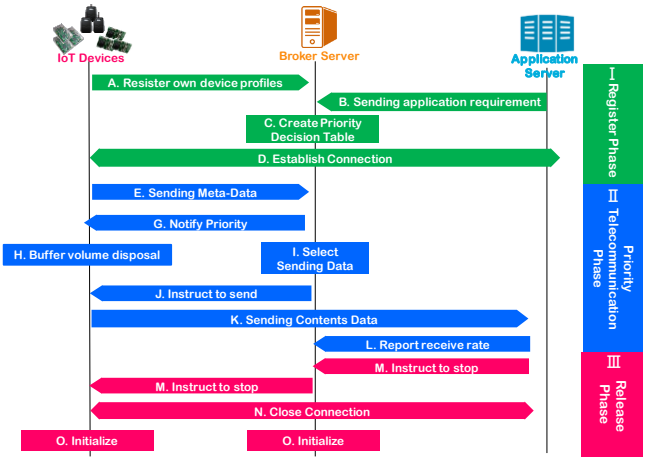

Figure 2. Flow of Priority Control Mechanism

support systems [15] that determine the conditions in places using images from a camera are desirable. However, using many cameras will degrade mobile network performance; therefore, systems cannot send environment data from sensors. In addition, generally mobile network rates vary dynamically. Due to the above, setting the granularity of sending data (for example, intervals and sizes) in advance is very difficult. System telecommunications that are controlled by priority control mechanism are desirable. A priority control mechanism enables the development of a best-effort IoT system that guarantees only telecommunication at the lowest limit.

3.3 Register Phase

First, **Table 1** shows the parts of the priority control mechanism. In this paper, we set Device ID to IoT devices beforehand. We set Device ID for the broker server to 0x00, for the application server to 0xff, and for the IoT devices to 0x01 to 0x0e. In addition, we call the profiles of content-data "meta data", and all of the data used the proposed mechanism "message". The broker server has a priority-decision table and manage table. The priority-decision table describes the rules to decide the priority of data from IoT devices. Manage table manages the connection state with IoT devices and the application server. The broker server performs priority control using these tables.

The register phase is the first phase in the proposed mechanism. **Figure 3** shows the details of the flow of the register phase. The register phase prepares information to launch the priority telecommunication phase. The required pieces of information

Table 1. Parts of Priority Control Mechanism

Word	Description
Device ID	Identifier of IoT devices.
Data Type	Class of content data (Decided by user.)
Content Data	Data that are sent by IoT devices.
Meta Data	Profile of content data. Meta data are associated with content data one-for-one.
Message	All data that were telecommunicated using priority control mechanism.
Priority-Decision Table	Table that describes rules of decision priority.
Manage Table	Table that manages connection state of IoT devices and application server.

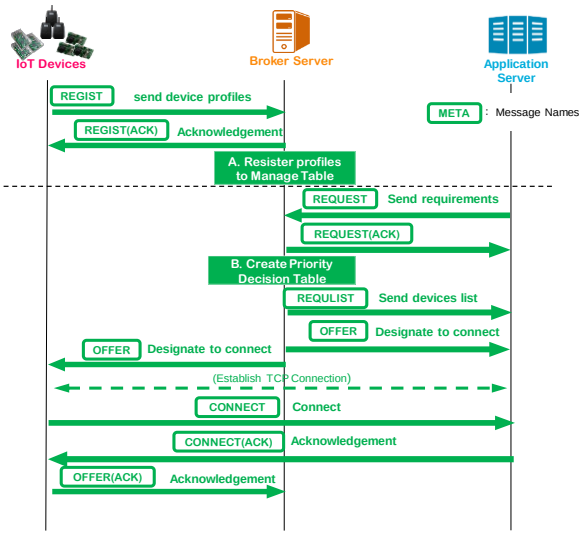

Figure 3. Details of flow of Register Phase

from the IoT devices are device ID and data type. The IoT devices register their information (A. in Figure 3.). The information required from the application server is the application's QoS requirement. The application server registers that information (B. in Figure 3.). After that, the IoT devices, application server, and broker server establish TCP connection with each other. Thus, the broker server collects information to launch the priority telecommunication phase.

