IJEEC
International Journal of Electrical Engineering and Computing
Vol. 8, No. 2 (2024)

Original research paper
UDC 004.65:621.395.721.5
DOI 10.7251/IJEEC2402053B

MQTT Protocol in Smart Home Environments: Principles of Operation and Application

Tijana Begović, Vasilije Čabarkapa, Milica Ivković, Božidar Popović

Faculty of Electrical Engineering, University of East Sarajevo, East Sarajevo, Bosnia and Herzegovina

E-mail address: tijana.begovic@etf.ues.rs.ba, vascabarkapa@gmail.com, milicaivkovic034@gmail.com, bozidar.popovic@etf.ues.rs.ba

Abstract — **Internet of Things as a revolutionary concept in technology of modern days has led to the popularization of "smart" things. It introduces low-cost and reliable solutions to provide independent operations of devices and machines. Smart home systems have become one of the most important implementations as they significantly improve the quality of people's lives. Since communication plays a key role in these systems, this article provides insights into Message Queuing Telemetry Transport protocol application in smart home systems. Although it is not the only protocol used, MQTT stands out in terms of reliability, efficiency, and simplicity. The article provides major MQTT characteristics, advantages and its application in a particular smart home system for ambient parameters monitoring and control. The system integrates NodeMCU based sensor node with mobile and web applications via MQTT protocol. Results of the performance analysis of MQTT use in this system are given. Tests have shown great connection establishment speed, a high percentage of successfully sent messages, and a large number of messages sent per minute. Thanks to the flexibility of the MQTT protocol, the system is easy to upgrade and expand.**

Keywords - smart homes; MQTT protocol; Internet of Things (IoT), monitoring and control systems, communication protocols;

I. Introduction

The Internet of Things (IoT) represents a revolutionary concept in modern technology, connecting physical objects to the Internet to enable seamless communication and data exchange. By embedding sensors, software, and network connectivity, IoT transforms everyday objects ranging from household appliances to industrial machinery into "smart" devices capable of autonomous operation and interaction [1]. IoT has become a reality due to advances in low-cost and reliable sensors, diverse connectivity technologies such as Wi-Fi, Bluetooth, Zigbee, and LoRaWAN, as well as the integration of cloud computing, big data analytics, machine learning, and security mechanisms [2,3]. This paradigm shift has positioned IoT as a cornerstone of the digital age, with applications spanning smart homes, wearable technology, industrial automation, smart cities, and precision agriculture [4].

One of the most impactful implementations of IoT is in smart home systems (SHS). These systems automate and optimize household functions by integrating IoT devices with home appliances, enhancing comfort, energy efficiency, security, and overall quality of life [5]. Research shows that smart homes can significantly reduce electricity costs while contributing to environmental sustainability [5,6]. Furthermore, they play a vital role in healthcare and accessibility, offering critical support for the elderly, individuals with disabilities, and patients, thereby increasing independence and improving living conditions [7].

Smart home systems typically function in three distinct phases. First, data is collected through sensors, cameras, and microphones installed throughout the home. This data is then transmitted to a centralized processing unit for storage and analysis. Finally, actionable insights are generated, enabling the system to provide automated services and improve household functionality [8]. However, achieving high performance and reliability in such systems requires careful consideration of the underlying communication protocols.

The MQTT (Message Queuing Telemetry Transport) protocol is a preferred communication mechanism for IoT-based systems due to its efficiency, scalability, and simplicity. Designed for constrained devices and low-bandwidth networks, MQTT uses a lightweight publish-subscribe model that facilitates seamless communication between devices. It supports multiple levels of Quality of Service (QoS) to ensure reliable message delivery and offers robust security features such as TLS/SSL encryption for data protection [9]. MQTT also enhances system resilience with its ability to retain the last published message, ensuring uninterrupted functionality in case of connection loss [10].

Despite its advantages, the implementation of IoT-based smart home systems poses challenges, including data privacy concerns, the potential for cyberattacks, and the need for interoperability between devices from different manufacturers. Addressing these issues requires the integration of advanced security measures, adherence to standardized protocols, and the development of user-friendly interfaces for seamless operation.

This paper aims to propose MQTT communication protocol application in a cost-effective smart home solution. The system integrates with mobile and web applications to provide a scalable, reliable, and user-friendly platform. The ultimate goal is to demonstrate how IoT and MQTT technologies can optimize smart home functionalities, enhance energy efficiency, and address the challenges of modern living.

II. State of the Art

Smart home concept has been highly present in recent times due to integration of modern computing and Internet of Things in everyday life. This concept is based on the applications of modern technologies in traditional appliances. It ensures improvements in life quality in aspects of comfort, security, surveillance and healthcare [5]. Smart home devices are highly integrated which extend home functionalities [8]. Applications that make home smart can differ and authors in [11] divided them into four groups: energy management and efficiency; entertainment; security; and healthcare. However, no matter which smart home approach is considered, communication plays a significant role. It is very important to provide reliable and secure ways of data and commands exchange. Different wireless protocols are used such as Wi-Fi, ATM [12], ZigBee, EnOcean, Z-wave, Thread [13].

Here, a review of commonly used protocols within smart home applications will be given. The Thread protocol is adjusted for IoT devices. It provides a robust and reliable network infrastructure as it is optimized for wireless networking. It is a suitable solution for deployment in smart home systems due to its operation on IPv6 [14]. Z-Wave, ZigBee, Bluetooth and Bluetooth Low Energy (BLE) are protocols designed for communication between low-power devices over short distances. The listed protocols are primarily intended for networks of small devices [15]. Another notable protocol is CoAP (the Constrained Application Protocol) which is suitable for use in devices with limited resources [16]. Probably the most common protocol is HTTP/HTTPS, often implemented through RESTful APIs. These protocols enable a great number of smart devices to interact with each other. Also, another client-server-based protocol that is the optimal solution for real-time communication is WebSocket [17].

