Performance Evaluation of AMQP and CoAP for Low-Cost Automation

Gustavo Caiza[1], Carlos S. Leon[2], Luis A. Campana[2], Carlos A. Garcia[2], and Marcelo V. Garcia[2,3](✉)

[1] Universidad Politecnica Salesiana, UPS, 170146 Quito, Ecuador
gcaiza@ups.edu.ec

[2] Universidad Tecnica de Ambato, UTA, 180103 Ambato, Ecuador
{cleon9397,la.campana,ca.garcia,mv.garcia}@uta.edu.ec

[3] University of Basque Country, UPV/EHU, 48013 Bilbao, Spain
mgarcia294@ehu.eus

Abstract. A lot of communication protocols have been developed to support the efficient communication of the Industrial Internet of Things (IIoT) devices. These kinds of applications are intended to run with constrained resources. However, the selection of a standard and effective industrial messaging protocol is a challenging task for any shop floor integration because it depends on the nature of the IoT system and its messaging requirements. In this paper, two IoT protocols like Advanced Message Queuing Protocol (AMQP) and Constrained Application Protocol (CoAP) are compared using a low-cost hardware device for factory integration. The results show that the CoAP protocol is designed to be so small that it fits inside a microcontroller, but it can be fully applied in cyber-physical environments, in another aspect the AMQP protocol is more complex, there is no official support and you need bigger installation packages; but it provides a higher communication speed.

Keywords: AMQP · CoAP · Industrial Internet of Things (IIoT) · Low-cost automation

1 Introduction

The relationship between the IoT and computer technologies in the cloud allow effective decision making to improve the productive capacity of a factory. This improvement in the industry is called the fourth industrial revolution or Industry 4.0, which brings new capabilities in the environment [2,4].

Industrial Internet of Things (IIoT) can be considered as the connection of industrial machine sensors and actuators to the Internet that can independently generate value [12]. IIoT protocols are used to develop Machine-to-Machine communication (M2M). One of the major factors that determine the performance of this M2M communication is the messaging protocol specially designed for M2M communications within the IoT applications. The selection of a communication standard and an optimized messaging protocol is a challenging task for any shop-floor integration. While selecting an appropriate messaging protocol for IoT systems, the pre-requisite is a better understanding of a target IoT system and its message/data sharing requirements [9].

Besides, IIoT applications that require a large network traffic, and consequently a greater bandwidth, would incur greater expenses to maintain the operation of the network infrastructure. Similarly, applications that generate a large amount of data require investments in computational resources (storage) proportional to the generated data [3]. Contrasted to the web systems, which use a single standard messaging protocol like HTTP. IIoT cannot rely on a single protocol for all its needs [5]. Consequently, hundreds of messaging protocols are available to choose for various types of requirements of the IoT system.

The aim of this research work is to evaluate and compared the use of two conventional communication protocols of IoT, AMQP, and CoAP, within an automated system. The main characteristics of this system are the requirement of real-time data transmission between the devices, as well as a diverse number of messages, due to the fact that the system consists of a sensor that partially sends the signal to activate the operation of the robotic manipulator arm (Scorbot). It should be noted that there is no intention of establishing which protocol is better than the other because the ideal protocol depends on the type of automation application being carried out. The aim is to capture the behavior of both protocols in the same automation environment [6,7].

This document is organized as follows: In Sect. 2 is analyzed related works where the performance of the AMQP and CoAP protocols is evaluated and their contributions to research are highlighted. It describes the environment and the different work devices that are used in the implementation of both protocols. Section 3 shows the state of the art in the work, as well as the automation environment in which the protocols will be used. In Sect. 4, the case study and the automation environment in which the work is developed are presented. In Sect. 5 the results obtained from the research are discussed. Finally, in Sect. 6 conclusions of the project are established.

2 Related Work

In the following investigations and works carried out with the communication protocol CoAP and AMQP, the usefulness and versatility provided by these protocols are analyzed, as well as the applications and uses that have been developed by them.

The research developed by Alvear et al. [1] relates the IoT technology with artificial vision concepts in which it is intended to obtain data in real-time and at the same time remotely, which will be stored in databases. The authors denote that the collected data will be used to carry out statistical and probability studies in certain areas based on the activities or characteristics of people in environments or environments where the application of electronic systems is possible, either to improve processes or identify weak points them. This whole process started with the analysis of image processing and the capture of videos to be used in an algorithm focused on IOT, for this reason the authors address the application of the CoAP protocol, because it is a specialized protocol for data transfer. To be able to directly relate to HTTP and at the same time integrate into the Web.

