Proposal of Gamma Which Is a Spatial Data Sharing Distributed MQTT System

Takahiro Ueda*, Tetsuro Katayama*, Yoshihiro Kita†,
Hisaaki Yamaba*, Kentaro Aburada*, Naonobu Okazaki*

** Department of Computer Science and Systems Engineering, Faculty of Engineering, University of Miyazaki,*
1-1 Gakuen-kibanadai nishi, Miyazaki, 889-2192 Japan

†Department of Information Security, Faculty of Information Systems, Siebold Campus, University of Nagasaki
1-1-1 Manabino, Nagayo-cho, Nishi-Sonogi-gun, Nagasaki, 851-2195 Japan

E-mail: uedat@earth.cs.miyazaki-u.ac.jp, kat@cs.miyazaki-u.ac.jp, kita@sun.ac.jp,
yamaba@cs.miyazaki-u.ac.jp, aburada@cs.miyazaki-u.ac.jp, oka@cs.miyazaki-u.ac.jp

Abstract

This paper proposes Gamma, which is a new distributed MQTT system, to improve the usefulness of distributed MQTT systems for sharing spatial data. In the experiment, it has been found that Gamma can process more messages than a single MQTT broker. It is confirmed that Gamma achieves scalability by increasing the number of Gateways and distributed MQTT brokers. Furthermore, it is confirmed that the efficiency of the distributed MQTT system can be improved by setting the responsibility area of Gateways.

Keywords: MQTT, Spatial Data, Load management, Information management, S2 Geometry

1. Introduction

Spatial data is "data that has as its elements both location information and semantic information (state, shape, size, etc.) of objects". For example, data such as "a bicycle is traveling at 30 km/h at point A."[1] In recent years, with improvements in the processing power of smartphones, the performance of cameras and other sensors, and improvements in image processing technology, it has become possible to acquire spatial data such as the type and location of objects captured by smartphone cameras in real time[2].

MQTT (Message Queuing Telemetry Transport)[3] is a well-known application layer communication protocol for the IoT (Internet of Things). Since the basic MQTT defines one system with one broker to mediate messages, the number of Publishers to send messages and Subscribers to receive messages is limited. Therefore, there are distributed MQTT implementations that work together among multiple brokers and perform load balancing.

A previous work on distributed MQTT for sharing spatial data is the work of Ryo Kawaguchi et al[4]. They introduced a topic structure that is suitable for handling spatial data. They also introduced a distributed MQTT broker and Gateway to reduce the load on the broker and to support different types of brokers. By numerical calculations, they confirmed that each broker received fewer messages per unit time than the existing distributed MQTT system, especially when the number of Subscribers was large or when Subscriber's topic was changed frequently.

However, there are two problems as below when their proposed distributed MQTT system is used actually.

- The configuration of the distributed MQTT broker cannot be changed while the system is running because IP address and port number of the distributed MQTT broker must be registered with the Gateway in advance.

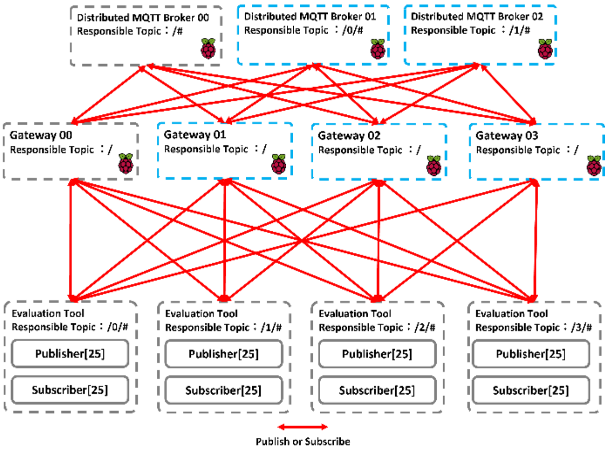

Fig. 1. System structure example (without Gateway's topic setting)

- Messages cannot be shared efficiently between the distributed MQTT broker and the Gateway because the Gateway is not configured with information on the topic it is responsible for.

