Open Access Journal on Engineering Applications (OAJEA) ISSN: 3105-7004
https://doi.org/10.64886/oajea.0101.001 Volume No: 01, Issue No: 01, Page: 1-13

A Comprehensive Review of M2M Communication Protocols

Md Hafizur Rahman[1,*] and M. Naderuzzaman[2]

[1]HafizLab, Dhaka, Bangladesh
[2]Department of Computer Science and Engineering, Sonargaon University, Dhaka, Bangladesh

***Abstract*—The rapid expansion of the Internet of Things (IoT) and smart device ecosystems has accelerated the development and deployment of Machine-to-Machine (M2M) communication protocols. M2M communication enables autonomous data exchange between devices without human intervention, forming the backbone of modern IoT applications in industries such as healthcare, transportation, energy, and manufacturing. This comprehensive review provides an in-depth analysis of the most widely adopted M2M communication protocols, including HTTP, WebSocket, XMPP, AMQP, MQTT, CoAP, DDS, LPWAN and LwM2M. The paper explores their architectural principles, operational mechanisms, and core features, highlighting their respective strengths and limitations in various application scenarios. Comparative analysis is presented based on critical performance metrics such as scalability, latency, security, energy efficiency, interoperability, and Quality of Service (QoS). The review also discusses the challenges associated with protocol selection, integration, and deployment in resource-constrained environments. Furthermore, emerging trends such as the convergence of edge computing, 5G connectivity, and enhanced security frameworks are examined for their impact on the evolution of M2M communication. The paper concludes by outlining future research directions, including the need for adaptive and intelligent protocol frameworks capable of addressing dynamic application requirements. By synthesizing recent advancements and providing actionable insights, this review aims to guide researchers, engineers, and practitioners in selecting and implementing optimal M2M communication protocols for next-generation IoT systems.**

***Index Terms*—M2M, IoT, HTTP, WebSocket, XMPP, AMQP, MQTT, CoAP, DDS, LPWAN, LwM2M**

Received: 30 June 2025, Revised: 10 July 2025
Accepted: 15 July 2025, Published: 30 July 2025
Email of corresponding author: hafizurfpbd@gmail.com

I. Introduction

Machine-to-Machine (M2M) communication enables direct data exchange between devices[1], sensors and systems without human intervention. The rapid expansion of the Internet of Things (IoT), coupled with advances in wireless technologies, has heightened the need for efficient, scalable, and secure communication protocols[2]. M2M protocols are at the core of industrial automation, smart homes, healthcare, and numerous IoT-driven applications. This paper presents a comprehensive review of key M2M communication protocols[3], focusing on their architecture, operating principles, comparative advantages, challenges, and future research trends.

A. Background and Evolution of M2M Communication

M2M communication originated from early telemetry and telematics applications, focusing primarily on remote monitoring and control via wired or cellular networks. The development of IP-based networking and wireless communication led to increased device connectivity and interoperability. Today, M2M forms the backbone of IoT systems, supporting applications across industry, healthcare, smart cities, and more. This evolution has necessitated standardized protocols to ensure reliability, scalability, and security.

B. Importance of M2M in IoT Ecosystem

Machine-to-Machine (M2M) communication is a foundational pillar of the Internet of Things (IoT) ecosystem[1]. It refers to the direct exchange of data between devices, sensors, and machines without human intervention[4]. The importance of M2M in the IoT ecosystem can be summarized as follows:

- **Enabling Autonomous Operations:** M2M allows devices to automatically collect, share, and act on data, reducing or eliminating the need for manual input. This automation is critical for real-time monitoring, remote control, and self-healing systems[1].
- **Scalability:** IoT ecosystems often involve thousands or millions of connected devices. M2M provides a scalable way for these devices to communicate and coordinate actions, making large-scale IoT deployments practical and efficient.
- **Real-Time Data Exchange:** M2M ensures instantaneous data transfer between devices. This is vital for applications such as industrial automation, smart grids, healthcare monitoring, and intelligent transportation systems, where timely response is essential[5].

- **Improved Efficiency and Cost Reduction:** By automating processes and enabling predictive maintenance, M2M minimizes downtime, reduces operational costs, and optimizes resource usage.
- **Enhanced Data Analytics:** Continuous M2M data streams feed powerful analytics engines, providing valuable insights, enabling smarter decision-making, and fostering innovation in areas like smart cities, agriculture, and logistics.
- **Interoperability and Integration:** M2M protocols and standards ensure seamless communication among diverse devices and platforms, supporting the integration of legacy systems into modern IoT networks.
- **Foundation for Advanced IoT Applications:** Many advanced IoT use cases—such as smart homes, autonomous vehicles, and connected healthcare—rely on robust M2M communication to function reliably and securely.

In summary, M2M communication acts as the "nervous system" of the IoT ecosystem, empowering devices to autonomously interact, collaborate, and deliver intelligent services that transform industries and improve quality of life.

C. Objectives and Scope of the Review

The primary objective of this review[11] is to provide a comprehensive analysis of the current landscape of Machine-to-Machine (M2M) communication protocols[7], with a particular focus on their application within the evolving Internet of Things (IoT) ecosystem[8]. This article aims to systematically explore the fundamental principles, architectural models, key features, strengths, and limitations of widely adopted M2M protocols. Additionally, the review seeks to compare and contrast these protocols based on critical parameters such as scalability, interoperability, security, energy efficiency, and real-time capabilities. The scope of this review encompasses both standardized and emerging M2M communication protocols, including but not limited to HTTP, WebSocket, XMPP, AMQP, MQTT, CoAP, DDS, LPWAN and LwM2M. By synthesizing recent research findings, industry practices, and technological trends, this article intends to identify current challenges, highlight potential research gaps, and suggest future directions for the development and deployment of robust M2M communication solutions in diverse IoT applications.

D. Organization of the Paper

The remainder of this paper is organized as follows. Section II provides an overview of M2M communication[9], its evolution, and its significance in the context of the Internet of Things (IoT)[10]. Section III-B describe the classification of M2M communication protocols based on network layer and communication paradigm. Section IV presents a detailed analysis of prominent M2M communication protocols, discussing their architectural models, operational principles and application scenarios. In Section V, a comparative evaluation of these protocols is conducted, focusing on key parameters such as scalability, security, interoperability, and energy efficiency. Section VI overview of security and privacy issues. Section VIII highlights the current challenges and research gaps associated with M2M communications and discusses emerging trends and future directions in the development of M2M protocols. Finally, Section IX concludes the paper by summarizing the key findings and potential implications for future research and practical deployments.

II. Fundamentals of M2M Communication

A. Definition and Key Concepts

Machine-to-Machine (M2M) communication refers to the automated exchange of data between devices, machines, or systems without direct human intervention[1]. This paradigm enables physical devices—such as sensors, actuators, controllers, and embedded systems—to connect, communicate, and coordinate their actions through wired or wireless networks[11]. At its core, M2M communication encompasses three main components: data acquisition from the physical environment, transmission of data across communication networks, and data processing for intelligent decision-making or control.

