Dynamic Real-Time MQTT Message Handling for Digital Twin Systems

Mohamed A. Dini, Dong-Seong Kim, Jae Min Lee
Department of IT Convergence Engineering, Kumoh National Institute of Technology, Gumi, South Korea
(mohamedabubakar407)@gmail.com (dskim, ljmpaul)@kumoh.ac.kr

***Abstract*—This paper presents a dynamic real-time MQTT message handling system for digital twin applications in Unreal Engine 5. The system connects to a local MQTT broker, receives JSON-formatted sensor data, and dynamically parses key-value pairs using a custom algorithm. Results show reliable real-time data synchronization within a local host environment. Future work includes integrating cloud-based MQTT brokers and developing a user-friendly interface for broader accessibility.**

***Index Terms*—Digital Twin, IoT, JSON, MQTT, Real-Time, Unreal Engine.**

I. Introduction

Message Queuing Telemetry Transport (MQTT) has also gained popularity in the field of the Internet of Things (IoT) as a messaging protocol due to its high reliability, minimal bandwidth consumption, and support for dynamic, asynchronous communication [1]. However, integrating dynamic MQTT message that handles capacities into game engine-based digital twin environments—specifically Unreal Engine 5 (UE5)—remains an underexplored area [2], [3].

Because of its real-time simulation capabilities, high-fidelity rendering, and modular architecture, Unreal Engine 5 has shown quite potential for immersive digital twin applications [4]. Nevertheless, the engine's basic architecture has native support for dynamic, real-time data transmission frameworks like MQTT, which is an issue to tackle and needs to be explored [5].

To tackle the discussed issue, we are proposing a Dynamic and Real-Time MQTT Message Handling System for the specific use of digital twin applications within UE5. The system ensures scalable, flexible, and low-latency communication between physical devices and their virtual counterparts using a custom MQTT parser and message routing framework [6]. The goal is to allow developers to dynamically handle JSON-based MQTT message types and payload structures at runtime, resulting in great and easy control and real-time data-driven decision-making in interactive 3D environments [7], [8].

This paper proposes a Dynamic Real-Time MQTT Message Handling System specifically designed for digital twin applications within Unreal Engine 5. The system enables flexible, scalable, and low-latency communication between physical devices and their virtual counterparts through a custom MQTT parser and message routing framework. The proposed system allows digital twin developers to dynamically handle a variety of MQTT message types and payload structures at runtime, promoting interoperability and real-time data-driven decision-making in interactive 3D environments.

II. Proposed System

The proposed system introduces a dynamic, real-time MQTT message handling framework integrated within Unreal Engine 5, enabling seamless data exchange between physical devices and their virtual representations in a digital twin environment. The system consists of three primary stages: MQTT message reception, dynamic message parsing, and key-value storage.

To enable MQTT communication within Unreal Engine 5, an experimental MQTT plugin is employed. The engine subscribes to a local broker configured using *Mosquitto*, and for testing purposes, messages are published via a command-line interface (CMD). When a message is received, it is stored as a raw string in Unreal Engine.

MQTT messages typically follow a JSON format containing key-value pairs. To simplify parsing, the message is first cleaned by removing unnecessary characters such as curly braces $\{\}$ and quotation marks "”." The cleaned message is represented as:

$$M = \{m_1, m_2, \ldots, m_n\}, \tag{1}$$

where each m_i is a key-value pair in the form:

$$m_i = (k_i : v_i). \tag{2}$$

The system defines:

- The set of keys:

$$K = \{k_1, k_2, \ldots, k_n\}, \tag{3}$$

- The set of values:

$$V = \{v_1, v_2, \ldots, v_n\}. \tag{4}$$

The cleaned message is split into individual key-value pairs using a comma (‘,‘) as a delimiter. Each pair is further divided by a colon (‘:‘) into a key k_i and its corresponding value v_i. These are sequentially stored in a **Map data structure**, defined as:

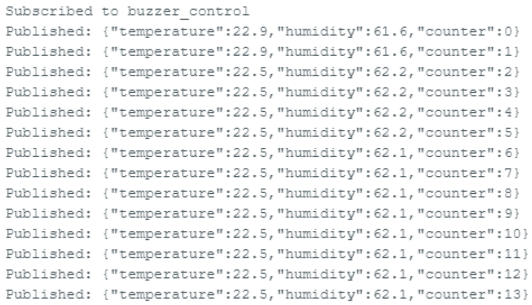
$$\text{Map} : K \rightarrow V. \tag{5}$$

This map allows for real-time data lookup, modification, and dynamic updates to the virtual environment without requiring predefined schema, offering high flexibility for diverse digital twin applications.

III. Result Discussion

In the initial implementation phase, the proposed system was tested within a controlled local network environment. The MQTT plugin in Unreal Engine 5 successfully established a connection with a Mosquitto broker, subscribed to a specific topic, and received real-time messages published from a command-line interface.