Therefore, this paper proposes and implements a new distributed MQTT system, Gamma, to improve the usefulness of distributed MQTT systems for sharing spatial data. Specifically, to solve the above two problems, this paper adds Manager to the previous study[4] and implements five functions.

2. Gamma functions

In this chapter, we describe the functionality of Gamma and its client library. The Gamma implemented in this research can add a distributed MQTT broker and Gateway when the load on the system becomes large. This makes it possible to handle a larger number of messages. The followings describe the five functions implemented in this paper.

- Function to add a distributed MQTT broker while the system is running:
  This function allows system administrators to add distributed MQTT brokers while the system is running. This function is mainly performed by the Manager which is newly added in Gamma. The Manager also performs the process of updating the connection information of the distributed MQTT broker held by Gateway when a distributed MQTT broker is newly added.
- Function to add a Gateway while the system is running:

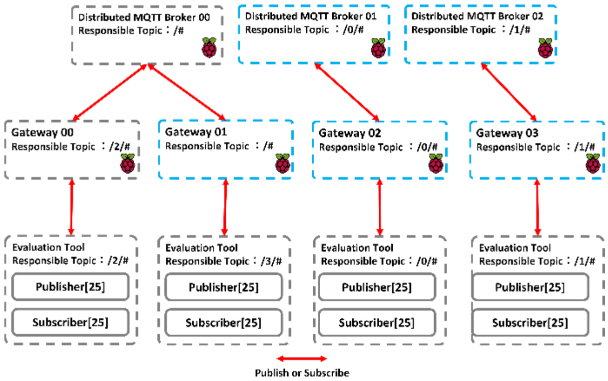

Fig. 2. System structure example (with Gateway's topic setting)

  This function allows the system administrator to add a Gateway while the system is running. The Manager receives the connection information of the newly added Gateway from it Gateway after the system starts and manages it. It is also responsible for notifying the newly added Gateway of the connection information of the distributed MQTT broker.
- Function to set the Gateway's assigned topic:
  This function allows the system administrator to set the topic for which the Gateway is responsible (topic is synonymous with area). This function is mainly performed by the Manager newly added in Gamma, which accepts the request from the system administrator to set the topic for which the Gateway is responsible and sets it. It is possible to set multiple topics for one Gateway.
- Function to select Gateway to connect based on location information:
  This function selects the Gateway to be connected based on the location information of the spatial data published/subscribed by the Client. The Client connects to the Manager before connecting to the Gateway and obtains the information about the connection to the Gateway and the topic that each Gateway is in charge of. Therefore, the Client only needs to know the connection information to the Manager. In addition, the Client can select and connect to the most suitable Gateway by comparing the topic of responsibility of each Gateway obtained from the Manager and the topic to be published/subscribed by the Client itself.

Table 1. Checking the scalability gained by increasing the number of distributed MQTT brokers

	Number of distributed MQTT brokers	1	2	3
Without responsibility area	Average delay time [ms]	14925.22	1163.05	118.92
With responsibility area	Average delay time [ms]	294.35	127.05	46.75

Table 2. Checking the scalability gained by increasing the number of Gateways

	Number of Gateways	1	2	3	4
Without responsibility area	Average delay time [ms]	11423.46	233.15	118.73	42.65
With responsibility area	Average delay time [ms]	11423.46	116.14	57.30	15.96

- Function to publish/subscribe to spatial data based on location information:
  This function allows the client to publish/subscribe to spatial data based on location information. The client calculates a topic to publish based on the latitude and longitude of the spatial data, and can publish the spatial data. It can also calculate topics for subscribing to spatial data in that range from the given latitude, longitude, and radius, and subscribe to it. This function uses S2 Geometry[5] for these calculation.

3. Evaluation of Gamma

We evaluate the usefulness of Gamma implemented in this paper.

