IIoT Application over an Energy Recovery System

1st Mario Valagao Sancho
depto. de electrónica
Facultad de Ingeniería, UNPSJB
Comodoro Rivadavia, Argentina
mario.valagao.s@gmail.com

2nd Ricardo R. Peña
depto. de electrónica
Facultad de Ingeniería, UNPSJB
Comodoro Rivadavia, Argentina
ramirop@unpata.edu.ar

3rd Roberto D. Fernández
depto. de electrónica
Facultad de Ingeniería, UNPSJB
Comodoro Rivadavia, Argentina

4th Marcelo A. Lorenc
depto. de electrónica
Facultad de Ingeniería, UNPSJB
Comodoro Rivadavia, Argentina

5th Sol Maldonado Betanzo
Instituto Balseiro (CNEA y UNCuyo)
San Carlos de Bariloche, Argentina

***Abstract*—In this paper, a low cost OPC UA/Modbus gateway for multiple Modbus devices is developed on a Raspberry Pi 2 computer. This computer runs a Slackware GNU/Linux operating system. In addition, Node-RED software is used to create a user-friendly interface with the developed system and store variables in a database for later analysis. This low cost tool is used to acquired data from an energy recovery system and to obtain different dynamic models.**

***Index Terms*—IIOT, Instrumentation, Automation, System identification, Energy recovery, Free software**

I. Introduction

Industry 4.0 concepts seeks the connection of all kind of devices to the Internet aiming to exchange data and contributing to collect and analyze data from production processes. For this reason, the expansion of the Internet of Things (IoT), particularly, the Industrial Internet of Things (IIoT) is favored [1]–[3]. It is well known that the concept of IoT is the beginning of the fourth industrial revolution being considered a key part of the future [4]. Nowadays, industrial communications systems connect only primary devices as PLCs (Programmable Logic Controllers) with SCADA (Supervisory Control And Data Acquisition) systems. A modern IoT-based system should communicate all kind of devices, databases, web resources and users. In addition, an IoT system may incorporate tools to transform and process data [4]. Currently, IoT applications are evolving to multi-sector systems. Hence, they are encouraged to the development of IIoT is centered in the expansion of IoT on industrial enviroments which require more complex features than a 'normal' application of IoT [5]. To expand IIoT it is appropriate to use the industrial communication protocol OPC UA (acronym for Open Process Communications Unified Architecture). OPC UA is scalable, secure, has good performance and is independent of the platform where it runs [2], [6]. Currently, OPC UA is the most widely used communication protocol on cloud-based platforms [7]. In this matter, in industrial environments the most accepted communication protocol to interconnect machines (M2M) and to link Supervisory Control And Data Acquisition (SCADA) systems is OPC UA. However, many devices and PLCs use the Modbus industrial communication protocol [8].

Node-RED is a tool created by IBM® in order to simplify the task of developing an application. This is achieved by dragging and dropping blocks and connecting them together [3], [9]. In recent years, this tool has been used in different applications related to IoT [10]–[12].

Two important functionalities in IIoT are the human machine interface (HMI) and a service called ”historian” provide a graphical interface for the system operator and collect and store data efficiently in a database from various devices. Different companies offer a complete solution for these two functionalities. In this sense, Node-RED and a relational database management system, such as MariaDB [13], can be used to build a low-cost HMI and a historian.

In the last time, energy recovery systems have gained impulse. Particularly, in this work, a pump used as a turbine (PAT) works as pressure regulating system but also as a recovery one in a water dsitribution system. In the framework of the project “Energy recovery and pressure control in the water distribution system of Comodoro Rivadavia” a PAT prototype was built in collaboration between “Sociedad Cooperativa Popular Limitada de Comodoro Rivadavia” (SCPL) and the Engineering School, Universidad Nacional de la Patagonia 'San Juan Bosco'. As part of this project, the modeling and control of a centrifugal pump working as a turbine, are showed in [14]. The development of a client for the Modbus industrial protocol and a first identification of the system are introduced in [15]. Furthermore, a first approximation of a gateway OPC UA/Modbus and a system identification is presented in [16]. This work is a continuation of the previous ones and it focuses on the addition of two or more devices to the gateway OPC UA/Modbus. In this way, one of the main goals of this work is providing an OPC UA interface for different devices and low-cost PLCs. In order to accomplish this objective, a Raspberry Pi 2 (RPi2) computer [17] running Slackware GNU/Linux operating system with support for ARM architecture [18] and Python language for the development of the gateway OPC UA/Modbus, is used.