3.4 Priority Telecommunication Phase

The priority telecommunication phase is the main phase in the proposed mechanism. This phase sets the priority for the content-data; after that, IoT device telecommunications with priority control can be performed. The priority is controlled by the broker server. The broker server decides the sending rate and sending timing to perform priority control using the application's required QoS. At this time, IoT devices save data to a buffer until they send the data. If the buffer capacity in an IoT device exceeds a threshold value, the IoT device deletes the content data that has the lowest priority from the buffer. Accordingly, this phase enables priority telecommunication to control the sending data amount by adjusting the telecommunication quality environment, which changes dynamically.

Figure 4 shows the details of the flow of the priority telecommunication phase. The IoT devices assign data ID and make meta data when IoT devices generate content data (A. in

Table 2. Structure of Meta Data

Field Name	Description	Type
Data ID	Identifier of content data	16 bits unsigned integer
Device ID	Identifier of devices	8 bits unsigned integer
Data Size	Volume of content data	16 bits floating point
Data Type	Types of content data	Strings

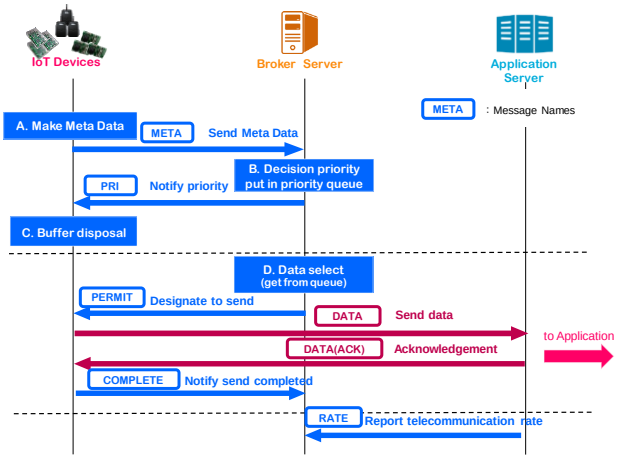

Figure 4. Details of flow of Priority Telecommunication Phase

Figure 4.). After that, IoT devices send meta data to the broker server as a meta message. At the same time, the IoT devices save their own content data to a buffer. When the broker server receives the meta message, the broker server checks the received message and priority-decision table to decide the priority (B. in Figure 4). The method of deciding the priority is described in the following section. After deciding the priority, the broker server sends the priority value to the IoT devices as a PRI message. At the same time, the broker server puts the meta data, data ID, and priority in its own priority queue. Meanwhile, when the IoT devices receive a PRI Message, the IoT devices save the priority to the buffer. If the buffer capacity in an IoT device exceeds a threshold value, the IoT device deletes the content data that has the lowest priority in the buffer while the buffer capacity exceeds a threshold value (C. in Figure 4.). Buffered content data is not sent until the PERMIT message from the broker server is received. When all IoT devices are not telecommunicating with the application server and meta data exists in the broker server's priority queue, the broker server fetches the meta data that has the highest priority from its own priority queue. If multiple meta data with the same priority exists, the maximum number of meta data fetched at the same time is equal to the number of IoT devices. Additionally, these meta data must have a different device ID value. After that, the broker server assigns the sending rate while considering the network bandwidth to the fetched data and sends that message to the IoT device as a PERMIT message (D. in Figure 4.). The sending rate is decided by the total baud rate per number of PERMIT messages. The IoT device that received the PERMIT message sends content data that has been designated by data ID. Application server calculations of the receiving rate are reported to the broker server while receiving content data. The priority telecommunication phase repeats the above operation to achieve control that satisfies the requirement from the application's QoS.

