Survey of the IoT Data Transmission Protocols

Andrii Polianytsia
Kyiv National Economic University
Kyiv, Ukraine
andyssey@gmail.com

Olena Starkova, Kostiantyn Herasymenko
Taras Shevchenko National University of Kyiv
Kyiv, Ukraine
National Technical University of Ukraine
"Igor Sikorsky Kyiv Polytechnic Institute"
Kyiv, Ukraine
elesta.tcs@gmail.com

Abstract – in this article are listed the most popular IoT data transmission protocols and their main features. These protocols are used to create modern IoT projects in startups, scientific projects and other useful devices for a smart house etc. The article formulates a basic review of IoT data transmission protocols based on user needs.

Key words: data transmission protocol, MQTT, COAP, MODBUS.

I. Introduction

In our time, a rapid growth IoT concept and computing capabilities provide us with powerful tools and give us the ability to connect things in different ways. But in IoT development, not only new technologies are important. So the key moment is the possibility of integration with the developments of the past. And the key moment of such integration is communication.

So now he has a big variate of data transmission protocols, which are working at different OSI levels, provides us with a different functionality with a big variate of libraries for almost all programming languages. So the problem of choosing the protocol that suits to your own need becomes very common[5].

II. Analysys of Data Transmission Protocols

A. Messasge Queue Telemetry Transport

MQTT is a "lightweight" messaging protocol for use on top of the TCP/IP protocol. It is designed for connections with remote locations where a "small code footprint" is required or the network bandwidth is limited. MQTT defines methods (sometimes referred to as verbs) to indicate the desired action to be performed on the identified resource [1-4]. What this resource represents, whether preexisting data or data that is generated dynamically, depends on the implementation of the server. Often, the resource corresponds to a file or the output of an executable residing on the server[2].

This methods are: Connect (Waits for a connection to be established with the server), Disconnect (Waits for the MQTT client to finish any work it must do, and for the TCP/IP session to disconnect), Subscribe (Waits for completion of the Subscribe or UnSubscribe method), UnSubscribe (Requests the server unsubscribe the client from one or more topics), Publish (Returns immediately to the application thread after passing the request to the MQTT client).

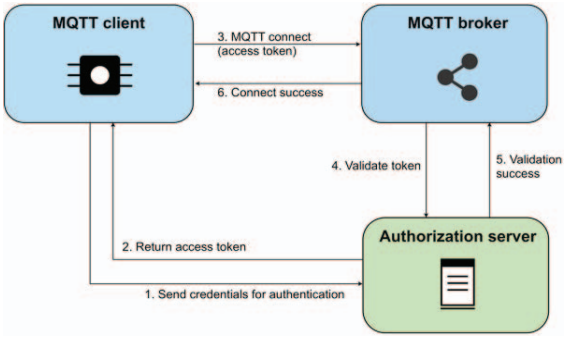

Fig. 1. MQTT scheme

B. Constrained Application Protocols

CoAP is an Internet Application Protocol [5-10] for constrained devices (defined in RFC 7228). It enables those constrained devices to communicate with the wider Internet using similar protocols. CoAP is designed for use between devices on the same constrained network, between devices and general nodes on the Internet, and between devices on different constrained networks both joined by an internet. CoAP is also being used via other mechanisms, such as SMS on mobile communication networks[1].

CoAP is a service layer protocol that is intended for use in resource-constrained internet devices, such as wireless sensor network nodes. CoAP is designed to easily translate to HTTP for simplified integration with the web, while also meeting specialized requirements such as multicast support, very low overhead, and simplicity.

Multicast, low overhead, and simplicity are extremely important for Internet of Things (IoT) and Machine-to-Machine (M2M) devices, which tend to be deeply embedded and have much less memory and power supply than traditional internet devices have. Therefore, efficiency is very important. CoAP can run on most devices that support UDP or a UDP analogue.