Additionally, the Node-RED software is used for the construction of an HMI. The acquired data are stored in a database server for later analysis and displayed in the graphical interface. This user interface can be used with any web browser.

Furthermore, a multiple input-single output (MISO) model of the system is obtained using least squares. It is worth noting that, all the stages of this work are developed using free software. This represents an advantage because of the proposal is modular, which means that, the database server or Node-RED can run on another computer connected to the network optimizing the resources of the Raspberry Pi computer.

The structure of this paper is as follows. Section II introduces a brief description of the protocols used in this paper. The development of the gateway OPC UA/Modbus is presented in section III. In Section IV the Node-RED development tool is presented and the construction of the developed software is explained in detail. In Section V the energy recovery system and associated devices are exposed. In addition, in Section VI, experimental results are showed. Finally, conclusions are summarized.

II. OPC/UA Modbus communication

This section briefly presents communication protocols Modbus and OPC UA. The first protocol is used to communicate a PLC and an instrument with the RPi2 and the second to generate a server that accepts multiple connections through the ethernet interface.

A. *Modbus*

Modbus, Modicon® (1979), is an industrial communication protocol designed for supervision and control of PLCs and devices. Despite of time, this protocol is widely used in industry due to its flexibility. Originally, the industrial protocol was designed as a master-slave architecture. There are numerous versions of the Modbus protocol, including the version for serial port RTU (acronym for Remote Terminal Unit) and TCP (acronym for Transmission Control Protocol) communications networks [8].

B. *OPC UA*

The OPC standard is a series of specifications developed by industry vendors, end-user and software developers. These specifications define the interface between clients and servers, as well as servers and servers, including things like access to real time data, historical data, monitoring of events and other useful applications.

Considering the successful adoption of OPC in thousand of automation systems and the advancement on industry technology, it was necessary to adapt this protocol to new ways of communication and interaction between systems and platforms. In 2006 OPC Foundation proposed OPC Unified Architecture (OPC UA) to acknowledge the problems and to create a better approach to new internet platforms, connection through firewall and other convenient characteristics without losing performance or old features. OPC UA has two fundamental components, which are transport mechanisms and data modeling. For transport mechanism OPC UA defines protocols for high performance internet communications as well as another internet standards such as Web Services, XML and HTTP for firewall-friendly internet communication. For data modeling part OPC UA defines necessary rules and base building blocks to make an information model [6].

OPC UA has several features, one of them offers an abstraction layer between devices (e.g. PLC, sensors and actuators) and plant-floor applications [6]. Furthermore, it also implements security, allowing data flow to be signed and encrypted, ensuring industrial information safety and reliability. Another distinctive feature is the possibility to add time stamps to OPC UA data protocol. One of the main advantages of this protocol is allowing to connect various operating systems and platforms [19].

III. OPC UA/Modbus gateway development

In this work, Slackware ARM Linux was chosen as the operating system for the RPi2 computer due to its ease of use and stability.

Figure 1 displays a connection diagram of devices. In the scheme, an Allen Bradley® Micro 820 PLC connected via ethernet to the RPi2 is shown and an energy meter from Schneider Electric® is connected via a RS-485/USB device to RPi2.

