Application of IoT concept on predictive maintenance of industrial equipment

Radu Constantin Parpala[1,*], and *Robert* Iacob[1]

[1]University Politehnica of Bucharest, Machine and Manufacturing Systems Department, Bucharest, Romania – 060042

Abstract. The Internet of Things (IoT) concept describes an intelligent connectivity of smart devices using the internet network. Nowadays, companies try different approaches for predictive maintenance as a solution to reduce costs and the frequency of maintenance activities. The IoT platforms provide a good support for predictive maintenance as it can integrate information from different machines and manufacturing systems. The main drawback in integrating production system with IoT dedicated platforms is the communication framework, knowing that the main industrial communication protocols are incompatible with modern communication protocols implemented on IoT platforms. In this context, the present paper proposes a new and simple method for on-line monitoring and predictive maintenance of industrial equipment. This method has two features of connected manufacturing. One of these is process monitoring for constant quality assurance, the other one is condition monitoring in order to prevent unplanned downtimes. A case study is presented to demonstrate the feasibility of the proposed method.

1 Introduction

The Internet of Things was described as the intelligent connectivity of smart devices by which objects can sense one another and communicate, thus changing how, where and by whom decisions about our physical world are made [1]. At first, the IoT concept was implemented on some commercial products: refrigerators, washing machines, control devices for smart homes, allowing them to directly connect to the internet using mobile or private networks. As the concept evolved, other types of products where equipped with the necessary technology and today many companies are trying to implement the IoT concept into modern manufacturing systems. Collecting usage data from different production equipment using an IoT platform is the main step in building a general, cloud based, predictive maintenance system, thus simplifying the factory upkeep. The main drawback of using the IoT technology in manufacturing systems is the communication (connection) between standard industrial devices and the web platforms. To solve this issue, the current paper proposes a new method to translate industrial equipment language to IoT ready devices like Arduino or Raspberry-Pi microcontrollers.

[*]Corresponding author: radu.parpala@gmail.com

2 IoT platforms

In the early 2000s, before the widespread of the IoT concept, different techniques were deployed in industry for machinery diagnostics and prognostics. A well-known method is Condition-Based Maintenance (CBM) which consists of three main steps: data acquisition, data processing and maintenance decision-making. It is a maintenance program that recommends maintenance decisions based on the information collected through condition monitoring [2].

Knowing that a single product is no longer sufficient to cope with the dynamic market environment, more recently, firms are trying to provide the integrated offering of products and services. In this regard, a generic structure of product-service integrated roadmap was developed [3].

Nowadays, the manufacturing paradigm changes towards predictive manufacturing, the role of maintenance function within manufacturing needs to be refined as a value creation function for achieving more sustainable operations. In this context, new techniques for maintenance are proposed [4]. Knowing that fault prediction represents a key technology to ensure sustainable operations, currently different fault prediction systems, based on IoT, are created [5].

The purpose of IoT devices is to connect with other devices or applications, this task being achieved by IoT platforms. Generally, an IoT platform is used as front end or a dashboard to present and process machines data. For simple IoT applications, there are several commercial or open source platforms that are already configured and ready to use. Most of the IoT platforms come with programming libraries for common developer boards like Arduino, Linux based devices (Raspberry Pi) or ARM based devices. For complex industrial application [6-7], developers can install an IoT platform like Easy-IoT, Kaa, Microsoft Azure IoT suite, PTC ThingWorx etc. These platforms come with a server installation and SDK libraries for different OS platforms.

Interoperability, which is dependent upon communication protocols and standardization level, is the ability of different IT systems and software to communicate, exchange, and use data. It's key to crafting a successful and resilient solution, regardless of whether an organization builds its IoT system in-house or engages a vendor. The communication between devices and IoT platforms is based on web protocols like HTTP GET and HTTP POST or Java Script data-interchange format (JSON). Given that most of those communication protocols are not implemented on industrial devices, one rapid solution is to use Arduino as a translator between industrial machines and IoT platforms (Figure 1).