In the study conducted by Naik [16] a comparison of IoT protocols is made, with communication standardization as a priority and, as an important factor, real-time transmission and transfer of data that are important aspects in IoT applications. The author mentions that the choice of a standard communication protocol that is also effective is a work that deserves considerable study because of the nature of the system to be implemented as well as the communication requirements that must be generated. The scientific article mentions an evaluation of communication protocols such as CoAP and AMQP used in IoT systems, to identify their strengths and limitations.

On the one hand, the author mentions that the communication systems that use the AMQP protocol are binary systems that generally use 8-byte headers with small or small messages, in addition that this protocol uses TCP as the default transport protocol and TLS/SSL and SASL for security. Similarly, the author mentions characteristics of the CoAP protocol that, unlike AMQP, uses fixed 4-byte headers with small messages and uses UDP as the transport protocol and DTLS for its security. An important point that the author considers is that these messaging protocols with the passage of time have been evolving according to the processes or needs that must cover, so it could be considered, devices, resources and the specific applications of IoT in which they will be employees [16].

On the other hand, a comparison made by the author refers to M2M/IoT compared to standardization, speaking of AMQP this study is successful worldwide and adopted the international standard ISO/IEC 19464: 2014 is currently used in projects of great importance as Nebula Cloud Computing from NASA and Indias Aadhar Project, however CoAP has not been left behind and in recent years has gained momentum and has been employed by large companies such as Cisco, Contiki, Erika and IoTivity in addition to having a specialized IETF standard to integrate IoT and the Web thanks to Eclipse Foundation. And when talking about security, AMQP presents a high level of security while CoAP uses DTLS and Ipsec useful tools for integrity, authentication and encryption [7].

Fernades [15] mentions that there are certain problems when talking about services or communications because a lot of data is commonly sent to databases and the agglomeration of data can affect the performance of the systems significantly; AMQP is a protocol that appears to address this problem and solve it. The study that the authors propose is the analysis of message exchange in a certain time, observing that when there is a high volume of message exchange the most favorable results are generated by AMQP because it can be connected with different applications and different platforms. In this analysis we use the AMQP protocol and the storage of data in a relational database created in MySQL, the exchange of data between clients and servers determined that when exchanging data in bulk the number of messages that can be sent per second it reduces with what causes a high consumption of resources and as a solution to this problem the AMQP is used.

As mentioned previously there are studies that to compared protocols IoT and certain characteristics presented by these protocols, varying according to the application or analysis that have been focused, is why the aim of this work is a comparison between CoAP and AMQP protocol as a way to help when making IoT systems and adopt the most appropriate protocol according to the requirements they must fulfill.

3 State of Art

3.1 Internet of Things

The IoT is based on three main foundations related to the capacity of objects that must have communication capabilities, computational capacity and may have interaction capacity [13]. It is called communication capacity because the IoT objects must have a minimum set of communication capacity. What we mean by this is not only a channel of communication, but also everything related to it, in order to make an efficient communication, such as an address, identifier and name. The objects can have all these characteristics or some of them [19]. The objects must have a basic or complex computational capacity to process data and network configurations. For example, receiving commands on the communications channel, administering network tasks, saving the status of a sensor, activating an effect, receiving signals and managing and controlling data.

3.2 CoAP (Restricted Application Protocol)

The CoAP communication protocol is used to communicate simple and inexpensive electronic devices such as PLC'S, RaspBerry and low power sensors. This protocol is a derivation of the HTTP protocol, but it is added several requirements such as multicast, overhead and simplicity, which are very important for the Internet of Things (IoT), reason why the protocol is applicable to develop the connectivity of intelligent objects using the Internet [6,12]. See Fig. 1.

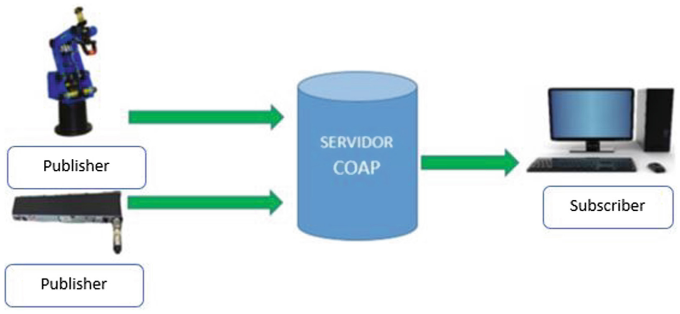


Fig. 1. CoAP architecture

It is a specialized protocol for the use of limited and limited low power wireless nodes that can communicate interactively through the internet, its client/server interaction model is similar to that of HTTP with the difference that CoAP performs these interactions (exchanges of messages) asynchronously by means of the UDP transport protocol [7].