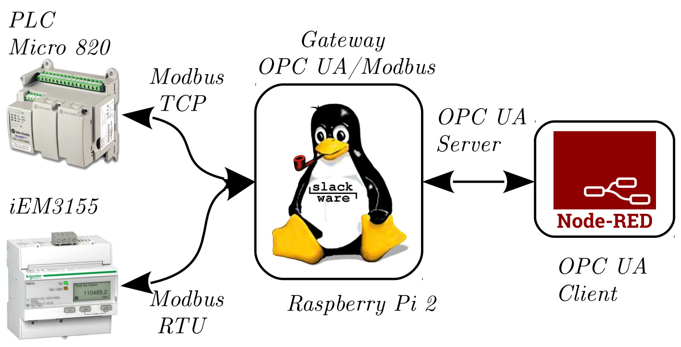

Fig. 1. Scheme of the communication between PLC, iEM3155 and RPi2

Meanwhile RPi2 is the Modbus master, devices connected via ethernet and RS-485 are Modbus slaves. Also, to complete the gateway, the OPC UA server accepts connections on a given port. Thus, the system can serve different customers. In this case, an OPC-UA client was built using Node-RED, Figure 1. For the development of the OPC UA/Modbus gateway, Python 3 language was used due to its abundant documentation and flexibility [20]. The developed OPC UA client is explained in detail in section IV.

There are different implementations of the Modbus protocol for Python, in this project *"pymodbus"* with the *"Free OPC-UA"* library are used to create a Modbus TCP/RTU to OPC UA gateway. In this implementation of the OPC UA/Modbus gateway, two devices are accessed, shown in Figure 1, with different data transmission rates. One of the devices, the energy meter provides a large number of variables and the data are unsigned 64-bit integers, 32-bit floating and 16-bit floating [21]. The second device connected to the RPi2 is the PLC Micro 820 with a Modbus map with 16-bit unsigned variables. Then, because of the energy meter uses a large number of variables and is slower than the PLC, a program executing one process cannot ensure a proper sampling time. As it is known, modern operating systems such as GNU/Linux can run multiple concurrent processes. To achieve this, the CPU switches from one process to another. A process is composed by its memory space, its system environment variables and I/O resources [20]. In many cases, there are activities that take place at the same time. These activities are divided into mini-processes called threads. A thread can be seen as a light process because threads are generally executed within the context of a parent process. An interesting feature of the threads is that they have the ability to share an address space and all its data with the main process. Furthermore, threads are lighter than processes but also the first ones are easly created and destroyed [22].

The Python language has support for concurrent programming and threads are implemented using the *"Threading"* library [20]. The pseudo-code of the OPC UA/Modbus gateway used in this paper is showed below:

Algorithm 1 Software that runs on the RPi2.

```
procedure OPC UA/MODBUS GATEWAY
    Modbus master configuration.
    Initialize OPC-UA Server.
    Initialize thread1
    while $i < NM$ do
        Read PLC variables.                     ▷ main program.
        Read energy meter Modbus RTU.           ▷ thread 1.
        Write variables to OPC-UA Server.
        $i = i + 1$;
        delay($t_d$);
    Close thread1.
    Close the Modbus connection.
    Close the OPC-UA server.
    end.
return ;
```

The first part of Algorithm 1, shows how the program is initialized: serial communication is first configured by Modbus RTU and Modbus TCP to read energy meter data and PLC data, respectively. The OPC UA server is then configured (to listen on port 4840 in this work). Then, *thread1*, which reads data from the energy meter and copies data to the OPC UA server, is started. Next, the program enters in a loop where PLC variables are read through the Modbus protocol, these variables are written in the OPC UA server. An i counter, which represents the amount of measurements made, is increased and a t_d delay occurs. The variable NM is the maximum value of measurements previously configured. When $i < NM$ is false, Algorithm 1 is finished.

IV. OPC UA Client and HMI using Node-RED

Node-RED is a flux-based development tool created by IBM® for wiring hardware devices together, Application Programming Interfaces (APIs) and online applications as a way to develop IoT and IIoT. Node-RED provides a browser-based flow editor allowing the creation of simple JavaScript programs with a wide variety of palettes and the additional advantage of testing and debugging within the browser. It also allows users to design their own blocks and palettes for functions not yet implemented. Both Node-RED development environment and Node-RED user interface (UI) can be reached through a web browser aslant the IP address of the machine where the software is running [9], [12]. Furthermore, Node-RED software allows users to communicate with external databases with the purpose of storage and management of variables, and also the retrieval of this data for other applications. A way to do this is using MySQL nodes.