Key concepts in M2M communication include device connectivity, network architecture, data protocols, and automation. Device connectivity ensures that machines can reliably interface with one another and with backend systems. Network architecture defines how devices are organized, connected, and managed within the M2M ecosystem, including point-to-point, star, mesh, or hybrid topologies. Communication protocols govern the rules and formats for data exchange, ensuring interoperability, reliability, and security among heterogeneous devices. Automation in M2M allows for real-time monitoring, remote control, predictive maintenance, and autonomous operation across a wide range of applications[1].

In the context of the Internet of Things (IoT), M2M communication serves as a fundamental building block, enabling the seamless integration of devices and the realization of intelligent, interconnected environments. By leveraging standardized communication protocols, M2M systems facilitate scalable, efficient, and secure information exchange, paving the way for innovations in industries such as smart cities, healthcare, transportation, manufacturing, and energy management[12].

B. M2M Architecture

The architecture of Machine-to-Machine (M2M) communication provides a structured framework for enabling efficient, scalable, and secure data exchange between devices and systems. A typical M2M architecture is composed of several key layers, each responsible for specific functions within the communication process. These layers are generally classified as the device layer, communication (or network) layer, and application (or service) layer.

- **Device Layer:** This foundational layer consists of a wide range of end devices such as sensors, actuators, embedded controllers, and smart meters. These devices are responsible for data acquisition, sensing environmental parameters, and performing control actions based on received instructions.
- **Communication Layer:** Acting as the backbone of M2M systems, the communication layer enables reliable data transmission between devices and backend systems. It incorporates various wired and wireless communication technologies such as cellular networks (e.g., 3G, 4G, 5G), Wi-Fi, ZigBee, Bluetooth, and Low-Power Wide-Area Networks (LPWAN). This layer also encompasses protocol stacks that ensure secure, interoperable, and real-time data exchange.
- **Application Layer:** The application or service layer processes the collected data, facilitates analytics, and delivers value-added services to end-users. It encompasses cloud platforms, data management systems, and user applications that enable functionalities such as remote monitoring, automation, visualization, and decision support[10].

Additionally, a robust M2M architecture often incorporates middleware components that handle interoperability, device management, data aggregation, security, and protocol translation between heterogeneous systems. The layered approach in M2M architecture allows for modularity, scalability, and ease of integration, which are essential for deploying large-scale IoT solutions across diverse application domains.

C. M2M vs IoT: Similarities and Differences

Machine-to-Machine (M2M) communication and the Internet of Things (IoT) are closely related concepts that share several foundational principles, yet they exhibit notable differences in scope, architecture, and application.

Similarities:

- **Device Connectivity:** Both M2M and IoT focus on enabling connectivity between physical devices, allowing them to exchange data autonomously.
- **Automation:** Both paradigms support automation by facilitating real-time data collection, remote control, and intelligent decision-making without direct human intervention.
- **Communication Technologies:** M2M and IoT utilize a wide range of wired and wireless communication technologies, including cellular networks, Wi-Fi, Bluetooth, and LPWAN[13].
- **Data-Driven Applications:** Both rely on data acquisition and processing to enable smart applications across domains such as healthcare, transportation, industry, and energy management.

Differences:

- **Scope:** M2M traditionally refers to point-to-point or networked communication between machines, typically within a closed system. In contrast, IoT represents a broader vision, encompassing not only device-to-device communication but also integration with cloud platforms, big data analytics, and end-user applications on a global scale.
- **Architecture:** M2M systems are often built using proprietary or industry-specific protocols, focusing on vertical integration. IoT architectures, however, emphasize horizontal integration, interoperability, and the use of standardized protocols to connect heterogeneous devices and platforms.
- **Scalability and Flexibility:** IoT systems are inherently designed for large-scale, flexible deployments, often connecting billions of devices across various domains. M2M solutions, while scalable, are generally more limited in scope and adaptability.
- **User Interaction:** While M2M primarily enables automated machine operations, IoT incorporates user interfaces, mobile apps, and data visualization tools to engage human users and enhance decision-making.
- **Data Utilization:** IoT leverages advanced analytics, artificial intelligence, and cloud computing to extract actionable insights from massive data streams, whereas M2M typically focuses on direct control, monitoring, and simple reporting.

In summary, M2M can be considered a foundational technology that paved the way for the broader and more versatile IoT ecosystem. While both aim to create smarter, more efficient systems through connectivity and automation, IoT expands upon the principles of M2M by enabling global integration, enhanced analytics, and user-centric applications.

III. Classification of M2M Communication Protocols

A. Protocols Based on Network Layer

Network layer protocols play a pivotal role in enabling reliable and efficient data transmission between devices in Machine-to-Machine (M2M) communication systems. These protocols are responsible for addressing, routing, and forwarding data packets across heterogeneous networks, ensuring seamless end-to-end connectivity and interoperability. In the context of M2M communications, network layer protocols are especially critical for managing large-scale deployments with diverse device types and communication technologies.

Key Network Layer Protocols in M2M Communication:

- **IPv4 and IPv6:** The Internet Protocol versions 4 and 6 (IPv4/IPv6) serve as the foundational network layer protocols, providing unique addressing and routing mechanisms necessary for global device connectivity. IPv6, in particular, offers an expanded address space, making it well-suited for massive-scale M2M and IoT environments.

- **6LoWPAN:** The IPv6 over Low-power Wireless Personal Area Networks (6LoWPAN) protocol enables the transmission of IPv6 packets over IEEE 802.15.4-based wireless networks. 6LoWPAN facilitates efficient addressing, fragmentation, and compression, making it ideal for resource-constrained M2M devices in wireless sensor networks[14].
- **RPL:** The Routing Protocol for Low-Power and Lossy Networks (RPL) is specifically designed for M2M and IoT networks characterized by constrained devices and lossy communication links. RPL constructs destination-oriented directed acyclic graphs (DODAGs) to provide scalable and adaptable routing in dynamic environments[15].
- **Mobile IP:** Mobile IP protocols support seamless mobility management, allowing M2M devices to maintain ongoing connections while moving across different network domains. This is particularly important for applications such as fleet management and asset tracking[1].

Network layer protocols not only ensure efficient routing and addressing but also contribute to network scalability, security, and quality of service in M2M communication. By supporting interoperability between heterogeneous devices and networks, these protocols enable the robust and flexible integration of M2M systems within the broader Internet of Things (IoT) ecosystem.

B. Protocols Based on Communication Paradigm

Machine-to-Machine (M2M) communication protocols can be classified according to their underlying communication paradigms, which define the manner and patterns in which devices exchange information. The choice of communication paradigm has a significant impact on the efficiency, scalability, and suitability of protocols for different M2M applications. The two most common paradigms are the *request-response (client-server)* and the *publish-subscribe* models.

