Lightweight messaging protocol for precision agriculture

1st Eng-Kee Tan
National Advanced IPv6 Centre
Universiti Sains Malaysia
Penang, Malaysia
engkee@student.usm.my

2nd Yung-Wey Chong
National Advanced IPv6 Centre
Universiti Sains Malaysia
Penang, Malaysia
chong@usm.my

3rd Raden Arief Setyawan
Faculty of Computer Science
Brawijaya University
Malang, Indonesia
rarief@ub.ac.id

4th Muhammad Niswar
Department of Informatics
Hasanuddin University
Makassar, Indonesia
niswar@unhas.ac.id

5th Khin Than Mya
Faculty of Computer Systems and Technologies
University of Computer Studies
Yangon, Myanmar
khinthanmya@ucsy.edu.mm

2021 International Conference on Information Networking (ICOIN) | 978-1-7281-9101-0/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICOIN50884.2021.9333986

***Abstract*—The deployment of precision agriculture is inevitable due to the scarcity of arable land. Many countries such as Japan, Singapore and Taiwan have deploy sensor networks and actuators to monitor growth of the crops and the surroundings to activate smart fertigation and irrigation system. Nevertheless, the choice of suitable messaging protocol is challenging due to the nature of precision agriculture system. For an efficient and effective solution, the precision agriculture system cannot rely on a single protocol because it need to work across several devices such as sensors, cameras, drones, robotics etc. In this paper, a lightweight messaging middleware (LMM) for precision agriculture is proposed. The purpose of the middleware is to facilitate the different devices that is used in rich sensing precision agriculture application. The modular approach of LMM can help to support the deployment of precision agriculture.**

***Index Terms*—IoT Systems, Messaging Protocol, Quality of Service, Interoperability**

I. Introduction

The amount of electronic devices connected to the Internet worldwide has been growing tremendously in recent years. Gatner [1] predicted that by 2020, there will be approximately 5.8 billion of IoT devices, with 21.5% increase from 2019. The proliferation of IoT has seen an increased in precision agriculture application. One of the main characteristics of precision agriculture is machine-to-machine (M2M) communications between multiple devices such as sensors, robots and drones and IoT gateway. These devices are usually installed in isolated area with limited Internet connectivity. In such environment, packet loss rate will be high, making it a challenge for massive deployment. One of the aspects that deciding the M2M communication performance is the application layer protocol, that required to be customized for M2M communications within constrained IoT related applications. It is always a daunting and challenging task for any organization to decide on a standard and effective messaging protocol. Unlike the Web which can works by using a single standard HTTP protocol, each IoT applications have distinct requirements. No protocol that can satisfy all of these requirements [2]. As a consequences, various messaging protocols are introduced for developers to select for satisfying different types of requirements in IoT system.

AMQP [3] was introduced as a corporate messaging protocol that designed for security, reliability, interoperability and provisioning [4]. AMQP provides flow control with quality of services (QoS). In AMQP, either the publisher or subscriber creates an ”exchange” with a given name. Subsequently, the name is being broadcast. Publishers and subscribers are then use the given name to find each other. Next, a consumer is then create a queue and subsequently attaches the queue to the ”exchange” simultaneously. Messages collected by the exchange is bind to the queue. AMQP is a connection-oriented protocol which runs on TCP as its transport protocol and TLS/SSL and SASL for security [5]. AMQP is lightweight with 8-bytes fixed header with no limit for the size of the message [6].

CoAP [7] was introduced by IETE CoRE (Constrained RESTful Environments) Working Group based on Representational State Transfer (REST) architecture to provide low power, low overhead interface in constrained network. CoAP supports request-response and publish-subscribe model using Universal Resource Identifier (URI) to send data from publisher to subscriber. When there is a publisher publishes a new data to the URI, all the subscribers will be notified about the new value as indicated by the URI. CoAP is a connectionless protocol that uses UDP as its transport protocol and DTLS for security [8]. CoAP supports content negotiation to express a preferred representation of a resource. This will isolate client with the server, so that they can evolve independently without affecting one another.

This publication is the output of the ASEAN IVO project FARMTAB: Precision Agriculture System using Internet of Things and Artificial Intelligence for Urban Farming and financially supported by NICT.