Table 3. Structure of Priority-Decision Table

	Field Name	Description	Type
Condition	Row ID	Identifier of rows	8 bits unsigned integer
	Device ID	Identifier of sent devices	8 bits unsigned integer
	Data Type	Types of content data	Strings
	Data Interval	Sending interval of content data	Unixtime
Output	Priority	Sending priority (1 is highest)	Unsigned integer (1 to 15)

Table 4. Example of Priority-Decision-Table

Row-ID (R)	Device-ID (D)	Data-Type (T)	Data Interval (I)	Priority
1	1	Jpg	10 minutes	2
2	1	Jpg	1 minute	6
3	2	Txt	10 seconds	1
4	2	Jpg	10 minutes	6

* Letters inside brackets are abbreviations used in Figure 5.

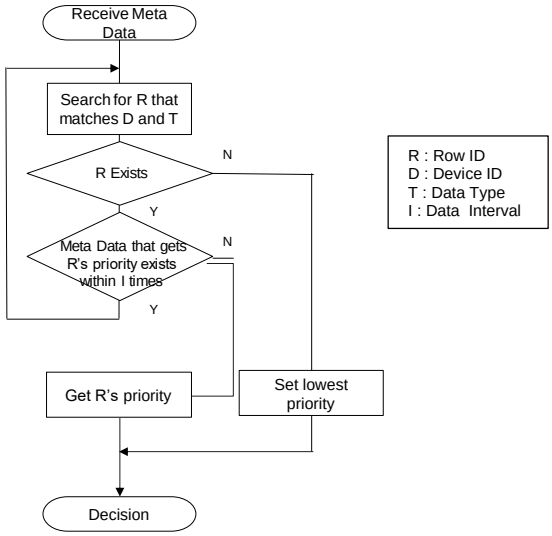

Figure 5. Algorithm of decision priority

3.5 Details of Deciding Priority

Priority in the proposed mechanism is determined in 15 steps. The highest priority is 1. Priority is decided in accordance with three elements; "sent device", "type of data", and "generated time". The proposed mechanism considers "when", "which", and "what".

The process of deciding priority uses meta data from the IoT device and priority-decision table in the broker server.

The meta data is the property of the content data. The meta data is assigned a 1-to-1 relationship with the content data. The meta data is generated when the IoT device generates content data. **Table 2** shows the structure of the meta data. The elements of the meta data are data ID, device ID, data size, and data type. The data size is the size of the content data. In contrast, the meta data does not have generated time because synchronizing the clock in each IoT device is difficult. The proposed mechanism regards the arrived time at the broker server as the generated time. The meta data expects a data type element with a size of 5 bytes. Accordingly, the proposed mechanism enables sending of meta data without occupying the sending bandwidth.

The priority-decision table is a table that describes the rules of deciding priority. **Table 3** shows the structure of the priority-decision table. The elements of the priority-decision table are row ID, device ID, data type, and data interval. The data interval expresses the sending interval of content data from the IoT device.

Table 4 shows an example of a priority-decision table. The priority-decision table recognizes different rows if either device ID, data type, or data interval has different values like Table 4's records. Therefore, the proposed mechanism enables control of more details than each session, such as "even if every device has the same data, assign different priority to each device", and "even if the device sent the same data every minute, assign high priority only every 10 minutes".

Figure 5 shows the decision priority algorithm. The broker server searches for the row that matches the condition in the priority-decision table in row ID order. The broker server decides the priority from the matched row first. If there is not match in the table, the broker server assigns the lowest priority.

4. IMPLEMENTATION & EVALUATION

4.1 Overview

In this section, we implemented and evaluated the effectiveness of the priority telecommunication phase in the proposed mechanism discussed in Section 3 as basic evaluation. **Table 5** shows the machinery used for this basic evaluation. Raspberry Pi 2 devices were used as IoT devices. Ubuntu was used as the platform for the application server and broker server.

