MQTT Protocol Extension for Real Time Location based Service

In Hwan Jung*
School of Computer Engineering, Hansung University, Korea. E-mail: ihjung@hansung.ac.kr

Jae Moon Lee
School of Computer Engineering, Hansung University, Korea. E-mail: jmlee@hansung.ac.kr

Kitae Hwang
School of Computer Engineering, Hansung University, Korea. E-mail: calafk@hansung.ac.kr

Received September 27, 2021; Accepted December 20, 2021
ISSN: 1735-188X
DOI: 10.14704/WEB/V19I1/WEB19314

Abstract

This paper introduces an extension of the MQTT protocol to reduce communication overhead in real-time location-based service system where mobile MQTT clients move and change their administrative district location frequently. The MQTT protocol was extended to handle UNSUBSCRIBE and SUBSCRIBE processing with a single step and the LBS server was revised to handle the extended protocol. Instead of directly processing SUBSCRIBE and UNSUBSCRIBE, mobile clients need to send their GPS location to the LBS server. LBS Server communicate with MQTT Broker to using extended protocol on behalf of each client. In an environment where large number of clients frequently move, the extended MQTT protocol can drastically reduce the number of messages between MQTT Broker and clients. This is mainly because clients do not need to issue SUBSCRIBE and UNSUBSCRIBE request to MQTT Broker. In addition, in contrast to our previous study where clients receive new administrative area name after sending its GPS location to LBS server, in this research, using extended MQTT protocol, mobile clients do not need to receive new administrative name even though they move across different district areas and as a result, the number of message exchanges can also be reduced. The LBS system applying the extended MQTT protocol can be used as a real-time location-based information service for large-scale mobile devices, such as real-time pedestrian population and vehicle traffic analysis, and location-based message delivery.

Keywords

MQTT, MQTT Protocol Extension, Location Based Service, Administrative District, GPS.

Introduction

With the emergence of mobility devices such as IoT for vehicles and smartphones, location based IoT application systems are emerging that process existing sensor information and GPS information of IoT devices together (Aslava et al., 2015; Sung, 2017; Ryoo et. al., 2019). The MQTT protocol effectively collects sensor information from the IoT devices based on Publish-Subscribe scheme (MQTT, n.d.; IBM, 2012). In our previous study, we have implemented Seoul Location base IoT Messaging System (SLIMS) that uses the administrative district name as an MQTT topic (Jung et at., 2018). In SLIMS, when the IoT device moves and the administrative district changes, the conventional MQTT protocol was used to unsubscribe from the old topic and subscribed to the new topic, which is identical to the administrative district name. In SLIMS, if the number of client increases and they move frequently across different district regions, the number of messages for changing topics inevitably increases and performance is degraded. In this paper, clients do not need to send UNSUBSCRIBE and SUBSCRIBE requests to MQTT Broker directly. Instead, clients only need to send their current location to the Location Based Service (LBS) Server. The MQTT protocol is extended so that the client is automatically subscribed to a new topic. The LBS server keeps track of the location of clients, and when the location changes, it sends an extended protocol request to the MQTT Broker that handles UNSUBSCRIBE and SUBSCRIBE of the client with a single request. The MQTT broker moves the client context from the old topic to the new topic in the subscription tree when the LBS server requests a topic change of the specific client. Furthermore, mobile clients do not need to receive administrative district name as a topic for SUBSCRIBE so that the number of message exchange can be reduced additionally. In this way, the number of messages exchanged between MQTT Client and Server is reduced and as a result performance degradation can be resolved.

Related Works

1. Related Works

One of the important features of MQTT is that topic can be organized into a hierarchical structure. In this paper as in SLIMS, we apply the hierarchical structure of MQTT Topic to administrative districts. Figure 1 shows the change of Client-A's topic when Client-A moves from “Seoul City / Seongbuk-GU / Jungnung-DONG” to “Seoul City / Seongbuk-GU / Gilum-DONG”. If Client-A's initial location is “Seoul City / Seongbuk-GU / Jungnung-DONG”, Client-A subscribes topics of “Seoul City”, “Seongbuk-GU” and “Jungnung-

DONG" respectively. If Client-A moves to "Gilum-DONG", Client-A unsubscribes from "Jungnung-DONG" and subscribes to "Gilum-DONG".