- **Request-Response (Client-Server) Paradigm:** In this model, one device (the client) initiates a request for data or action, and another device (the server) responds accordingly. This paradigm is straightforward and well-suited for applications that require direct querying and immediate feedback. A prominent example is the Constrained Application Protocol (CoAP)[16], which is optimized for resource-constrained devices and supports lightweight, RESTful communication similar to HTTP[17].
- **Publish-Subscribe Paradigm:** In the publish-subscribe model, devices (publishers) send messages to a central broker or intermediary, which then distributes the messages to interested devices (subscribers). This decouples senders and receivers, enabling greater scalability and flexibility, especially in dynamic and large-scale deployments. Message Queuing Telemetry Transport (MQTT) is a widely used publish-subscribe protocol that offers lightweight communication with minimal overhead, making it ideal for low-bandwidth and high-latency environments. Advanced Message Queuing Protocol (AMQP) and Data Distribution Service (DDS) are other examples supporting publish-subscribe messaging for robust and reliable M2M interactions[18].
- **Hybrid Paradigms:** Some protocols, such as Extensible Messaging and Presence Protocol (XMPP)[19], support both client-server and publish-subscribe mechanisms, offering flexibility to accommodate various communication patterns within M2M systems[20].

The selection of communication paradigm depends on the specific requirements of the M2M application, such as scalability, latency, resource constraints, and network topology. By leveraging the appropriate paradigm and corresponding protocols, M2M systems can achieve efficient, reliable, and scalable information exchange across diverse use cases.

IV. Overview of Major M2M Communication Protocols

A. HTTP/HTTPS

The **Hypertext Transfer Protocol (HTTP)** is one of the most widely used application layer protocols on the Internet and has been instrumental in enabling client-server communication. Initially designed for human-to-human interaction through web browsers and servers, HTTP has also found significant use in Machine-to-Machine (M2M) communication due to its simplicity, ubiquity, and ease of implementation[21].

a) Fundamentals and Working Principle: HTTP operates as a request-response protocol in the client-server architecture. In a typical scenario, a client (such as a sensor node or gateway) initiates a connection by sending an HTTP request to a server, which then processes the request and responds accordingly. HTTP utilizes the stateless TCP transport protocol, ensuring reliable delivery but without retaining any session information between requests.

b) Key Features:

- **Statelessness:** Each HTTP request is independent, making it simple but less efficient for persistent M2M sessions.
- **Text-based:** HTTP messages are human-readable, which facilitates debugging but increases overhead compared to binary protocols.
- **Portability and Interoperability:** Being universally supported, HTTP is easily integrated across diverse platforms and programming languages.
- **Support for RESTful APIs:** Modern M2M applications often leverage RESTful architectural style, utilizing HTTP methods such as GET, POST, PUT, and DELETE for resource manipulation[22].

c) HTTP in M2M and IoT Applications: In the context of M2M and IoT, HTTP is commonly used for data retrieval, device management, and cloud connectivity. Devices such as sensors and gateways frequently use HTTP to send

telemetry data to centralized servers or cloud platforms. The protocol's widespread adoption ensures compatibility with existing infrastructure, including firewalls and proxies, which is crucial for enterprise deployments.

However, HTTP also presents some limitations for M2M applications:

- **Overhead:** HTTP headers are relatively large, leading to increased bandwidth consumption—an important consideration for low-power or low-bandwidth devices.
- **Real-Time Limitations:** HTTP's request-response model is less suited for real-time or bidirectional communication, where protocols like WebSocket or MQTT may be preferred.
- **Resource Consumption:** Maintaining persistent TCP connections and handling verbose messages can be resource-intensive for constrained devices.

d) Security Considerations: HTTP can be secured using HTTPS (HTTP over TLS/SSL), ensuring confidentiality and integrity of transmitted data. This is especially important for sensitive M2M applications such as healthcare, industrial automation, or financial services.

e) Conclusion: While HTTP may not be the most efficient protocol for all M2M scenarios, its universality, ease of use, and mature ecosystem make it a viable choice, particularly for applications where interoperability and rapid development are prioritized over low latency and minimal resource usage. The continued evolution of HTTP/2 and HTTP/3 further enhances its performance characteristics, making it relevant for certain M2M and IoT use cases.

B. WebSocket

The **WebSocket** protocol is a standardized network protocol that provides full-duplex, bidirectional communication channels over a single, long-lived TCP connection. Standardized by the IETF as RFC 6455 in 2011, WebSocket addresses the limitations of traditional HTTP, particularly for applications requiring real-time data exchange[23].

a) Fundamentals and Working Principle: WebSocket begins with an HTTP handshake to establish the connection, after which it upgrades the protocol to a persistent WebSocket connection. This enables the server and client to send messages to each other at any time without the overhead of repeatedly opening new HTTP connections. Data can be transmitted in both directions as either text or binary frames, enabling low-latency, real-time communication.

b) Key Features:

- **Full-Duplex Communication:** Allows simultaneous, bidirectional data exchange between client and server.
- **Low Latency:** Maintains a persistent connection, reducing overhead and enabling near real-time data transfer.
- **Efficient Resource Usage:** Eliminates the need for repeated HTTP requests and responses, optimizing bandwidth and server resources.
- **Support for Text and Binary Data:** Facilitates a wide range of M2M communication scenarios, including sensor data, control signals, and multimedia.
- **Cross-Platform Compatibility:** Supported by modern web browsers, IoT devices, and server platforms.

c) WebSocket in M2M and IoT Applications: WebSocket is increasingly used in M2M and IoT applications where low latency and real-time interaction are critical. Common use cases include remote monitoring, live sensor data streaming, industrial automation, online gaming, and collaborative applications. WebSocket is also useful for implementing lightweight event-driven architectures in distributed IoT environments.

d) Security Considerations: WebSocket can be secured using the *wss://* protocol, which leverages TLS/SSL to ensure encrypted communication between endpoints. Authentication and authorization mechanisms should be implemented at the application level to prevent unauthorized access, as WebSocket itself does not define these by default.

e) Conclusion: WebSocket protocol overcomes the inherent limitations of HTTP for real-time, bidirectional M2M communication. Its efficiency, scalability, and support for persistent connections make it a robust choice for modern IoT and M2M scenarios requiring instant data transfer and interactive control.

C. XMPP

The **Extensible Messaging and Presence Protocol (XMPP)** is an open, XML-based communication protocol originally developed for instant messaging but increasingly used in Machine-to-Machine (M2M) and Internet of Things (IoT) applications. Standardized by the IETF as RFC 6120, XMPP enables real-time, extensible, and interoperable message exchange across distributed networks[24].

a) Fundamentals and Working Principle: XMPP uses a client-server architecture where clients connect to an XMPP server to exchange XML-formatted messages, presence information, and structured data. The protocol supports both direct client-to-client (C2C) and client-to-server (C2S) communication. XMPP's extensibility is achieved through "XMPP Extension Protocols" (XEPs), which allow custom features and functions to be added as needed.

b) Key Features:

- **Real-Time Messaging:** Supports low-latency, asynchronous message exchange for real-time M2M interactions.
- **Extensibility:** Flexible protocol architecture enables integration of new features using standardized XEPs.
- **Presence Information:** Devices can advertise and subscribe to presence states, enabling context-aware applications.
- **Decentralized Architecture:** Federated network structure allows communication across different servers and domains.
- **Security and Authentication:** Built-in support for TLS encryption and SASL authentication mechanisms.

c) XMPP in M2M and IoT Applications: XMPP is used in a variety of M2M and IoT scenarios, such as device control, remote monitoring, data collection, and collaborative applications. Its support for publish/subscribe messaging (via XEP-0060) and group communication enables scalable and event-driven architectures. XMPP is also well-suited for scenarios requiring interoperability and integration with other systems.

d) Security Considerations: XMPP provides robust security features, including end-to-end encryption, transport layer security (TLS), and strong authentication (SASL). These mechanisms help protect data integrity and privacy in sensitive M2M environments.

e) Conclusion: XMPP's extensible, real-time, and interoperable nature makes it a compelling choice for a wide range of M2M and IoT applications. Its federated architecture, support for presence, and strong security capabilities enable reliable and scalable communication across distributed devices and networks.

