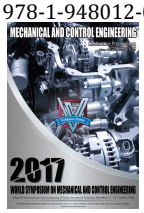
ISBN: 978-1-948012-06-5

Contents List available at VOLKSON PRESS

World Symposium on Mechanical and Control Engineering (WSMCE)



DESIGN AND IMPLEMENTATION OF MESSAGE PUSH SYSTEM BASED ON MQTT PROTOCOL

Yelei Guan, Xiaohui Cheng*, Congcong Xia

College of Information Science and Engineering, Guilin University of Technology Jiangan road No.12, Guilin, China.
Corresponding Author Email: cxiaohui@glut.edu.cn

ARTICLE DETAILS

Article History:

Received 02 october 2017
Accepted 06 october 2017
Available online 11 november 2017

Keywords

MQTT, publish/subscribe, Android, Apache Apollo, Eclipse PAHO

ABSTRACT

The huge growth has been made in mobile devices and smart terminals represented by Android mobile phones, and the number of users ushered in explosive growth. In order to adapt to the practical application of information acquisition timeliness, power consumption, and network environment and other requirements, the need to push the way to replace the traditional way to pull the message transmission. MQTT is a lightweight based on publish/subscribe messaging protocol, its emergence provides a new way for the realization of message push on Android platform. According to the requirement of Android platform message push protocol, the structure and message format of MQTT protocol are researched. Apache Apollo is used as the message push server, Eclipse PAHO API is used to write the client code and a unified character code is selected to solve the text of different platforms Compatibility problem, a message push system based on MQTT protocol is designed and implemented. The result shows that this system can better meet the basic functions and requirements of message push.

1. INTRODUCTION

MQTT (Message Queuing Telemetry Transport) is a messaging technology which is designed for Internet of thing. It is a protocol which is open, simple, lightweight and easy to achieve. It is particularly suited to the low bandwidth, the network which is not stable or expensive and the embedded devices or mobile terminal which processor and memory resources are limited [1].

The MQTT protocol developed by IBM and Eurotech company in 1999, it is a lightweight cross-platform messaging protocol based on publish/subscribe. MQTT is designed for the low bandwidth, the unstable network and the devices which calculation and processing are limited [2]. The protocol using small transmission which can low power consumption, greatly reduce the network flow, it is very suitable for the application of mobile systems, IBM has been successfully applied to the smart lab, St.Jude remote medical center and other projects. March 2013, OASIS (Organization for the Advancement of Structured Information Standards) announced MQTT as a First-choice standard of emerging Internet messaging protocol, with the advent of the Internet of things, the MQTT protocol will get unprecedented attention and widely used.

MQTT message content mainly consists of three parts: fixed head, variable head and payload, only fixed head is must contained in all message content. Its structure as shown in table 1.

Table 1: MQTT Fixed Head

Bit	7	6	5	4	3	2	1	0
byte1	Message Type				DUP flag	QoS level		Retain
byte2	Remaining Length							

2. MESSAGE MIDDLEWARE OVERVIEW

Message middleware integrates distributed system through data communication technology and realizes data exchange on different platforms through efficient and reliable message transmission mechanism. Through the messaging model, it can extend inter-process communication in distributed environment and is especially suitable for communication in complex network environments and frequent network changes [3].

Apache ACTIVEMQ is Apache's most popular open source messaging integration server. Apache ACTIVEMQ is fast and supports multilingual and cross-client protocols with easy and full support for both JMS 1.1 and 1.4 with J2EE Enterprise Integration Mode and many advanced features, and Apache ACTIVEMQ is released under Apache 2 license. ACTIVEMQ is one of the most popular open source messaging middleware and supports a wide range of programming languages [4,5]. Most notably, Apache ACTIVEMQ has become quite stable and many well-known companies in the retail, banking, ecommerce and government sectors have successfully used ACTIVEMQ to develop applications. However, ACTIVEMQ can only implement one-to-one communication between the server and the client. So, Apache is launching a new project called Apache Apollo, a sub-project of ACTIVEMQ that better enables communication between the server and multiple clients.

3. SYSTEM DESIGN

3.1 Message Push Plan Design

Publish/Subscribe (the Publish/Subscribe) is a kind of forward model, message between multiple publishers and multiple subscribers through the establishment of subject in the message broker mechanism as an intermediary to communicate with one another. Publishers will need interaction messages sent to the middle of the agent, and don't need to know what subscribers exists, then the message broker sends a message to the corresponding subscriber [6]. Subscribers can express their interest in one or more categories news, receive only interested in news, also don't need to know is which publishers of the message. The decoupling relationship between publishers and subscribers to make the system has better expansibility.