3.3 AMQP (Advanced Message Queue Protocol)

The Advanced Message Queuing Protocol is also a publication/subscription protocol based on a reliable message queue. It has been commonly used in the financial sector. This community uses services such as commerce and banking systems that often require extremely high levels of performance, scalability, reliability and manageability [17]. AMQP uses TCP as its main transport protocol for the exchange of messages. Application level messages have a header to route them to the respective queue (see Fig. 2). The AMQP architecture is composed of two main components: Queues and Exchanges [17,18].

Queues represent the main concept of AMQP. All messages end in a queue that stores them before forwarding them to recipients. These queues can be organized by service levels with respect to implementation performance characteristics such as latency and availability [19].

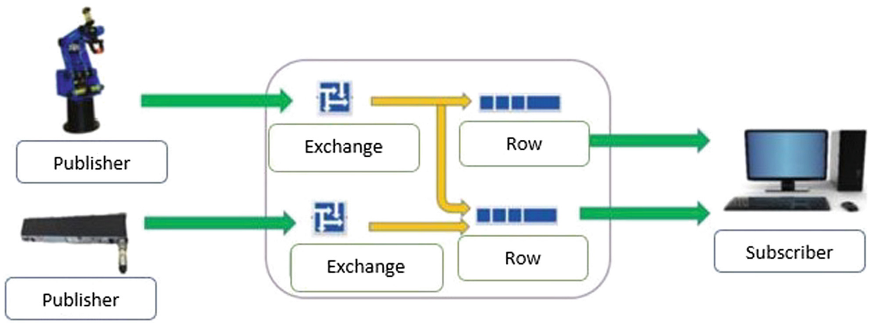


Fig. 2. Architecture AMQP

3.4 RabbitMQ Broker

Both AMQP and MQTT are communication protocols based on intermediaries. As discussed in Sect. 2, they contain a central entity, called an agent, in charge of managing peer-to-peer communication in the network. In this work, we use RabbitMQ, a popular open source message agent. RabbitMQ is an Erlang based technology that allows asynchronous communication between devices. Initially, it was developed to implement AMQP and then to support MQTT [12].

The exchanges distribute the messages to the respective queues according to the predefined rules. As a new message arrives at the intermediary, the exchange evaluates the message and stores it in a queue, ready to be forwarded. Figure 2 represents the main messaging process in an environment based on AMQP. First, editors who want to send messages to potential subscribers, send them to a broker. The intermediary has exchanges and queues. As mentioned above, the exchanges receive the messages and forward them to the respective queues. In turn, the queues send these messages to the clients that previously subscribed to the given queue [11].

3.5 Raspberry Pi 3

It is a low cost hardware platform that includes all the elements offered by a computer. Nowadays it has acquired great importance in the market due to its diversity of options for projects in computer networks, electronic circuits, robotics, domotics, security, programming, among other technological areas. Even some authors like Saari et al. [10] have used the Raspberry Pi as a solution for the Internet of Things (IoT).

The most current model of Raspberry Pi 3 is B, which has a storage unit with MicroSD Card Slot and is equipped with 35000 packages and pre-compiled programs in a format that facilitates installation. In addition, despite being adapted to the perfection of the board, it is not an operating system affiliated with the Raspberry Pi foundation [7,14].

4 Case Study

The aim of this research work is to communicate a factory process made by a ScorBot ER-4U robot arm and a conveyor belt to control a palletizing process. In order to get the industrial information and compared IIoT protocols characteristics the control of industrial devices is made re-using a PLC industrial architecture. The PLC selected is an S7-1200. The ScorBot ER-4U robot arm is controlled by a PLC1 this PLC sends commands to the USB Controller that provides advanced control features for the ScorBot ER-4U robotic arm. The conveyor belt is controlled by a PLC2 that controls the electric motor to move pieces from a deposit to the robotic arm. The synchronization of the process is made by PLC1, this PLC sends commands to run or stop the conveyor. To read PLCs memories Raspberry Pi boards use the Snap7 library.

To implement the stack of AMQP and CoAP protocols the authors of this paper use low-cost boards like Raspberry Pi. The first Raspberry PI is used as a server (AMQP publisher or CoAP server) that sends information to all clients. The second one is used as the client (AMQP consumer or CoAP client) to integrate the information of the conveyor belt. Furthermore, a web client is developed to monitor the operation of the palletizing process.