MQTT [9] is a publish/subscribe based M2M messaging protocol that is used on top of the TCP/IP protocol stack. It was designed as a lightweight broker-based protocol with small packet size, low bandwidth and high latency connection. In MQTT, a message is published by a publisher into a channel, which is known as a 'topic'. Whoever interested to a particular topic can subscribe to it. All subscribers subscribed to a topic will receive every messages that have been published to that topic. A centralised broker server handles the communication between publishers and subscribers and distributes messages between them. The centralised broker collects all the topics and handles all the message forwarding between publishers and subscribers. However, such centralised protocol suffers from poor scalability problem [10]. In addition, they hardly have any locality-awareness because the centralised broker needs to manage all the subscribers, and the location of this centralised broker cannot be changed [11]. In order to enhance the scalability of publish/subscribe messaging, there are various approaches with no centralised broker have been proposed.

Several middlewares have been proposed to address the interoperability of IoT protocols. The common design patterns used in these protocols are service oriented, publish/subscribe oriented, software defined network-based and virtual machine based design. Cheng [12] proposed a lightweight IoT service mashup middleware based on REST-style architecture for IoT applications. The design comprises six components that connected in a pipeline to provision the IoT services. These components are namely physical sensor layer, uniform resource access framework layer, uniform message space layer, service mashup layer, open service interface layer and applications layer. The proposed middleware is able to dynamically manage the physical sensory devices and its corresponding protocols. In addition, a distributed publish/subscribe-based messages distribution service was shown to provide better response time with an increase of concurrent instances. Although it provides a way to create applications for IoT, it does not support data analytics [13], an important features required for precision agriculture. In addition, it does not address the communication protocols in the application layer for video streaming [14], which can be used to understand the growth rate of the crops.

II. Related Work

Application layer plays an important roles to provide services and ensures there is an effective communication between application programs within a network. Thus, the process of selecting an appropriate application layer protocol is a prerequisite to understand a specific IoT application based on its data-sharing demands. Traditionally, most of the web applications depends on HTTP. However, IoT applications cannot depend on a single protocol to meet all its use case's requisites. In the past, many emerging protocols have been made available to developers which varies due to different needs of the IoT-based applications. Middlewares that allow integration of loosely coupled distribution systems with advanced message oriented approach have been proposed to provide versatility and modularity to IoT applications. Impala [15] implemented non virtual machine based middleware that allows functionality of application to be adaptable easily. It handles requests and responses asynchronously. Milan [16] filled in the gap in providing management tools required by different network applications. The middleware allows applications to specify the policy to manage and adjust the network characteristics. PRISMA [17] is proposed to address the needs of programminng abstraction through the use of RESTful APIs and QoS mechanisms to meet applications constraints. By providing an open RESTful APT for data access, PRISMA supports interoperability of the heterogeneous devices based on network technologies. In order to cope with the dynamic execution of IoT domain, PRISMA uses publish/subscribe paradigm to implement the asynchronous communication between devices.

Although many protocols and middleware have been proposed in the past to ensure effective communication in IoT environment, most of the proposed work are not specific to precision agriculture. As such Farmbeats [18], an IoT platform for data-driven precision agriculture was developed. The platform was developed to collect data from various sensors and ensure that the system is available even during power outages. It was then transition to cloud platform built on top of Azure. Although the platforms offer web-based environments for precision agriculture, users need to subscribe to Azure cloud. In order to satisfy the protocol requirements, system designer should be taking accounts of the devices (which can be ranged from the resource-constrained IoT edge-nodes to the resourceful clouds), the data (which can be produced either at the IoT edge, fog, or cloud layers), and the communication architecture of the system.

III. Lightweight messaging middleware (LMM)

In precision agriculture, it is important to couple IoT sensors for nutrient monitoring with image processing for disease and growth analysis. Thus, the system should be have the capability in supporting devices such as robots, sensors and cameras as shown in Fig. 1. Nevertheless, most of the proposed protocols are meant for IoT sensors only. The proposed lightweight messaging middleware (LMM) is designed so that the end nodes can be used to capture both images and sensor date in IoT gateway and send them to server for post-processing. The main features of LMM are:

- Interconnections between connection-oriented and connectionless services and data streams.
- Provides authentication of data source and confidentiality of data.
- Small data overhead and ability to support multiple application layer protocols
- Able to deployed in large precision agriculture system.

To integrate these features, LMM interconnect two (2) application layer protocols namely CoAP is used to stream video images to analyse growth rate and MQTT, which is used for sensor data as shown in Fig. 2. The reason CoAP is used to stream video images is because it is lightweight protocol stack.