In this evaluation, we experimented with data sending to verify that the proposed mechanism enables telecommunication by considering an application's QoS. In this evaluation, we send content data from four IoT devices. We evaluated two plans: applying the proposed mechanism and not applying it. We used two evaluation indicators: sending ratio of each priority (ratio of data that were able to be transmitted in evaluation times) and average delay time of each priority (required time from generation of data in IoT devices to complete transition). We compare two plans to confirm that the proposed mechanism enables telecommunication considering the applications QoS. If the proposed mechanism improves the sending ratio and average delay time more than when not applying the plan, we can say the proposed mechanism enables telecommunication considering an applications QoS.

Table 6, 7, and 8 show the scenarios of this evaluation. These scenarios were made by using requirements from the agriculture support system in our laboratory. This system has a minimum requirement that the IoT devices send data once per 5 minutes (however, in nighttime it sends only sensor data, which is acceptable). Therefore, the scenario in Table 7 is in daytime. In that scenario, large image data were sent with high priority once per 10 minutes. Small image data and sensor data were sent with high priority once per 5 minutes. The scenario of Table 8 is in nighttime. In that scenario, large image data were sent with high priority once per 10 minutes. Similarly, small image data and sensor data were sent with high priority once per 5 minutes. The sending data in scenarios are generated by cameras and sensors once per minute. **Table 9** shows the environment of this evaluation.

Table 5. Equipment used in this evaluation

	IoT devices (Raspberry Pi 2)	Application Server Broker Server
OS	Raspbian 8.0	Ubuntu 14.04 LTS
CPU	ARM Cortex-A7	Intel Corei7 5820K
Main Memory	1 GB	16 GB
Programing Language	C++11(GCC4.8) + Boost1.55	
Buffer Size	32MB	-

Table 6. Scenarios of this evaluation (structure of IoT devices)

Device-ID	Device Name (Data Type)	Data amount	Data Generate Interval
101	Camera (large: JPG, small: SJPG)	2 MB / 200 KB	1 minute
102			
103	Sensor GW (TXT)	80B	
104			

Table 7. Scenarios of this evaluation (daytime)

Row ID	Device ID	Data Type	Data Interval	Priority
1	101,102	SJPG	5 minutes	1
2	101,102	JPG	10 minutes	2
3	103,104	TXT	5 minutes	3
4	101,102	SJPG	1 minute	4
5	101,102	JPG	1 minute	5
6	103,104	TXT	1 minute	6

Table 8. Scenarios of this evaluation (nighttime)

Row ID	Device ID	Data Type	Data Interval	Priority
1	103,104	TXT	5 minutes	1
2	103,104	TXT	1 minute	2
3	101,102	SJPG	5 minutes	3
4	101,102	SJPG	1 minute	4
5	101,102	JPG	10 minutes	5
6	101,102	JPG	1 minute	6

4.2 Results and Discussions

Figures 6 and **7** show the result of the sending ratio of each priority in daytime and nighttime. In the daytime results, the sending ratio rose by an average of 12.0% up to 29.2% with a priority level of 1 to 5. On the other hand, in the nighttime results, the sending ratio fell by an average of 14.5%. However, the priority levels that led to a fall in the sending ratio are level 5 and 6 in each scenario. Priority levels 5 and 6 are lower priority than levels 1 to 4. For this reason, the fact that the sending ratio fell with priority levels 5 and 6 is not a problem. If the results are limited to priority levels 1 to 4, the average sending ratio rose 17.7% in daytime and 1.2% in nighttime. In addition, all the sending rates with a higher priority were higher than those with lower priority.

