# Simulated Test Environment for Low Voltage and Medium Voltage Distribution Cell Monitoring Systems

Radu Fechet
Ștefan cel Mare University of Suceava
Computers, Electronics and Automation Department
Suceava, Romania
radu.fechet@usm.ro

Adrian Graur
Ștefan cel Mare University of Suceava
Computers, Electronics and Automation Department
Suceava, Romania
adrian.graur@usm.ro

Marius Prelipceanu
Ștefan cel Mare University of Suceava
Computers, Electronics and Automation Department
Suceava, Romania
marius.prelipceanu@usm.ro

Dragos Vicoveanu
Ștefan cel Mare University of Suceava
Computers, Electronics and Automation Department
Suceava, Romania
dragos.vicoveanu@usm.ro

***Abstract*—The present paper proposes tested solutions for simulation of environmental sensors in Low Voltage (LV) / Medium Voltage (MV) microclimate monitoring. The sensor simulations were implemented by three different methods, using both IoT technologies and x86 architectures, with the help of dedicated software. A monitoring system reduces the danger of power plant failures due to unfavourable environmental conditions, which allows both predictive and preventive maintenance measures. As set forth, the least friendly combinations are those where the dust, humidity and temperature can bring synergistic effects, which can determine unplanned events (i.e., electrical discharges) within insulators, inside of a voltage distribution cell. The absence of predictive and preventive maintenance measures can lead to both significant technological and financial losses.**

***Keywords—sensors; simulation; low voltage switchgear cells; medium voltage switchgear cells; modbus; monitoring; preventive maintenance; IoT; Node-RED***

## I. Introduction

Both predictive and preventive maintenance are scheduled maintenance methods that aim to bring confidence into accuracy and responsiveness of the monitored assets, increasing at the same time the reactivity to their failures. The costs can vary depending on the maintenance type, considering the proactive subscription, plan, or period, but also of the predictive or preventive type. Here, it is worth to mention that in the industry, the most expensive subscription plan covers a reactive level of the companies. If we refer to the predictive maintenance, this covers a set of inbound scheduling orders of the actual execution of the maintenance. Predictive maintenance has also been one of the driving forces for improving productivity [1]. The aim is to focus on this type of maintenance, which envisages the continuous (i.e., online) or periodic (i.e., offline) monitoring for the functioning status of the equipment. The remote devices that are tracking by permanent data collections have an important role, identifying possible complications faster or providing help in preventive procedures. Predicting the remaining useful life of an asset using supervised machine learning (ML) is the most common technique in predictive maintenance [2], [3]. To be able to test and simulate field like various behaviours on the communication paths that range from sensors to monitoring system (server), an external equipment is needed. The external simulation equipment should be able to transmit values in the normal interval similar to the field sensors. Moreover, the equipment needs to send values at the limits of the measuring range, and values outside the measuring range, thus simulating the sensors failure (i.e., lack of response to interrogation).

The switchgear is a significative piece with both control and protective roles. If switchgears are properly maintained, failures are usually rare, meaning that they can reach a lifetime period more than 15 years. (i.e., Fig. 1).

Fig. 1. Example of MV (Medium Voltage) switchgear with 8 (eight) cells *(Source: https://electroalfa.ro/referinte-infrastructura, accessed on 2022-02)*

978-1-6654-8162-5/22/$31.00 ©2022 IEEE

In general, the main cause of LV/MV failures is outlined below, according to the highest percentage of occurrence rate:

- The increase of the temperature due to increase of resistance usually occurs with jumper cables, buses, cable terminals or unsuccessful short circuit interruption.
- The degradation of insulation where partial discharge (PD) phenomena is generally accepted as the predominant cause for the long-term degradation (the PD on the surface, or corona discharges).
- In the cases of rare attacks of small animals (mice, rats, squirrels, snakes, etc.) or birds (crows, storks, etc.).

Fig. 2 shows an example of damaged cell due to high temperature.

Fig. 2. Example of a damaged distribution cell due to high temperatures *(Source:[4], accessed on 2022-02)*

## II. THE PROPOSED SENSORS FOR THE SIMULATION TASK

Considering the many decades of our partner experience, the several parameters analysis has been conducted to attain the early degradation of the conductor insulations. These factors include the dust formation by the nitric acid ($HNO_3$) action within condensation phase followed by ozone release ($O_3$). It is known that when electrical discharges occur inside of LV/MV cells, the short phase of plasma formation can be accompanied by a set of chemical reactions which can produce ozone ($O_3$) due to a tripartite recombination of $O_2$ species, $NO_2$, $N_2O_5$ along with other compounds. Nitric acid is formed due to $NO_2$ gas based on the moisture and oxygen [5], [6], [7] and [8]. A IEEE study [9] shows that the breakdown of the switching bus was caused by the atmosphere composed of 30% humidity, where the exposure to dust further contributes to the total percentage of 19% of the switching bus failure in a total damage situation. A similar study on the occurrence of PD phenomena with the increase of humidity is presented in [10]. Fig. 3 shows a detailed schematic of the effect of Partial Discharge (PD) phenomena.

The lowest possible cost implementation regarding the sensors and the monitoring software was considered. Therefore, the ModBus RS_485 communication protocol has been chosen. The sensors to be used for LV/MV environment evaluation are the temperature, the humidity, the dew point calculation, the atmospheric pressure, the dust, the $O_3$ (i.e., ozone), and the IAQ (i.e., Index Air Quality). PD sensors are generally expensive and, therefore, they are not included in the present analysis. On the other hand, one of our major concerns relates to the appearance and formation of condensation inside LV/MV cells, therefore our sensor simulation focuses to cover this requirement and to calculate the dew point inside the equipment.