The communication of all the components is given with a data transmission speed of 10 Mbps to have a minimum error rate in the communication, managing the processes in real-time and monitoring. Figure 3 presents the structure of the proposed communication architecture.

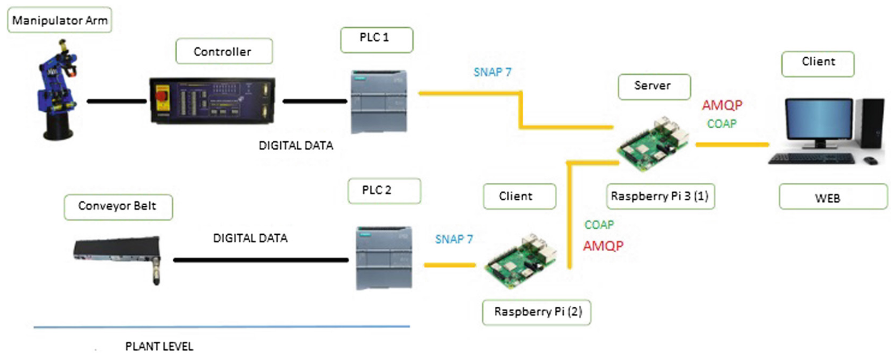


Fig. 3. Case study hardware architecture platform

4.1 SNAP 7 Implementation

Applying the Snap 7 (C++) library, it is intended to get data from the S7-1200 PLCs through the reading of the internal databases of the PLC using the Raspberry Pi 3. This library uses the Communication interface. S7 Siemens Ethernet for reading and writing PLC data (inputs, outputs, memories, timers, counters) and has three independent components: client, server and partner.

All Snap7 functions completely hide this concept, the data that a system can transfer in a single call depends only on the size of the available data. The Snap 7 stack library provides two main functions: Cli_FullUpload() to upload a full block from the PLC CPU and Cli_Upload() to upload only data from a data block, depending on the need of the software. These functions are asynchronous and executed in the same thread of the caller, i.e. it exists only when its job is complete. These functions consist of two parts, the first, executed in the same thread of the caller, which prepares the data (if any), triggers the second part and exits immediately. The second one is executed in a separate thread and performs the body of the job requested, simultaneously to the execution of the caller program. See Fig. 4.

The advantages of using the Snap7 library are many, because it is written in C++, reading data from Ethernet compatible PLCs, as long as the requests to Ethernet are not restricted. In the work done, the data is written and read through bytes, but you can use variables of type: Word, Double Word and Real used extensively in the programming of the Siemens language [8].

4.2 CoAP Protocol Implementation

The implementation of CoAP is based on libcoap library, an opensource C-library that is specifically targeted at low-cost embedded systems with constrained resources. This library provided support for the current working group drafts of CoAP as well as its optional extensions for block-wise transfer, resource observation.

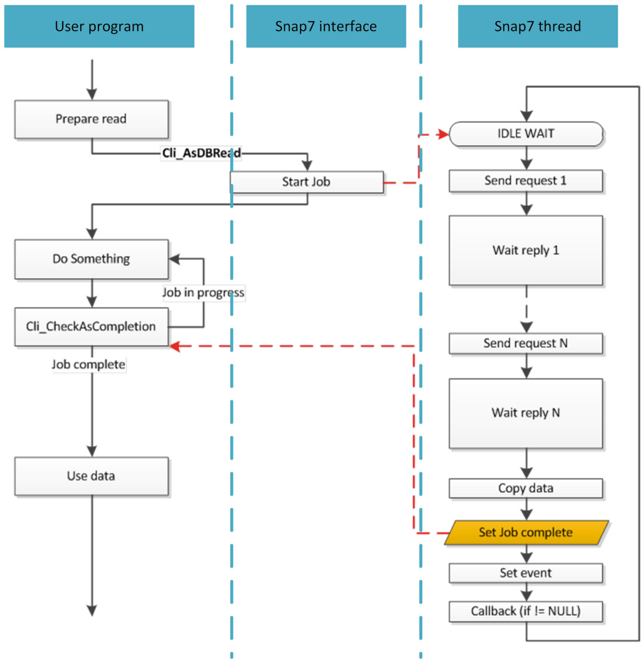


Fig. 4. Sanp7 communication function model

The library provides functions and data structures for parsing and in-place editing of CoAP protocol data units (PDUs) to minimize memory overhead in embedded systems. An additional application server and a multi-purpose command-line client built upon this library demonstrate the use of the API in stand-alone CoAP-enabled applications.