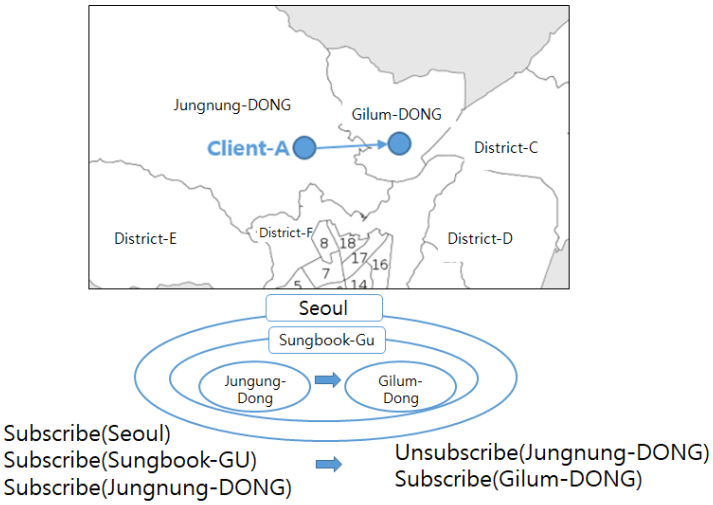

Figure 1. MQTT Topic and District Name

Because the clients are subscribed to a specific topic which is same as administrative district name, the application will be able to determine which clients are staying in which administrative areas, and to specify a specific administrative area to deliver messages to the clients. We use data that can divide administrative districts in Seoul city. In Seoul, there are total of 25 GU's and 424 DONG's (Seoul, 2018).

In this paper, clients do not need to translate its current location to administrative district name by itself. Instead, the LBS Server plays a role to keep track of every client's location and mange administrative district name according to LBS system's business logic.

2. MQTT Topic Change in SLIMS

Figure 2 depicts system architecture and topic change processing in SLIMS. As shown in the Figure 2, in SLIMS, once a client identifies its location, it subscribes a specific topic, which is identical to administrative district name, and send its current location to LSBS server periodically. When the client moves while changing the administrative area, it must request UNSUBSCRIBE and SUBSCRIBE respectively. Figure 2 also describes how many message exchange is necessary for topic change processing in SLIMS. Due to conventional MQTT protocol is used for this case, at least 8 message exchanges is observed for UNSUBSCRIBE and SUBSCRIBE processing. In this respect, when there are many clients

that frequently move, performance degradation is inevitable because the number of message exchanges increases for topic change.