D. AMQP Protocol in M2M Communication

The **Advanced Message Queuing Protocol (AMQP)** is an open standard application layer protocol for message-oriented middleware, designed to enable reliable, secure, and interoperable messaging across distributed systems. Standardized by OASIS and ISO/IEC (ISO/IEC 19464:2014), AMQP is widely adopted in enterprise systems, cloud platforms, and increasingly in Machine-to-Machine (M2M) and Internet of Things (IoT) applications[25].

a) Fundamentals and Working Principle: AMQP employs a message broker architecture that decouples message producers and consumers. The protocol defines standardized components such as *exchanges*, *queues*, and *bindings* to manage message routing and delivery. AMQP supports various messaging patterns, including point-to-point, publish/subscribe, and request/response, over reliable, persistent connections—typically using TCP.

b) Key Features:

- **Reliable Messaging:** Guarantees message delivery with mechanisms for acknowledgments, persistence, and transactions.
- **Flexible Messaging Patterns:** Supports point-to-point, publish/subscribe, and topic-based communication.
- **Interoperability:** Vendor-neutral and platform-independent, enabling integration across heterogeneous systems.
- **Security:** Provides support for authentication, encryption (TLS/SSL), and access control.
- **Message Routing:** Sophisticated routing through exchanges and bindings allows dynamic, scalable message distribution.

c) AMQP in M2M and IoT Applications: AMQP is employed in scenarios where high reliability, scalability, and interoperability are required. It is suitable for industrial automation, smart grids, financial systems, and other mission-critical M2M/IoT environments. Its broker-based architecture makes it ideal for aggregating and distributing sensor data, command messages, and alerts in distributed networks.

d) Security Considerations: AMQP ensures secure messaging by supporting TLS/SSL encryption, strong authentication, and access controls. These features are essential for protecting sensitive data in enterprise-grade M2M and IoT deployments.

e) Conclusion: AMQP's robust, flexible, and interoperable messaging capabilities make it a strong candidate for M2M and IoT applications that demand reliability and scalability. Its comprehensive feature set, support for complex messaging patterns, and strong security ensure dependable operation in diverse and distributed communication environments.

E. MQTT Protocol in M2M Communication

The **Message Queuing Telemetry Transport (MQTT)** protocol is a lightweight, publish-subscribe messaging protocol designed specifically for low-bandwidth, high-latency, or unreliable networks, making it ideally suited for Machine-to-Machine (M2M) and Internet of Things (IoT) applications. MQTT was originally developed by Andy Stanford-Clark and Arlen Nipper in 1999 for monitoring oil pipelines over satellite links[26].

a) Fundamentals and Working Principle: MQTT operates on a client-server model where the *broker* acts as the central communication point. Devices, referred to as *clients*, publish messages to specific *topics* on the broker, and clients interested in those topics subscribe to receive messages. The broker is responsible for efficiently distributing messages to all subscribers. MQTT uses the TCP protocol for reliable data delivery, and its lightweight header minimizes overhead.

b) Key Features:

- **Publish/Subscribe Model:** Enables scalable and flexible message distribution, decoupling data producers and consumers.
- **Low Bandwidth Consumption:** Compact header and payload make MQTT highly efficient for resource-constrained devices.
- **Quality of Service (QoS) Levels:** Offers three QoS levels (0: at most once, 1: at least once, 2: exactly once) to balance reliability and performance.
- **Retained Messages and Last Will:** Supports retained messages for new subscribers and last will testament for abnormal client disconnections.
- **Lightweight and Simple:** Simple protocol structure and small code footprint suitable for embedded systems.

c) MQTT in M2M and IoT Applications: MQTT is widely used in scenarios where devices require real-time communication, low power consumption, and efficient use of network resources. Typical use cases include remote sensing, industrial automation, smart homes, telemetry, and mobile applications. The protocol's publish/subscribe model enables

one-to-many communication, making it suitable for scalable M2M networks and group notifications.

d) Security Considerations: Although the basic MQTT protocol does not mandate security features, it can be secured using TLS/SSL for encrypted communication and username/password authentication for client verification. These security mechanisms are crucial for protecting sensitive data in M2M applications.

e) Conclusion: MQTT's lightweight nature, flexible architecture, and reliable message delivery make it a preferred choice for many M2M and IoT scenarios, especially where device resources are limited, and network conditions are challenging. Its ability to decouple publishers and subscribers enhances scalability and interoperability in diverse M2M ecosystems.

F. CoAP Protocol in M2M Communication

The **Constrained Application Protocol (CoAP)** is a specialized web transfer protocol designed by the IETF for constrained devices and networks, as typically found in Machine-to-Machine (M2M) and Internet of Things (IoT) applications. Standardized as RFC 7252, CoAP is optimized for use in resource-constrained environments, providing a simple, lightweight, and efficient protocol for reliable communication[27].

a) Fundamentals and Working Principle: CoAP is based on the REST architecture, similar to HTTP, and supports methods such as GET, POST, PUT, and DELETE for resource manipulation. However, unlike HTTP, CoAP runs over UDP instead of TCP, enabling reduced overhead, low latency, and better suitability for devices with limited processing power and memory. CoAP supports both synchronous request/response and asynchronous communication patterns, making it versatile for diverse M2M use cases.

b) Key Features:

- **Lightweight Design:** Compact binary headers and messages minimize network and device resource usage.
- **UDP-Based Communication:** Provides lower latency and better efficiency for lossy or unreliable networks.
- **RESTful Architecture:** Enables interoperability with HTTP-based web systems through proxies and gateways.
- **Asynchronous Messaging:** Supports observe mode and resource discovery for event-driven M2M applications.
- **Built-in Reliability:** Implements confirmable and non-confirmable message types for flexible reliability.

c) CoAP in M2M and IoT Applications: CoAP is widely used in applications where devices are highly constrained and operate over unreliable or low-power networks. Typical scenarios include environmental monitoring, smart home automation, industrial control, and remote sensing. Its compatibility with HTTP and REST allows seamless integration with existing web services and cloud platforms.

d) Security Considerations: CoAP employs Datagram Transport Layer Security (DTLS) to provide encryption, authentication, and integrity for data transmitted over UDP. This is crucial for safeguarding sensitive data and ensuring secure communication in M2M and IoT deployments.

e) Conclusion: CoAP's lightweight nature, RESTful paradigm, and efficient UDP-based communication make it particularly suitable for resource-constrained M2M and IoT environments. Its support for both synchronous and asynchronous operations, along with robust security features, positions CoAP as a leading protocol for next-generation M2M communication networks.