LibCoAP is compiled in Debian kernel into Raspberry Pi. The architecture of the software algorithm is described as follows: (i) When the client is initialized the CoAP-client class starts, this client has GET or PUT methods. The GET method is used to retrieve resources from PLCs. The resource is identified by the requested Uniform Resource Identifier (URI). The PUT method is used to modify an existing resource on PLC. CoAP uses the datagram-oriented UDP transport protocol to exchange messages.

(ii) When a request arrives at a resource from the client, the module of the server Server.on takes care of it according to what type of information requirement is, for example, the GET method receives information of the I0.5 input of the selected device, while the PUT method writes the value of each input and output of the PLC in which it is being executed. See Fig. 5.

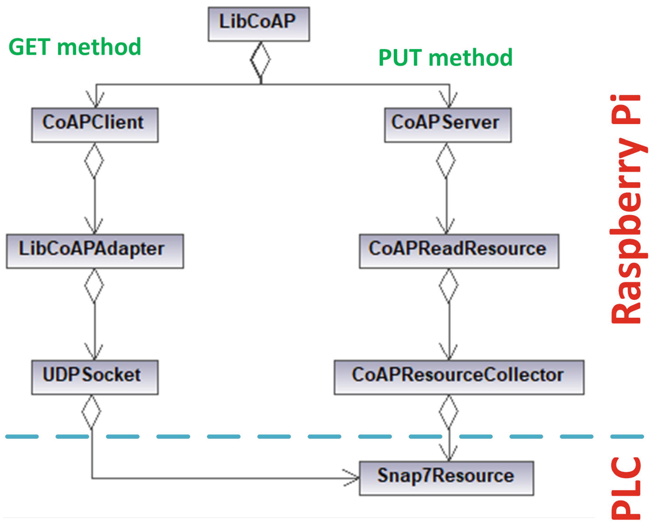


Fig. 5. Class diagram for CoAP PUT and GET methods

4.3 AMQP Implementation

The second protocol implemented in the low-cost hardware is AMQP. For this communication, rabbitMQ library was used. The RabbitMQ implementation of a sample dev/test event bus is boilerplate code. This library is a message-queueing software also known as a message broker or queue manager. The queue-manager software stores the messages until a receiving application connects and takes a message off the queue. The receiving application then processes the message.

The AMQP producer queueing up new messages of Snap7 library The consumer takes a message off the queue and starts processing the shop-floor information. Messages with shop-floor information are not published directly to a queue; instead, the producer sends messages to an exchange. An exchange is responsible for routing the messages to different queues.

The Message flow for AMQP protocol developed into low-cost hardware is as follows (see Fig. 6): (1) The producer publishes a message with ScorBot ER-4U robot arm to an exchange. (2) The exchange receives the message and is now responsible for routing the message. The exchange takes different message attributes into account, such as the routing key, depending on the exchange type. (3) Bindings must be created from the exchange to queues. In this case, there are two bindings to two different queues from the exchange. The exchange routes the message into the queues depending on message attributes. (4) The messages stay in the queue until they are handled by a consumer (conveyor device). The consumer handles the message.

The Listing 1.1 shows the function that sends the PLC states in AMQP, besides the String that stores the message, a simple message is created so that

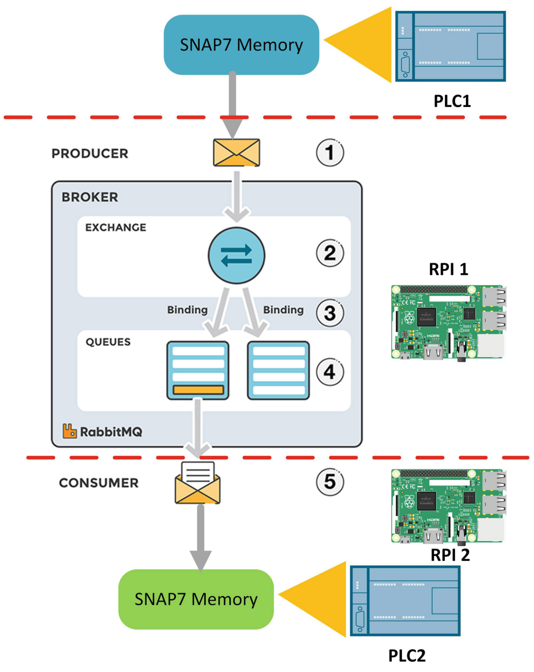


Fig. 6. Message flow for AMQP protocol

AMQP can send it. The created message is published to the corresponding device, which is PLC1 or PLC2.

```
void publishstatesPLC(string ipserver, string plc, string io,
    int number, bool data){
    message = "{\"protocol\": \"AMQP\", \"";
    AmqpClient::BasicMessage::ptr_t msg = AmqpClient::
    BasicMessage::Create(mensaje);
    connection->BasicPublish("", plc, msg);
}
```