In the publish/subscribe system, subscriber usually receive only a subset of the announcement, choose to receive messages a subset of the interactive process is called filtering. According to different forms of filtering, the divided into publish-subscribe system based on themes and publish-subscribe system based on content. System, based on the theme of the publisher in a particular subject name identifies the message and the message is posted to the message broker [7]. Subscribers can send subscription request to the message broker, subscription conditions are identified by the subject name. Subscribers will receive the distributor all of the messages sent by the theme. Message broker message queue management, implement the function of store and forward. As shown in

Figure 1.

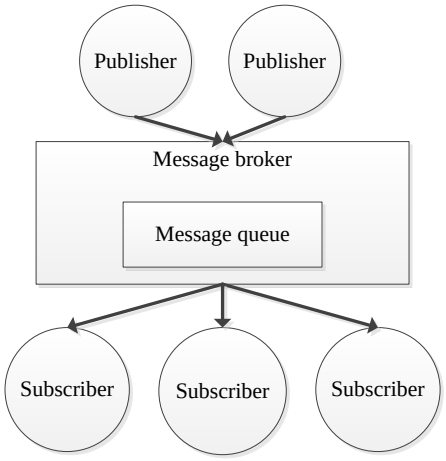


Figure 1: Publish/Subscribe Pattern

Message broker is system server, publishers put messages to the message broker message queue, the message broker sends a message to the subscriber in turn. Topic-based publish subscribe is one of the widely used model, still have a lot of push system in use. As already discussed, as is shown in the MQTT protocol is used based on the theme of the publish-subscribe model.

3.1 Server Design

Described by the second chapter, the Apache Apollo server has certain advantages, it can better realize the server and client communication. So this system USES the Apache Apollo as being pushed to the server. Server functions include listening all client connections (the publisher and subscribers), to monitor each client message into and output, and listening to the news topic of detailed information, etc. The server through the message topic relate the publisher and subscribers, to save the theme of the publishers publish messages to the message queue, and, in turn, sends a message to subscribe to the topic subscribers. The server function includes three parts: client management, message management and subject management [8].

First, The server must listen to all clients' (publisher and subscriber) connection conditions, listen to each client's message input and output, and display the details of the client in the form of a list. add client to the client list when it is online, delete client from the client list when it is offline, the number of messages sent and received by the client will be displayed in the details of the client. The server connects the publisher and subscriber through the message theme, save the publisher's topic message to the message queue, and sent message to the subscribers who subscribed the theme.

The message management module is responsible for the matching of messages and the message's store-and-forward function, the server needs to listen to all the subject information in the system and displayed in the form of a list, when publishers publish multiple content under the same theme, The server needs to update the amount of data sent under the theme, The information in the theme list also includes the number of publishers and subscribers under the theme, real-time monitor this subject detailed information.

3.2 Client Design

The client includes the publisher and the subscriber, the publisher is used to publish the message, the subscriber is used to receive the message of interest, and subscribers subscribe to the message of interest through the topic subscription.

The system uses J2SE technology to achieve the publisher function, using Android technology to achieve subscriber function, because Eclipse PAHO to achieve the MQTT protocol support, and MQTT official also recommended Eclipse PAHO, based on Eclipse PAHO API programming to achieve the publisher function and subscription Function. For the sake of simplicity, the system publisher and subscriber use the unified topic "test / topic", the publisher publishes the message to the server under the topic "test / topic", and the subscribers passively receive the message that the server sends the theme [9].

Because the publisher client is developed on the windows platform and the subscribers' client is developed on the Android platform, the default encoding of the two platforms is not the same. When the publisher publishes Chinese or other special language message content, the subscribers are at When receiving the message appeared on different platforms, text compatibility issues. UTF-8 (8-bit Unicode Transformation Format) is a variable-length character encoding for Unicode, also known as universal code. UTF-8 uses 1 to 4 bytes to encode Unicode characters.

4. SYSTEM IMPLEMENTATION

The application uses the MQTT protocol, the server application uses Apollo Apache to build the message push system server, the client application uses Eclipse PAHO API. Specific implementation steps are as follows:

Downloads and decompresses the Apollo serve, run apache-apollo-1.7.1\bin\apollo.cmd, enter create mybroker to create server instance. It will Generate mybroker folder under the bin directory after the create mybroker, the folder contains a lot of information, etc\apollo.xml file is used to configure the server information, and etc\users.properties contains username and password which are used to connect MQTT server, Finally start the command, enter apache-apollo-1.7.1\bin\mybroker\bin\apollo-broker.cmd run to start Apollo server, enter http://127.0.0.1:61680/ in any browser to check whether the installation is successful, the browser shows the theme, the number of client connection and a lot of information. After the above steps, the server has been basically completed, the server is shown in Figure 2.