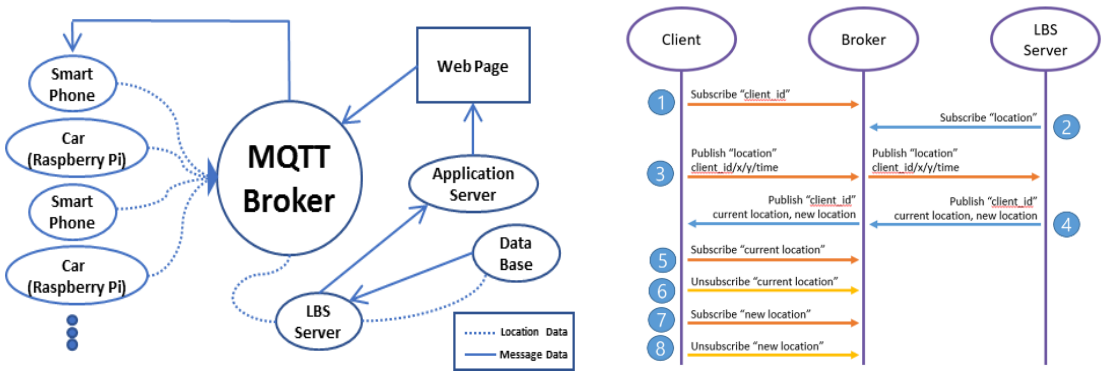

Figure 2. System Architecture and Topic Change Processing in SLIMS

MQTT Protocol Extension Design

1. Extended Protocol Flow

In contrast to using conventional MQTT protocol for UNSUBSCRIBE and SUBSCRIBE for topic change, in this research, the PUBLISH message is extended to process topic change with a single request issued by the LBS Server. As a result, the number of communication between the client and the MQTT broker did not increase for topic change, and the performance degradation was reduced. Figure 3 shows the implemented MQTT protocol extension. In Figure 3, the client transmits the current coordinates to the LBS server through MQTT PUBLISH message. The LBS server, which keeps track and manages the client's location, converts the client's coordinates into the administrative district name and requests the MQTT broker to change the new topic, which is identical to new location of the client. In this case, instead of requesting UNSUBSCRIBE and SUBSCRIBE respectively by clients, the PUBLISH message format was extended to change the topic with a single request issued by the LBS Server. The LBS server sends a single extended PUBLISH message using a special topic “CHANGE” and attach payload data including client ID, old topic and new topic. Figure 4 depicts context change flow for topic change. When there is a topic change request in a PUBLISH message sent by the LBS server, the MQTT broker moves the client context from old topic to new topic in the topic subscription. In this way, the messages exchange between MQTT Client and Server for topic change can be eliminated and as a result performance degradation can be reduced. The extended MQTT protocol was newly implemented in SLIMS's MQTT Broker and LBS Server.

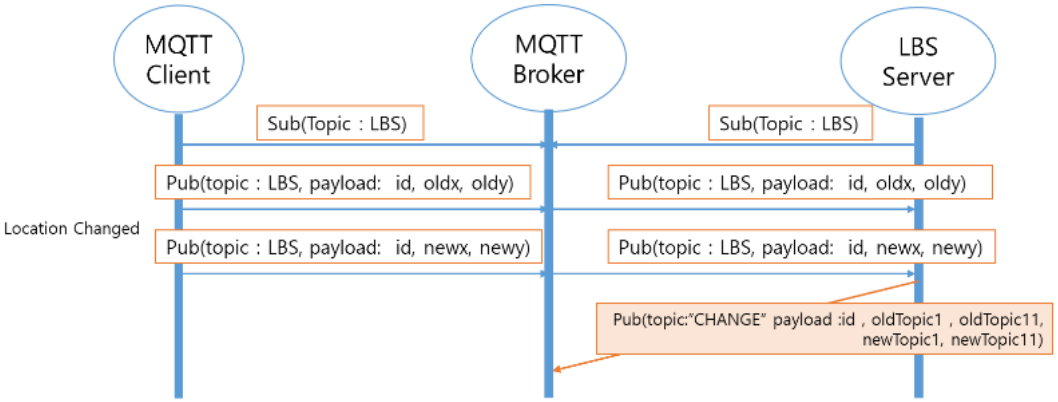

Figure 3. Extended Protocol Flow for Topic Change

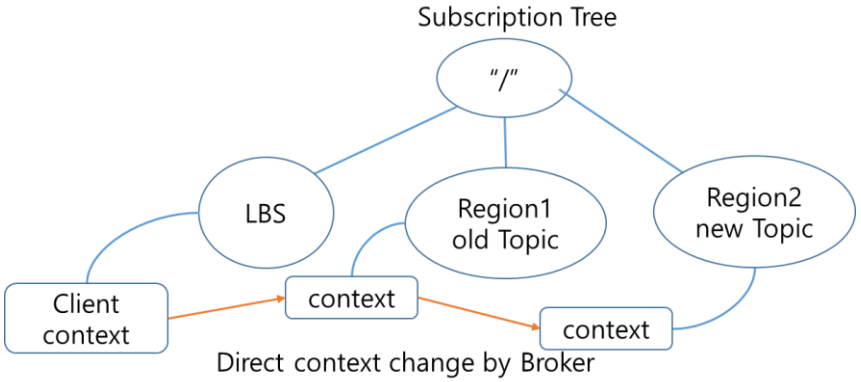

Figure 4. Client Context Change Flow for Topic Change by MQTT Broker

2. Extended Protocol Format

Figure 5 describes extended MQTT protocol. The MQTT PUBLISH message is used for extension. A special topic name "CHANGE" and QoS level 3 are used for specify topic change protocol extension. In addition, the payload is included containing client ID, old topic name and new topic name.