Firstly, we compare it with the single MQTT broker. As a result, we confirmed that Gamma is capable of processing more messages than the single MQTT broker.

Next, we will conduct an experiment to evaluate the scalability that can be obtained by increasing the number of Gateways. Fig. 1 shows the configuration of the system when no topic is assigned to the Gateway, and Fig. 2 shows the configuration of the system when a topic is assigned to the Gateway.

The experiments shown below were conducted under the following conditions: the length of the transmitted message was 150 characters, the number of Publishers was 100, the number of Subscribers was 100, the publish interval for each Publisher was 100 milliseconds, and the measurement time was 100 seconds. In addition, Publishers and Subscribers were equally placed in four areas.

3.1. *Experiments and Discussion on Scalability Gained by Increasing the Number of Distributed MQTT Brokers*

Check the scalability by varying the number of distributed MQTT brokers; the number of Gateways is fixed to four.

Here, when the upper limit of the CPU utilization of the distributed MQTT broker is viewed in the entire command line tool called cputool[6] during the experiment, the percentage of the load occupied by the Gateway is higher than that of the distributed MQTT broker. In addition, there were only four Gateways that could be prepared in this paper. However, even when there are four Gateways for only one distributed MQTT broker, the problem happens that the processing capacity of the Gateway became insufficient before that of the distributed MQTT broker. Therefore, in this experiment, we will intentionally reduce the processing capacity of the distributed MQTT broker using cputool and confirm the scalability obtained by increasing the number of distributed MQTT brokers.

From the experimental results in Table 1, we can see that the average delay time decreases with the number of distributed MQTT brokers. Because of this, we can see that increasing the number of distributed MQTT brokers improves the processing capacity of the entire system. Therefore, we were able to confirm that scalability is ensured by increasing the number of distributed MQTT brokers.

In addition, Table 2 shows that the average delay time is smaller in the case of "with topic setting" than in the case of "without topic setting," regardless of the number of Gateways. From this, it can be said that the setting of the topic to the Gateway, which is one of the functions of

Gamma implemented in this paper, has improved the efficiency of the entire system.

3.2. *Experiments and Discussion on Scalability Obtained by Increasing the Number of Gateways*

Check the scalability by varying the number of Gateways. The number of distributed MQTT brokers is fixed to one.

Table 2 shows the results of the experiment when the number of Gateways is varied, and it can be confirmed from Table 2 that the average delay time decreases with the number of Gateways regardless of whether the topic for which the Gateway is responsible is set or not. In other words, by increasing the number of Gateways, the processing capacity of the entire system can be improved. It can be confirmed that Gamma implemented in this paper ensures scalability by increasing the number of Gateways.

From Table 2, we can see that the average delay time is smaller in the case of "with topic setting" as shown in Fig. 2 than in the case of "without topic setting" as shown in Fig. 1. Therefore, it can be said that the setting of the assigned topic in the Gateway, one of the functions of Gamma implemented in this paper, has improved the efficiency of the entire system.

4. Conclusion

This paper has proposed and implemented a new distributed MQTT system, Gamma, to improve the usefulness of distributed MQTT systems for sharing spatial data.

A previous study of distributed MQTT for sharing spatial data is the work of Ryo Kawaguchi et al[4]. This previous research has two problems, which are described in Chapter 1.

In order to solve these two problems, Gamma adds Manager to the previous study[4], and is implemented five functions. We have confirmed that the five functions of Gamma properly are run. This confirms that Gamma solves the problem of not being able to change the configuration of the distributed MQTT broker while the system is running.

The experiment also showed that Gamma can process more messages than a stand-alone MQTT broker, and it has been confirmed that scalability can be ensured by increasing the number of Gateways and distributed MQTT brokers. Furthermore, it has been confirmed that the efficiency of the entire system can be improved by setting the topic of responsibility of the Gateway so that the load is not unevenly distributed to a specific Gateway. From this, it has been confirmed that the problem of not being able to efficiently share messages between the distributed MQTT broker and the Gateway because the information on the topic of responsibility is not set in the Gateway could be solved.