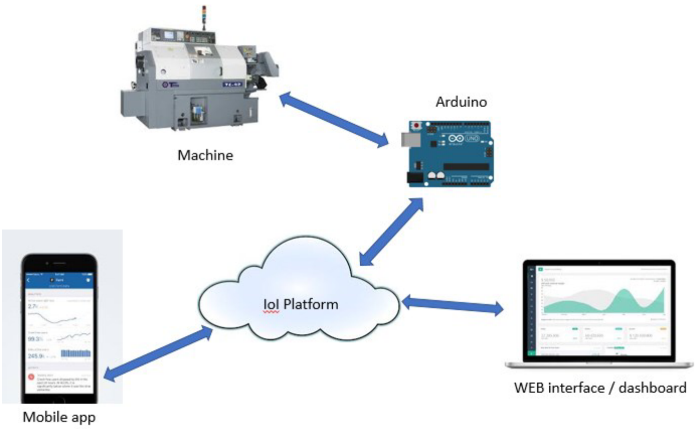


Fig. 1. Simplified scheme of an IoT platform that uses an Arduino board

Table 1. Serial communication standards

Serial Standard	Operation mode	Total nr. of devices	Cable length	Speed	Wires
RS-232	Single Ended	1 Sender / 1 Receiver	15 m	20 Kbits/s	min. 3
RS-422	Differential	1 Sender / 10 Receiver	1200 m	10 Mbit/s	4
RS-485	Differential	32 Sender / 32 Receiver	1200 m	10 Mbit/s	2

3 Industrial communication protocol

Industrial devices use several serial protocols to communicate one with another. A serial communication protocol can only send one bit at a time. This is slower than parallel communication standards, but it can be used over longer distances. When using serial communication, two aspects should be considered: the hardware specification and the software format of the data.

3.1 Hardware standards used in industrial communications

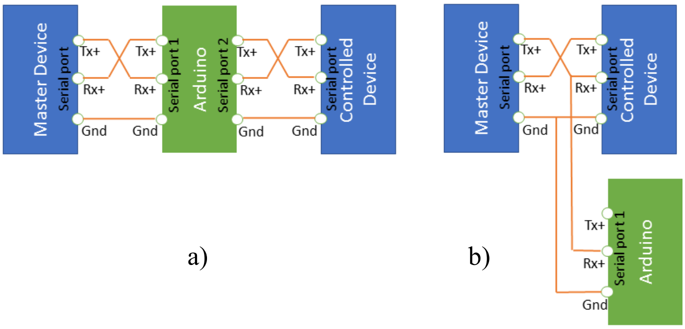
The main difference between different communication standards, as can be seen in the Table 1, consists in the number of devices that can communicate in the same time, the speed and the maximum cable length.

Because of specific hardware limitation on industrial machines, RS-232 standard is usually used to connect to a computer or to some types of HMI devices.

3.1.1 Standard RS-232

The RS-232 protocol is the most common serial interface mainly because almost every computer has at least one RS-232 serial port. The main disadvantages of this standard are: limited cable length, possibility of noise occurrence and limited number of devices that can be connected at the same time.

Usually, the RS-232 standard connects only two devices. Thus, in order to be able to capture usage information that is sent over the serial port for the IoT platform, a new device could be connected as an echo device – if it has two serial ports (Figure 2.a.), or it can be connected only to a send line of a master device (Figure 2.b.) – in this case it can only receive information from a single device.



a) b)

Fig. 2. Arduino interface in RS-232 serial networks [8]: a. In line connection; b. Serial connection