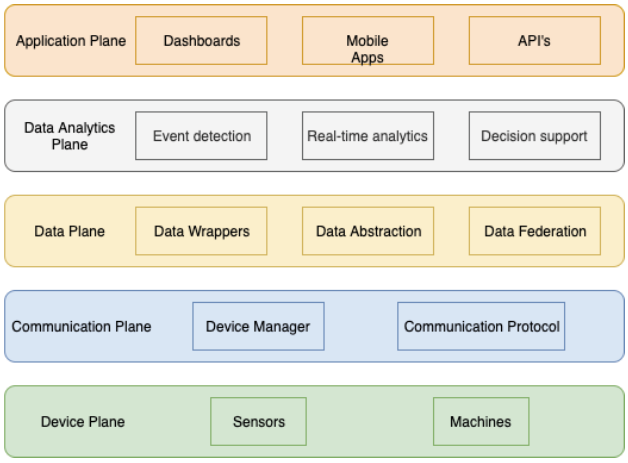

Fig. 1. Overview of precision agriculture system

Besides, CoAP also conserves energy and save bandwidth. CoAP that relies on UDP as its transport protocol, which neither provides guarantee for message delivery nor congestion control mechanism is suitable for precision agriculture. Unlike IoT sensors that capture small size data, IoT camera produces a huge capacity of data packets before it can be process the image for disease detection or growth rate analysis. The streaming of video/image feed is time consuming as the transmission of many data packets, which is also consuming high bandwidth. Although the video/image compression techniques implemented has significantly reduced the required bandwidth, the MTU size and data rate is still high as compared to IoT sensor. As such IoT camera will require UDP whereas sensor will use publish/subscribe architecture with minimal bandwidth requirements, message data overhead and power consumption. MQTT is used as application layer protocol for IoT sensor because it has lower delay than CoAP messages at lower packet loss rate and the message size is very small. LMM will published messages to the broker via an address known as 'topic'. The client that resides in server will then receive the published messages by subscribing to that topic.

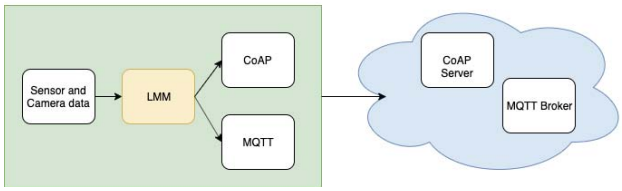

Fig. 2. Lightweight Messaging Middleware (LMM)

LMM contains common API calls to support publish-subscribe model for MQTT and CoAP namely *Connect()*, *ConnectAck()*, *publishMessage()* and *PublishAck*. *Connect()* and *ConnectAck()* is used by LMM to request connection between IoT gateway and cloud server or broker. The connection will determine whether the IoT gateway has adequate access rights to publish messages over that specific topic to the server/broker. *Connect()* API call contains *destination* and *secretkey* as arguments. No additional information is added to simplify the connection authentication process so that the traffic overhead is small. The *destination* varies, depending on whether it is CoAP or MQTT message. Through *Connect()* and *ConnectAck()* API call, the precision agriculture system can ensure authentication of data source and confidentiality of data.

publishMessage() API call contains arguments such as *topic*, *destination* and *message*. When LMM received sensor data from precision agriculture system, it will published the topic through the broker, which is the destination IP address. Similarly, LMM will published the topic through URI when LMM received video/image data from the system. *PublishAck()* contains acknowledgement message sent from server to client as shown in Fig. 3.

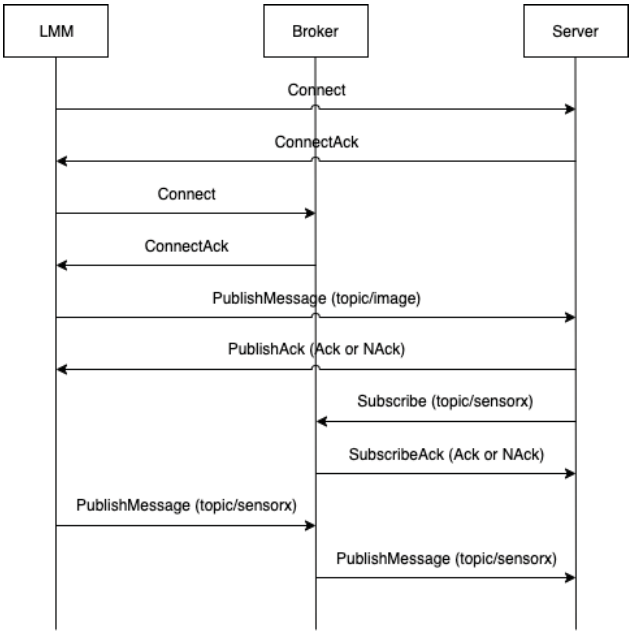

Fig. 3. LMM sequence diagram

There are three (3) parts in LMM, publish/subscribe interface, middleware database and application functions. The publish/subscribe interface handles the incoming data, the topic it subscribed, published, cancelled or registered. Information such as message sets, that contains latest message related to the topics and topic subscription table are stored in middleware database. Each item in topic subscription table contains *topic*, *destination* and *message*.

