Comparative Study of Messaging Protocols Used in Mobile Software Architecture

A State-of-the-art Literature Review

Lukas Gruber
Department of Media and Digital Technologies
University of Applied Sciences St.Pölten
3100 St.Pölten, Austria
it231519@fhstp.ac.at

***Abstract* — Messaging protocols are essential components of modern software architectures, enabling efficient communication between various software systems and devices. This paper presents a comparative study of several prominent messaging protocols, including HTTP, WebSockets, WebTransport, WebRTC, MQTT, AMQP, CoAP, STOMP, Matrix and XMPP. These protocols are evaluated and compared based on key criteria such as performance, flexibility, security, and suitability for different use cases. The results of this study aim to guide software architects and developers in selecting the most appropriate protocol for their specific needs.**

Keywords — SOTA, Messaging Protocols, HTTP/3, WebSockets, WebTransport, WebRTC, MQTT, AMQP, CoAP, STOMP, Matrix, XMPP

I. Introduction / Area of research

The IoT (Internet of Things) is growing both in terms of the number of connected devices as well as in the variety of use cases, e.g. smart phones, cars and automation systems [1]. None of these devices have an unlimited amount of battery life or computing power nor can they guarantee permanent network availability. In addition, safety concerns concerning critical environments such as the medicine or industry sector must be considered [2]. This is where protocols come into play. Message protocols allow us to exchange data effectively and reliably between software services respectively devices.

Since 1984, the ISO/OSI (Open Systems Interconnection Model) has been the standard reference model for describing communication across several technical system levels [3]. It consists of seven successive layers. As the interfaces between these layers are clearly defined, the protocol used within a layer is interchangeable. For the web respectively the internet, however, the TCP/IP reference model based on the OSI model is more decisive [4]. It combines several OSI layers, as shown in Fig. 1. The application layer represents the highest level of abstraction and includes all protocols used for exchanging application data such as HTTP or FTP.

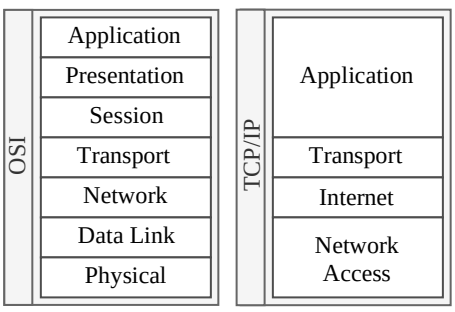

Fig. 1 – Layers of OSI and TCP/IP [3], [4]

This paper focuses solely on protocols belonging to the application layer. Though they depend on protocols that are used on lower layers. Protocols were chosen depending on the research taken for openly available protocols that can be used for message exchange between software services/devices.

In general, the protocols examined can be divided into different communication types. Some can be used for Peer-to-Peer (P2P) communication. Others might be classified as message-oriented middleware (MOM) [5]. MOM means that there is no direct data transfer between two clients. Instead, the communication between distributed systems is handled via a message broker or likewise.

II. Related Work

There are studies that specifically aim to compare IoT protocols at the application layer. The most comprehensive work found is “Investigating Messaging Protocols for the Internet of Things (IoT)” [6]. This paper compares and contrasts the HTTP, MQTT, CoAP, AMQP, XMPP and DDS protocol. No papers were found that would provide a more complete analysis featuring further protocols. Referenced works such as “Choice of Effective Messaging Protocols for IoT Systems: MQTT, CoAP, AMQP and HTTP” [7], take a similar approach for the comparison. Both works collect the protocol properties in a table containing for example the release year, the application purpose and its architecture. However, comparisons that do also include other web protocols such as WebTransport are missing.

III. Methodology

Protocols of interest were determined through browsing for related work starting with the term “messaging protocols”. Sources for the research were IEEE, Google Scholar and Google Search. After a protocol was identified as significant for this paper the associated specification was searched up. Additionally, at least two further trustworthy sources from IEE or Google Scholar were determined per protocol.

IV. comparison analysis

This chapter serves to present the investigated protocols. Mentioned RFC (Requests for Comments) refer to the technical documentations published by the Internet Engineering Task Force (IETF) [8]. The IETF is the premiere standards development organization for the Internet. The international organization for standardization (ISO) on the other hand is a more formal and not internet specific institution that has 169 national standardization bodies as members [9].