MQTT protocol is widely used in IoT applications. Compared to other protocols, it shows better results in aspects of energy efficiency, performance, viability and simplicity of integration [18-20]. As it is already implemented in many applications, authors in [21] referred to MQTT as the protocol of IoT. According to that MQTT integration in the smart home concept presents itself as the most suitable solution.

Research has shown its effectiveness in enabling real-time monitoring, control, and communication among devices in resource-constrained environments [22]. Systems have been developed utilizing MQTT for enhanced security through integration with cryptographic protocols, to address inherent vulnerabilities like brute-force attacks and unauthorized access [23].

Over the last few years, several MQTT-based IoT systems have been proposed. MQTT as the main communication protocol has found its role in systems for environment monitoring. Authors in [24] proposed an IoT platform for detection and classification of contaminants, and MQTT was used for data exchange between different networking levels within platform. A multi-sensor system for air and water quality was described in [25]. The data collected is sent to a cloud-based IoT platform via MQTT, which enables remote access to the provided information. Universal system for data collection and monitoring in agriculture require efficient protocol to collect data from large number of sensors to unique collection unit, and MQTT proved to be a suitable solution [26].

Other studies [27-30] have focused on leveraging MQTT for home automation, demonstrating its versatility in controlling appliances via mobile apps or voice commands [31, 32], while maintaining low bandwidth usage. NodeMCU and Raspberry Pi [33] have been popular platforms for implementing these systems, given their compatibility with MQTT and capacity to connect a range of sensors and actuators.

MQTT is widely used in many systems proposed in literature. Most of the works explain its application principle, but focus should be also given on performance analysis. In this paper, analysis will be given in the system that builds upon these advancements, integrating MQTT with NodeMCU to collect and regulate ambient parameters such as temperature, humidity, and lighting. By leveraging real-time data, it achieves precise control of environmental conditions, offering a robust, scalable, and user-friendly solution.

III. Basics of the MQTT Protocol

MQTT protocol is a communication protocol based on a “*publish/subscribe*” messaging pattern. This protocol is a simple binary protocol, which excels in data transmission compared to other protocols like HTTP. Its simplicity makes it ideal for use in constrained environments such as M2M (Machine-To-Machine) and Internet of Things applications [34]. MQTT is an application layer protocol that relies on the TCP transfer protocol [9].

Publish/subscribe messaging pattern (Fig. 1) differs from traditional client – server architecture. Pub/sub model includes two types of clients, ones that send messages (publishing) and others that receive messages (subscribing). Clients do not communicate directly, but the connection is handled by the broker. The broker’s task is to filter incoming messages and forward them to the proper clients [35].

The protocol defines three levels of Quality of Service (QoS), which defines how much effort the client/broker will put into ensuring that the message is delivered. Messages can be sent regardless of the QoS level. Table 1 shows a list of the QoS levels and their descriptions [36].

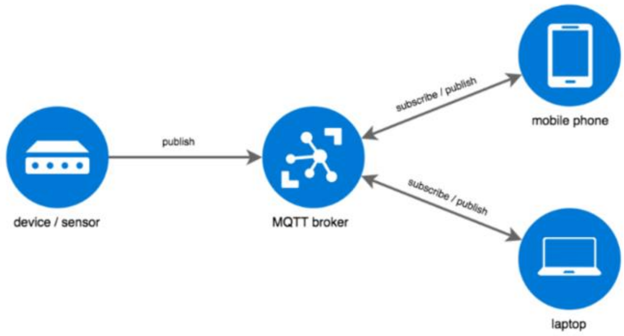

Figure 1. Publish/Subscribe pattern [35]

TABLE I. QOS LEVELS DESCRIPTION

QoS level	Description
0	Message is sent once without acknowledgment
1	Message is sent at least once, with acknowledgment required
2	Message is sent exactly once, using a four-step handshake

All devices that use MQTT protocol are clients; both publishers and subscribers depending on whether they send or receive messages [37]. Basically, any device, from small devices with limited resources to very powerful machines, can be MQTT with the proper TCP/IP connection [35]. MQTT broker is the core of the protocol. It is responsible for receiving and filtering all messages. Broker is easy to extend and customize, but also to integrate in the backend systems. It can handle millions of MQTT clients simultaneously. Every message exchanged between two or more clients must pass through the broker, so it needs to be scalable, easy to implement and monitor [9].

The MQTT broker uses the message topic to filter messages and decide which client receives which incoming message. A topic can consist of one or more levels which are separated by a forward slash ("/"). A simple example of a hierarchically organized topic is shown in Fig. 2 [37]. Each message published using the MQTT protocol is "addressed" to a specific topic. Clients subscribe to these topics and thus can receive messages. Subscriptions can be explicit, which limits the messages a client receives to a specific topic [34].

Since clients do not communicate directly, they only need to connect to the broker. Once the connection is established, the broker keeps it open as long as the client does not disconnect. The communication process involves sending and receiving messages based on subscribing, publishing and unsubscribing. A client can publish messages as soon as it connects to the broker. The 'publish' message contains, among other parts, topic name, QoS level and payload. Broker filters the received messages and forwards it to the clients that are subscribed to the proper topic [35]. The subscription process is done by sending a 'subscribe' message which contains all desired topics and QoS levels [34]. 'Unsubscribe' message is opposite to the 'subscribe' and it deletes proper subscriptions on the broker [37].

IV. ARCHITECTURE AND IMPLEMENTATION OF THE SYSTEM

A. Hardware Architecture of the Sensor Node

In order to monitor and control ambient parameters in smart home, sensor node has been developed [38]. It is based on NodeMCU development board, and includes several sensors for collecting data about temperature, air humidity and lighting indoors.