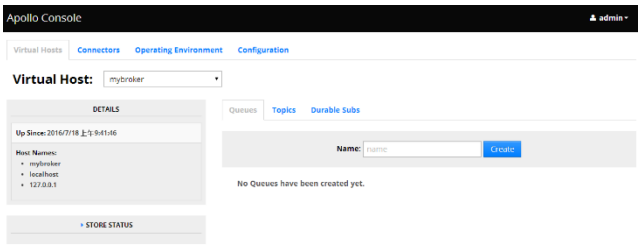


Figure 2: Apollo Server Display

The client application is divided into the Android client and J2SE client, Android client is used to receive the subscribe message, J2SE client is used to publish news, write the client code based on Eclipse Paho API.

The J2SE client will be shown in Figure 3 after running it, fill in the content of the text box and publish topics. the Android client will be shown in Figure 4 after running it, Click publish topic button in Figure 3 and the topic content comes from figure 3 will be shown in Figure 4, publish message many times, it will also receive message many times, many times' test show that the application can better satisfy the basic function of message push.

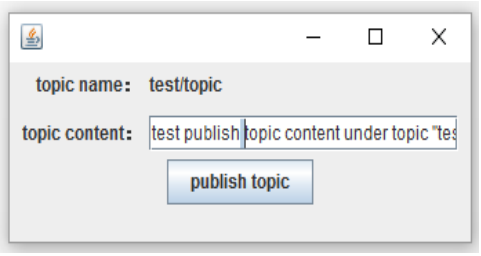


Figure 3: J2SE Client Message Publish Display

Figure 4: Android Client Message Receive Display

5. CONCLUSION

This paper studies the structure, message format, and characteristics of the MQTT protocol, And use this protocol to implement a message push system on the android platform, using unified character encoding method solve the text compatibility problem of different platforms, And test the application's using effect, the result satisfies the message push demand. Select the efficient and low resource occupancy rate message transmission protocol MQTT, and using publish / subscribe mechanism, it can automatically send the message to the corresponding mobile terminal in time and effectively. Not only satisfy the mobile terminal limited resources characteristics, but also avoid the cause effect of pressure on the performance of the server, greatly save manpower deployment cost and network resources, it is a light weight, simple and efficient message push system. MQTT as a light-weight message publish/subscribe protocol, its development prospect is very considerable on mobile phone, the application about Android push also has the very big development space, especially, characteristics of low bandwidth of this protocol are very popular in the place of limited bandwidth. But the security of the application is not considered in this paper, the next step of this paper will focus on the study of security aspects .

ACKNOWLEDGMENTS

As the research of the thesis is sponsored by National Natural Science Foundation of China (No: 61662017)，major scientific research project of Guangxi higher education (No: 201201ZD012), Guilin Science and Technology Project Fund(No： 2016010408)and Guangxi Graduate Innovation Project (No:SS201607),we would like to extend our sincere gratitude to them.

REFERENCES

[1] Collina, M., Corazza, G.E., Vanelli-Coralli, A. 2012. Introducing the QEST broker: Scaling the IoT by bridging MQTT and REST[C]// IEEE International Symposium on Personal Indoor & Mobile Radio Communications. IEEE, 36-41.

[2] Ren, H., Ma, Y., Yang, H.B. 2014. Message push server based on MQTT protocol [J]. Application of computer system, 23 (03), 77-82(In Chinese).

[3] Warren, I., Meads, A., Srirama, S., Weerasinghe, T., Paniagua, C. 2014. Push Notification Mechanisms for Pervasive Smartphone Applications[J]. Published in Pervasive Computing, IEEE, 13 (2), 61-71.

[4] Ju, H., Kim, H., Lee, S., Hong, D.K. 2013. Correlation analysis of MQTT loss and delay according to QoS level[C]// International Conference on Information NETWORKING. IEEE Computer Society, 714-717.

[5] Jiang, N., Zhang, Y., Zhao, Z.J. 2015. The push system based on message queue telemetry transmission [J]. Computer engineering, 41 (9), 1-6(In Chinese).

[6] Guan, Q.Y., Li, H.B., Yu, B. 2014. Research and application of MQTT protocol on Android platform [J]. Application of computer system, 23 (04), 197-200(In Chinese).

[7] Chen, T., Li, J. 2016. Push technology research based on the MQTT protocol [J]. Software Guide, 15 (3), 18-21(In Chinese).

[8] Chen, Y., Zhang, M.P., Xu, L. 2015. WSN application layer protocol MQTT-SN and the analysis and improvement of CoAP [J]. Application of computer system, 24 (02), 229-234(In Chinese).

[9] Liu, Y.L., Liu, W., Guo, K.H. 2013. A mobile terminal oriented adaptive news push strategy [J]. computer engineering and science, 35 (12), 114-119(In Chinese).