A. *HTTP/3 (RFC9114 [10])*

Today, the Hypertext Transfer Protocol (HTTP) is the basis for communication across the internet [11]. It is a stateless protocol based on a request-response model. A client sends a message, and a host/server generates a response message. The first version of HTTP was standardized in 1997, followed by HTTP/2 in 2014. Finally, 2022 the standard of HTTP/3 was introduced. Under the hood HTTP/3 uses QUIC and the UDP protocol on the transport layer and therefore is able to solve line blocking problems that occurred in previous HTTP versions that use TCP for multiplexing [12].

QUIC (Quick UDP Internet Connections), standardized in RFC 9000 [13], was initially developed by Google in 2012 and became a standard in 2021 [14]. While TCP allows reliable ordered and error-checked delivery of data, UDP sends datagrams without establishing a connection. Therefore, it is more lightweight and faster than TCP but at the same time cannot guarantee package delivery or order. QUIC however builds up on top of UDP solving these issues and trying to replace TCP in a faster, more secure and reliable way.

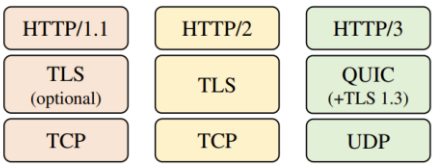

Fig. 2 – HTTP Protocol Stack [11]

With HTTP there are several popular choices available for building API's (Application Programming Interfaces) [15]. These include SOAP, REST, gRPC and GraphQL. According to the Postman state of the API 2023 survey report [16] REST is the most used pattern with 86% of respondents using it.

B. *WebSocket (RFC6455 [17]), WebTransport (RFC draft [18]) & WebRTC (RFC8831 [19])*

WebSockets can be used to establish bi-directional data channels between clients and a server [20]. The WebSocket standard was introduced in 2012. Like HTTP/2 it depends on TCP. Messages are handled via a single ordered reliable stream. This means that messages must be sent and received in order. The consequence is that WebSocket's are a bad choice for latency-sensitive applications.

A kind of successor to WebSockets is the WebTransport protocol [21]. Since 2021 the standard for WebTransport has been under development. The big advantage over WebSockets is the ability to create a bidirectional multiplexed communication channel. Through datagrams unreliable unordered data such as real-time audio or video frames can be sent and received. Additionally, streams can be used to send and receive reliable ordered data.

WebSocket and WebTransport can be used for real-time client-to-server communication. However, in some cases it might be necessary to create such a connection for p2p (browser to browser) data transmission [22]. For this scenario, WebRTC can be used. A possible use case is for example a video call.

C. *CoAP (RFC7252 [23])*

The CoAP standard was published in 2014. It was designed for the use within constrained (e.g., low-power, lossy) networks and low performance devices [24]. Machine-to-machine (M2M) applications like smart energy and building automation are the preferred fields of application. However, the protocol can easily interoperate with HTTP via a proxy to provide a web interface. Therefore, the protocol realizes a subset of the REST (Representational State Transfer) architectural pattern and is based on a request/response interaction model between a server and clients. In addition, features like asynchronous message delivery, device discovery or multicast support are implemented. UDP is used as the default transportation method but also TCP or SMS are possible options. Endpoints are defined by an URI (e.g., coap://localhost:5683/device_name/parameter). Messages can be secured with DTLS.

D. *MQTT (ISO/IEC 20922:2016 [25])*

After the protocols already presented, we now for the first-time encounter with MQTT a MOM (Message Oriented Middleware) based approach that can be used for many-to-many communication [26]. The invention of MQTT dates to 1999. Nevertheless, it was not freely available until 2010. 2014 MQTT became an official OASIS standard. Furthermore, MQTT v3.1.1 is an international standard (ISO/IEC 20922:2016). In 2019 the MQTT v5 standard [27] was ratified. MQTT is a platform-oriented, simple to implement and lightweight protocol that can be used in many different situations [28]. Because of its small footprint MQTT is a perfect match for low-power and low-memory devices. Use case examples are manufacturing systems, logistics, enterprise chat applications and mobile apps. In earlier versions MQTT referred to MQ Telemetry support, but nowadays it is no longer considered an acronym. The protocol is based on a publish/subscription model where clients subscribe or publish to a specific topic. A Message Broker is used to handle the incoming requests [29]. Also see Fig. 3 – MQTT Publish/Subscribe Model [28]. A broker is available from, for example, EMQX, HiveMQ, RabbitMQ or Mosquito. As an underlying protocol there are several options available. The default is TCP, WebSocket's can be used for connecting over a web browser and QUIC is the latest available option [14].