Figure 2. Hierarchically organized message topic [37]

The block scheme of the proposed node is given in Fig. 3. Sensor node is integrated with mobile and web applications using MQTT protocol.

NodeMCU (Fig. 4.a) is an ESP8266 microcontroller-based board which is used to collect data from sensors, establish communication with both applications and control proper devices to keep ambient parameters in desired limits. This board is suitable for any purpose that requires Internet connection as it has a built-in Wi-Fi module [39]. It has a significant number of digital pins and can communicate with other devices (e.g. sensors) via many different protocols (UART, SPI, I2C) [40]. The board is compatible with Arduino boards and can easily be programmed by developing firmware in Arduino or similar IDEs [41]. Today, ESP boards are widely used in IoT applications due to their simplicity and features-richens [39].

As one of the purposes of sensor node is to collect data about ambient parameters, several sensors are used. DS18B20 sensor (Fig. 4.b) is used for temperature measurement. It communicates with master device (NodeMCU in this case) via 1-Wire protocol and sends temperature reading as 9-bit or 12-bit data. With resolution of ±0.5 °C, it is highly accurate and precise temperature sensor [42]. Air humidity is measured with low-power integrated DHT11 sensor (Fig. 4.c). This sensor has a resolution of ±5% and can measure humidity from 20% to 90%. Request/response communication is done via one wire that should be pulled-up. When master device initiates communication (request), sensor sends response that carry humidity and temperature readings. DHT11 is suitable for both indoor and outdoor applications [43]. The role of the light-dependent resistor module integrated into the sensor node is to notice light presence in the room. The module is based on a voltage divider with photoresistor. Voltage change across the photoresistor is measured and converted into corresponding light intensity [44].

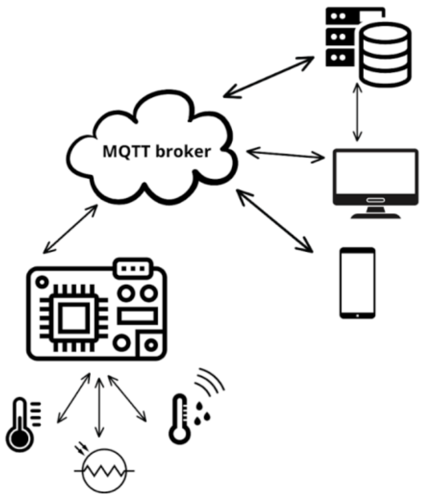

Figure 3. Block scheme of a system

Figure 4. a) NodeMCU [40] b) DS18B20 [42] c) DHT11 [43] d) 4-channel relay [45]

Besides measuring ambient parameters, the sensor node is responsible for turning on/off proper devices (heaters, dehumidifiers, light) to maintain parameters within desired limits. It is enabled by using 4-channel relay with optical isolation. Relay module is given in Fig. 4.d. The module is designed to block voltages up to 250 V and conduct currents up to 10 A. It includes four SPDT switches which are controlled by four digital pins (active low). Relay module is highly efficient due to its simplicity and ability to work with high current and voltages [45].

A system built from basic electronic components, such as the DHT11, NodeMCU ESP8266, DS18B20, a four-channel relay module, and a photoresistor module, represents a cost-effective solution in the field of IoT applications and automation. The total cost of building this system varies depending on market prices and suppliers, with an estimated range of 15–20 euros. The individual prices of key components are as follows: the DHT11 sensor for measuring air temperature and humidity costs approximately 2–3 euros, the NodeMCU ESP8266 microcontroller 5–6 euros, the DS18B20 digital temperature sensor 2–3 euros, while the four-channel relay module is priced between 2 and 4 euros. The photoresistor module can be purchased for 2–3 euros [46]. In addition to these hardware components, the system utilizes the HiveMQ MQTT broker, which is free to use [34].

B. System Communication and Management

The MQTT protocol and the HiveMQ broker are used to establish reliable and efficient communication between the hardware and software components of the system. The HiveMQ broker provides users with the ability to send, receive, and view detailed message flows, enabling full transparency, real-time monitoring, and thorough analysis of device communication within the system. This enhanced capability is crucial for analyzing, diagnosing, and optimizing data traffic, ensuring robust performance, improved scalability, security, and overall reliability in a modern IoT environment [9] [34].

The development of the system utilized the free version of the HiveMQ Cloud platform, which proved to be fully suitable for meeting the project's requirements. The free version supports up to 100 connected devices, provides a monthly data transfer limit of 10 GB, and retains data for up to three days. The maximum size of an individual message is 5 MB, which sufficiently meets the system's operational needs [34].

The functioning of the system is based on the performance of several key actions and functionalities. Initially, a few one-time tasks are executed, followed by a sequence of cyclic operations. After installation, the first step is establishing a Wi-Fi connection to enable remote control and monitoring, which are the main features of the proposed system. The microcontroller acts as a Wi-Fi station, connecting to a predefined network. Once the Wi-Fi connection is established, NodeMCU, as an MQTT client, connects to the broker and subscribes to the relevant topics, allowing communication with both mobile and web applications. The next step involves configuring the microcontroller peripherals and sensors to prepare the node for uninterrupted operation. A continuous MQTT connection is required to maintain real-time communication between the sensor node and applications. If the connection is lost, the system automatically attempts to re-establish it to avoid prolonged interruptions.

The primary role of the sensor node (Fig. 5) is to monitor and control ambient parameters. The microcontroller continuously reads sensor data on temperature, humidity, and light levels in the room while simultaneously checking for potential messages from mobile or web applications. These sensor readings are compared to user-defined upper and lower limits (set via the mobile app) to determine the appropriate actions. Relays are controlled to activate or deactivate devices such as heaters, dehumidifiers, or lights to achieve the desired ambient conditions in the room [45]. Users can directly control the lighting or enable an automatic mode, where the system independently manages the light based on the room's illumination level by adjusting the state of the relay.