G. LPWAN Protocols in M2M Communication

Low Power Wide Area Network (LPWAN) protocols represent a class of wireless communication technologies specifically designed to enable long-range connectivity with low power consumption for resource-constrained devices. LPWAN protocols have become crucial for large-scale Machine-to-Machine (M2M) and Internet of Things (IoT) deployments, particularly where devices are geographically dispersed and required to operate for years on battery power[29].

a) Fundamentals and Working Principle: LPWAN technologies operate in both licensed and unlicensed spectrum bands and are optimized for low data rate transmissions over long distances, often exceeding several kilometers. These protocols typically use simple modulation schemes and adaptive data rates to balance range, reliability, and power efficiency. LPWAN devices periodically transmit small packets of data, making them ideal for applications like remote sensing, asset tracking, and environmental monitoring.

b) Key Features:

- **Long-Range Communication:** Supports connectivity over several kilometers, ideal for rural, industrial, or large urban deployments.
- **Low Power Consumption:** Enables multi-year operation on battery-powered devices.
- **Low Data Rate:** Optimized for infrequent, small payload transmissions, typically in the range of tens to hundreds of bytes.
- **Scalability:** Supports a massive number of connected devices within a single network.
- **Cost-Effectiveness:** Utilizes simple hardware and unlicensed spectrum to minimize deployment and operational costs.

c) Popular LPWAN Protocols: Several LPWAN technologies are widely adopted in M2M and IoT scenarios:

- **LoRaWAN:** Utilizes Chirp Spread Spectrum (CSS) modulation in unlicensed ISM bands; features star-of-stars network topology and adaptive data rates.
- **Sigfox:** Employs ultra-narrowband (UNB) modulation for extremely low power, long-range, and simple connectivity.
- **NB-IoT (Narrowband IoT):** A 3GPP cellular-based LPWAN technology operating in licensed spectrum,

supporting secure, reliable, and massive IoT connectivity.

- **LTE-M:** Another 3GPP standard, offering higher data rates and mobility support compared to NB-IoT, with similar low power features.

d) LPWAN in M2M and IoT Applications: LPWAN protocols are extensively used in applications such as smart metering, agriculture, logistics, environmental monitoring, city infrastructure, and industrial automation. Their ability to cover wide areas with minimal infrastructure and power usage makes them foundational technologies for scalable M2M systems.

e) Security Considerations: Security features in LPWAN protocols vary, but most include device authentication, encryption (e.g., AES in LoRaWAN), and secure key management to protect data confidentiality and integrity over wide areas.

f) Conclusion: LPWAN protocols address the fundamental challenges of long-range, low-power, and low-cost connectivity for large-scale M2M and IoT deployments. Their scalability, energy efficiency, and robustness position them as essential components in the rapidly expanding IoT ecosystem.

H. DDS

The **Data Distribution Service (DDS)** is an advanced, real-time, publish-subscribe communication protocol standardized by the Object Management Group (OMG). Designed for scalable, high-performance, and low-latency data exchange, DDS is particularly suited for mission-critical Machine-to-Machine (M2M) and Internet of Things (IoT) applications requiring deterministic communication[**?**].

a) Fundamentals and Working Principle: DDS utilizes a decentralized, broker-less architecture where nodes, known as *publishers* and *subscribers*, communicate directly through a shared data space. DDS is based on the concept of *Topics*, which describe the type and structure of data to be exchanged. It supports Quality of Service (QoS) policies that allow fine-grained control over data delivery guarantees, reliability, latency, and resource usage. DDS operates over standard transport protocols such as UDP and TCP.

b) Key Features:

- **Real-Time Publish/Subscribe:** Supports efficient, scalable, and asynchronous message distribution for real-time applications.
- **Quality of Service (QoS):** Offers extensive QoS policies for reliability, durability, deadline, and latency control.
- **Decentralized Architecture:** Eliminates the need for a central broker, increasing fault tolerance and scalability.
- **Automatic Discovery:** Dynamically discovers new publishers and subscribers without manual configuration.
- **Interoperability:** Standardized by OMG, allowing for vendor-neutral and cross-platform implementation.

c) DDS in M2M and IoT Applications: DDS is widely used in safety-critical and time-sensitive domains such as industrial automation, autonomous vehicles, robotics, aerospace, and healthcare. Its ability to provide reliable, deterministic, and high-throughput data delivery makes it ideal for distributed M2M systems requiring robust communication and strict timing constraints.

d) Security Considerations: DDS includes a security model defined by OMG, which supports authentication, access control, encryption, and logging to ensure data confidentiality, integrity, and auditability in critical environments.

e) Conclusion: DDS's rich QoS controls, real-time capabilities, and decentralized design make it a powerful protocol for demanding M2M and IoT applications. Its suitability for mission-critical systems with stringent requirements distinguishes it from many other M2M communication protocols.

I. LwM2M

The **Lightweight Machine to Machine (LwM2M)** protocol is an application layer protocol developed by the Open Mobile Alliance (OMA) specifically for device management and telemetry in resource-constrained M2M and IoT environments. LwM2M is designed to provide secure, efficient, and interoperable communication for remote device provisioning, monitoring, firmware updates, and sensor data reporting[30].

a) Fundamentals and Working Principle: LwM2M operates using a client-server architecture, where the *LwM2M Client* resides on the end device and the *LwM2M Server* performs management and monitoring tasks. The protocol is built atop the CoAP (Constrained Application Protocol) for lightweight messaging and UDP transport, benefiting from CoAP's low overhead and support for constrained networks. Device resources and operations are organized as objects and instances, described using a standardized data model.

b) Key Features:

- **Efficient Device Management:** Provides remote provisioning, configuration, monitoring, firmware updates, and fault management.
- **Lightweight Messaging:** Utilizes CoAP for communication, ensuring low bandwidth and minimal power consumption.
- **Security:** Implements DTLS for secure transport, supporting device authentication and data confidentiality.
- **Standardized Object Model:** Defines reusable data objects for sensors, actuators, connectivity, and device information.
- **Scalability:** Supports large-scale IoT/M2M deployments, including support for over-the-air (OTA) operations.

c) LwM2M in M2M and IoT Applications: LwM2M is widely adopted in device management for utilities (smart meters), asset tracking, smart cities, remote industrial monitoring, and other large-scale IoT/M2M environments. Its

efficient protocol stack and standardized object model enable interoperability across devices from different manufacturers.

d) Security Considerations: Security in LwM2M is provided through DTLS (Datagram Transport Layer Security), which ensures encrypted, authenticated, and tamper-resistant communication between clients and servers. The protocol also supports bootstrap mechanisms for secure initial device configuration.

e) Conclusion: LwM2M offers a robust, efficient, and secure protocol framework for managing and monitoring constrained devices in M2M and IoT deployments. Its integration with CoAP, lightweight design, and strong device management capabilities make it a preferred solution for large-scale, resource-limited, and heterogeneous environments.