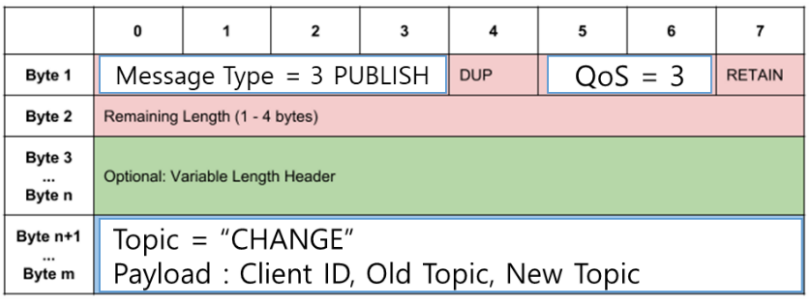

Figure 4. Extended MQTT PUBLISH Protocol Format

Implementation

1. Client Implementation

In SLIMS, as shown in Figure 6(a), if the client moves and the administrative area changes, a new administrative district name is received from the server and it unsubscribes from the old topic and subscribes to the new administrative name as a topic. On the other hand, in this paper, as shown in Figure 6(b), the client algorithm is revised such that it simply sends its GPS location to LBS server and do not need to receive any message related to topic change.

```
// Check GPS location
CurGPS = Get_Current_GPS();
// Send GPS information to the server
Send_Current_GPS_to_Server(CurGPS);
// Subscribe to the new administrative area
// sent by the server as a new Topic
NewMsg = Receive_Msg_from_Server();
NewTopic = NewMsg.NewTopic;
if (NewTopic!=CurTopic) { // Location is changed
      Unsubscribe(CurTopic);
      Subscribe(NewTopic);
      CurTopic = NewTopic;
}
```

(a) Existing Algorithm in SLIMS

```
// Check GPS location
CurGPS = Get_Current_GPS();

// Send GPS information to the LBS server and
Send_Current_GPS_to_Server(CurGPS);

// do not need to receive topic name and subscribe
```

(b) Revised Algorithm

Figure 6. Revision of Client Algorithm for Location Change

(a) App for Pedestrians

(b) App for Drivers

Figure 7. Client Applications

Figure 7(a) shows the implemented smartphone application for pedestrians and Figure 7(b) shows the smartphone application of vehicle driver connected via Bluetooth to the Raspberrypi (Raspberrypi, n.d.) device installed at the vehicle.

2. LBS Server

Figure 8 shows the algorithm of LBS Server. When clients send their GPS location, LBS Server converts it into a topic name according to the scheme determined by the business logic. In this case, the topic name compared to the location information is stored in its own database. On the other hand, if there is no predetermined rule, the general administrative district name is used as the topic. In this case, the LBS server uses a map service API such as Google map. In any case, the LBS server converts the GPS location information to the topic name sent by the client, and if the topic name is changed, the LBS Server requests the MQTT Broker to change the topic of the client using the extended protocol.

```
Client.NewMsg = Client.ReceiveMsg();// Receive GPS data from clients
NewTopic = Convert_GPS_to_Topic(Client.NewMsg); // Convert GPS to Administrative name
if (NewTopic!=Client.CurTopic) { // Location is changed
      MQTTMsg NewMsg.Type  = PUBLISH;
      NewMsg.QoS = 3; // Use QoS 3 to specify extended protocol
      NewMsg.Topic = "CHANGE";
      NewMsg.Payload = Client.ID + Client.CurTopic + NewTopic;
      MQTTSend(NewMsg); // Send topic change request to MQTT Broker
      Client.CutTopic = NewTopic; // Remember current location
}
```