For the sake of clarity, Figure 2 shows part of the flow created in Node-RED. The OPC UA client was developed using a Node-RED toolbox. The first part of the flow inside the yellow box shows the OPC UA client, in which OPC UA server is accessed through IP address and port. On the left side of the chart the timestamp node identify OPC variables for later use in the next node, whose function is to retrieve the actual values of those variables and then send them to the last filter node, to compress and simplify data to be used and stored. In addition, this block is executed every certain time interval. The interval which is adjusted from the development environment. Then the OPC UA connector is configured and the server variables are read.

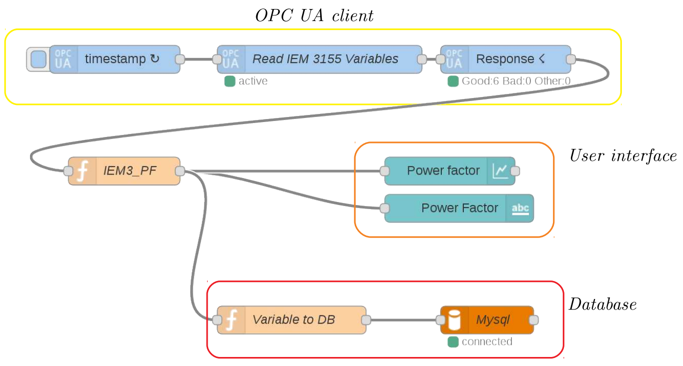

Fig. 2. A part of the flow created with Node-RED

Afterwards, the data goes through a function node that extract the value and then the path is splitted in two, one of them goes to the MySQL database communication nodes (red box) and the other path goes to the user interface creation nodes (orange box). Orange box in Figure 2 shows the block "Variable to DB". This block takes the message with the variable and saves them in the specified database table. The next block "MySQL" is the block that manages the connection to a database server, local or remote.

User interface nodes possess several options for data visualization customization, making possible a really useful HMI.

In this way, users can analyze real time information obtained from different devices in a set of tabs or even different screens, which highlights the benefits of this platform in an IIoT implementation. The throughout results of this HMI application are shown in the next section.

It should be noted that implementation using free software allows modularity of this proposal. For example, the database server and Node-RED can run on a dedicated server. This server may have the ability to make backups with which the information can be protected. Then, the RPi2 load can be optimized and system operation data can be backed up.

V. Test Bench

In this section, the energy recovery test bench, built at the Control and Automation Laboratory of the UNPSJB and its main components are presented. A schematic of the prototype is displayed in Figure 3. The diagram shows a hydraulic pump working as a turbine (PAT) connected to a 4 quadrant variable speed drive. Also, the drive is connected to the electrical grid by means of a filter and the aforementioned energy meter is connected in between the line filter and the grid. The variable speed drive receives the set point velocity from the PLC and, the four quadrant drive, returns an estimate of the speed to the PLC. The PLC measures, also, input and output PAT pressures and flow rates with a 4-20 mA current loop. Also, other current loops communicate the PLC and the variable speed drive.

Both, PLC and RPi2 are connected to the ethernet network of the laboratory. RPi2 is also connected to the energy meter via an USB/RS-485 converter. As indicated, the RPi2 1) takes data from the energy meter and the PLC and 2) executes the client developed eith Node-RED. It should be noted that, the OPC UA client can run on another computer. On the other side, the energy meter iEM3155 from Schneider Electric® measures different system parameters and incorporates an RS-485 communications port with a transfer rate of 9600-38400 bps.