V. Comparative Analysis of M2M Protocols

Table **??** presents a comparative analysis of the major M2M communication protocols based on parameters such as scalability, bandwidth usage, security, latency, and suitability for different applications.

A. Communication Model Comparison

The communication model is a fundamental aspect of each M2M protocol, determining how devices interact, exchange data, and coordinate activities within the network. Communication models directly impact scalability, interoperability, and system complexity. The primary models observed in M2M protocols include client-server, publish/subscribe, peer-to-peer, and hybrid approaches.

Table II summarizes the communication models adopted by major M2M protocols.

Different communication models offer distinct benefits:

- **Client-Server:** Simplicity, wide support, and suitability for synchronous requests.
- **Publish/Subscribe:** Decouples producers and consumers, enabling scalable and flexible many-to-many communication.
- **Peer-to-Peer / Decentralized:** Enhanced robustness, reduced single points of failure, and real-time responsiveness.
- **Hybrid/Advanced:** Protocols like CoAP and XMPP offer both synchronous and asynchronous interaction modes.

Selecting the appropriate communication model is essential for matching protocol capabilities to application requirements, ensuring efficient and reliable M2M interactions.

B. Performance Metrics

A comprehensive comparative analysis of M2M communication protocols requires the evaluation of key performance metrics that directly impact the efficiency and effectiveness of M2M systems. The primary metrics considered are as follows:

1) **Latency:** Latency refers to the time delay between the transmission and reception of a message. Low latency is critical for real-time or mission-critical applications such as industrial automation and healthcare monitoring. Protocol overhead, transport mechanisms, and network topology significantly influence overall latency.
2) **Scalability:** Scalability denotes the protocol's ability to support an increasing number of devices, users, or data flows without a loss in performance. Protocols with lightweight headers, decentralized architectures, and efficient resource management are generally more scalable and better suited for large-scale IoT or smart city deployments.
3) **Security:** Security assesses how well a protocol protects data confidentiality, integrity, and authenticity during transmission. This includes support for encryption, authentication, authorization, and resilience to common attacks. The level of security offered is a key determinant for protocol selection in applications handling sensitive or private information.
4) **Reliability:** Reliability measures the protocol's ability to deliver messages accurately and consistently, even under adverse network conditions. Features such as message acknowledgments, retransmissions, and quality of service (QoS) levels are crucial for maintaining reliable communication, particularly in critical infrastructure and industrial environments.
5) **Energy Efficiency:** Energy efficiency reflects the protocol's impact on device power consumption, which is especially important for battery-powered or resource-constrained M2M nodes. Protocols that minimize message overhead, support sleep modes, and optimize transmission intervals contribute to prolonged device lifetime and reduced operational costs.

These metrics provide a standardized basis for comparing the suitability and performance of different M2M protocols across various application scenarios.

C. Protocol Selection Criteria

Selecting an appropriate M2M communication protocol is critical for ensuring optimal system performance, reliability, and scalability in diverse IoT and M2M applications. The following criteria are commonly considered in comparative analysis and practical deployment:

- **Application Requirements:** Protocol selection should align with the specific use case, such as telemetry, real-time control, device management, or bulk data transfer.
- **Network Topology and Scale:** The architecture (e.g., client-server, publish/subscribe, peer-to-peer) and scalability requirements influence the choice of protocol, especially for large-scale or distributed deployments.
- **Resource Constraints:** Protocols must be matched to the processing power, memory, and battery life of the end devices. Lightweight protocols are preferred for resource-constrained environments.

TABLE I: Deep Comparison of M2M Communication Protocols Including LwM2M

Protocol	Est. Year / Org.	Transport Layer	Architecture / Model	Security	Device Management	QoS / Reliability	Scalability	Typical Applications
HTTP[21]	1991 / IETF	TCP	Client-Server (REST)	TLS/SSL (HTTPS)	No native support	Low (stateless)	High	Web APIs, cloud, device data
WebSocket[23]	2011 / IETF	TCP	Full-duplex, Persistent	TLS/SSL (wss)	No native support	TCP reliability, app-level	High	Real-time UI, dashboards, games
XMPP[24]	1999 / IETF	TCP	Client-Server, Pub/Sub, Federation	TLS, SASL, end-to-end (OMEMO)	Limited (extensions)	Reliable, extensible (XEPs)	High	Messaging, real-time, device alerting
AMQP[25]	2003 / OASIS	TCP	Broker, Queue, Pub/Sub	TLS/SSL, access ctrl	Limited (management APIs)	High (delivery, persistence)	Very high	Enterprise IoT, cloud, finance
MQTT[26]	1999 / IBM	TCP	Broker-based Pub/Sub	TLS/SSL, user/pass	Limited (extensions)	3 QoS levels (0,1,2)	Very high	IoT telemetry, sensing, mobile
CoAP[27]	2014 / IETF	UDP	Client-Server, Proxy, Multicast	DTLS	Basic via objects	Confirmable / Non-confirmable	High	Sensor nets, constrained IoT
DDS[28]	2004 / OMG	TCP/UDP	Decentralized Pub/Sub, P2P	DDS Security (OMG), encryption, access	Limited	Extensive QoS policies	Very high	Critical systems, automation, robotics
LPWAN (LoRaWAN, NB-IoT)[29]	2013 / LoRa Alliance, 3GPP	MAC, Radio	Star, Mesh, PtM	AES, SIM, DTLS	No native support	Low–Mod (conf./unconf.)	Extremely high	Wide-area sensor nets, metering
LwM2M[30]	2014 / OMA	UDP (via CoAP), SMS, TCP	Client-Server (Mgmt. and Telemetry)	DTLS (UDP), TLS (TCP)	Full device mgmt., OTA, provisioning	CoAP confirmable, async, notification	Very high	Smart metering, asset tracking, remote mgmt.

TABLE II: Communication Model Comparison of Major M2M Protocols

Protocol	Communication Model(s)
HTTP	Client-Server (Request/Response, RESTful)
MQTT	Publish/Subscribe (Broker-mediated)
CoAP	Client-Server (Request/Response), Proxy, Multicast (Observe for notifications)
WebSocket	Full-duplex, Persistent, Bidirectional (Client-initiated handshake)
XMPP	Client-Server, Publish/Subscribe, Federated (extensions via XEPs)
AMQP	Publish/Subscribe, Message Queue (Broker-mediated), Point-to-Point
DDS	Data-centric Publish/Subscribe, Decentralized Peer-to-Peer
LPWAN	Star (central gateway to device), Point-to-Multipoint, sometimes Mesh
LwM2M	Client-Server (Device management and telemetry), supports asynchronous notifications via Observe

- **Quality of Service (QoS):** Requirements for reliability, delivery guarantees, latency, and data integrity determine whether advanced QoS features (e.g., message acknowledgments, prioritization) are needed.
- **Security and Privacy:** The protocol should provide adequate mechanisms for authentication, encryption, and access control, especially for sensitive or mission-critical applications.
- **Interoperability:** Protocol compatibility with existing systems, platforms, and standards is essential for seamless integration and future scalability.
- **Network Environment:** The underlying network type (wired, wireless, cellular, LPWAN) and its constraints (e.g., bandwidth, latency, reliability) affect protocol suitability.
- **Device Management Support:** Native or extensible support for device provisioning, monitoring, and firmware updates can simplify large-scale IoT deployments.
- **Ecosystem and Maturity:** The availability of development tools, community support, and proven deployments are practical considerations that can influence protocol adoption.