Additionally, the microcontroller sends the current values of temperature, humidity, and illumination to mobile and web applications every 60 seconds. This allows users to monitor environmental conditions and system performance in real-time. The system's design ensures robust communication, leveraging MQTT for reliable message delivery and operational efficiency in an IoT setting.

C. Software Support for Control and Parameter Monitoring

The integrated system, called *eRegulation*, includes a mobile application for regulating ambient conditions (temperature, humidity, and lighting) and a web application for monitoring parameters measured by the ESP microcontroller. Both applications were developed using the MERN stack, with MongoDB as the database. The mobile application, built with React Native, is available for both Android and iOS devices, ensuring efficient code sharing across platforms and streamlined development.

The mobile application (Fig. 6) features a home page displaying a welcome message, the current air temperature (°C), and humidity (%). Below, a menu offers four options: Temperature and Humidity for regulation settings, Light for lighting adjustments, and Statistics for monitoring parameters via a web application.

Upon launching, the application sends a "*welcome*" message to the MQTT broker, followed by a configuration message from the ESP microcontroller. This message provides live data about temperature, humidity, and lighting values in a compact format, such as "*t-22.50-h-66.75-l-0*", designed for efficient transmission. A simplified "*ping*" message is sent periodically or upon user request to refresh data. Icons for temperature, humidity, and lighting flash or change color based on active regulation modes.

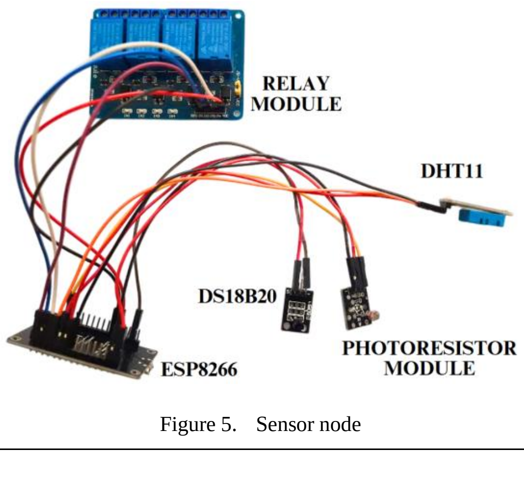

Figure 5. Sensor node

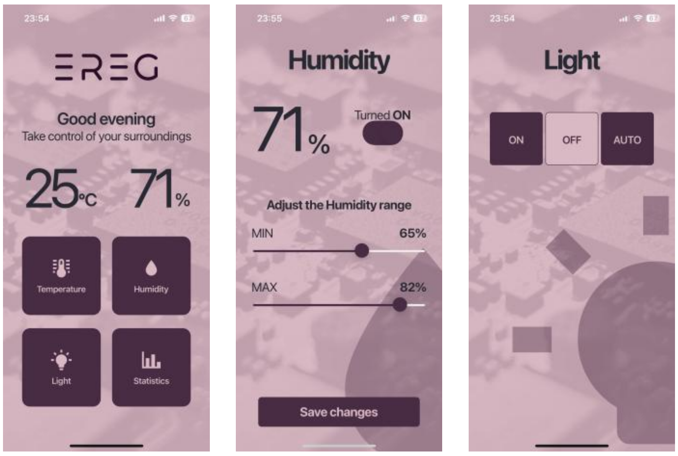

Figure 6. Mobile application

The temperature control screen displays current values and includes a switch to toggle regulation on/off. When activated, sliders appear for setting temperature ranges. The air humidity control screen operates in a similar manner. The light control screen offers On, Off, and Auto options, enabling either manual control or automatic adjustments to ambient light.

During operation, the system continuously monitors changes in regulation values, assessing whether the heating and ventilation are functioning correctly. If a malfunction is detected, the user is alerted to verify the heater or fan's operation. However, this assessment may not always be precise due to various environmental factors, such as room openness or the placement of the sensor node. Using previous temperature data stored in the database, the temperature growth trend over the last 15 minutes can be expressed through a linear regression formula. Linear regression is a mathematical model describing the linear relationship between an independent variable x and a dependent variable y [47]. The formulation of the linear model is represented by the equation (1):

$$y = mx + b \qquad (1)$$

where y is temperature, x is time, m is the slope of the line (slope coefficient) and b is the intercept of the y-axis. Then, it is possible to calculate m and b and use them to estimate the temperature growth trend.

If $m > 0$, it indicates a positive slope, which means an increase in temperature, and if $m < 0$ or $m \approx 0$, it could indicate a decrease or stagnation of temperature. An identical calculation is carried out for the trend of increasing/decreasing air humidity.

On the other hand, the web application (Fig. 7) allows users to view and analyze key parameters such as temperature, humidity, and lighting through a statistical representation of the data. Its purpose is to provide insight into the current and historical values of these parameters, with the option to filter data by time periods. The application ensures secure user authentication and intuitive data management, enabling efficient monitoring and real-time analysis of changes. Additionally, users have access to a centralized overview of all information, facilitating easier decision-making and optimization of system management.

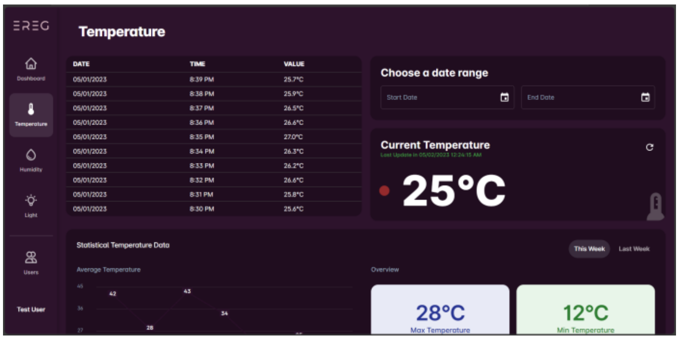

Figure 7. Web application

The development of the system requires a certain amount of time for design and configuration, but the code is implemented in a way that ensures universality and flexibility. After the initial installation, the system allows for easy expansion by adding new elements or sensors, thereby adapting to the specific needs of the user. This feature makes the system suitable for scaling and long-term use in various IoT scenarios.