Listing 1.1. Sending PLC states using AMQP protocol

5 Results Discussion

In the present study, different tests have been performed to compare both protocols and define in which situations they behave better. The first test is about the execution time introducing different bandwidths (5, 2.5, 1 Mbit), latencies (0, 5, 10 ms) and lost packet rates (0, 2.5 Y5)%, it should be noted that the tests were measured in seconds. Figure 7 shows a uniform behavior over time, with a variation between tests no greater than 7% the time. See Fig. 7(a).

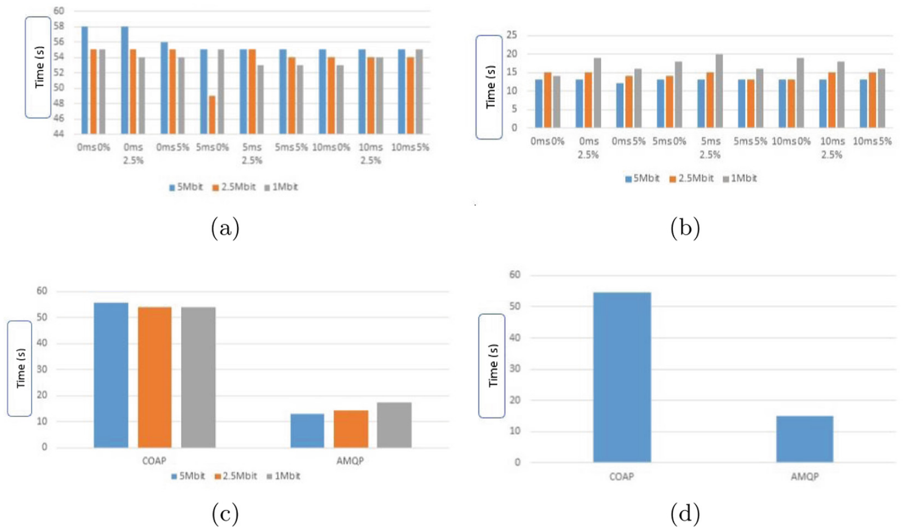
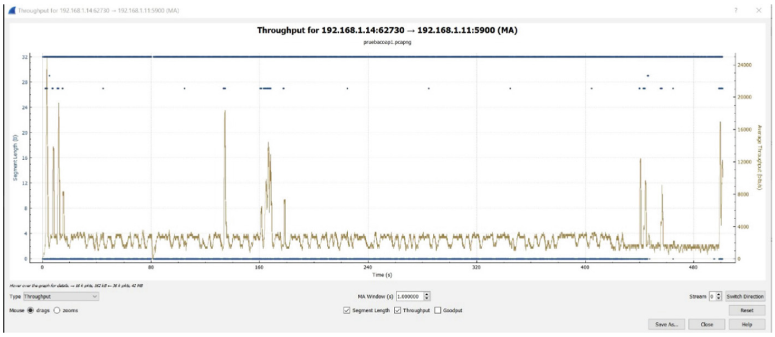


Fig. 7. Comparison of AMQ and CoAP protocols. (a) CoAP protocol execution time. (b) Execution time AMQP protocol. (c) Execution time per bandwidth. (d) Average execution times

As the main reason to be implemented the CoAP protocol is to run on hardware and minimal infrastructure, the execution time is clearly greater than AMQP due to the implementation problems required by this library. In the case of the AMQP protocol, the speed of execution is unparalleled, with an average of 0.09 s being much faster than CoAP, under normal conditions the protocol is optimal. The problem arises when the network conditions are less favorable, it shows a 17% variability with respect to the average. See Fig. 7(b).