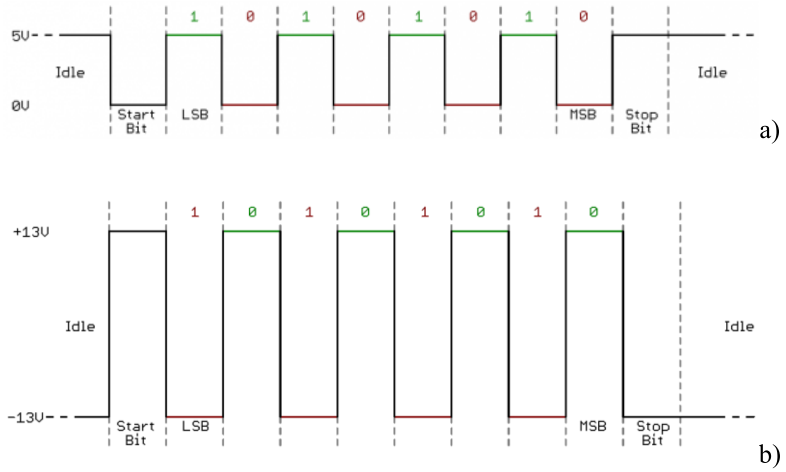


Fig. 3. Serial port voltages [9]: a. TTL Serial; b. Standard Serial

Another drawback of interconnecting industrial serial and Arduino consists in the difference between voltages used by communication ports. Arduino uses TTL serial and industrial equipment uses standard serial. The Arduino serial port uses a voltage of 3.3v or 5v (Figure 3.a.), while all industrial equipment use a voltage between -13v and 13v (Figure 3.b.), so, obviously, connecting the Arduino to a machine's serial port will result in burning the Arduino port. In this case a convertor is required to transform Standard serial to TTL serial.

3.1.2 Standard RS-422

The RS-422 protocol was initially implemented on Macintosh as the serial standard interface. It became fast an industrial standard because it uses differential electrical signal, thus permitting more devices to use the same bus, and it also has greater noise immunity, thus allowing longer cable lengths. With RS-422 every device has a unique ID, data is sent from the master device to all the devices that are connected in the same network, but only the destination station, with the required ID, will process the request.

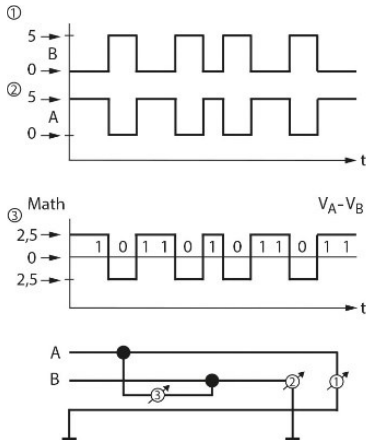


Fig. 4. RS-485 voltage levels [9]

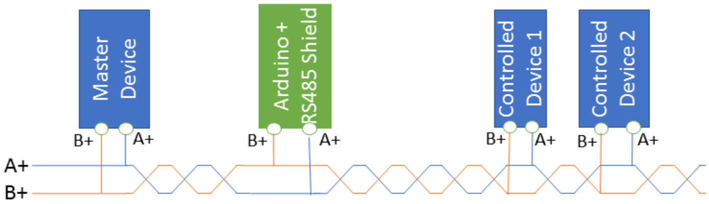


Fig. 5. Arduino interface in a RS-485 network [8]

3.1.3 Standard RS-485

The RS-485 protocol is an improvement of the RS-422 protocol, as it can handle more devices and it accepts more than one master device as long as only one master device will send data at one time. RS-485 is a superset of RS-422, so any device that supports RS-485 will be able to communicate with a device that only supports RS-422. As the voltage level between the two differential wires usually has a value lower than 5v (Figure 4) it is possible to connect an Arduino board directly to an RS-485 line (Figure 5).

3.2 Software format of the serial standards

The RS-485 data is sent using the ModBus protocol. ModBus was first developed for Programmable Logic Controllers (PLCs) and it became a standard communication protocol for many industrial devices. ModBus enable data communication between several devices connected in the same network. A typical ModBus data packet consists of: a destination station address, function code, data area and cyclic redundancy check (CRC). There are two protocol versions used over RS-485 serial: ModBus RTU and ModBus ASCII.

ModBus RTU is the most common implementation used in serial communication. It uses a compact binary representation of data. In ModBus RTU data is sent without spaces, the messages being marked at the beginning and at the end with a silence period of 31/2 characters. In ModBus RTU each byte is sent as a string of eight binary characters enclosed by a start bit and a stop bit.

ModBus ASCII marks the start of each message with a colon character and the end of the message with a carriage return and line feed characters, thus allowing spaces inside data messages. In ModBus ASCII there are only 7 data bits, but there is one extra parity bit before the ending bit.