C. Modbus communication protocol

Modbus is a serial communications protocol originally published by Modicon (now Schneider Electric) in 1979 for use with its programmable logic controllers (PLCs). Simple and robust, it has since become a *de facto* standard communication protocol and is now a commonly available means of connecting industrial electronic devices [11-15].

Each device intended to communicate using Modbus is given a unique address. In serial and MB+ networks, only the node assigned as the Master may initiate a command.

978-1-5386-0983-5/17/$31.00 ©2017 IEEE

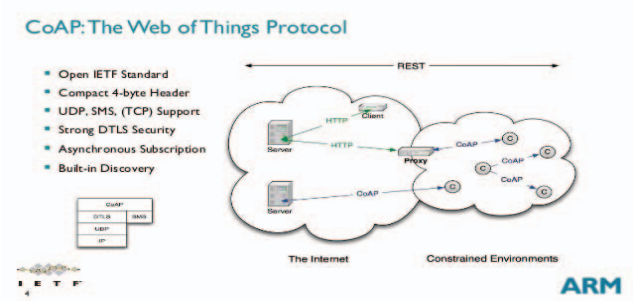

Fig. 2. COAP scheme

On Ethernet, any device can send out a Modbus command, although usually only one master device does so. A Modbus command contains the Modbus address of the device it is intended for (1 to 247). Only the intended device will act on the command, even though other devices might receive it (an exception is specific broadcastable commands sent to node 0, which are acted on but not acknowledged). All Modbus commands contain checksum information, to allow the recipient to detect transmission errors. The basic Modbus commands can instruct an RTU to change the value in one of its registers, control or read an I/O port, and command the device to send back one or more values contained in its registers.

There are many modems and gateways that support Modbus, as it is a very simple protocol and often copied. Some of them were specifically designed for this protocol.

Different implementations use wireline, wireless communication, such as in the ISM band, and even Short Message Service (SMS) or General Packet Radio Service (GPRS). One of the more common designs of wireless networks makes use of mesh networking. Typical problems that designers have to overcome include high latency and timing issues[3].

The Modbus communication interface is built around messages. The format of these Modbus messages is independent of the type of physical interface used. On plain old RS232 are the same messages used as on Modbus/TCP over ethernet. This gives the Modbus interface definition a very long lifetime. The same protocol can be used regardless of the connection type. Because of this, Modbus gives the possibility to easily upgrade the hardware structure of an industrial network, without the need for large changes in the software. A device can also communicate with several Modbus nodes at once, even if they are connected with different interface types, without the need to use a different protocol for every connection.

On simple interfaces like RS485 or RS232, the Modbus messages are sent in plain form over the network. In this case the network is dedicated to Modbus. When using more versatile network systems like TCP/IP over ethernet, the Modbus messages are embedded in packets with the format necessary for the physical interface. In that case Modbus and other types of connections can co-exist at the same physical interface at the same time. Although the main Modbus message structure is peer-to-peer, Modbus is able to function on both point-to-point and multidrop networks.

Each Modbus message has the same structure. Four basic elements are present in each message. The sequence of these elements is the same for all messages, to make it easy to parse the content of the Modbus message. A conversation is always started by a master in the Modbus network. A Modbus master sends a message and—depending of the contents of the message—a slave takes action and responds to it. There can be more masters in a Modbus network. Addressing in the message header is used to define which device should respond to a message. All other nodes on the Modbus network ignore the message if the address field doesn't match their own address.