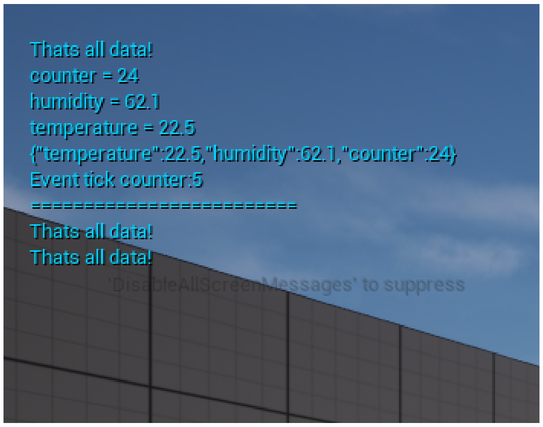
Figure 1 illustrates the serial monitor output from an Arduino microcontroller equipped with a DHT11 sensor, periodically sending temperature and humidity readings formatted in a JSON-like structure.



```
Subscribed to buzzer_control
Published: {"temperature":22.9,"humidity":61.6,"counter":0}
Published: {"temperature":22.9,"humidity":61.6,"counter":1}
Published: {"temperature":22.5,"humidity":62.2,"counter":2}
Published: {"temperature":22.5,"humidity":62.2,"counter":3}
Published: {"temperature":22.5,"humidity":62.2,"counter":4}
Published: {"temperature":22.5,"humidity":62.2,"counter":5}
Published: {"temperature":22.5,"humidity":62.1,"counter":6}
Published: {"temperature":22.5,"humidity":62.1,"counter":7}
Published: {"temperature":22.5,"humidity":62.1,"counter":8}
Published: {"temperature":22.5,"humidity":62.1,"counter":9}
Published: {"temperature":22.5,"humidity":62.1,"counter":10}
Published: {"temperature":22.5,"humidity":62.1,"counter":11}
Published: {"temperature":22.5,"humidity":62.1,"counter":12}
Published: {"temperature":22.5,"humidity":62.1,"counter":13}
```

Fig. 1. Arduino serial monitor displaying temperature and humidity readings.

Upon receiving the MQTT message, Unreal Engine processed the string using the dynamic parsing algorithm. The system successfully extracted key-value pairs from the message, which were stored within a Map data structure for real-time access. Figure 2 demonstrates the debug output in Unreal Engine 5, showing the separated variables and their values.



```
Thats all data!
counter = 24
humidity = 62.1
temperature = 22.5
{"temperature":22.5,"humidity":62.1,"counter":24}
Event tick counter:5
========================
Thats all data!
Thats all data!
'DisableAllScreenMessages' to suppress
```

Fig. 2. Unreal Engine 5 Print String debug output displaying parsed message values.

The parsing algorithm had good accuracy. It worked with varied message formats. Real-time digital twin synchronization happened with little delay. Limitations are there. The system does not handle errors well. The system uses string-based parsing.

IV. Conclusion

For digital twin applications, a system did MQTT message handling. The system is in Unreal Engine 5. It works in real-time. The system connected to a local broker. The system received sensor data. It parsed messages into key-value pairs. The key-value pairs became available for use in the virtual environment. Tests showed dependable reception. Parsing worked well. Data became accessible in real-time with minimal delay. At present the system only functions on a local network. It uses command-line publishing. It gives a good base for later improvements. Improvements include cloud-based MQTT support. There will be an in-engine graphical interface. This will make configuration and data visualization simpler.

V. Acknowledgement

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(2018R1A6A1A03024003, 33%). Additionally, this work was supported by Innovative Human Resource Development for Local Intellectualization program through the Institute of Information & Communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (IITP-2025-RS-2020-II201612, 33%). Lastly, was supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2025-RS-2024-00438430, 34%) supervised by the IITP(Institute for Information & Communications Technology Planning & Evaluation).

References

[1] L. A. C. Ahakonye, C. I. Nwakanma, J. M. Lee, and D.-S. Kim, "Time-efficient deep learning-based energy consumption prediction for smart factory," in *2022 13th International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, 2022, pp. 879–882.

[2] J. N. Njoku, E. C. Nkoro, R. M. Medina, C. I. Nwakanma, J.-M. Lee, and D.-S. Kim, "Leveraging digital twin technology for battery management: A case study review," *IEEE Access*, 2025.

[3] Y. Cho and S. D. Noh, "Design and implementation of digital twin factory synchronized in real-time using mqtt," *Machines*, vol. 12, no. 11, p. 759, 2024.

[4] N. Pereira, A. Rowe, M. W. Farb, I. Liang, E. Lu, and E. Riebling, "Arena: The augmented reality edge networking architecture," in *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 2021, pp. 479–488.

[5] P. Zema, C. Suraci, A. Molinaro, and G. Araniti, "A comparative analysis of mqtt and coap to test transmission performance in medical digital twin applications," in *2024 IEEE 10th World Forum on Internet of Things (WF-IoT)*. IEEE, 2024, pp. 589–594.

[6] M. A. Dini, M. J. A. Shanto, G.-H. Kwon, D.-S. Kim, and T. Jun, "Emotion type recognition scheme using eeg based signals," , pp. 489–490, 2023.

[7] A. M. Tayeb, M. A. Khatun, M. Golam, M. F. Rahaman, A. Aouto, O. P. Angelo, M. Lee, D.-S. Kim, J.-M. Lee, and J.-H. Kim, "Smartrsd: An intelligent multimodal approach to real-time road surface detection for safe driving," *arXiv preprint arXiv:2406.10128*, 2024.

[8] M. A. Dini, D.-S. Kim, J. M. Lee, and T. Jun, "Tf-idf empowered content-based recommendation system for labor complaints and service operations," , pp. 866–867, 2023.