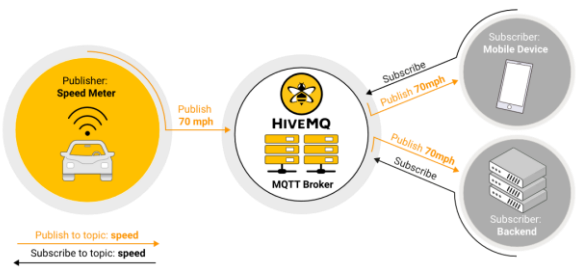

Fig. 3 – MQTT Publish/Subscribe Model [28]

E. AMQP (ISO/IEC 19464:2014 [30])

OASIS introduced the first version of the Advanced Message Queuing Protocol (AMQP) standard in 2012 [31]. Like for MQTT there exists an international standard (ISO/IEC 19464:2014). It defines itself as an internet protocol for business messaging and uses a MOM based approach [32]. It was specifically designed for the finance sector to address business processes, message transactions and applications. For this a reliable protocol on the transport layer such as TCP or QUIC is assumed. Similar to MQTT, messages are exchanged via a broker [33]. The main difference lies in the usage of message queues. In MQTT a published message would be directly routed to subscribers. In AMQP though published messages are first handled by an exchange component. Depending on the configuration (like routing keys and message type) messages are added to the corresponding message queues [34]. There it will be stored until a consumer consumes it. See Fig. 4 – AMQP core concept [34]. Examples for AMQP message brokers are RabbitMQ, SwiftMQ, Azure Service Bus and Apache Artemis. Message queues are an essential concept to make software scalable, resilient and working asynchronously.

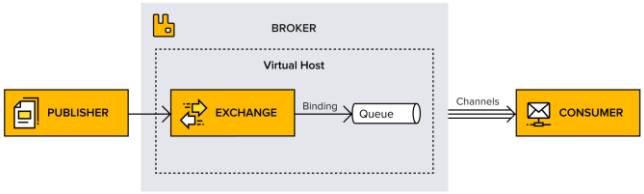

Fig. 4 – AMQP core concept [34]

F. STOMP

STOMP (Simple Text Oriented Message Protocol) [35] is designed to work with message-oriented middleware [36]. Whereas MQTT and AMQP are binary protocols, STOMP is text-based. Due to the human-readable text-format STOMP is simpler and therefore easier to implement than other messaging protocols. Commonly it is used for real-time messaging in distributed systems. Supported messaging servers that support STOMP are for example RabbitMQ or EMQX via a Gateway. The communication between a client and the server is handled through a frame modelled on HTTP. The first line of the frame contains the command, followed by headers like username and password. The last line is the message body. Destination addresses, transportation protocol and security depend on the used server respectively message broker.

G. XMPP (RFC6120 [37])

XMPP (eXtensible Messaging and Presence Protocol) or formerly Jabber enables near-real-time exchange of data [38]. Popular use cases are instant messaging, multi-party chat, voice and video calls as seen in applications like WhatsApp or Facebook. The protocol was mainly developed in 1999 and was standardized the first time in 2004. In 2012 the latest revision RFC6120 was introduced. XMPP uses XML (Extensible Markup Language) as the data-exchange format and is based on a client/server architecture. Furthermore, it uses TCP on the transportation layer. For the XMPP Server there are many options available, e.g., Tigase and ejabberd. Users on the network are addressed by email like identifiers called JID's. One of its strengths compared to MQTT is the build in end-to-end encryption.