Careful evaluation of these criteria helps ensure that the chosen protocol meets both current and future needs, optimizes system resources, and aligns with organizational and technical objectives.

TABLE III: Protocol-wise Performance Metric Scores for M2M Communication Protocols

Protocol	Latency	Scalability	Security	Reliability	Energy Efficiency
HTTP[21]	↓	↑	↑	∼	↓
WebSocket[23]	↑	↑	∼	↑	∼
XMPP[24]	∼	↑	↑	↑	∼
AMQP[25]	∼	↑	↑	↑	∼
MQTT[26]	↑	↑	∼	↑	↑
CoAP[27]	↑	↑	∼	∼	↑
DDS[28]	↑	↑	↑	↑	∼
LPWAN[29]	↓	↑	∼	∼	↑
LwM2M[30]	∼	↑	↑	↑	↑

VI. Security and Privacy Issues

Security is a critical aspect of M2M communication. Protocols implement security features such as encryption (TLS/DTLS), authentication, and access control to protect against threats like eavesdropping, unauthorized access, and data breaches. However, resource constraints in devices and heterogeneity of networks pose significant security challenges, necessitating lightweight and robust security solutions.

VII. Application Scenarios of M2M Communication

Machine-to-Machine (M2M) communication protocols play a critical role in various domains including Smart Home & Building Automation, Industrial Automation, Healthcare, Transportation and Logistics, and Agriculture. In Smart Home and Building Automation, protocols like **MQTT** and **CoAP** are extensively utilized for their lightweight architecture, ideal for integrating numerous constrained devices. Additionally, **HTTP** and **WebSocket** are commonly used to facilitate interactive web-based user interfaces and real-time monitoring of smart appliances. For Industrial Automation, the reliability, scalability, and real-time capabilities of **DDS** make it suitable for mission-critical environments, whereas **AMQP** supports guaranteed message delivery in industrial control systems. In Healthcare applications, **MQTT**, **CoAP**, and **LwM2M** protocols enable secure and efficient data exchange between medical devices, wearable sensors, and cloud-based healthcare systems, providing remote patient monitoring and telemedicine functionalities. Transportation and Logistics rely heavily on **XMPP**, **MQTT**, and **AMQP** to ensure real-time tracking, fleet management, and instant communication between distributed transportation nodes. Furthermore, the wide-area coverage, energy efficiency, and long-range capabilities of **LPWAN** protocols facilitate tracking and management of logistic assets across expansive geographical regions. In Agriculture, **LPWAN** technologies support precision farming through long-range connectivity, while lightweight protocols such as **MQTT**, **CoAP**, and **LwM2M** enable remote environmental monitoring and control of resource-constrained agricultural sensors and actuators. Collectively, these M2M protocols form a comprehensive communication infrastructure that effectively addresses the unique challenges and requirements of each application domain.

VIII. Challenges and Future Directions

Despite significant advancements, M2M communication faces challenges including protocol standardization, interoperability, scalability, and energy efficiency. The integration of artificial intelligence and machine learning for adaptive communication, enhanced security mechanisms, and the emergence of new wireless technologies are shaping the future of M2M systems. Further research is needed to address the open issues and optimize protocol performance for diverse IoT applications.

A. Interoperability Issues

Despite the proliferation of M2M communication protocols, achieving seamless interoperability among heterogeneous devices and platforms remains a significant challenge. Diverse protocol stacks, proprietary implementations, and varying data models often result in compatibility issues, limiting integration across different vendors and ecosystems. Middleware solutions, protocol gateways, and the adoption of standardized data models are being explored to bridge these gaps, but comprehensive interoperability at scale is yet to be realized.

B. Security and Privacy Concerns

Security and privacy are critical concerns in M2M communications, especially as devices transmit sensitive data across public and private networks. Many protocols were not initially designed with strong security in mind, leading to vulnerabilities such as weak authentication, lack of encryption, and susceptibility to attacks like eavesdropping and spoofing. Enhanced security frameworks, device authentication mechanisms, end-to-end encryption, and privacy-preserving data management are ongoing research and development areas.

C. Scalability Challenges

The exponential growth of connected devices poses substantial scalability challenges for M2M protocols. Networks must handle massive numbers of nodes, frequent data exchanges, and dynamic topologies, all while maintaining performance, reliability, and low latency. Efficient resource allocation, hierarchical architectures, lightweight messaging, and adaptive protocols are being investigated to address these scalability concerns, especially in large-scale deployments like smart cities and industrial IoT.

D. Standardization Efforts

Standardization is vital for ensuring interoperability, security, and long-term sustainability of M2M systems. Numerous organizations, including IETF, OMA, ETSI, and 3GPP, are actively developing and harmonizing protocol standards. However, the coexistence of multiple standards and ongoing technological evolution create fragmentation and slow down widespread adoption. Collaborative efforts and the convergence of protocols are necessary to streamline standardization and foster ecosystem growth.

E. Research Gaps and Emerging Trends

Despite significant progress, several research gaps remain in areas such as energy-efficient protocol design, real-time communication for critical applications, semantic interoperability, and intelligent resource management. Emerging trends include the integration of artificial intelligence for autonomous network management, the application of blockchain for secure M2M transactions, and the evolution of 5G and beyond for ultra-reliable low-latency communications. Continuous research and innovation are essential to address existing limitations and unlock the full potential of M2M communications in the future[17].

ACKNOWLEDGMENT

The author(s) would like to express their sincere gratitude to ChatGPT (OpenAI) for its valuable assistance in literature review, technical clarification, and manuscript preparation during the development of this article. The interactive suggestions and information provided by ChatGPT significantly contributed to improving the quality and clarity of the work.

IX. CONCLUSION

This paper presented a comprehensive review of the major M2M communication protocols, discussing their architecture, strengths, weaknesses, and application suitability. The selection of an appropriate protocol depends on specific application requirements, including scalability, security, and device capabilities. Future work will focus on overcoming current challenges and exploring emerging solutions for robust, intelligent, and scalable M2M communication.

REFERENCES

[1] Gunjal, P.R., Jondhale, S.R., Lloret Mauri, J., & Agrawal, K. (2024). Internet of Things: Theory to Practice (1st ed.). CRC Press. https://doi.org/10.1201/9781003282945

[2] Patil, S., Bhende, M., & Sharma, S. (2025). Optimising IoT Networks: Energy-Efficient Resource Management through Machine Learning (1st ed.). Chapman and Hall/CRC. https://doi.org/10.1201/9781003606062

[3] Tolani, M., Balodi, A., Bajpai, A., Jain, V., & Kovintavewat, P. (Eds.). (2025). Security and Privacy Issues for IoT and WSN-based Real-time Applications (1st ed.). Chapman and Hall/CRC. https://doi.org/10.1201/9781003491910

[4] Geng, Hwaiyu, ed. Internet of things and data analytics handbook. John Wiley & Sons, 2017.

