Industrial IoT Smartbox for the Shop Floor

Sérgio Malhão
Instituto Politécnico de Castelo Branco
Castelo Branco, Portugal
smalhao@ipcbcampus.pt

Rogério Dionísio
Instituto Politécnico de Castelo Branco
Castelo Branco, Portugal
rdionisio@ipcb.pt

Pedro Torres
Instituto Politécnico de Castelo Branco
Castelo Branco, Portugal
pedro.torres@ipcb.pt

***Abstract*— Constant search for efficiency and productivity has led to innovation on the factory shop floor, representing an evolution of the current production systems combined with new technologies of industrial automation and information technology. This work presents an experimental demo of a smartbox for Industry 4.0 scenarios, allowing sensing, monitoring and data acquisition. We have tested two different approaches, depending on the communication protocol used for real time applications: OPC UA or MQTT. Raspberry Pi's platform act as an OPC UA server or MQTT broker, respectively. From the measurements, data stored in a cloud server can be accessed remotely with improved security and visualized from a computer dashboard. One of the conclusions that can be drawn is that both protocols allow data from the smartbox to be stored and easily monitored from a smartphone application or a computer web interface. MQTT is a good option in communications requiring very low bandwidth. However, there is a lack of suitable libraries to program alarm features for OPC UA Servers.**

Keywords*— *Industrial IoT, OPC UA, MQTT, Raspberry PI, Node-RED

I. Introduction

Recent years witnessed the forthcoming of Machine-to-Machine (M2M) networks as an efficient means to provide automated communications between distributed devices. The idea of Smart Industry is gaining increasing importance in the current context because of its ability to automate industrial environments with great effectiveness using Smart systems. These systems incorporate detection, actuation and control functionalities.

The Portuguese industrial sector follows this trend and searches to technologically re-adapt its production processes to include sensing, monitoring and control functionalities, allowing efficient communication with the equipment on the shop floor with Industrial Internet of Things (IIOT) platforms [1].

This paper focuses on the development and testing of a Smartbox demonstrator, which allows sensing, monitoring and storage of equipment parameter's data from industrial environments through Wireless and Ethernet networks, using OPC UA (OPC Unified Automation), Message Queuing Telemetry Transport (MQTT) protocols.

II. Protocols

A. MQTT

MQTT, Created by IBM and by Eurotech, is a standard detailed by the MQTT v3.1 specification [2].

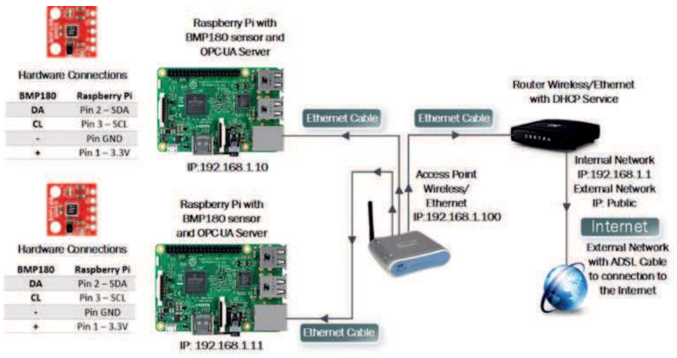
Hardware Connections
BMP180 Raspberry Pi
DA Pin 2 – SDA
CL Pin 3 – SCL
- Pin GND
+ Pin 1 – 3.3V
Raspberry Pi with BMP180 sensor and OPC UA Server
IP:192.168.1.10
Ethernet Cable
Ethernet Cable
Router Wireless/Ethernet with DHCP Service
Access Point Wireless/ Ethernet IP:192.168.1.100
Internal Network IP:192.168.1.1 External Network IP: Public
Internet
External Network with ADSL Cable to connection to the Internet
Hardware Connections
BMP180 Raspberry Pi
DA Pin 2 – SDA
CL Pin 3 – SCL
- Pin GND
+ Pin 1 – 3.3V
Raspberry Pi with BMP180 sensor and OPC UA Server
IP: 192.168.1.11
Ethernet Cable

Fig. 1 Communication with OPC UA protocol in a local network.

This communication standard follows a Publisher / Subscriber model, where a Publisher publish information to the Broker (Middleware). The Broker is responsible for receiving and centralizing messages from one or more Publishers, and to dispatch information to any authorized Subscriber requesting access to it, using applications for both PC and Smartphones.