Figure 8. LBS Server Algorithm

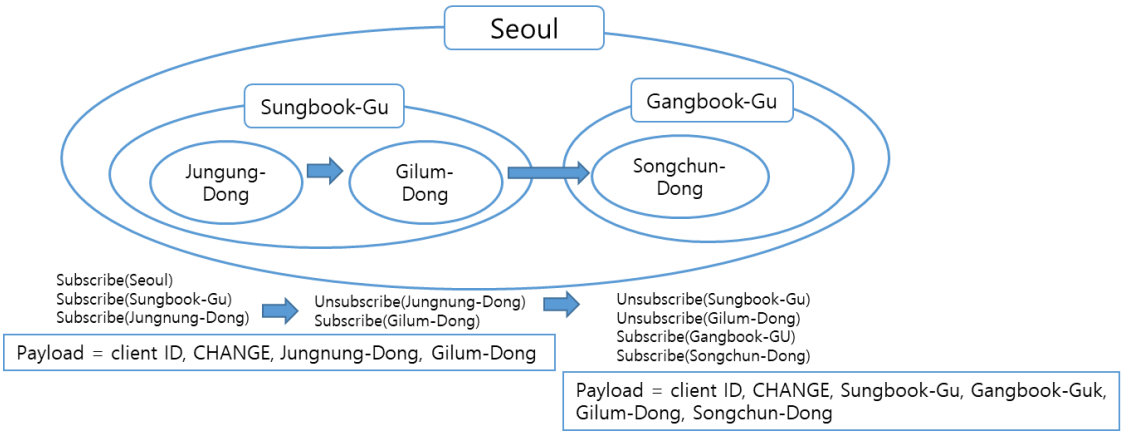

Figure 9. Payload Example for Topic Change

When the location of the client is changed in the lowest unit of the administrative district, the payload includes one set of old and new topics. However, if the location change occurs more than two levels based on the hierarchical structure of the administrative district, the payload includes two or more {old topic, new topic} sets. As shown in Figure 9, when the client location is changed from Jungnung-Dong to Gilum-Dong, only {Jungnung-Dong,

Gilum-DONG} is included in the payload. However, if the client location is changed from Gilum-Dong to Songchun-Dong, the payload requires two level change in the administrative district hierarchy. Therefore, {Songbook-Gu, Gangbook-Gu} and {Gilum-Dong, Songchun-Dong} two set are included.

```
MQTTClient LBSServer; // LBS Server is one of MQTT Client
MQTTMsg NewMsg = LBSServer.ReceiveMsg(); // Receive Topic Change Request from LBS Server
If (NewMsg.Type == PUBLISH and NewMsg.Topic == "CHANGE") {
        ClientContext Client = FindClientContext(NewMsg.Payload.Client_Id);
        For All Sets of {Old Topic, New Topic} {
                RemoveContextTree(Client, Old Topic);
                AddContextTree(Client, New Topic)
        }
}
```

Figure 10. MQTT Broker Topic Change Algorithm

3. MQTT Broker

As show in Figure 10, a new algorithm has been added to the MQTT Broker to handle the extended protocol. When there is a message sent by the LBS Server with the topic name "CHANGE", including Client ID and a set of {old topic, new topic}, the Broker finds the Client ID in the context table, removes the client context from the old topic subscription tree, and adds it to the new topic tree. This means that a client context is moved from old topic to new topic without communicating with the client. In this paper, the extended MQTT protocol was implemented by modifying the moquitto MQTT broker (Mosquitto, n.d.). The implemented MQTT Broker is mounted on a high-performance MQTT Appliance. The development of high-performance MQTT Appliance is the main objective of this research.

Experiment

An experiment was conducted to check whether the implemented MQTT protocol extension works properly and improves performance. Table 1 shows the experimental configuration. In order to simulate MQTT client traffic, the MQTT Load Test System (Jung et al., 2021) was used. The number of virtual clients was set to increase by 100 from 100 to 1000, and each client was set to change the region every 10 secs on average. Application Server transmits messages at an average rate of 100 messages/sec for each region. The main figure of merit is the average message reception rate (msg/sec) of clients. Figure 11 shows the performance evaluation results. The SLIMS represents conventional method while the SLIMS-Ex means extended protocol case implemented in this paper. If the number of clients is small, there is little difference in performance between the existing method and the proposed method. However, as the number of clients gradually increases, the SLIMS

shows the reception rate decreases due to performance degradation caused by large amount of UNSUBSCRIBE and SUBSRIBE message exchange. On the other hand, when the extended protocol is used, it shows that the message reception rate does not decrease significantly despite the increase in the number of clients.