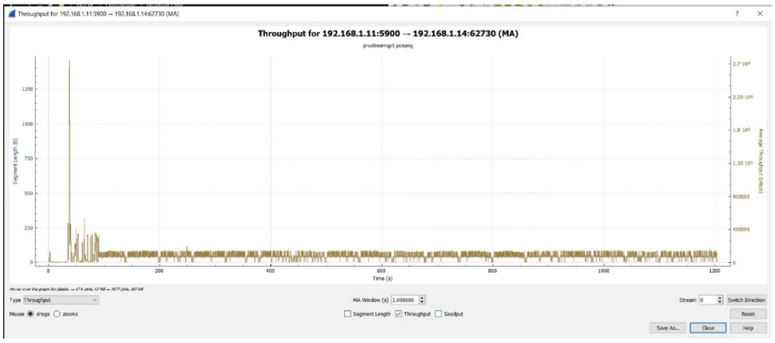
Regarding the execution time, it can be noted that the AMQP protocol significantly exceeds the CoAP protocol with an average of 0.09 s under normal conditions, given the implementation problems required by the CoAP library. See Fig. 7(c). About the variability presented by the protocols, it can be noted that the CoAP protocol being restricted remains constant over time (variability of 1%), while AMQP suffers when the network conditions are not optimal (variability up to 30%). Figure 7(d) shows that AMQP is superior to CoAP.

In the 200 samples that were taken from the Scorbot Robot, it demonstrates a speed of 380% with respect to the process by CoAP. Throughput performance graphs, as they are done in a controlled and noise-free environment, have similar characteristics, so if you want to obtain a successful transfer of packets, both protocols fulfill their purpose. Due to the lack of compatibility, AMQP would not be compatible with small devices. In contrast, AMQP has reception feedback, so if it is implemented in the code, the rate of lost packets is zero in critical processes. See Fig. 8.



Hover over the graph for details. → 16 k pkts, 162 kB ← 36 k pkts, 42 MB
Type Throughput
Mouse [✓ drags] zooms
MA Window (s) 1.000000
☑ Segment Length ☑ Throughput ☐ Goodput
Stream 0

(a)



Hover over the graph for details. → 17 k pkts, 13 MB ← 9177 pkts, 187 kB
Type Throughput
Mouse [✓ drags] zooms
MA Window (s) 1.000000
☐ Segment Length ☑ Throughput ☐ Goodput
Stream 0

(b)

Fig. 8. AMQ and CoAP protocols performance. (a) CoAP protocol performance. (b) AMQP protocol performance.

6 Conclusion and Future Work

In the present work presents a study in order to make a comparison between two IoT communication protocols; AMQP and CoAP, due to the constant revolution in the industry and the implementation of Process Automation, where the Industrial Internet of Things (IoT) has been inserted as part of Industry 4.0, its development is on track.

In this way, the authors have developed research with the purpose of comparing both protocols and to give a verdict of which one is the most convenient in different types of scenarios, measuring the total time of the process (TPT) in the same conditions and for each protocol, for the analysis of AMQP the data is stored in MySQL, we have found that AMQP was faster than CoAP with an average of 0.09 s in normal conditions, however CoAP does not stay behind, since it does not require a broker and in the analysis of the variation in bandwidth, being a restricted protocol, it remains constant over time, for the case of AMQP, it presents variation is when the network conditions are not optimal.

When implementing the two protocols for the application of the palletizing of the Scorbot Er-4u Manipulator Arm, it was noted that AMQP is superior to the CoAP protocol in the 200 samples taken, showing a 380% higher benefit. However, each IoT protocol has its own characteristics, so each one of them in different scenarios can be of vital importance in terms of communication and in different cases, the CoAP protocol would be the best option.

As a future work, we have proposed to implement as an application the analysis of the FESTO parts sorting station with IoT protocols, since it makes use of sensors and actuators allowing us to change the environment and open ourselves to the field of pneumatic together with the PLCs and the mechanics.

Acknowledgment. This work was financed by Universidad Técnica de Ambato (UTA) and their Research and Development Department (DIDE) under project CONIN-P-256-2019.