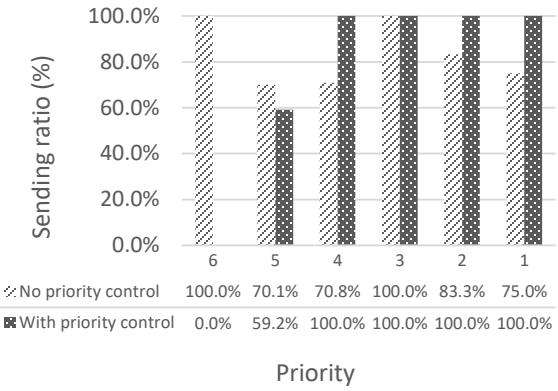

Figure 6. Results of sending ratio of each priority in daytime

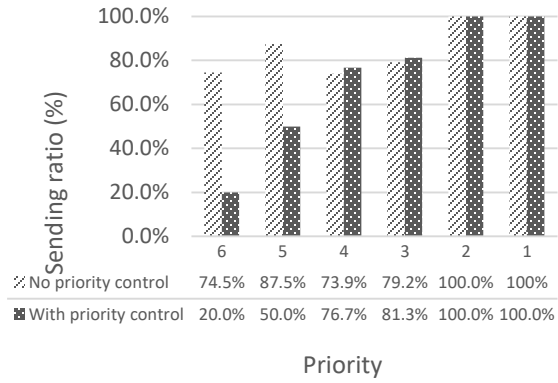

Figure 7. Results of sending ratio of each priority in nighttime

Figures 8 and **9** show the results of the average delay time of each priority in daytime and nighttime. With the daytime results, the proposed mechanism takes longer than when not applying the proposed mechanism. We presume the reason for this is the priority telecommunication phase in the proposed mechanism does not reassign the sending rate in real time because the broker server assigns the sending rate to the content data only when the broker server sends a PRI message. Therefore, the proposed mechanism cannot follow the bandwidth variation in real time completely. If the bandwidth rises, the sending rate of content data in the middle of sending does not rise. For this reason, the proposed mechanism takes longer than when used the normal telecommunication (not applying the proposed mechanism). This is future work. On the other hand, with the nighttime results, the proposed mechanism takes a shorter time: an average of 81.2% up to a maximum of 96.0% compared to when not applying the proposed mechanism, except for priority levels 1 and 2. The level 1 and 2 results are longer by about 8.5 seconds due to the overhead time of exchanges with the broker server. In addition, all average delay times with higher priority were shorter than those with lower priority.

From the above, the proposed mechanism could satisfy the minimum requirement of the agriculture support system. Therefore, the proposed mechanism functions as a heterogeneous remote monitoring IoT system. However, we must improve the assigning sending rate algorithm in the priority telecommunication phase to renew it in real time. This is future work. As a result, the proposed mechanism considers the application's QoS for telecommunication.