From the above, Gamma realized the following two points and solved the two problems of the previous studies.

- The ability to add a distributed MQTT broker while the system is running
- The ability to notify the Client of the information of multiple Gateways and the topics for which the Gateway is responsible.

In summary, we have confirmed that Gamma proposed and implemented in this paper is useful.

Future issues are as follows.

- Support for automatic addition of Gateways and distributed MQTT brokers based on load.
- Add the ability to monitor the activity and death of Gateway and distributed MQTT brokers
- Support for single-level wildcards

References

1. "Utilizing Spatial Data: The Future Visible through Objects and Locations" (in Japanese), https://www.nttdata.com/jp/ja/data-insight/2020/033001/ (Accessed 2021-12-14)
2. "DRiVR", https://drivr.app/ (Accessed 2021-12-14)
3. "MQTT Version 3.1.1", http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html (Accessed 2021-12-14)
4. Ryo Kawaguchi and Masaki Bandai, "A Distributed MQTT Broker System for Location-based IoT Applications", 2019 IEEE International Conference on Consumer Electronics (ICCE), pp.1-4, 2019.
5. "S2 Geometry", https://s2geometry.io/ (Accessed 2021-12-14)
6. "cputool", http://manpages.ubuntu.com/manpages/bionic/man8/cputool.8.html (Accessed 2021-12-14)

Authors Introduction

Takahiro Ueda

Takahiro Ueda received the Bachelor's degree in engineering (computer science and systems engineering) from the University of Miyazaki, Japan in 2021. He is currently a Master's student in Graduate School of Engineering at the University of Miyazaki, Japan. His research interests software development support method and networking.

Tetsuro Katayama

Tetsuro Katayama received a Ph.D. degree in engineering from Kyushu University, Fukuoka, Japan, in 1996. From 1996 to 2000, he has been a Research Associate at the Graduate School of Information Science, Nara Institute of Science and Technology, Japan. Since 2000 he has been an Associate Professor at the Faculty of Engineering, Miyazaki University, Japan. He is currently a Professor with the Faculty of Engineering, University of Miyazaki, Japan. His research interests include software testing and quality. He is a member of the IPSJ, IEICE, and JSSST.

Yoshihiro Kita

Yoshihiro Kita received a Ph.D. degree in systems engineering from the University of Miyazaki, Japan, in 2011. He is currently an Associate Professor with the Faculty of Information Systems, University of Nagasaki, Japan. His research interests include software testing and biometrics authentication.

Hisaaki Yamaba

Hisaaki Yamaba received the B.S. and M.S. degrees in chemical engineering from the Tokyo Institute of Technology, Japan, in 1988 and 1990, respectively, and the Ph D. degree in systems engineering from the University of Miyazaki, Japan in 2011. He is currently an Assistant Professor with the Faculty of Engineering, University of Miyazaki, Japan. His research interests include network security and user authentication. He is a member of SICE and SCEJ.

Kentaro Aburada

Kentaro Aburada received the B.S., M.S, and Ph.D. degrees in computer science and system engineering from the University of Miyazaki, Japan, in 2003, 2005, and 2009, respectively. He is currently an Associate Professor with the Faculty of Engineering, University of Miyazaki, Japan. His research interests include computer networks and security. He is a member of IPSJ and IEICE.

Naonobu Okazaki

Naonobu Okazaki received his B.S, M.S., and Ph.D. degrees in electrical and communication engineering from Tohoku University, Japan, in 1986, 1988 and 1992, respectively. He joined the Information Technology Research and Development Center, Mitsubishi Electric Corporation in 1991. He is currently a Professor with the Faculty of Engineering, University of Miyazaki since 2002. His research interests include mobile network and network security. He is a member of IPSJ, IEICE and IEEE.