V. Performance Analysis of the MQTT Protocol

The MQTT protocol represents an optimal solution for developing mobile applications within the IoT ecosystem, thanks to its network efficiency and ability to operate with minimal resource consumption. These characteristics make it particularly suitable for smartphones with limited bandwidth and resources. The protocol supports various Quality of Service levels, enabling adjustable reliability in message delivery. By utilizing the "*pub/sub*" model, where devices exchange information via predefined topics, MQTT offers significant flexibility in managing device communication. Its resilience to connection interruptions ensures communication continuity even in dynamic environments. As an open-standard protocol, MQTT facilitates interoperability in applications requiring the integration of diverse devices and services [48].

The HiveMQ broker offers three primary plans for the MQTT platform, tailored to different user needs. Among these, the free Serverless plan has been identified as the most efficient solution for the requirements of an experimental project, given its accessibility and low implementation costs. This plan supports up to 100 connected devices and 10 GB of monthly data throughput, with compatibility for MQTT versions 3.1, 3.1.1, and 5.0, basic authentication options, and the ability to use the WebSocket protocol. The maximum message size of 5 MB significantly exceeds the typical message sizes used in the system, further ensuring operational flexibility. The lack of a guaranteed Service Level Agreement (SLA) and limited technical support have been deemed acceptable considering the project goals. For more complex applications, the Starter plan offers unlimited connected devices, throughput of up to 1 MB/s, advanced authentication options, data streaming integrations, and a guaranteed 99.95% uptime SLA, along with priority technical support. For the most complex business requirements, the Enterprise plan provides customizable solutions with a high level of scalability, including managed cloud services or self-hosted options, as well as additional integration capabilities tailored to specific user needs [34].

In addition to HiveMQ, popular MQTT brokers such as Eclipse Mosquitto [49], EMQX [50], and RabbitMQ [51] offer similar core functionalities, including support for thousands of connections and standard MQTT protocols. However, their free versions are limited by technical support and Service Level Agreements (SLA). For medium-scale projects, performance and scalability are nearly identical across all brokers, with HiveMQ standing out due to its advanced integrations and guaranteed SLA.

The speed of MQTT message transmission and reception is influenced by several factors, including network latency, which pertains to delays in device communication; message size, as larger messages take longer to transmit; the Quality of Service (QoS) level, which determines the reliability of message delivery; broker performance, which impacts the processing and routing of messages; and the efficiency of both hardware and software involved in the communication process. Furthermore, the implementation specifics of MQTT client libraries can play a crucial role in optimizing communication speed to align with particular requirements. These combined factors [52] collectively influence the overall performance of the MQTT protocol. Test results for a specific MQTT protocol and broker are presented in Table 2.

The testing results of average performance for each of the 10 MQTT clients demonstrate high efficiency and reliability under various operating conditions. The testing covered different distances between clients and brokers, the use of various Wi-Fi networks with varying bandwidths and stability, as well as operation on devices with diverse hardware and software configurations. The connection establishment time was 0.0735 seconds, indicating a fast and stable connection process regardless of network conditions. All clients successfully published 100 out of 100 messages without any failures, confirming the protocol's reliability even in fluctuating network environments. The average message transmission time was 0.037 seconds, indicating extremely low latency, while the system's throughput was 2718.4706 messages per second, demonstrating the protocol's capability to efficiently handle large volumes of data in various scenarios. These results confirm that the MQTT protocol is reliable, fast, and adaptable, even in demanding and dynamic IoT environments [53].

Based on the analysis, individual speed tests were conducted for different message types, with results presented in Table 3, further substantiating the previous observations. The table highlights the key characteristics of different message types, including their size in bytes and corresponding response times. The analysis indicates that message size influences response time, but in all cases, the response time remains low, showcasing the protocol's efficiency in processing messages of varying complexities. The data in the table further confirm the consistency of system performance, regardless of variations in message size.

TABLE II. MQTT COMMUNICATION PERFORMANCE

Parameter	**Value**
Average time to establish a connection	0.0732 seconds
Successfully published messages	100/100
Unsuccessfully published messages	0/100
Publishing duration	0.38 seconds
Average throughput	2718.4689 messages/sec

TABLE III. ANALYSIS OF MESSAGES IN MQTT COMMUNICATION

Type of Message	**Size [B]**	**Response Time [ms]**
ping	4	29
welcome	7	31
t-on	4	29
h-off	5	29
t-20.50-h-52.35-l-1	19	32
t-20.50-1-18-22-h-52.35-0-60-85-l-1	35	34

Additionally, the results show that using a compact message format is a better choice in scenarios where bandwidth, speed, and resource limitations are critical factors, which is often the case in IoT environments. The compact format reduces message size, leading to lower transmission times and reduced network load while ensuring efficient data processing. Compared to the JSON format, the compact format offers advantages in terms of reduced message size and improved performance, making it more suitable for applications with limited resources and high-speed transmission requirements. This analysis further emphasizes that choosing the message format tailored to the specific requirements of IoT applications is essential for achieving optimal system performance.

VI. APPLICATION OF THE SYSTEM IN SMART HOMES

The proposed system can be effectively implemented in real-life smart home environments to monitor and control ambient parameters such as temperature, humidity, and lighting. By ensuring efficient communication between sensor nodes, mobile applications, and web platforms, the system provides a reliable solution for maintaining optimal environmental conditions and energy efficiency in residential spaces.