The data collected by middleware database is then processed by application functions to match the traffic type and route it accordingly. LMM will perform the route messages from sensor/camera to relevant API calls. In order to maintain the data distribution consistency, the application function will synchronize the information based on the message routing policy. Fig. 4 illustrates the LMM architecture.

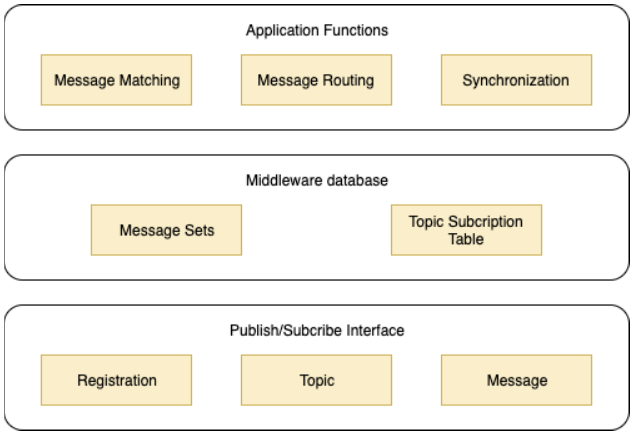

Fig. 4. LMM architecture

IV. Use Case: Precision Agriculture System

LMM is built on Raspberry Pi 4 using node.js framework that are event-driven, allowing it to support high concurrency. The proof of concept (POC) is conducted on precision agriculture system that collects pH, temperature, electrical conductivity (EC), total dissolved solution (TDS) and camera images. Sensor/camera nodes will send/publish sensor data to LMM and LMM will aggregate these information before sending it to cloud services. Fig. 5 illustrates the proof of concept (POC) of the middleware in precision agriculture system. The web server subscribes to the camera feed using CoAP and sensor data using MQTT. Fig 6 shows the print out logs of the script running in Raspberry Pi.

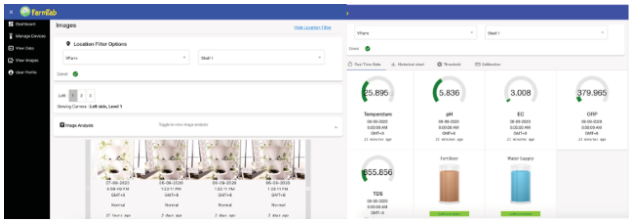

Fig. 5. Use case of LMM in precision agriculture system

```
1|lmm_script | INFO - 2020-09-20 12:00:02+0800
1|lmm_script | SENSOR DATA :
1|lmm_script | {"data_datetime": "2020-09-20 12:00:04+0800", "temp": 27.2819444444444445, "ph":
3.3025, "ec": 2.1077777777777786, "orp": 508.3825, "tds": 1037.1302777777776, "fertilizer":
1.0, "water": 1.0}
1|lmm_script | [ MQTT ] Message Published... mid: 48
1|lmm_script | Updated CAM lvl3] - [ Slot_C] ->lvl1:true, lvl2:true, lvl3:true
1|lmm_script | CAM_STATUS:  Updated all camera stream
1|lmm_script | 2020-09-20 12:00:07+0800
1|lmm_script | [ CoAP ] - Sent camera feed for processing
1|lmm_script | ARDUINO ==>
b'PH@3.31#TEMP@27.50#TDS@1037.43#EC@2.12#ORP@511.72#WLVL1@1#WLVL2@1#WLVL3@0\r\n'
1|lmm_script | {'data_datetime': '2020-09-20 12:00:20+0800', 'serial_number':
'dc:a6:32:46:04:be', 'waterPumpStat': False, 'ferPumpStat': False, 'pumpStat': False, 'temp':
27.5, 'ph': 3.31, 'ec': 2.12, 'orp': 511.72, 'tds': 1037.43, 'fertilizer': 1.0, 'water': 1.0}
1|lmm_script | Store [ sensor_data ] col : 1
1|lmm_script | SUCCESS store [ sensor_data ] col : [ObjectId('5f66d6e0c8b9007b674b8b11')]
1|lmm_script | ARDUINO ==>
b'PH@3.31#TEMP@27.50#TDS@1037.43#EC@2.09#ORP@514.65#WLVL1@1#WLVL2@1#WLVL3@0\r\n'
1|lmm_script | {'data_datetime': '2020-09-20 12:10:23+0800', 'serial_number':
'dc:a6:32:46:04:be', 'waterPumpStat': False, 'ferPumpStat': False, 'pumpStat': False, 'temp':
27.5, 'ph': 3.31, 'ec': 2.09, 'orp': 514.65, 'tds': 1037.43, 'fertilizer': 1.0, 'water': 1.0}
```

Fig. 6. LMM PM2 logs

V. Conclusion

In this paper, a middleware for precision agriculture system is proposed to support multi-protocol implementation needed in the system. The integration of MQTT and CoAP as application layer protocol in the middleware support detailed analysis of impact of environment versus growth of the crops. As future work, the performance of LMM will be evaluated.

