Study on Communication Protocols for IoT and Edge Computing

Hoza Violeta Maria

1 Introduction to IoT and Edge Computing

The Internet of Things (IoT) represents a global ecosystem of interconnected physical devices embedded with sensors, software, and communication interfaces that enable them to collect, transmit, and process data. These smart devices enable various applications across multiple sectors, including healthcare, agriculture, space, manufacturing, construction, water, and mining [1].

IoT devices generate vast amounts of data that require efficient processing, leading to the emergence of edge computing as a complementary technology. Edge computing addresses the limitations of cloud-centric architectures by performing data computing and storage near the data source, thereby reducing latency, minimizing bandwidth usage, and improving responsiveness [2]. Edge computing is crucial for time-sensitive applications such as autonomous vehicles, smart grids, and industrial control systems, where very short response times are required and traditional cloud computing-based services cannot satisfy the demand [2].

For IoT and edge computing to function properly, devices require efficient and reliable methods of communication. Many communication protocols have been developed for this purpose. Each protocol has different features, benefits, and limitations, so selecting the right one is crucial.

This report provides an overview of the primary IoT communication protocols, examines the computing continuum paradigm, compares key protocol features, and explains how protocol selection varies across different layers of the distributed computing infrastructure.

2 The Computing Continuum

The modern IoT ecosystem extends beyond simple device-to-cloud architectures, encompassing what is known as the computing continuum. This continuum represents a seamless integration of computational resources spanning from IoT devices at the network edge to powerful cloud data centers, creating an integrated distributed computing infrastructure that optimizes latency, performance, and resource utilization throughout the IoT ecosystem [3].

The computing continuum addresses the evolving requirements of IoT applications by providing a flexible framework where computational tasks can be dynamically allocated based on application needs, resource availability, and performance constraints. This approach recognizes that different types of processing are better suited to different layers of the infrastructure. For instance, time-critical sensor data processing may be performed locally on IoT devices or nearby edge nodes, while complex analytics and machine learning models may be executed in the cloud where greater computational resources are available [3].

The computing continuum typically comprises several interconnected layers, each serving specific purposes in the IoT ecosystem:

- **Device layer:** Consists of sensors and actuators that generate data and may perform basic preprocessing.

- **Edge layer:** Gateways and local servers that provide real-time processing and reduce latency and bandwidth usage by handling data close to the source.
- **Cloud layer:** Large-scale data centers offering scalable storage, complex analytics, and orchestration for broader IoT deployments.

The effective implementation of the computing continuum relies heavily on selecting appropriate communication protocols for each layer and interaction within the distributed infrastructure. Protocol selection must consider the specific requirements of device-to-edge, edge-to-edge, and edge-to-cloud communications, each with distinct constraints regarding power consumption, latency, bandwidth, and reliability.

3 Classification of IoT Communication Protocols

IoT communication protocols can be classified based on multiple criteria, including by communication medium (wired vs. wireless), network range, and OSI model layer.

3.1 Wired vs. Wireless Communication

Wired protocols, such as **Ethernet** (governed by the IEEE 802.3 standard) and **Power Line Communication** (PLC), offer reliable high-bandwidth connections suitable for stable environments, such as industrial settings [4].

Wireless protocols dominate IoT implementations and can be further categorized by range:

- **Short-range**: Bluetooth Low Energy (BLE), Zigbee, Z-Wave - low power, ideal for personal area networks (PANs), wearables, and smart homes [5]
- **Medium-range**: Wi-Fi (IEEE 802.11), Wi-Fi HaLow - facilitate long-distance connectivity, provide high bandwidth but consume more power, making it suitable for edge gateways [4]
- **Long-range**: LoRaWAN, Sigfox, NB-IoT - enable connectivity across kilometers [4]

3.2 OSI Layer Classification

From a networking perspective, IoT protocols operate across different layers of the communication stack.

At the **physical and data link layers**, technologies such as IEEE 802.15.4 (used in Zigbee [5,6] and 6LoWPAN [4, 6]) define how devices physically connect and transmit raw data.

Network layer protocols such as 6LoWPAN (IPv6 over Low-Power Wireless Personal Area Networks) adapt IPv6 for low-power devices with limited processing capabilities, enabling seamless integration with the internet [6].

Transport layer options range from traditional TCP to lightweight UDP variants optimized for constrained devices [6, 7] .

The **application layer** features specialized protocols, such as MQTT, XMPP, AMQP, and CoAP, that define how devices structure, interpret, and exchange messages [7].

4 Analysis of Key IoT Protocols

4.1 MQTT (Message Queuing Telemetry Transport)

MQTT is a lightweight, publish/subscribe (pub/sub) messaging protocol optimized for low-bandwidth and constrained environments [8, 9]. Its architecture consists of clients (publishers, e.g., sensors, and subscribers, e.g., edge analytics systems) and a central broker (i.e., a server) that manages message distribution between publishers (devices sending data) and subscribers (devices requesting data) [9]. MQTT supports three Quality of Service (QoS) levels to manage message delivery guarantees:

- **QoS 0 (At most once)**: the message is sent at most once, and the delivery of the message is not guaranteed [9]
- **QoS 1 (At least once)**: guaranteed delivery but may result in duplicates [7]
- **QoS 2 (Exactly once)**: ensures message delivery without duplication, but with higher overhead [1, 7].