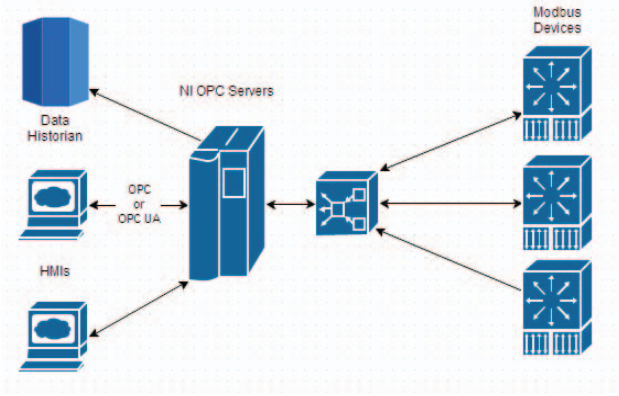

Fig. 3. Modbus scheme

III. ARCHITECTURE OF IOT

There is no single consensus on architecture for IoT, which is agreed universally. Different architectures have been proposed by different researchers. The most basic architecture is a three-layer architecture. It has three layers, namely, the perception, network, and application layers.

1) The perception layer is the physical layer, which has sensors for sensing and gathering information about the environment. It senses some physical parameters or identifies other smart objects in the environment.

2) The network layer is responsible for connecting to other smart things, network devices, and servers. Its features are also used for transmitting and processing sensor data.

3) The application layer is responsible for delivering application specific services to the user. It defines various applications in which the Internet of Things can be deployed, for example, smart homes, smart cities, and smart health.The three-layer architecture defines the main idea of the Internet of Things, but it is not sufficient for research on IoT because research often focuses on finer aspects of the Internet of Things. That is why, we have many more layered architectures proposed in the literature[4].

One is the five-layer architecture, which additionally includes the processing and business layers [3–6]. The five layers are perception, transport, processing, application, and business layers. The role of the perception and application layers is the same as the architecture with three layers. We outline the function of the remaining three layers.

1) The transport layer transfers the sensor data from the perception layer to the processing layer and vice versa through networks such as wireless, 3G, LAN, Bluetooth, RFID, and NFC.

2) The processing layer is also known as the middleware layer. It stores, analyzes, and processes huge amounts

of data that comes from the transport layer. It can manage and provide a diverse set of services to the lower layers. It employs many technologies such as databases, cloud computing, and big data processing modules.

3) The business layer manages the whole IoT system, including applications, business and profit models, and users' privacy. [10]

Most architectures proposed for the SIoT have a server side architecture as well. The server connects to all the interconnected components, aggregates (composes) the services, and acts as a single point of service for users.

The server side architecture typically has three layers. The first is the base layer that contains a database that stores details of all the devices, their attributes, metainformation, and their relationships. The second layer (Component layer) contains code to interact with the devices, query their status, and use a subset of them to effect a service. The topmost layer is the application layer, which provides services to the users.[11]

On the device (object) side, we broadly have two layers. The first is the object layer, which allows a device to connect to other devices, talk to them (via standardized protocols), and exchange information. The object layer passes information to the social layer. The social layer manages the execution of users' applications, executes queries, and interacts with the application layer on the server.

IV. CONLUSION

In this article described basic IoT data transmission protocols and how to use them in different situations. As we can see, the problem of choosing a good protocol for IoT is very common now. So the decision about which of them would meet the requirements of the specific project in IoT requires additional knowledge about platforms, embedded OS's, sensors etc. Anyway, the main hint is to compare these protocols and decide which protocol to use only in real projects, with real tasks and cases. Such an approach gives us the ability to choose right solutions and does not follow for the modern tendencies.

REFERENCES

[1] "Constrained Application Protocol", RFC 7252 [On-line] Constrained Application Protocol, Available: http://coap.technology/

[2] "MQTT official web site". [On-line] MQTT community, Available: http://mqtt.org/

[3] "Modbus official web site". [On-line] Modbus Organization, Available: http://www.modbus.org/

[4] Schatz, G. (2017). *The Complete List Of Wireless IoT Network Protocols.* [online] Link-labs.com. Available at: https://www.link-labs.com/blog/complete-list-iot-network-protocols [Accessed 28 Jul. 2017].

[5] "Mikrotik.kpi.ua. (2017). Vivchaêmo distantsíyne keruvannya dlya Arduino [Studying distance control for Arduino]. [online] Available at: http://mikrotik.kpi.ua/index.php/%20courses-list/category-arduino/24-learning-remoute-control-for-arduino [Accessed 28 Jun. 2017].