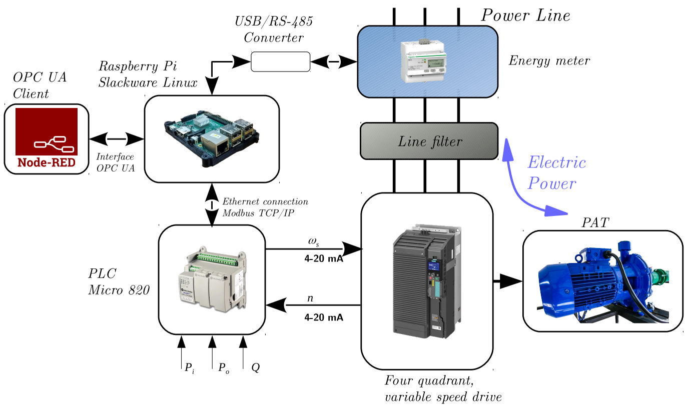

Fig. 3. Complete system scheme

To analyze the complex power of the energy recovery system, the power factor register is read. The energy meter interprets the energy sent or received according to the direction of the power flow. In this sense, the positive power flow real power (P(+)) and imaginary power (Q(+)) indicates that power is flowing from the power supply to the load. In addition, the direction of the real power flow is related to the sign of the power factor [21]. Modbus maps of the power meter and the PLC are presented in Tables I and II, respectively.

Overview	Registers	Type	Unit
Real Power	3060	Float 32	kW
Reactive Power	3068	Float 32	kVAr
Power Factor	3192	Float 32	-
Frequency	3110	Float 32	Hz

TABLE I
iEM3155 Modbus map

Overview	Registers	Type	Unit
Input Pressure	40001	Unsigned 16 bit	kg/cm^2
Output pressure	40002	Unsigned 16 bit	kg/cm^2
Speed	40003	Unsigned 16 bit	RPM
Flow	40004	Unsigned 16 bit	$m^3/hour$

TABLE II
PLC Modbus map

VI. Experimental Results

In this section some tests performed with the OPC UA/Modbus gateway, the HMI over the energy recovery system are presented.

Previous papers showed experimental measures and mathematical models utilized to characterize the test bench system [14]–[16], hence this work tries to continue in that way using new tools to find a better system characterization, friendly interfaces and looking for real time applications.

The flow, developed in Node-RED, in Figure 2 has the double purpose of visualizing all the variables in a web browser and also to save the data in a database. Figures 4 and 5 taken from screen captures of a computer web browser. Both of them show graph and texts related to the variables. These data are later processed to obtain a mathematical model. In the figures it can be observed how different speed steps are performed and how the system variables evolve. Figure 4 shows PLC variables being read in real time, they are output pressure, input pressure, speed and flow rate. Furthermore, a gauge chart allows a better visualization of the speed of the PAT to operate the system.

The other tab of the user interface, Figure 5, shows the electrical variables acquired by the energy meter. In this work, only active and reactive powers, frequency and power factor variables are read. Those represents the interconnection of the energy recovery system with the electrical grid. As can be seen in Figure 5, the active power and power factor is negative below 2000 RPM. Then it becomes positive due to the change in power flow.

Because the objective of the test bench and its future implementation in the water distribution system, is to control PAT's output pressure and, at the same time, to recover energy from the input/output drop of pressures, the PAT models are, briefly, introduced. In this sense, it is relevant to write a model that takes into account the input and output pressures to the system, the flow rate and the PAT shaft speed. Then, considering that the system output is the output pressure, the following model can be written:

$$y(k) = \begin{bmatrix} H_{1,i}(z) & H_{2,i}(z) & H_{3,i}(z) \end{bmatrix} \begin{bmatrix} n \\ Q \\ P_i \end{bmatrix}, \qquad (1)$$

where $H_{1,i}(z)$, $H_{2,i}(z)$ $H_{3,i}(z)$ are the transfer functions, i are different models proposed (for steps at 1000-1500, 1500-2000 and 2000-2500 RPM reference speed), n is the pump shaft speed, Q is the flow rate and P_i is the input pressure.