B. OPC UA

OPC UA is a standard for communication and information modeling in industrial automation [3]. The specifications are standardized by IEC 62541. This standard allows the implementation of servers in embedded systems. This communication standard can use two communication models, the Request / Reply model and the Publisher / Subscriber model. Messages can be sent in binary format + TCP or Extensible Markup Language (XML) + HTTP / Simple Object Access Protocol (SOAP).

III. Prototype Development

A. Implementation with OPC UA

The OPC UA demonstrator's architecture is built upon Ethernet and Wi-Fi networks, as can be seen in Fig. 1. For the Raspberry Pi's to act as OPC UA servers, several required libraries were previously installed [4].

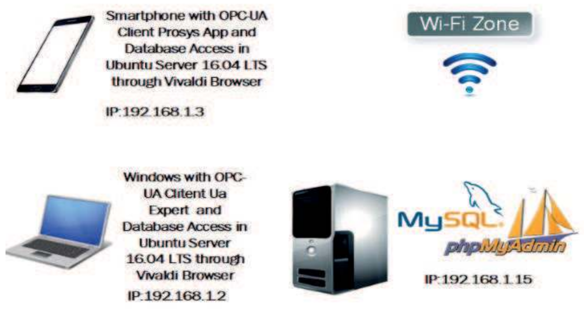
Smartphone with OPC UA Client Prosys App and Database Access in Ubuntu Server 16.04 LTS through Vivaldi Browser
IP:192.168.1.3
Wi-Fi Zone
Windows with OPC-UA Client Ua Expert and Database Access in Ubuntu Server 16.04 LTS through Vivaldi Browser
IP:192.168.1.2
MySQL
phpMyAdmin
IP:192.168.1.15

Fig. 2 Software and Hardware to subscribe and store OPC UA data through a WiFi network.

This work is funded by the Portuguese programme PORTUGAL 2020, under grant agreement n.º 024541 (PRODUTECH SIF - Soluções para a Indústria do Futuro).

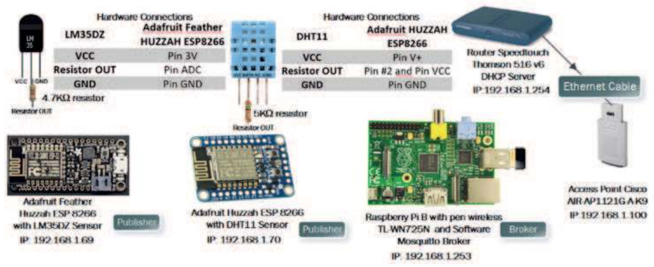

Fig. 3 Communication with MQTT protocol in a local network.

A PYTHON script was written to enable the acquisition of three environmental parameters (temperature, altitude and pressure) available in each BMP180 sensor, through the OPC UA Servers. It was also necessary to set up a *set_endpoint* consisting of an IP address, communication port, library and class, and also to define the object parameters, such as temperature, altitude and pressure, as well as access and permission rules of the OPC UA Clients.

To subscribe and store all data sent from the OPC UA server, we install a MySQL database on UBUNTU Server 16.04 LTS together with Unified Automation UA Expert software [5] on a WINDOWS 10 virtual machine, as shown in Fig. 2. On the Client side, we use a PROSYS Smartphone application [6] to subscribe to the sensor data parameters.

B. *Implementation with MQTT*

The network architecture presented in Fig. 3, demonstrates data acquisition using MQTT protocol [2]. Two ESP8266 modules were configured as Publishers and connected to a temperature sensor (LM35DZ) and a humidity/temperature sensor (DHT11). Topics (data) are sent from Publishers to a Broker (MOSQUITO) [7] installed on a Raspberry Pi platform. All authorized Subscribers can subscribe to the Broker platform for Topics; for the LM35DZ module the topics is "esp8266" and for the DHT11 module, the topics are "dht11humidity" and "dht11temperature".

Each Publisher module has received a script in C/C++ with network SSID and authentication credentials to access the wireless network and export the measured values from the sensors to the Broker. All Subscribers are been set up with Authentication to avoid unwanted access to the network and to the Broker. Several applications with graphical user interfaces (GUI) are available to interact with the parameters of the sensors. We use MyMQTT app for smartphone [8], MQTT.fx for laptop [9] with BODHI LINUX operating system and finally Node-RED [10] on WINDOWS 10 operating system in another Laptop, as shown in Fig. 4.