References

1. Alvear, V.: Internet de las cosas y visión artificial, funcionamiento y aplicaciones: revisión de literatura (Internet of Things and artificial vision, performance and applications: literature review). Enfoque UTE **8**(1), 244–256 (2017). http://ingenieria.ute.edu.ec/enfoqueute/
2. Ancillotti, E., Bruno, R., Vallati, C., Mingozzi, E.: Design and evaluation of a rate-based congestion control mechanism in CoAP for IoT applications. In: 19th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, WoWMoM 2018, pp. 14–15 (2018). https://doi.org/10.1109/WoWMoM.2018.8449736
3. Andrade, L., Serrano, M., Prazeres, C.: The data interplay for the fog of things: a transition to edge computing with IoT. In: 2018 IEEE International Conference on Communications (ICC), pp. 1–7 (May 2018). https://doi.org/10.1109/ICC.2018.8423006
4. Bahashwan, A.A.O., Manickam, S.: A brief review of messaging protocol standards for Internet of Things (IoT). J. Cyber Secur. Mobil. **8**(1), 1–14 (2018). https://doi.org/10.13052/jcsm2245-1439.811
5. Batista, E., Andrade, L., Dias, R., Andrade, A., Figueiredo, G., Prazeres, C.: Characterization and modeling of IoT data traffic in the fog of things paradigm. In: 2018 IEEE 17th International Symposium on Network Computing and Applications (NCA), pp. 1–8 (November 2018). https://doi.org/10.1109/NCA.2018.8548340
6. Bhatia, R., et al.: Massive machine type communications over 5G using lean protocols and edge proxies. In: IEEE 5G World Forum, 5GWF 2018 - Conference Proceedings, pp. 462–467 (2018). https://doi.org/10.1109/5GWF.2018.8517086
7. Bolettieri, S., Tanganelli, G., Vallati, C., Mingozzi, E.: pCoCoA: a precise congestion control algorithm for CoAP. Ad Hoc Netw. **80**, 116–129 (2018). https://doi.org/10.1016/j.adhoc.2018.06.015
8. Din, S., Paul, A., Hong, W.H., Seo, H.: Constrained application for mobility management using embedded devices in the Internet of Things based urban planning in smart cities. Sustain. Cities Soc. **44**, 144–151 (2019). https://doi.org/10.1016/j.scs.2018.07.017

9. Gohar, M., Choi, J.G., Koh, S.J.: CoAP-based group mobility management protocol for the Internet-of-Things in WBAN environment. Future Gener. Comput. Syst. **88**, 309–318 (2018). https://doi.org/10.1016/j.future.2018.06.003
10. Granjal, J., Silva, J.M., Lourenço, N.: Intrusion detection and prevention in CoAP wireless sensor networks using anomaly detection. Sensors **18**(8), 2445 (2018). https://doi.org/10.3390/s18082445. (Switzerland)
11. Halabi, D., Hamdan, S., Almajali, S.: Enhance the security in smart home applications based on IOT-CoAP protocol. In: 6th International Conference on Digital Information, Networking, and Wireless Communications, DINWC 2018, pp. 81–85 (2018). https://doi.org/10.1109/DINWC.2018.8357000
12. Kumar, A., Mozar, S. (eds.): ICCCE 2018. LNEE, vol. 500. Springer, Singapore (2019). https://doi.org/10.1007/978-981-13-0212-1
13. Herrero, R.: Dynamic CoAP mode control in real time wireless IoT networks. IEEE Internet Things J. **6**(1), 801–807 (2019). https://doi.org/10.1109/JIOT.2018.2857701
14. Iglesias-Urkia, M., Orive, A., Urbieta, A., Casado-Mansilla, D.: Analysis of CoAP implementations for industrial Internet of Things: a survey. J. Ambient Intell. Humaniz. Comput. **00**(2016), 1–14 (2018). https://doi.org/10.1007/s12652-018-0729-z
15. Fernandes, J.L., Lopes, I.C., Rodrigues, J.J.P.C., Ullah, S.: Performance evaluation of RESTful web services and AMQP protocol. In: 2013 Fifth International Conference on Ubiquitous and Future Networks (ICUFN), Da Nang, pp. 810–815 (2013). https://doi.org/10.1109/ICUFN.2013.6614932
16. Naik, N.: Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP. In: 2017 IEEE International Symposium on Systems Engineering, ISSE 2017 - Proceedings (2017). https://doi.org/10.1109/SysEng.2017.8088251
17. Rathod, D., Patil, S.: Security analysis of constrained application protocol (CoAP): IoT protocol. Int. J. Adv. Stud. Comput. Sci. Eng. **6**(8), 37–41 (2017). https://search.proquest.com/docview/1947842952?accountid=17242
18. Talaminos-Barroso, A., Estudillo-Valderrama, M.A., Roa, L.M., Reina-Tosina, J., Ortega-Ruiz, F.: A Machine-to-Machine protocol benchmark for eHealth applications - use case: respiratory rehabilitation. Comput. Methods Programs Biomed. **129**, 1–11 (2016). https://doi.org/10.1016/j.cmpb.2016.03.004
19. Vallati, C., Righetti, F., Tanganelli, G., Mingozzi, E., Anastasi, G.: ECOAP: experimental assessment of congestion control strategies for CoAP using the wishful platform. In: Proceedings - 2018 IEEE International Conference on Smart Computing, SMARTCOMP 2018, pp. 423–428 (2018). https://doi.org/10.1109/SMARTCOMP.2018.00040