H. Matrix

An alternative to XMPP for instant messaging is Matrix [39]. It was introduced in 2014 and is an open standard. Compared to XMPP it is generally seen more usable for group organization platforms like Slack [40]. Matrix supports bridging to other messaging platforms such as XMPP servers, Email and SMS enabling a unified way of communication. Matrix is also considered to be more secure than, for example, WhatsApp. The French government uses Matrix as base for their own communication platform called Tchap that is used for the communication of government officials and civil servants. The world wide web currently lacks on scientific papers observing this protocol.

V. Conclusion

In summary, all explored protocols have their unique strengths and weaknesses. In practice, this is why multiple protocols are used together to achieve the development goals of a new application. With the rise of new technologies and the gain in the number of internet devices more and more protocols will emerge. Furthermore, the existing protocol standards will evolve as well. QUIC is the best example for this. It was standardized in 2021 and was then used as transportation method of HTTP/3. Other technologies like MQTT start profiting from the advantages of QUIC over TCP. These developments leave room for future research in this sector. In addition, detailed research of the protocol usage in different programming environments can be conducted to identify missing links and gain a deeper understanding of the inner workings of web technologies and protocols in general. To conclude Fig. 5 - Covered Protocols Summary contains the summary of the protocols that were covered by this paper.

Acknowledgment

This paper was written as part of the master class Mobile in the master program Interactive Technologies at the UAS St.Pölten. It was revised based on the feedback from fellow students.

References

[1] J. Mesnil, *Mobile and Web Messaging: Messaging Protocols for Web and Mobile Devices*. O'Reilly Media, Inc., 2014.

[2] J. C. Talwana and H. J. Hua, "Smart World of Internet of Things (IoT) and Its Security Concerns," in *2016 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, Dec. 2016, pp. 240–245. doi: 10.1109/iThings-GreenCom-CPSCom-SmartData.2016.64.

[3] Y. Li, D. Li, W. Cui, and R. Zhang, "Research based on OSI model," in *2011 IEEE 3rd International Conference on Communication Software and Networks*, May 2011, pp. 554–557. doi: 10.1109/ICCSN.2011.6014631.

[4] F. Y. Aslan and B. Aslan, "Comparison of IoT Protocols with OSI and TCP/IP Architecture," *Int. J. Eng. Res. Dev.*, vol. 15, no. 1, Art. no. 1, Jan. 2023.

[5] L. Qilin and Z. Mintian, "The State of the Art in Middleware," in *2010 International Forum on Information Technology and Applications*, Jul. 2010, pp. 83–85. doi: 10.1109/IFITA.2010.118.

[6] E. Al-Masri *et al.*, "Investigating Messaging Protocols for the Internet of Things (IoT)," *IEEE Access*, vol. 8, pp. 94880–94911, 2020, doi: 10.1109/ACCESS.2020.2993363.

[7] N. Naik, "Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP," in *2017 IEEE International Systems Engineering Symposium (ISSE)*, Oct. 2017, pp. 1–7. doi: 10.1109/SysEng.2017.8088251.

Protocol	Architecture/Pattern	Release date of latest standard	Suitability	Use Case Examples
HTTP/3	request/response	2022 (IETF)	Default web data transfer	API's (SOAP, REST, GraphQL, gRPC, ...)
WebSocket	bi-directional (server-client) channel	2011 (IETF)	Two-way single ordered reliable server-client data transfer channel	Live Chat, Data synchronization, …
WebTransport	bi-directional (server-client) channel	Draft (IETF)	Two-way multiplexed un-/ordered /un-/reliable server-client data transfer channel	Everything from WebSocket's + Games; Stream, …
WebRTC	bi-directional (p2p) channel	2021 (IETF)	p2p two-way data communication	Audio/Video Call, Screen Sharing, …
CoAP	request/response	2014 (IETF)	Same as HTTP but with smaller package size, multicast support and asynchrony	Constrained Networks (M2M, IoT)
MQTT	publish/subscribe (MOM)	2019 (OASIS) v5 2016 (ISO/IEC) v3	lightweight Many-to-Many communication	Constrained Networks, Real-time apps, Service Bus, …
AMQP	publish/queue/consume	2014 (ISO/IEC)	Resilient, scalable and asynchronous message transfer between software services	Service Bus, Real-time apps, Notifications, , …
STOMP	client/server	2012 (STOMP)	Simple text-based messaging	Simple interaction with existing message servers/brokers
XMPP	client/server	2011 (IETF)	End-to-end encrypted many-to-many messaging	Instant Messaging, Chat, Video/Audio, …
Matrix	client/server	2023 (Matrix)	End-to-end encrypted many-to-many messaging	Same as XMPP but specialized for security and organized groups