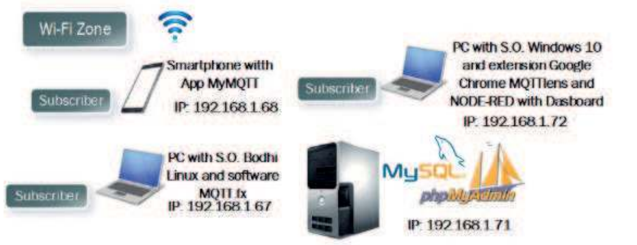
Wi-Fi Zone
Subscriber
Smartphone with App MyMQTT
IP: 192.168.1.68
Subscriber
PC with S.O. Windows 10 and extension Google Chrome MQTTlens and NODE-RED with Dasboard
IP: 192.168.1.72
Subscriber
PC with S.O. Bodhi Linux and software MQTT.fx
IP: 192.168.1.67
MySQL
phpMyAdmin
IP: 192.168.1.71

Fig. 4 Software and Hardware to subscribe and store MQTT data through a WiFi network.

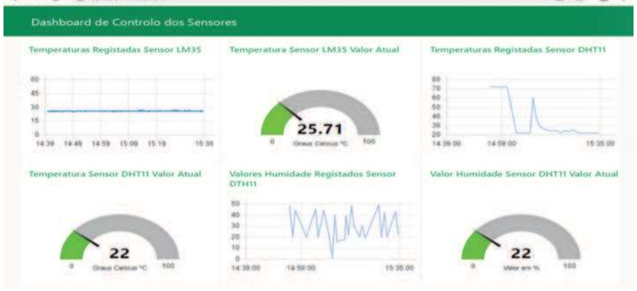
Dashboard de Controlo dos Sensores
Temperaturas Registadas Sensor LM35
Temperatura Sensor LM35 Valor Atual
25.71
Graus Celsius °C
Temperaturas Registadas Sensor DHT11
Temperatura Sensor DHT11 Valor Atual
22
Graus Celsius °C
Valores Humidade Registados Sensor DTH11
Valor Humidade Sensor DHT11 Valor Atual
22
Valor em %

Fig. 5 Sensor monitoring dashboard with Node-RED UI.

Node-RED flow is accessible through a localhost with IP address 127.0.0.1:1880 and the corresponding dashboard is accessible through IP address 127.0.0.1:1880/ui/#0 as can be seen in Fig. 5. With Node-RED, we can observe the sensor data and simultaneously send it to a MySQL database configured on UBUNTU Server 16.04 LTS, and then export it to PDF, Excel or CSV files.

IV. Conclusions

The development of this demonstrator has created a affordable solution of a smartbox for industrial applications, based on the Raspberry Pi platform. It can monitor critical parameters of the shop-floor factory through open-source software, both for smartphone and Desktop / Laptop computers, as well as storing data for remote analysis. Some limitations remain to find libraries suitable to program alarm features in the OPC UA Servers. MQTT protocol is a lightweight an easy to implement M2M solution in shop-floor with limited or inexistent Ethernet network infrastructure. Future work will include additional sensors for industrial environment (vibration and angular rotation speed), alternative hardware platforms with more computational capabilities, and the implementation of Machine Learning algorithms for predictive maintenance based on the collected sensor data.

References

[1] M. Schleipen, "OPC UA supporting the automated engineering of production monitoring and control systems", Emerging Technologies and Factory Automation 2008. ETFA 2008. IEEE International Conference, pp. 640-647, 2008.

[2] MQTT Version 3.1.1. Edited by Andrew Banks and Rahul Gupta. 29 October 2014. OASIS Standard. Retrieved from: http://docs.oasis-open.org

[3] W. Mahnke, S. H. Leitner, and M. Damm. 2009. OPC Unified Architecture (1st ed.). Springer Publishing Company, Incorporated.

[4] FreeOPCUA [Computer Software]. (2018). Retreived from: https://github.com/FreeOpcUa/python-opcua

[5] Unified Automation [Computer Software]. (2018). Retreived from: https://www.unified-automation.com/

[6] PROSYS [Computer software]. (2018). Retrieved from: https://www.prosysis.com

[7] MOSQUITTO [Computer software]. (2018). Retrieved from: https://mosquitto.org

[8] MyMQTT [Computer software]. (2018). Retrieved from: https://play.google.com

[9] MQTT.fx [Computer software]. (2018). Retrieved from: https://mqttfx.jensd.de

[10] Node-RED Programming Guide [Online]. Retrieved from: http://noderedguide.com