[6] "Mikrotik.kpi.ua. (2017). Yak kontrol'vati Arduino z PK cherez Bluetooth | Zanyattya 8-9. [How to control Arduino from PC through Bluetooth]. [online] Available at: http://mikrotik.kpi.ua/index.php/courses-list/category-arduino/32-how-to-control-the-arduino-from-the-pc-via-bluetooth [Accessed 28 Aug. 2017]."

[7] Arduino.ua. (2017). "SPI Biblioteka" [SPI library] [online] Available at: http://arduino.ua/ru/prog/SPI [Accessed 28 Jul. 2017].

[8] Gaw.ru. (2017). Posledovatel'nyy interfeys SPI (3-provodnoy). [Serial interface SPI (3-wire)] [online] Available at: http://www.gaw.ru/html.cgi/txt/interface/spi [Accessed 28 Jul. 2017].

[9] Kostiantyn V. Herasymenko and Olena V. Starkova. "Improved Interference Suppression BFGS-Algorithm in Modern Satellite, Radio Relay and Tropospheric systems" *Everant. Engineering And Technology Journal*, № 1(4), pp. 107-112, 2016.

[10] Dariia Ivanova, Olena Starkova and Kostiantyn Herasymenko. "Realization of the remote power management system based on the concept of Internet of Things" *Problems of Infocommunications Science and Technology* (PIC S&T), 2016 Third International Scientific-Practical Conference, 2016, pp. 96-98.

[11] Roman Kaporin, Alla Kogan, Olena Starkova and Kostiantyn Herasymenko. "Organization of secure multipath routing" *Problems of Infocommunications Science and Technology (PIC S&T),* 2016 Third International Scientific-Practical Conference, 2016, pp. 103-104.

[12] D. V. Ageyev and F. Wehbe, "Parametric synthesis of enterprise infocommunication systems using a multi-layer graph model," *2013 23rd International Crimean Conference "Microwave & Telecommunication Technology"*, Sevastopol, 2013, pp. 507-508.

[13] A. A. Ignatenko and D. V. Ageyev, "Structural and parametric synthesis of telecommunication systems with the usage of the multi-layer graph model," *2013 23rd International Crimean Conference "Microwave & Telecommunication Technology"*, Sevastopol, 2013, pp. 498-499.

[14] D. Ageyev, A. Ignatenko and F. Wehbe, "Design of information and telecommunication systems with the usage of the multi-layer graph model," *2013 12th International Conference on the Experience of Designing and Application of CAD Systems in Microelectronics (CADSM)*, Polyana Svalyava, 2013, pp. 1-4.

[15] D. Ageyev, D. Yarkin and Q. Nameer, "Traffic aggregation and EPS network planning problem," *2014 First International Scientific-Practical Conference Problems of Infocommunications Science and Technology*, Kharkov, 2014, pp. 107-108.

[16] K. Kovalchuk, O. Starkova and K. Herasymenko, "IoT device for object's power remote control," *2016 Third International Scientific-Practical Conference Problems of Infocommunications Science and Technology (PIC S&T),* Kharkiv, 2016, pp. 112-113. doi: 10.1109/INFOCOMMST.2016.7905351

[17] A. Polianytsia, O. Starkova and K. Herasymenko, "Survey of hardware IoT platforms," *2016 Third International Scientific-Practical Conference Problems of Infocommunications Science and Technology (PIC S&T),* Kharkiv, 2016, pp. 152-153. doi: 10.1109/INFOCOMMST.2016.7905364

[18] G.Vlasyuk, K.Herasymenko, Y.Kravchenko and A.Polianytsia. "Implementation of the Internet of things concept for remote power management", *2nd International conference on Advanced Information and Communication Technologies-2017 (AICT-2017*), 2017, pp.1-3.