Fig. 5 - Covered Protocols Summary

[8] "Introduction to the IETF," IETF. Accessed: Nov. 12, 2023. [Online]. Available: https://www.ietf.org/about/introduction/

[9] "ISO - International Organization for Standardization," ISO. Accessed: Nov. 15, 2023. [Online]. Available: https://www.iso.org/home.html

[10] M. Bishop, "HTTP/3," Internet Engineering Task Force, Proposed Standard RFC 9114, Jun. 2022. doi: 10.17487/RFC9114.

[11] M. Trevisan, D. Giordano, I. Drago, and A. S. Khatouni, "Measuring HTTP/3: Adoption and Performance," in *2021 19th Mediterranean Communication and Computer Networking Conference (MedComNet)*, Jun. 2021, pp. 1–8. doi: 10.1109/MedComNet52149.2021.9501274.

[12] J. Koch and E. K. Gyamfi, "Securing HTTP/3 Web Architecture in the Cloud," in *2023 IEEE World AI IoT Congress (AIIoT)*, Jun. 2023, pp. 0158–0166. doi: 10.1109/AIIoT58121.2023.10174337.

[13] J. Iyengar and M. Thomson, "QUIC: A UDP-Based Multiplexed and Secure Transport," Internet Engineering Task Force, Request for Comments RFC 9000, May 2021. doi: 10.17487/RFC9000.

[14] *MQTT over QUIC: Revolutionizing IoV Messaging with the Next-Gen Standard Protocol*. EMQ. Accessed: Oct. 23, 2023. [Online]. Available: https://www.emqx.com/en/resources/mqtt-over-quic-revolutionizing-iov-messaging-with-the-next-gen-standard-protocol

[15] D. Gurus, "REST vs GraphQL vs gRPC," Design Gurus: One-Stop Portal For Tech Interviews. Accessed: Nov. 12, 2023. [Online]. Available: https://www.designgurus.io/blog/REST-GraphQL-gRPC-system-design

[16] "2023 State of the API Report | API Technologies," Postman API Platform. Accessed: Nov. 12, 2023. [Online]. Available: https://www.postman.com/state-of-api/api-technologies/

[17] A. Melnikov and I. Fette, "The WebSocket Protocol," Internet Engineering Task Force, Request for Comments RFC 6455, Dec. 2011. doi: 10.17487/RFC6455.

[18] V. Vasiliev, "The WebTransport Protocol Framework," Internet Engineering Task Force, Internet Draft draft-ietf-webtrans-overview-06, Sep. 2023. Accessed: Nov. 12, 2023. [Online]. Available: https://datatracker.ietf.org/doc/draft-ietf-webtrans-overview-06

[19] R. Jesup, S. Loreto, and M. Tüxen, "WebRTC Data Channels," Internet Engineering Task Force, Request for Comments RFC 8831, Jan. 2021. doi: 10.17487/RFC8831.

[20] S. Springer, *Node.js - Das umfassende Handbuch*, 4th ed. Bonn: Rheinwerk, 2022. [Online]. Available: https://www.rheinwerk-verlag.de/nodejs-das-umfassende-handbuch/

[21] D. Williamson and R. O'Reilly, "WebTransport and WebSockets: An Empirical Analysis of Connection Time, Message Response, and Payload Efficiency," in *2023 34th Irish Signals and Systems Conference (ISSC)*, Jun. 2023, pp. 1–6. doi: 10.1109/ISSC59246.2023.10162060.

[22] K. I. Zinnah Apu, N. Mahmud, F. Hasan, and S. H. Sagar, "P2P video conferencing system based on WebRTC," in *2017 International Conference on Electrical, Computer and Communication Engineering (ECCE)*, Feb. 2017, pp. 557–561. doi: 10.1109/ECACE.2017.7912968.