Table 1

Experiment Configuration

Parameters	Value
Publishing Speed (# of Mgs/Region/sec)	100
# of Virtual Clients	From 100 to 1,000 step by 100
Test duration (sec)	60
Size of Message (bytes)	100
QoS Level	0
Region Change Interval for each client	10 sec

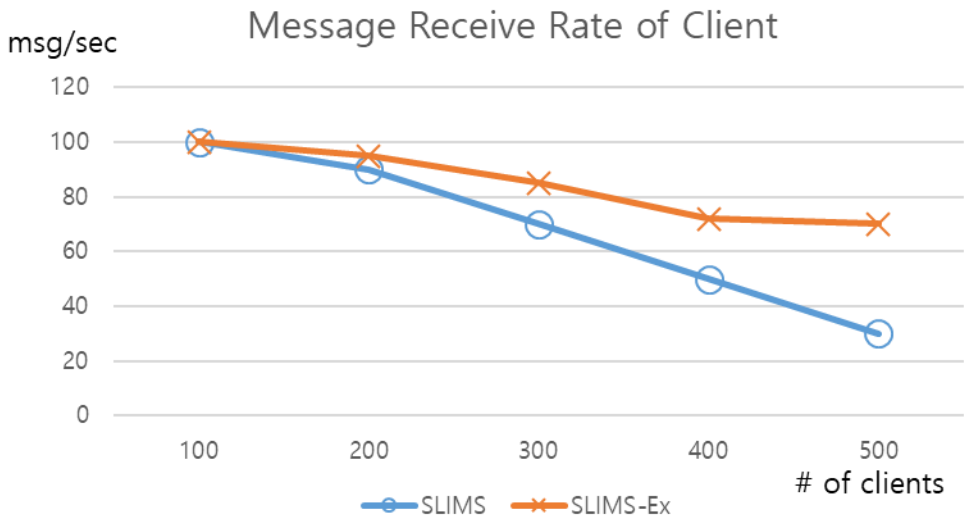

Figure 11. Experimental Results

Conclusion

In this paper, we have designed and implemented an extension of MQTT protocol for LBS system. In the implemented system, each client does not need to directly subscribe and unsubscribe when its administrative district changes as it moves. The client needs to send only its GPS information to LBS server. As a result, the extended MQTT protocol can effectively support location-based services especially when clients move frequently across different administrative regions. Extended MQTT protocol has been implemented into existing SLIMS and the experimental results shows observable performance enhancement

was achieved. Future research is to conduct performance evaluation for real world application and improve related algorithms under an environment in which a large number of IoT devices move more frequently.

Acknowledgment

This research was financially supported by Hansung University.

References

Aslava, H., Rojas, L.A. & Pereira. R. (2015). Implementation of Machine-to-Machine Solutions Using MQTT Protocol in Internet of Things (IoT) Environment to Improve Automation Process for Electrical Distribution Substations in Colombia. *Journal of Power and Energy Engineering*, *9*(11), 92-96.

IBM. (2012). Building smarter planet solutions with MQTT and IBM WebSphere MQ telemetry, *IBM Redbooks*, https://www.redbooks.ibm.com/abstracts/sg248054.html

Jung, I.H., Hwang, K.T., & Lee J.M. (2018). An MQTT based Real Time LBS System for Vehicles and Pedestrians, *International Journal of Engineering & Technology*, *6*(3), 125-139.

Jung, I.H., Hwang, K.T., & Lee J.M. (2021). Design and Implementation of MQTT Load Test System, *Turkish Journal of Computer and Mathematics Education*, *12*(6), 564-572.

Mosquitto (n.d.). Eclipse Mosquitto, https://mosquitto.org/

MQTT. (n.d.). MQTT: The Standard for IoT Messaging, https://mqtt.org/

Raspberrypi (n.d.). Raspberrypi Foundation, https://www.raspberrypi.org/

Ryoo, H.G. & Kwang-Ki Ryoo K.K. (2019), Kindergarten school bus notification service using IoT network, *Journal of Next-generation Convergence Technology Association*, *3*(1), 21-28.

Seoul. (2018). Seoul Mobile Platform, *Seoul Metropolitan Government Public Data*. https://data.seoul.go.kr/.

Sung, K.K. (2017) A Study on the IoT Technology Trend and Utilization Plan, *Journal of Next Generation Convergence Technology Association*, *17*(3), 121-127.