A system identification method (least squares) is carried out by using all the presented applications in this work and evaluating time responses in Figure 4. The models are:

$$H_{1,i}(z) = z^{-r_1} \frac{n_1}{1 + d_{11}z^{-1} + d_{12}z^{-2}}, \qquad (2)$$

$$H_{2,i}(z) = z^{-r_2} \frac{n_2}{1 + d_{21}z^{-1} + d_{22}z^{-2}}, \qquad (3)$$

$$H_{3,i}(z) = z^{-r_3} \frac{n_3}{1 + d_{31}z^{-1} + d_{32}z^{-2}}. \qquad (4)$$

Due to the energy recovery system complex mathematical model three tests are carried out leading to model parameters presented in Table III. As can be seen from Table III, the system parameters strongly change in different operating conditions. The percentage of adjustment of the experimental data and the proposed models is observed in Table IV. In contrast to [16] in this paper, a more complex model was proposed with the aim of improving the percentage of adjustment.

Figure 6 shows the locus of poles and zeros in the z-plane for each system model $H_{1,1}(z)$, $H_{1,2}(z)$ and $H_{1,3}(z)$. Due to the zeros at the origin of the z-plane correspond to delays presented in transfer functions, zeros remain unchanged. However, poles are highly dependent on the operating conditions. It is important to note that the first two speed steps in Figure 4 are related to the poles on the real axis of the z-plane of Figure 6, i.e $H_{1,1}(z)$ and $H_{1,2}(z)$. Yet, the last speed step (2000-2500 RPM) in Figure 4 correspond to the pair of complex poles in Figure 6, $H_{1,3}(z)$. As a consequence, a highly variable behavior of the energy recovery system can be observed with the change on the operating conditions. Additionally, the location of poles can be used as a measure about how far or close, from a change in the power flow direction, the energy recovery system is working.

TABLE III
MODEL PARAMETERS

Step	Parameters of the proposed models					
	d_{11}	d_{12}	d_{21}	d_{22}	d_{31}	d_{32}
1	0.004	-0.73	-0.15	0.72	-0.10	-0.86
2	-0.006	-0.68	-1.43	0.43	0.009	0.3712
3	0.108	0.006	-1.82	0.825	0.006	-0.946
	r_1	r_2	r_3	n_1	n_2	n_3
1	7	3	14	-6.8e−5	-0.06	-0.01
2	7	16	1	-1.1e−4	2.3e-8	-0.2
3	13	3	1	-4e−4	2.3e-7	-0.009