Development of the system has included devices that have minimal power demands. Maximum currents that sensors consuming do not exceed 1 mA in active mode (measuring and data transmission) [42, 43]. The most demanding device is ESP8266 board, but its maximum consumption is about 100 mA [39] in transmission mode, so it does not present a large problem. MQTT usage, as communication protocol for devices with limited resources, also contributes to the system efficiency.

In smart homes, this system can automate heating, cooling, and lighting based on real-time sensor data, reducing energy consumption and enhancing user comfort. For instance, the temperature sensor can regulate heating or cooling devices to maintain a predefined range, while the photoresistor module can control the lighting system to match the natural light level in a room. Using the DALL-E 3 model [54], an image of the future prototype of the hardware device was generated, reflecting the specifications and settings outlined earlier in this study (Fig. 8).

Similar devices are available on the market that, although not open-source or fully customizable, offer some of the functionalities of this system. Devices available on Amazon

[46], which enable remote control via Wi-Fi using an app and accompanying software, are generally priced between 30 and 40 euros for basic versions. More advanced models, which provide additional features or higher build quality, often exceed 100 euros. However, these commercial devices often have limitations in terms of customization and further development by end users.

In contrast, the proposed system, based on basic electronic components and software developed for universal application, offers a significantly more economical and flexible alternative. With an estimated cost of just over 15 euros for the complete hardware and software, the system meets all technical and functional requirements for efficient smart home management. On alternative online shops, the hardware components can often be found at even lower prices. Its openness to customization and scalability makes it an ideal solution for users seeking long-term sustainable, adaptable, and cost-effective IoT solutions in the field of automation.

To expand the functionality and improve user interaction, several enhancements can be proposed for the system. One potential upgrade is the integration of an LCD display, which would provide users with a local interface to monitor real-time temperature, humidity, and lighting levels directly within their homes. This feature would be particularly beneficial in situations where mobile or web applications are unavailable. Additionally, the incorporation of extra sensors, such as motion detectors, CO2 sensors, or air quality monitors, could broaden the system's applications.

For instance, motion detectors could enable automatic lighting in unoccupied rooms, while air quality sensors might trigger ventilation systems when pollutant levels exceed safe thresholds. Another valuable enhancement is the integration with voice assistants, allowing users to control devices or access sensor readings using voice commands, thereby increasing convenience and accessibility. Furthermore, upgrading the web application to include advanced data analytics and predictive modeling would enable users to gain deeper insights into long-term trends in their home's environmental parameters, helping them make informed decisions to optimize energy usage and improve comfort. By implementing these upgrades, the system could become a more robust, user-friendly, and versatile solution tailored to the evolving needs of modern smart homes.

Figure 8. Prototype of the hardware device

VII. CONCLUSION

The development of modern technologies and the increasing presence of the Internet of Things in everyday life have led to the popularization of the smart homes concept. Smart homes applications significantly improve quality of life, introducing upgrades in everyday life. Improvements are highly noticeable in the fields of surveillance, security, healthcare and energy management. One of the main aspects of smart home concept is communication. It is very important to choose an adequate transfer protocol to ensure reliable and efficient communication.

MQTT stands out as a very good solution for integration into smart home applications. It is simple, easy to integrate and suitable for systems with limited resources. MQTT is already used in many IoT applications, and it has shown better results compared to other commonly used protocols. Based on various studies, a high efficiency of this protocol in the real-time monitoring and control of key parameters in smart environments.

In this article, an application of the MQTT protocol in a system for indoor ambient parameters control is proposed. System integrates sensor node with Android/iOS mobile application and web application. Sensor node is based on the NodeMCU development board with an integrated Wi-Fi module. DS18B20 temperature sensor, DHT11 humidity sensor and LDR module are used for parameters measurements. Developed mobile application is user-friendly and allows users to monitor and control temperature, humidity and light in the room by setting upper and lower limits of proper parameters. As it was said earlier, communication between applications and node is achieved with MQTT protocol. It provides reliable and protected real-time message exchange. Both sides send and receive messages in predefined formats, which significantly facilitates the analysis of the messages themselves as well as taking the necessary actions.

Performance analysis of the MQTT protocol leads to the conclusion that it represents an optimal solution, as it achieves high efficiency with low resource consumption. Tests were done in various scenarios (different bandwidths of Wi-Fi networks, different distances between clients and broker, different system configurations). Tests results showed that protocol is fast, reliable and adaptable. Compactness of the message format ensures efficient data transfer and enables reliable and uninterrupted communication.

REFERENCES

[1] J. Salazar and S. Silvestre, "Internet of Things", European Virtual Learning Platform for Electrical and Information Engineering, 2017. ISBN 978-80-01-06232-6

[2] A. S. Abdul-Qawy, P. J. Pramond, E. Magesh, T. Srinivasulu, "The Internet of Things (IoT): An overview," International Journal of Engineering Research and Applications, vol. 5, issue 12, 2015, pp. 71-82

[3] L. Atzori, A. Iera, and G. Morabito, "Understanding the internet of things: Definition, potentials, and societal role of a fast-evolving paradigm", Ad Hoc Networks, vol 56, 2017,pp. 122-140

[4] G. Velivela, A. Srija, and S. Krishna Rao, "Smart homes: Steps, components, utilities and challenges", International Journal of Engineering & Technology, 2018, pp. 436-440

[5] M. Hasan, M. Hossain Anik, S. Chowdhury, S. Alam Chowdhury, T. Islam Bilash and S. Islam, "Low-cost appliance switching circuit for discarding technical issues of microcontroller controlled smart home", International Journal of Sensors and Sensor Networks, vol. 7, no. 2, 2019, pp. 16-22