Acknowledgment

This publication is the output of ASEAN IVO (http://www.nict.go.jp/en/asean_ivo/index.html) project, FARMTAB: Precision Agriculture System using Internet of Things and Artificial Intelligence for Urban Farming and financially support by NICT (http://www.nict.go.jp/en/index.html).

References

[1] Gartner. (2019) Gartner says 5.8 billion enterprise and automotive IoT Endpoints will be in use in 2020. [Online]. Available: https://www.gartner.com/en/newsroom/press-releases/2019-08-29-gartner-says-5-8-billion-enterprise-and-automotive-io

[2] N. Naik and P. Jenkins, "Web protocols and challenges of Web latency in the Web of Things," in *2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN)*, July 2016.

[3] OASIS. (2012) OASIS Advanced Message Queuing Protocol (AMQP) Version 1.0. [Online]. Available: http://docs.oasis-open.org/amqp/core/v1.0/os/amqp-core-complete-v1.0-os.pdf

[4] A. Foster, *Messaging technologies for the industrial internet and the internet of things whitepaper*. PrismTech, 2015.

[5] N. Han, *Semantic service provisioning for 6LoWPAN: powering internet of things applications on Web*. Institut National des Tel'ecommunications, 2015.

[6] J. Luzuriaga *et al.*, "A comparative evaluation of AMQP and MQTT protocols over unstable and mobile networks," in *2015 12th Annual IEEE Consumer Communications and Networking Conference (CCNC)*, Jan 2015.

[7] C. B. Z. Shelby, K. Hartke. (2014) The Constrained Application Protocol (CoAP). [Online]. Available: https://tools.ietf.org/html/rfc7252

[8] D. Thangavel *et al.*, "Performance evaluation of MQTT and CoAP via a common middleware," in *2014 IEEE Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing ISSNIP*, April 2014.

[9] OASIS. (2019) MQTT Version 5.0. [Online]. Available: https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html

[10] K. Paridel, E. Bainomugisha, Y. Vanrompay, Y. Berbers, and W. Meuter, "Middleware for the Internet of Things, Design Goals and Challenges," *Electronic Communications of the EASST*, vol. 28, pp. 1–6, 2010.

[11] V.-N. Pham, V. Nguyen, T. Nguyen, and E.-N. Huh, "Efficient Edge-Cloud Publish/Subscribe Broker Overlay Networks to Support Latency-Sensitive Wide-Scale IoT Applications," *Symmetry*, vol. 12(1), pp. 1–18, 2019.

[12] B. Cheng, S. Zhao, J. Qian, Z. Zhai, and J. Chen, "Lightweight Service Mashup Middleware with REST Style Architecture for IoT Applications," *IEEE Transactions on Network and Service Management*, vol. 15 (3), pp. 1063–1075, 2018.

[13] F. Moreno, *Modularizing Flink programs to enable stream analytics in IoT Mashup tools*. Technical University of Munich, 2018.

[14] T. Sultana and K. A. Wahid, "Choice of Application Layer Protocols for Next Generation Video Surveillance Using Internet of Video Things"," *IEEE Access*, vol. 7, pp. 41 607–41 624, 2019.

[15] T. Liu and M. Martonosi, "Impala: a middleware system for managing autonomic, parallel sensor systems," in *Proceedings of the ninth ACM SIGPLAN symposium on Principles and practice of parallel programming*, June 2003.

[16] W. Heinzelman, A. Murphy, H. Carvalho, and M. Perillo, "Middleware to support sensor network applications," *IEEE Network*, vol. 18 (1), June 2004.

[17] J. Silva *et al.*, "PRISMA: A Publish-Subscribe and Resource-Oriented Middleware for Wireless Sensor Networks," in *AICT2014: The Tenth Advanced International Conference on Telecommunications*, June 2004.
[18] D. Vasisht *et al.*, "FarmBeats: An IoT Platform for Data-Driven Agriculture," in *Proceedings of the 14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 2017)*, March 2017.