4 Case study

To demonstrate the viability of the proposed method, the operating parameters of an automatic machine were monitored (Figure 6). The equipment used for testing is an automatic polishing and sanding machine for high gloss lacquered furniture components (painted MDF – medium-density fibber). The automatic machine offers the possibility to obtain quality products in a short time with limited human intervention. The equipment is accessible to small firms, has a reduced size: 1500×1100×1200 [mm] ideal for workshops with a reduced floor space, allows rapid reconfiguration – depending on the size of the plate, and simple operation. The machine has three types of moves: the motion of the conveyor table, with the fixed wood panel; the motion of the brush head in a transversal direction to the wood panel; continuous rotation of the brush.

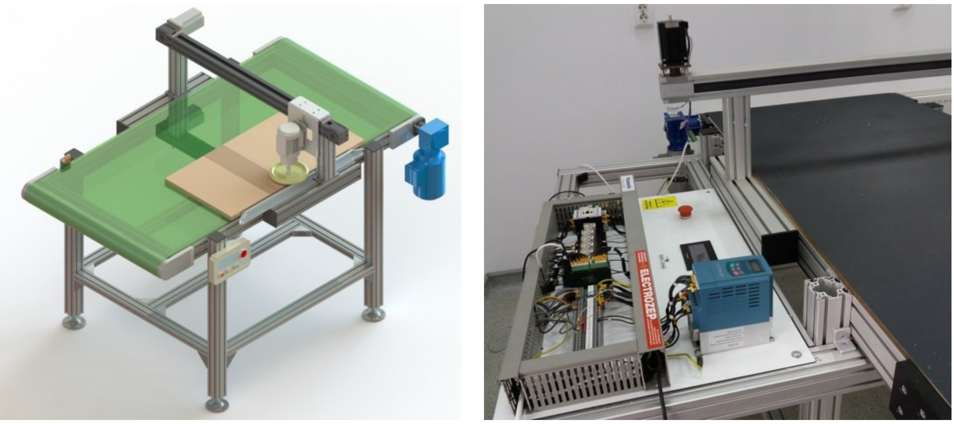


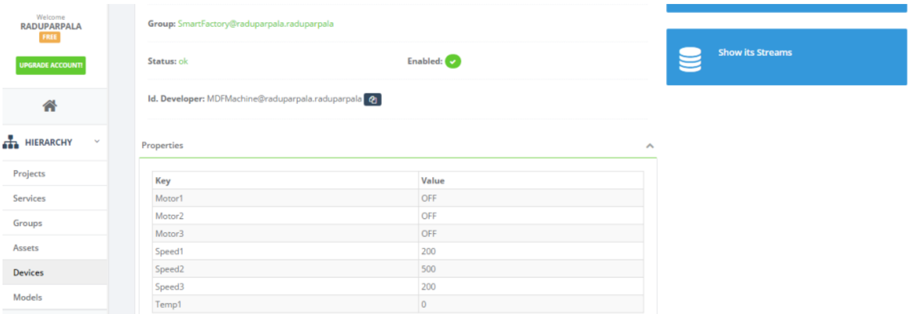
Fig. 6. Automatic polishing and sanding machine: a. Virtual prototype; b. Equipment

The machine's PLC uses RS-485 / ModBus RTU communication standard to control the 3 motor drivers. For preventive maintenance purposes, the speed and the functioning time of all motors are recorded and exported to an IoT platform.

For this case study, an open source IoT platform named Carriots (Figure 7) was chosen. It offers some predefined functions that are indispensable for the proposed predictive maintenance system. In the first step, only 7 parameters where included in the developed monitoring system: working status of the motors, speed of the motors and the temperature measured near the main (first) motor.

The hardware required to connect the polishing and sanding machine to the IoT platform include an Arduino Uno board, equipped with an Ethernet shield, and one serial RS-485 shield. The Arduino waits continuously for the RS-485 serial commands and, if a Start or a Stop command is received, based on the station number, it then sends the data to the IoT platform (Figure 8).