[6] I. Zenginis, J. Vardakas, N. E. Koltsaklis and C. Verikoukis, “Smart home’s energy management through a clustering-based reinforcement learning approach”, IEEE Internet of Tings Journal, vol. 9, no. 17, 2022, pp. 16363-16371

[7] P. Mtshali and F. Khubisa, “A smart home appliance control system for physically disabled people”, in Proceedings of the 2019 Conference on Information Communications Technology and Society (ICTAS), 2019, pp. 1-5

[8] A. Chakraborty, M. Islam, F. Shahriyar, S. Islam, H. U. Zaman and M. Hasan, “Smart home system: A comprehensive review”, Journal of Electrical and Computer Engineering, Hindawi, 2023, pp. 1-3

[9] MQTT: The standard for IoT messaging, Official documentation, [Online] Available: https://mqtt.org, Accessed 20 Oct 2024

[10] I. L. Paris, M. H. Habaebi, and A. M. Zyoud, “Implementation of SSL/TLS security with MQTT protocol in IoT environment”, Wireless Personal Communications, vol. 132, no. 1, 2023, pp. 163-182

[11] T. D. P. Mendes, R. Godina, E. M. G. Rodrigues, J. C. O. Matias and J. P. S. Catalão, “Smart home communication technologies and applications: Wireless protocol assessment for home area network resources”, Energies, vol. 8, no. 7, pp. 2015, 7279-7311

[12] H. Zemrane, Y. Baddi and A. Hasbi, “IoT smart home ecosystem: Architecture and communication protocols”, 2019 International Conference of Computer Science and Renewable Energies (ICCSRE), Agadir, Morocco, 2019, pp. 1-8

[13] S. Marksteiner, V. J. Exposito Jimenez, H. Valiant and H. Zeiner, “An overview of wireless IoT protocol security in the smart home domain”, 2017 Internet of Things Business Models, Users, and Networks, Copenhagen, Denmark, 2017, pp. 1-8

[14] G. Sittampalam, R. Ragel and S. Karunarathna, “Two-factor commissioning for thread protocol”, 2023 3rd International Conference on Advanced Research in Computing (ICARC), 2023, pp. 190-195

[15] R. Toulson and T. Wilmshurst, “Wireless communication – Bluetooth and Zigbee”, Fast and Effective Embedded Systems Design, 2017, pp. 257-290

[16] S. Bansala and D. Kumar, “A reliable CoAP protocol for IoT communication”, Research Square, 2022

[17] N. Sharma and R. Agarwal, “HTTP, Websocket, and SIGNALR: A comparison of real-time online communication protocols”, Mining Intelligence and Knowledge Exploration, 2023, pp. 128-135

[18] B. Mishra and A. Kertesz, “The use of MQTT in M2M and IoT systems: A survey”, in IEEE Access, vol. 8, 2020, pp. 201071-201086

[19] M. Esposito, A. Belli, L. Palma and P. Pierleoni, “Design and implementation of a framework for smart home automation based on cellular IoT, MQTT, and serverless functions”, Sensors, vol. 23, no. 9, 2023, p. 4459

[20] N. Naik, “Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP”, 2017 IEEE International Systems Engineering Symposium (ISSE), 2017. pp. 1-7

[21] D. Soni and A. Makwana, “A survey on MQTT: A protocol of Internet of Things (IoT)”, International conference on telecommunication, power analysis and computing techniques (ICTPACT-2017), vol. 20, 2017, pp. 173-177

[22] F. Yalçinkaya, H. Aydilek, M. Y. Erten, N. İnanç, “IoT based smart home testbed using MQTT communication protocol”, Uluslararası Muhendislik Arastirma ve Gelistirme Dergisi, 2020, p. 317

[23] Z. Yushaniza M. Yusoff, M. K. Ishak, L. A. B. Rahim, M. S. M. Asaari, “Improving smart home security via MQTT: Maximizing data privacy and device authentication using elliptic curve cryptography”, Computer Systems Science and Engineering, vol. 48, no. 6, 2024, pp. 1669-1667

[24] G. Betta, G. Cerro, M. Ferdinandi, L. Ferrigno and M. Molinara, “Contaminants detection and classification through a customized IoT-based platform: A case study”, IEEE Instrumentation & Measurement Magazine, vol. 22, no. 6, 2019, pp. 35-44

[25] M. Simić, M. Stojanović, L. Manjakkal and K. Zaraska, “Multi-sensor system for remote environmental (air and water) quality monitoring”, 2016 24th Telecommunications Forum (TELFOR), 2016, pp. 1-4

[26] K. Grgić, I. Špeh and I. Heđi, “A web-based IoT solution for monitoring data using MQTT protocol”, 2016 International Conference on Smart Systems and Technologies (SST), 2016, pp. 249-253

[27] P. K. Sayannagari and S. P. Manikanta, “IoT based smart home applications using cloud based MQTT protocol”, International Journal Of Research In Electronics And Computer Engineering, vol 7, issue 2, 2019, pp. 2769-2773

[28] V. Thirupathi and K. Sagar, “Implementation of home automation system using MQTT protocol and ESP32”, International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249-8958 (Online), vol. 8, issue 2C2, 2018, pp. 111-113

[29] A. Gharat and S. Patil, “IOT based smart home using MQTT protocol”, International Journal of Science Technology and Management, ISSN 2394-1537, vol 10, no. 3, 2021, pp. 27-34

[30] S. Balakrishnan, B. Madhurekha, N. Shobana, S. S. Selshiya and G. Sathyabama, “Home automation system using ESP8266 based MQTT”, International Journal of Research in Engineering, Science and Managemen, vol 2, issue 2, 2019, pp. 489-492