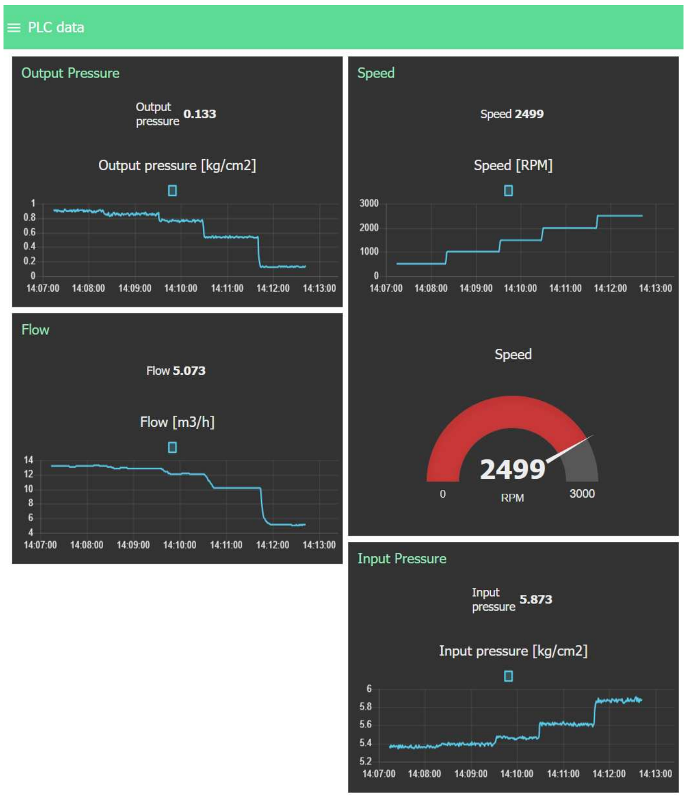

Fig. 4. PLC measurements user interface tab

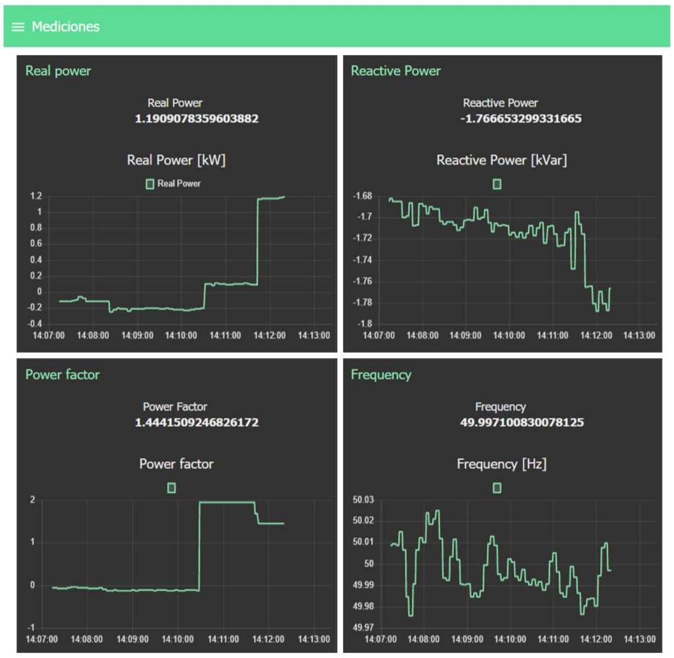

Fig. 5. iEM3155 measurements user interface tab

VII. Conclusions

This work present a OPC UA/Modbus gateway, a HMI and a historian completely implemented with open source

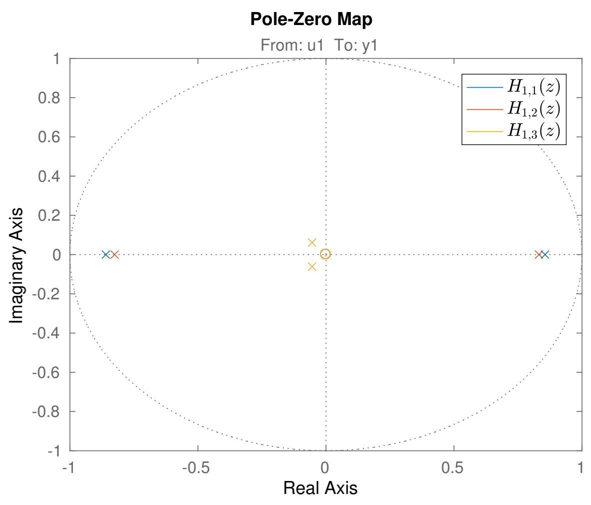

Fig. 6. z-plane.

TABLE IV
PERCENTAGES OF FIT OF THE PROPOSED MODELS.

Steps	Percentages of fit
1	69.04 %
2	78.95 %
3	88.46 %

software and low cost devices. In this sense it should be noted that open source software can economize the development of an investigation project. This proposal also allows to save resources and increase the reliability of the information due to the modularity of the system.

Additionally, the HMI produced good results, taking into consideration that the measurements not only can be seen in a specific computer, but also in any device with a web browser and internet connection. This shows the possibilities IIoT brings to an investigation group, but also it opens the field to possible applications in an industrial setting.

Furthermore, different MISO models of the system were obtained. But also, it can be seen how the change in power flow is reflected in the model. First results are promising and they encourage to synthesize controllers using adaptive techniques.

ACKNOWLEDGMENT

This work has been performed through UNPSJB (PI-UNPSJB Sigeva number 80020180100026UP).

REFERENCES

[1] H. Boyes, B. Hallaq, J. Cunningham, and T. Watson, "The industrial internet of things (IIoT): An analysis framework," *Computers in Industry*, vol. 101, pp. 1 – 12, 2018.

[2] M. Schleipen, S.-S. Gilani, T. Bischoff, and J. Pfrommer, "OPC UA & industrie 4.0 - enabling technology with high diversity and variability," *Procedia CIRP*, vol. 57, pp. 315 – 320, 2016, factories of the Future in the digital environment - Proceedings of the 49th CIRP Conference on Manufacturing Systems.