For this proof-of-concept application, the Arduino device was used because it packs considerable power on a very small board and it provides many opportunities for automation, networking and data collection. It should be noted that we are aware of the existing drawbacks that generally limit the use of Arduino based devices in industrial application: low level voltage, robustness and safety. However, nowadays, more and more research systems are based on Arduino and similar devices as the main disadvantages are overcomes: underwater sound recorders for marine research, robots for assembling gene-editing tools and even a system developed for an outreach project on the International Space Station [10].



Group: SmartFactory@raduparpala.raduparpala

Status: ok

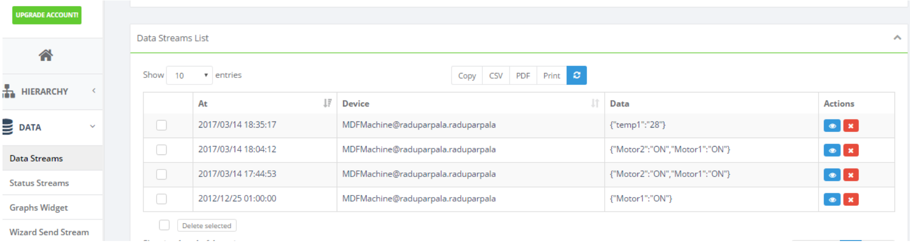
Enabled: ✓

Id. Developer: MDFMachine@raduparpala.raduparpala

Properties

Key	Value
Motor1	OFF
Motor2	OFF
Motor3	OFF
Speed1	200
Speed2	500
Speed3	200
Temp1	0

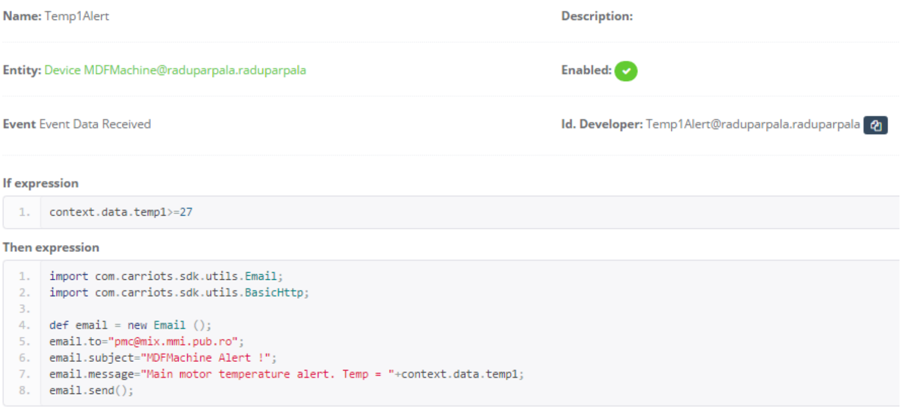
Fig. 7. Carriots Platform



Data Streams List

	At	Device	Data	Actions
□	2017/03/14 18:35:17	MDFMachine@raduparpala.raduparpala	{"temp1":"28"}	
□	2017/03/14 18:04:12	MDFMachine@raduparpala.raduparpala	{"Motor2":"ON","Motor1":"ON"}	
□	2017/03/14 17:44:53	MDFMachine@raduparpala.raduparpala	{"Motor2":"ON","Motor1":"ON"}	
□	2012/12/25 01:00:00	MDFMachine@raduparpala.raduparpala	{"Motor1":"ON"}	

Fig. 8. IoT Stream data



Name: Temp1Alert

Description:

Entity: Device MDFMachine@raduparpala.raduparpala

Enabled: ✓

Event Event Data Received

Id. Developer: Temp1Alert@raduparpala.raduparpala

If expression

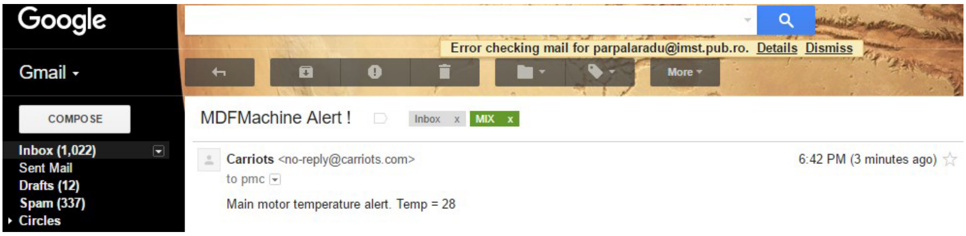
```
context.data.temp1>=27
```

Then expression

```
import com.carriots.sdk.utils.Email;
import com.carriots.sdk.utils.BasicHttp;

def email = new Email ();
email.to="pmc@mix.mmi.pub.ro";
email.subject="MDFMachine Alert !";
email.message="Main motor temperature alert. Temp = "+context.data.temp1;
email.send();
```

Fig. 9. E-mail rule definition



Error checking mail for parpalaradu@imst.pub.ro. Details Dismiss

MDFMachine Alert !

Carriots <no-reply@carriots.com>
to pmc

6:42 PM (3 minutes ago)

Main motor temperature alert. Temp = 28

Fig. 10. E-mail alert

The main function of the developed predictive maintenance system is to send usage alert via e-mail or other messaging services. For testing purposes, a new alert was defined for a temperature of 27 °C (Figure 9). Using this function, each time the defined threshold is reached, an e-mail alert is send to the user defined (Figure 10). In a similar way, other parameters can be added and different alerts can be defined. Thus, using the proposed method, an on-line monitoring and predictive maintenance system can be easily developed for any type of industrial equipment.

5 Conclusions

Currently, IoT is proving to be a game-changer for many companies as a variety of industries begin employing IoT-enabled architectures and experimenting with how IoT solutions can bring new benefits. From self-driving cars to smart homes filled with voice activated devices, the innovation of IoT's connectivity never fails to impress. However, many of the machines used in skilled trades or manufacturing are still not connected to IoT platforms because they lack sensors, software and connections to IT systems [11].

In this context, the paper presented a new and simple method for on-line monitoring and predictive maintenance of industrial equipment. This method is based on an integration of devices with software and it contains a procedure to translate different industrial equipment language into web protocols, thus creating a way to efficiently implement predictive maintenance in industrial systems.

In order to demonstrate the feasibility of the developed method, a case study is presented. The operating parameters of an automatic polishing and sanding machine for high gloss lacquered furniture components were monitored and a threshold alert was defined. In a similar way, using the presented method, an on-line monitoring and predictive maintenance system can be easily developed for any type of industrial equipment. It should be noted that for complex industrial applications a more robust device is envisaged – a similar system based on a FPGA process computer.

This work has been funded by University Politehnica of Bucharest, through the “Excellence Research Grants” Program (UPB-GEX). Identifier: UPB-EXCELENTA-2016. Research project title: “Monitorizarea inteligenta integrata a sistemelor de fabricatie pentru asigurarea mentenantei preventive a acestora”. Contract number: 34 / 26.09.2016.

References

1. M. Albert, Modern Machine Shop Magazine (2015)
2. A.K.S. Jardine, D. Lin, D. Banjevic, Mechanical Systems and Signal Processing, **20** (2006)
3. Y. Geum, S. Lee, D. Kang, Y. Park, Journal of Engineering and Technology Management, **28** (2011)
4. J. Lee, M. Holgado, H.A. Kao, M. Macchi, IFAC Proceedings, **47** (2014)
5. X. Xu, T. Chen, M. Minami, Computers & Mathematics with Applications, **64** (2012)
6. Q. Wang, J. Gao, Journal of Loss Prevention in the Process Industries, **25** (2012)
7. L. Dong, R. Mingyue, M. Guoying, Procedia Engineering, **174** (2017)
8. *** https://www.arduino.cc/en/Main/Products, Accesed 10.02.2017
9. *** http://www.omega.co.uk/techref/das/rs-232-422-485.html, Accesed 15.02.2017
10. D. Cressey, Nature journal, **544** (2017)
11. ***http://www.bosch-presse.de/pressportal/de/en/69632.html, Accesed 18.03.2017