[31] S. Nandhini, D. Pragatheswaran, S. Priyadharshini and L. B. Tennison, “Smart home automation using Adafruit and MQTT protocol using voice control”, International Journal of Advance Research and Innovative Ideas in Educationm, vol. 6, issue 2, 2020, pp. 274-277

[32] F. Alfiah, B. Rahman and Imelda, “Control system prototype smart home IoT based with MQTT method using Google asisstant”, Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi), vol. 4, no. 2, 2020, pp. 303-310

[33] C. Patel and N. Doshi, “A novel MQTT security framework in generic IoT model”, Procedia Computer Science, vol. 171, 2020, pp. 1399-1408

[34] HiveMQ, Official Documentation, [Online] Available: https://docs.hivemq.com/hivemq/latest/user-guide/index.html, Accessed 12 Nov 2024

[35] D. Košutar, “Izvedba i testiranje komunikacijskog sustava baziranog na MQTT protokolu”, Završni rad, Sveučiište Sjever, Sveučilišni centar Varaždin, Odjel za elektrotehniku, 2018

[36] ISO/IEC 20922:2016 standard, “Information technology - Message Queuing Telemetry Transport (MQTT)”, v3.1.1, [Online] Available: https://www.iso.org/standard/69466.html, Accessed 13 Nov 2024

[37] U-Blox, “MQTT beginner’s guide”, Tech, [Online] Available: https://www.u-blox.com/en/blogs/insights/mqtt-beginners-guide, Accessed 14 Nov 2024

[38] T. Begović, V. Čabarkapa, M. Ivković and B. Popović, “Implementation of a sensor node for ambient conditions monitoring and control in smart homes”, 23rd International Symposium INFOTEH-JAHORINA (INFOTEH), 2024, pp. 1-6

[39] Espressif Systems IOT Team, “ESP8266EX datasheet”, Datasheet v4.3, 2015, pp. 1-31

[40] Handson Technology, “ESP8266 NodeMCU Wi-Fi development board”, User Manual v1.3, [Online] Available: https://handsontec.com/dataspecs/module/esp8266-V13.pdf, Accessed 16 Nov 2024

[41] Using the Arduino Software (IDE), [Online] Available: https://docs.arduino.cc/learn/starting-guide/the-arduino-software-ide, Accessed 16 Nov 2024

[42] Dallas Semiconductor, “DS18B20 programmable eesolution 1-Wire Digital thermometer”, pp. 1-27, [Online] Available: https://cdn.sparkfun.com/datasheets/Sensors/Temp/DS18B20.pdf, Accessed 16 Nov 2024

[43] D-Robotics, “DHT11 humidity & temperature sensor”, 2018, pp. 1-10

[44] Joy – IT, “KY – 018 Photoresistor module”, 2017, pp. 74-78

[45] Handson Technology, “4 Channel 5V optical isolated relay module”, User Guide, [Online] Available: https://www.handsontec.com/dataspecs/4Ch-relay.pdf, Accessed 17 Nov 2024

[46] Amazon, “Amazon.com: Online Shopping for Electronics, Apparel, Computers, Books, DVDs & More”, [Online] Available: www.amazon.com, Accessed 20 Dec 2024

[47] Newcastle University, “Simple linear regression”, [Online] Available: https://www.ncl.ac.uk/webtemplate/ask-assets/external/maths-resources/statistics/regression-and-correlation/simple-linear-regression.html, Accessed 18 Nov 2024

[48] C. Bernstein, “MQTT (MQ telemetry transport)”, TechTarget, IoT Agenda, [Online] Available: https://www.techtarget.com/iotagenda/definition/MQTT-MQ-Telemetry-Transport, Accessed 18 Nov 2024

[49] Eclipse Mosquitto, Official Documentation, [Online] Available: https://mosquitto.org/documentation, Accessed 14 Dec 2024

[50] EMQX, Product Documentation, [Online] Available: https://docs.emqx.com/en, Accessed 14 Dec 2024

[51] RabbitMQ 4.0 Documentation, [Online] Available: https://www.rabbitmq.com/docs, Accessed 14 Dec 2024

[52] Flespi Platform, "HTTP vs MQTT performance tests", Medium, [Online] Available: https://medium.com/@flespi/http-vs-mqtt-performance-tests-f9adde693b5f, Accessed 20 Nov 2024

[53] V. Čabarkapa, T. Begović, M. Ivković and B. Popović, "Integration of system for ambient conditions monitoring and regulation in smart homes using the MQTT protocol", 2024 23rd International Symposium INFOTEH-JAHORINA (INFOTEH), 2024, pp. 1-6

[54] DALLE-3 Model Research, OpenAI, [Online] Available: https://openai.com/index/dall-e-3, Accessed 22 Nov 2024

Tijana Begović received the B.Sc. degree in Electrical Engineering from the University of East Sarajevo, Bosnia and Herzegovina. She is working as a teaching assistant and researcher at the Faculty of Electrical Engineering. The field of research interest include embedded systems and IoT.

Vasilije Čabarkapa received the B.Sc. degree in Electrical Engineering, specializing in Computer Science and Informatics, from the University of East Sarajevo, Bosnia and Herzegovina. He works as a Backend Developer at Dwelt Software. His areas of interest include web and mobile development, with a focus on innovating digital solutions.

Milica Ivković received the B.Sc. degree in Electrical Engineering, specializing in Computer Science and Informatics, from the University of East Sarajevo, Bosnia and Herzegovina. She works as a Fullstack developer at Lanaco Company. Her areas of interest include artificial intelligence and software development.

Božidar Popović, PhD in Electrical Engineering at the University of East Sarajevo, Bosnia and Herzegovina. He works as an associate professor at the Faculty of Electrical Engineering. His research interests include Sensors and Measurements, Electronics and Electronic Systems, and Embedded Systems.