[5] Debruyne, C., Panetto, H., Weichhart, G., Bollen, P., Ciuciu, I., Vidal, M., & Meersman, R. (2009). On the move to meaningful Internet systems. In OTM 2017 Workshops-Confederated International Workshops, EI2N, FBM, ICSP, Meta4eS, OTMA 2017 and ODBASE Posters 2017, Rhodes, Greece, October 23-28, 2017, Revised Selected Papers (Vol. 10697).

[6] Nsoh B, Katimbo A, Guo H, Heeren DM, Nakabuye HN, Qiao X, Ge Y, Rudnick DR, Wanyama J, Bwambale E, et al. Internet of Things-Based Automated Solutions Utilizing Machine Learning for Smart and Real-Time Irrigation Management: A Review. Sensors. 2024; 24(23):7480. https://doi.org/10.3390/s24237480

[7] Escamilla-Ambrosio, P. J., et al. "Distributing computing in the internet of things: cloud, fog and edge computing overview." NEO 2016: Results of the Numerical and Evolutionary Optimization Workshop NEO 2016 and the NEO Cities 2016 Workshop held on September 20-24, 2016 in Tlalnepantla, Mexico. Cham: Springer International Publishing, 2017.

[8] Pielli, Chiara, Daniel Zucchetto, Andrea Zanella, Lorenzo Vangelista, and Michele Zorzi. "Platforms and Protocols for the Internet of Things." EAI Endorsed Transactions on Internet of Things 1, no. 1 (2015): 1-15.

[9] Di Pasquale, A., Becker, J. K. M., Kettner, A. M., & Paolone, M. (2024). Ensuring solution uniqueness in fixed-point-based harmonic power flow analysis with converter-interfaced resources: Ex-post conditions. IEEE Transactions on Smart Grid.

[10] Qureshi, K.N., & Newe, T. (Eds.). (2024). Artificial Intelligence of Things (AIoT): New Standards, Technologies and Communication Systems (1st ed.). CRC Press. https://doi.org/10.1201/9781003430018

[11] Nsoh, Bryan. "Development of Crop2Cloud: An IoT-integrated Platform for Automated Irrigation Scheduling

Using Multi-source Stress Indices." (2024).

[12] Feng, X., Yan, F. & Liu, X. Study of Wireless Communication Technologies on Internet of Things for Precision Agriculture. Wireless Pers Commun 108, 1785–1802 (2019). https://doi.org/10.1007/s11277-019-06496-7

[13] Sinha, S. R., & Park, Y. (2017). Building Network Services. In Building an Effective IoT Ecosystem for Your Business (pp. 49-61). Cham: Springer International Publishing.

[14] Mukhopadhyay, S.C., Suryadevara, N.K. (2014). Internet of Things: Challenges and Opportunities. In: Mukhopadhyay, S. (eds) Internet of Things. Smart Sensors, Measurement and Instrumentation, vol 9. Springer, Cham. https://doi.org/10.1007/978-3-319-04223-7_1

[15] P. Di Marco, C. Fischione, G. Athanasiou and P. -V. Mekikis, "Harmonizing MAC and routing in low power and lossy networks," 2013 IEEE Global Communications Conference (GLOBECOM), Atlanta, GA, USA, 2013, pp. 231-236, https://doi.org/10.1109/GLOCOM.2013.6831076.

[16] Sarkar, C., Das, A. & Jain, R.K. Development of CoAP protocol for communication in mobile robotic systems using IoT technique. Sci Rep 15, 9269 (2025). https://doi.org/10.1038/s41598-024-76713-2

[17] Verma, P. K., Verma, R., Prakash, A., Agrawal, A., Naik, K., Tripathi, R., & Abogharaf, A. (2016). Machine-to-Machine (M2M) communications: A survey. Journal of Network and Computer Applications, 66, 83-105 http://dx.doi.org/10.1016/j.jnca.2016.02.016i

[18] Tightiz, L., & Yang, H. (2020). A Comprehensive Review on IoT Protocols' Features in Smart Grid Communication. Energies, 13(11), 2762. https://doi.org/10.3390/en13112762

[19] Alkwiefi, M. (2024, September). M2M Protocols: An Overview on LwM2M and XMPP Machine-to-Machine Protocols in IoT Context. In 2024 IEEE/ACIS 24th International Conference on Computer and Information Science (ICIS) (pp. 209-215). IEEE. https://doi.org/10.1109/ICIS61260.2024.10778343

[20] Shyam, P. R. (2025). A Survey of Communication Protocols in IoT: MQTT, COAP, and Beyond. https://philarchive.org/archive/RAGASO

[21] Berners-Lee, T., Fielding, R., & Frystyk, H. (1996). Hypertext Transfer Protocol – HTTP/1.0 (RFC 1945). Internet Engineering Task Force (IETF). https://datatracker.ietf.org/doc/html/rfc1945

[22] Rahman, M. H., Reza, M. M., Ferdous, R., Naderuzzaman, Kashem, M. A. (2025). Development of a IoT Based Thermologger for Real-Time Temperature Monitoring. Internet of Things and Cloud Computing, 13(2), 28-37. https://doi.org/10.11648/j.iotcc.20251302.11

[23] Fette, I., & Melnikov, A. (2011). The WebSocket Protocol (RFC 6455). Internet Engineering Task Force (IETF). https://datatracker.ietf.org/doc/html/rfc6455

[24] Saint-Andre, P. (2011). Extensible Messaging and Presence Protocol (XMPP): Core (RFC 6120). Internet Engineering Task Force (IETF). https://datatracker.ietf.org/doc/html/rfc6120

[25] OASIS Advanced Message Queuing Protocol (AMQP) Working Group. (2012). OASIS Advanced Message Queuing Protocol (AMQP) Version 1.0. https://docs.oasis-open.org/amqp/core/v1.0/os/ amqp-core-complete-v1.0-os.pdf

[26] Stanford-Clark, A., & Nipper, A. (2014). MQTT Version 3.1.1. OASIS Standard. https://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html

[27] Shelby, Z., Hartke, K., & Bormann, C. (2014). The Constrained Application Protocol (CoAP) (RFC 7252). Internet Engineering Task Force (IETF). https://datatracker.ietf.org/doc/html/rfc7252

[28] Object Management Group. (2015). Data Distribution Service (DDS) Specification (Version 1.4). https://www.omg.org/spec/DDS/1.4/

[29] Raza, U., Kulkarni, P., & Sooriyabandara, M. (2017). Low Power Wide Area Networks: An Overview. IEEE Communications Surveys & Tutorials, 19(2), 855–873. https://doi.org/10.1109/COMST.2017.2652320

[30] Open Mobile Alliance. (2017). Lightweight Machine to Machine Technical Specification (OMA-TS-LightweightM2M-V1_0-20170208-A). https://www.openmobilealliance.org/release/LightweightM2M/V1_0-20170208-A/OMA-TS-LightweightM2M-V1_0-20170208-A.pdf