MQTT is widely adopted in healthcare, smart homes, and telemetry systems due to its simplicity and minimal overhead [1, 4]. It operates over TCP/IP and supports TLS (Transport Layer Security) for secure communication, making it ideal for edge computing where low latency and security are critical [10].

One of MQTT's main strengths in edge environments is its small packet size (as low as 2 bytes), which helps reduce bandwidth consumption and extend battery life in constrained devices [7]. Despite these benefits, MQTT can face performance issues under high message volume or large payload sizes; for instance, under experimental conditions involving 1000 messages, MQTT showed higher latency and lower reliability compared to CoAP and DDS [7]. Nonetheless, its widespread platform support, ease of implementation, and efficient pub/sub model make MQTT a default choice in many edge-to-cloud scenarios.

4.2 CoAP (Constrained Application Protocol)

CoAP is a specialized web transfer protocol designed for constrained devices, that operates over UDP instead of TCP [7, 10]. It mirrors RESTful architecture, using GET, POST, PUT, and DELETE methods like HTTP, but with reduced overhead, making it ideal for IoT applications.

CoAP is preferred in smart homes and M2M communications due to its low latency and compatibility with web standards [1]. It also supports DTLS (Datagram Transport Layer Security) for encryption, ensuring secure communication in edge environments [8]. Its decentralized and UDP-based nature makes it effective in edge networks with tight resource constraints. CoAP's lightweight design allows it to operate efficiently even in networks with frequent disruptions or variable connectivity. In edge computing scenarios, it outperforms MQTT in energy efficiency and bandwidth use when handling small, frequent messages [7]. However, its reliance on UDP may lead to packet loss in unreliable networks unless complemented with robust retransmission mechanisms. Additionally, CoAP can face challenges when deployed behind NATs or firewalls and lacks native support for multicast over DTLS [7]. These factors must be weighed when selecting it for edge applications.

4.3 AMQP (Advanced Message Queuing Protocol)

AMQP is a wire level protocol [10] developed for message-oriented middleware and enterprise systems [1]. It supports features such as reliable delivery, transaction management, and message queuing with routing. Unlike MQTT, AMQP ensures message delivery with acknowledgment

mechanisms and is ideal for applications requiring high reliability. However, AMQP has significant overhead and requires more resources, making it less suitable for low-power or memory-constrained devices. It typically operates over TCP and supports encrypted connections via TLS [11] .

4.4 DDS (Data Distribution Service)

DDS is a data-centric, publish/subscribe protocol suitable for IoT applications that require communication in real-time [10]. It eliminates the need for a centralized broker by enabling direct peer-to-peer communication among nodes, reducing latency. DDS supports Quality of Service (QoS) policies for timing, reliability, and resource management. This protocol is ideal for mission-critical edge applications, such as autonomous vehicles and industrial control systems, that require high throughput and real-time guarantees.

4.5 XMPP (Extensible Messaging and Presence Protocol)

XMPP is an XML-based protocol originally designed for instant messaging but adapted for IoT to provide real-time communication [7]. It operates over TCP and supports extensions (XEPs) to enhance its functionality. XMPP is advantageous for applications needing interactive communication or federation between servers [7] (it is better suited for messaging-intensive applications rather than typical edge IoT sensor data exchange). Its open standard nature ensures flexibility and interoperability. However, XML encoding and persistent connections increase bandwidth and memory demands, making it less efficient for constrained IoT devices [8].

Table 1: Comparison of IoT Application Layer Protocols

Feature	**MQTT**	**CoAP**	**AMQP**	**DDS**	**XMPP**
Transport Protocol	TCP	UDP	TCP	UDP or TCP	TCP
Architecture	Client/Broker/Subscriber	Client/Server	Client/Broker/Server/Subscriber	Client/Subscriber	Client/Server/Subscriber
Transmission Model	Publisher/Subscriber	Request/Response	Request/Response	Publisher/Subscriber	Publisher/Subscriber
Minimum Header Size	2 bytes	4 bytes	8 bytes	–	–
QoS Support	QoS 0, 1, 2	Confirmable / Non-confirmable	Settle / Unsettle Format	23 QoS Policies	Not specified
Message Size	Up to 256 MB	$\leq$1024 KB	Undefined	Undefined	Undefined
Security	TLS/SSL	DTLS	TLS/SSL, IPSec, SASL	SSL, DTLS	TLS

5 Protocol Selection in the Computing Continuum Context

The computing continuum paradigm significantly influences the selection and deployment of communication protocols across different layers of the IoT infrastructure. Each layer of the continuum has distinct requirements that favor specific protocol characteristics and implementation strategies.