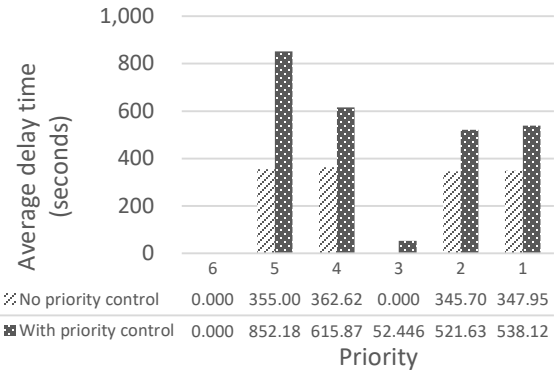

Figure 8. Results of average delay time of each priority in daytime

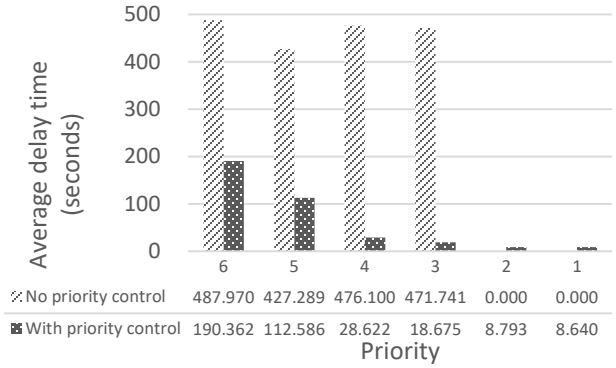

Figure 9. Results of average delay time of each priority in nighttime

5. CONCLUSION

In this paper, we propose a priority control mechanism for a heterogeneous remote monitoring IoT system to enable a best-effort IoT system that guarantees only telecommunication at the lowest limit. The proposed mechanism controls sending rates using application requirements and data characteristics. We implemented and evaluated the effectiveness of the priority telecommunication phase in the proposed mechanism. The results of sending ratio of each priority rose to 29.2% in daytime. Similarly, the results of average delay time of each priority shortened to 96.0% in nighttime. As a result, we propose a priority control mechanism that considers the application's QoS for telecommunication. For future work, we will launch and evaluate an actual agriculture support system. Furthermore, we will improve the assigning sending rate algorithm.

6. ACKNOWLEDGMENT

This work was supported by JSPS Grant-in-Aid for Scientific Research B Grant Number JP26280028 and JP15H02697.

7. REFERENCES

[1] Towards a Definition of the Internet of Things (IoT), IEEE, 2016(online) , available from <http://iot.ieee.org/images/files/pdf/IEEE_IoT_Towards_Definition_Internet_of_Things_Revision1_27MAY15.pdf>, (accessed 2016-02-15).

[2] Atzori, L., Iera, A. and Morabito, G. 2010. The internet of things: A survey. *Computer networks*, *54*(15), 2787-2805.

[3] Gubbi, J., Buyya, R., Marusic, S. and Palaniswami, M. (2013). Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, *29*(7), 1645-1660.

[4] Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2014-2019. Technical report, Cisco, 2015.

[5] Sheng, Z., Mahapatra, C., Zhu, C. and Leung, V. C. 2015. Recent advances in industrial wireless sensor networks toward efficient management in IoT. *IEEE Access*, *3*, 622-637.

[6] Decuir, J. 2015. The Story of the Internet of Things: Issues in utility, connectivity, and security. *IEEE Consumer Electronics Magazine*, *4*(4), 54-61.

[7] Zorzi, M., Gluhak, A., Lange, S. and Bassi, A. (2010). From today's intranet of things to a future internet of things: a wireless-and mobility-related view. *IEEE Wireless Communications*, *17*(6), 44-51.

[8] Palattella, M. R., Accettura, N., Vilajosana, X., Watteyne, T., Grieco, L. A., Boggia, G. and Dohler, M. 2013. Standardized protocol stack for the internet of (important) things. *IEEE Communications Surveys & Tutorials*, *15*(3), 1389-1406.

[9] Sheng, Z., Yang, S., Yu, Y., Vasilakos, A., V., McCann, J. A. and Leung, K. K. 2013. A survey on the ietf protocol suite for the internet of things: Standards, challenges, and opportunities. *IEEE Wireless Communications*, *20*(6), 91-98.

[10] MQ Telemetry Transport, MQTT.org(online), available from <http://mqtt.org/> (accessed 2015-04-01).

[11] Nishioka, T., Machida, T., Arai, D., Ogishi, T. and Mineno, H., 2016, Proposal of Mobile Data Offloading Protocol to Reduce Temporal Locality of Mobile Data Traffic. IPSJ Journal (in Japanese), Vol.58, No.1, 2017 (in press).

[12] Shelby, Z., Hartke, K. and Bormann, C. 2014. The constrained application protocol (CoAP). No. RFC 7252.

[13] Advanced Message Queueing Protocol, Organization for the Advancement of Structured Information Standard (online), available from < http://www.amqp.org/ > (accessed 2016-08-08).

[14] 3GPP. 2013. System improvements for Machine-Type Communications （MTC）. TR 23.888.

[15] Ibayashi, H., Kaneda, Y., Imahara, J., Oishi, N., Kuroda, M. and Mineno, H. 2016. A Reliable Wireless Control System for Tomato Hydroponics, MDPI Sensor, Vol.16.