[3] P. Ferrari, A. Flammini, E. Sisinni, S. Rinaldi, D. Brandão, and M. S. Rocha, "Delay estimation of industrial IoT applications based on messaging protocols," *IEEE Transactions on Instrumentation and Measurement*, vol. 67, no. 9, pp. 2188–2199, Sep. 2018.

[4] H. Geng, *Internet of Things and Data Analytics Handbook*. Somerset: John Wiley & Sons, Incorporated, 2016.

[5] S. Chen, H. Xu, D. Liu, B. Hu, and H. Wang, "A vision of IoT: Applications, challenges, and opportunities with china perspective," *IEEE Internet of Things Journal*, vol. 1, no. 4, pp. 349–359, Aug 2014.

[6] W. Mahnke, *OPC unified architecture*. Berlin Heidelberg: Springer-Verlag, 2009.

[7] P. Ferrari, A. Flammini, S. Rinaldi, E. Sisinni, D. Maffei, and M. Malara, "Evaluation of communication delay in IoT applications based on OPC UA," in *2018 Workshop on Metrology for Industry 4.0 and IoT*, April 2018, pp. 224–229.

[8] "Modbus Foundation," 2019. [Online]. Available: http://www.modbus.org/

[9] "Node-RED official webpage," 2019. [Online]. Available: https://nodered.org/

[10] A. Nicolae and A. Korodi, "Node-red and OPC UA based lightweight and low-cost historian with application in the water industry," in *2018 IEEE 16th International Conference on Industrial Informatics (INDIN)*, July 2018, pp. 1012–1017.

[11] M. Tabaa, B. Chouri, S. Saadaoui, and K. Alami, "Industrial communication based on modbus and node-red," *Procedia Computer Science*, vol. 130, pp. 583 – 588, 2018, the 9th International Conference on Ambient Systems, Networks and Technologies (ANT 2018) / The 8th International Conference on Sustainable Energy Information Technology (SEIT-2018) / Affiliated Workshops.

[12] M. Lekić and G. Gardašević, "IoT sensor integration to node-red platform," in *2018 17th International Symposium INFOTEH-JAHORINA (INFOTEH)*, March 2018, pp. 1–5.

[13] "MariaDB official webpage." [Online]. Available: https://mariadb.org/

[14] A. Cadiboni, F. Yncio, R. Fernández, G. Ahrtz, R. Peña, C. Sosa Tellechea, and M. Vásquez, "Control and modeling of a centrifugal pump used as a turbine in an energy recovery system," in *2018 IEEE 9th Power, Instrumentation and Measurement Meeting (EPIM)*, Nov 2018, pp. 1–6.

[15] F. Yncio, R. Peña, A. Cadiboni, R. Fernández, G. Ahrtz, and C. Sosa Tellechea, "A modbus client for the identification of an energy recovery system for a water distribution network," in *2018 IEEE 9th Power, Instrumentation and Measurement Meeting (EPIM)*, Nov 2018, pp. 1–6.

[16] R. R. Peña, R. D. Fernández, M. Lorenc, and A. Cadiboni, "Gateway opc ua/modbus applied to an energy recovery system identification," in *2019 XVIII Workshop on Information Processing and Control (RPIC)*, Sep. 2019, pp. 235–240.

[17] "Raspberry Pi Foundation." [Online]. Available: https://www.raspberrypi.org/

[18] "The slackware arm linux project," 2019. [Online]. Available: http://arm.slackware.com/

[19] S. Cavalieri and F. Chiacchio, "Analysis of OPC UA performances," *Computer Standards & Interfaces*, vol. 36, no. 1, pp. 165 – 177, 2013.

[20] J. Hughes, *Real world instrumentation with Python*. Boston, MA: O'Reilly Media, 2010.

[21] "iem3155 energy meter," 2019. [Online]. Available: https://www.se.com/ar/es/product/A9MEM3155

[22] A. Tanenbaum, *Modern operating systems*. Boston: Pearson, 2015.