At the device layer, protocols must prioritize energy efficiency and minimal resource consumption. CoAP and lightweight MQTT implementations are particularly well-suited for this layer due to their small overhead and ability to operate on constrained devices. The protocol choice at this level directly impacts device battery life and operational costs [3].

Edge computing nodes in the continuum require protocols that can handle higher message volumes while maintaining low latency for real-time processing. MQTT brokers and DDS implementations are commonly deployed at edge nodes to aggregate data from multiple IoT devices and facilitate local decision-making. The pub/sub model of these protocols enables efficient data distribution to multiple edge applications.

Communication between edge and cloud layers demands protocols that can handle variable network conditions and ensure reliable data transmission over potentially unreliable wide-area networks. AMQP's robust delivery guarantees and MQTT's QoS levels make them suitable for edge-to-cloud communication, where message persistence and delivery confirmation are critical.

6 Conclusion

IoT and edge computing architectures rely on an intricate stack of communication protocols to enable seamless data collection, transmission, and processing. The selection of a communication protocol depends on factors such as power and bandwidth constraints, latency requirements, scalability, and the application context [2, 4].

MQTT and CoAP remain widely adopted in constrained edge devices. DDS, while more complex, is optimal for scalable and high-performance deployments.

Wired protocols, such as Ethernet and PLC, offer stable and high-throughput connections, making them well-suited for industrial and infrastructure-centric applications. On the other hand, wireless protocols such as LoRaWAN, Zigbee, and NB-IoT offer scalable, flexible, and cost-effective connectivity across a wide range of use cases, from smart agriculture to smart cities [4].

Ongoing developments aim to improve interoperability, security, and support for heterogeneous environments. Future trends include hybrid protocol stacks, tighter integration with AI at the edge, and evolution toward 6G-ready IoT frameworks [4].

References

[1] N. Batra and S. Goyal, *IoT Fundamentals with a Practical Approach*. CRC Press, 2025.

[2] W. Yu, F. Liang, X. He, W. G. Hatcher, C. Lu, J. Lin, and X. Yang, "A survey on the edge computing for the internet of things," *IEEE Access*, vol. 6, pp. 6900–6919, 2018. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8123913

[3] A. Al-Dulaimy, M. Jansen, B. Johansson, A. Trivedi, A. Iosup, M. Ashjaei, A. Galletta, D. Kimovski, R. Prodan, K. Tserpes, G. Kousiouris, C. Giannakos, I. Brandic, N. Ali, A. B. Bondi, and A. V. Papadopoulos, "The computing continuum: From iot to the cloud," *Internet of Things*, vol. 27, p. 101272, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2542660524002130

[4] M. Mansour, A. Gamal, A. I. Ahmed, L. A. Said, A. Elbaz, N. Herencsar, and A. Soltan, "Internet of things: A comprehensive overview on protocols, architectures, technologies, simulation tools, and future directions," *Energies*, vol. 16, no. 8, 2023. [Online]. Available: https://www.mdpi.com/1996-1073/16/8/3465

[5] J. M and A. Prakash, "A study of communication protocols for internet of things (iot) devices: Review," 01 2021. [Online]. Available: https://www.researchgate.net/publication/354887907_A_Study_of_Communication_Protocols_for_Internet_of_Things_IoT_Devices_Review

[6] Microsoft Azure, "Iot protocols — azure iot reference," 2024, accessed: 2025-07-15. [Online]. Available: https://azure.microsoft.com/en-us/solutions/iot/iot-technology-protocols

[7] E. Al-Masri, K. R. Kalyanam, J. Batts, J. Kim, S. Singh, T. Vo, and C. Yan, "Investigating messaging protocols for the internet of things (iot)," *IEEE Access*, vol. 8, pp. 94 880–94 911, 2020. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9090208

[8] A. Gerodimos, L. Maglaras, M. A. Ferrag, N. Ayres, and I. Kantzavelou, "Iot: Communication protocols and security threats," *Internet of Things and Cyber-Physical Systems*, vol. 3, pp. 1–13, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2667345222000293

[9] D. Soni and A. Makwana, "A survey on mqtt: a protocol of internet of things (iot)," in *International conference on telecommunication, power analysis and computing techniques (ICTPACT-2017)*, vol. 20, no. April, 2017, 2017.

[10] C. Bayılmış, M. A. Ebleme, Ünal Çavuşoğlu, K. Küçük, and A. Sevin, "A survey on communication protocols and performance evaluations for internet of things," *Digital Communications and Networks*, vol. 8, no. 6, pp. 1094–1104, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352864822000347

[11] L. Tightiz and H. Yang, "A comprehensive review on iot protocols' features in smart grid communication," *Energies*, vol. 13, no. 11, 2020. [Online]. Available: https://www.mdpi.com/1996-1073/13/11/2762