[23] Z. Shelby, K. Hartke, and C. Bormann, "The Constrained Application Protocol (CoAP)," Internet Engineering Task Force, Request for Comments RFC 7252, Jun. 2014. doi: 10.17487/RFC7252.

[24] S. Hamdani and H. Sbeyti, "A Comparative study of COAP and MQTT communication protocols," in *2019 7th International Symposium on Digital Forensics and Security (ISDFS)*, Jun. 2019, pp. 1–5. doi: 10.1109/ISDFS.2019.8757486.

[25] "ISO/IEC 20922:2016(en), Information technology — Message Queuing Telemetry Transport (MQTT) v3.1.1." Accessed: Nov. 15, 2023. [Online]. Available: https://www.iso.org/obp/ui/en/#iso:std:69466:en

[26] C. B. Gemirter, Ç. Şenturca, and Ş. Baydere, "A Comparative Evaluation of AMQP, MQTT and HTTP Protocols Using Real-Time Public Smart City Data," in *2021 6th International Conference on Computer Science and Engineering (UBMK)*, Sep. 2021, pp. 542–547. doi: 10.1109/UBMK52708.2021.9559032.

[27] "MQTT Version 5.0." Accessed: Nov. 12, 2023. [Online]. Available: https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html

[28] *MQTT Essentials*. HiveMQ. Accessed: Oct. 18, 2023. [Online]. Available: https://www.hivemq.com/download-mqtt-ebook/

[29] *A Practical Guide to MQTT Broker Selection*. EMQ. Accessed: Oct. 23, 2023. [Online]. Available: https://www.emqx.com/en/resources/a-practical-guide-to-mqtt-broker-selection

[30] "ISO/IEC 19464:2014(en), Information technology — Advanced Message Queuing Protocol (AMQP) v1.0 specification." Accessed: Nov. 15, 2023. [Online]. Available: https://www.iso.org/obp/ui/en/#iso:std:iso-iec:19464:ed-1:v1:en

[31] "Advanced Message Queueing Protocol (AMQP) v1.0," OASIS Open. Accessed: Nov. 12, 2023. [Online]. Available: https://www.oasis-open.org/standard/amqp/

[32] N. Basavaraju, N. Alexander, and J. Seitz, "Performance Evaluation of Advanced Message Queuing Protocol (AMQP): An Empirical Analysis of AMQP Online Message Brokers," in *2021 International Symposium on Networks, Computers and Communications (ISNCC)*, Oct. 2021, pp. 1–8. doi: 10.1109/ISNCC52172.2021.9615705.

[33] N. Q. Uy and V. H. Nam, "A comparison of AMQP and MQTT protocols for Internet of Things," in *2019 6th NAFOSTED Conference on Information and Computer Science (NICS)*, Dec. 2019, pp. 292–297. doi: 10.1109/NICS48868.2019.9023812.

[34] L. Johansson and D. Dossot, *RabbitMQ Essentials - Second Edition*. Packt, 2020. Accessed: Nov. 15, 2023. [Online]. Available: https://www.packtpub.com/product/rabbitmq-essentials-second-edition/9781789131666

[35] "STOMP Specification v1.2." Accessed: Nov. 15, 2023. [Online]. Available: https://stomp.github.io/stomp-specification-1.2.html

[36] "STOMP Protocol," GeeksforGeeks. Accessed: Nov. 15, 2023. [Online]. Available: https://www.geeksforgeeks.org/stomp-protocol/

[37] P. Saint-Andre, "Extensible Messaging and Presence Protocol (XMPP): Core," Internet Engineering Task Force, Request for Comments RFC 6120, Mar. 2011. doi: 10.17487/RFC6120.

[38] S. Watkin and D. Koelle, *Practical XMPP*. Packt, 2016. Accessed: Nov. 15, 2023. [Online]. Available: https://www.packtpub.com/product/practical-xmpp/9781785287985

[39] "Matrix Specification." Accessed: Nov. 15, 2023. [Online]. Available: https://spec.matrix.org/v1.8/

[40] "Matrix (protocol)," *Wikipedia*. Oct. 08, 2023. Accessed: Nov. 15, 2023. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Matrix_(protocol)&oldid=1179142671