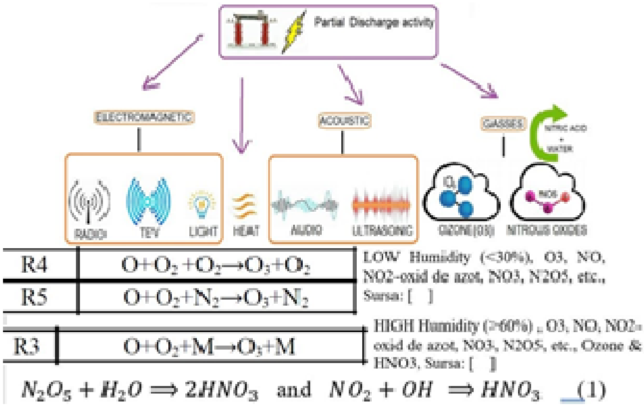


Fig. 3. Partial discharge energy emissions in low and high humidity.

The major concern relates to the appearance and formation of condensation inside LV/MV cells, so our sensors simulation focuses to cover this requirement and to calculate the dew point.

With this purpose, a portable testbench equipped with sensors for determining environmental parameters, has been built, as shown in Fig. 4 (designed for up to 512 environmental sensors).

Fig. 4. Testbench with different environmental sensors that will be tested in the LV/MV distribution cells.

It is known that the RS_485 is a robust electrical communication interface that can operate in heavy industrial environments up to 1200 m and a maximum speed of 100 mbps. The Modbus protocol is robust and easy to implement in a configuration with low-cost microcontrollers. In this case, the advantage of the testbench equipped with real sensors permits to test the same lengths and types of communication cables as in the case of the field installed distribution cells. The sensors can be physically heated or cooled individually on the testbench to be able to simulate various behaviours.

One disadvantage, which it is worth to mention, is the impossibility to simulate the electromagnetic fields that exist inside a distribution cell, which can influence and disrupt the operation of the sensors. Another downside is the operating range up to 80% relative humidity as compared with 100% relative humidity in operation (where an IP68 enclosure is required).

The first proposal of the sponsor of the scientific project regarding sensor part, was to design a prototype sensor with two measuring points. The second measuring point aims to determine the condensation conditions and it is based on the dew point calculation using the first sensor. The sensors have been chosen to work precisely at 100% humidity, and the housing has been carefully chosen (see Fig. 5).

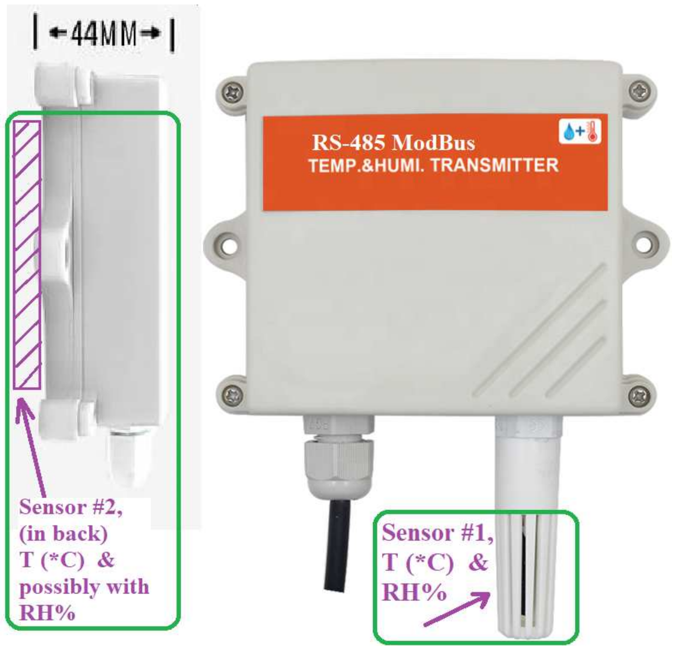


Fig. 5. Sensor design proposal with two environmental measurement points for the LV/MV distribution cells.

Regarding the Dew Point formula used to calculate the dew point, the medium complexity formula from [11] has been chosen out of many dedicated formulas.

Regarding the real sensors used in this project, the ModBus registers are obtained in integer values, and then scaled by a factor of 10. For example, if a temperature value equal to 213 means a value of 21.3 °C in the SCADA system. Therefore, in order not to affect the values in SCADA system panels and server panels, the data have been scaled by 10 in all registers. The ModBus simulated registers are presented in TABLE I.

TABLE I. SIMULATED MODBUS REGISTERS (FUNCTION 03, HOLDING REGISTERS)

| ModBus REG | Simulated ModBus Registers | | |
|---|---|---|---|
| | ***Sensor Type*** | ***min.*** | ***Max.*** |
| 4001 | Temperature Sensor_1 (signed integer scaled by 10) | -20°C | +100°C |
| 4002 | Relative Humidity (integer scaled by 10) | 0% | +100% |
| 4003 | Dew Point calculation Sensor_1 (signed integer scaled by 10) | -120°C | +100°C |
| 4004 | Temperature Sensor_2 (signed integer scaled by 10) | -20°C | +100°C |
| 4005 | Temperature Delta (Temperature Sensor_2 – DewPoint_Sensor_1) | -20°C | +100°C |

a. *(Table 1)*

As in SCADA systems, data simulated in ModBus registers can be incremental numbers, random, brownian, white noise, etc. In our case, random values for Temperature Sensor 1, Temperature Sensor 2 and Relative Humidity Sensor 1 (see TABLE I. ) were used.

The proposed switchgear remote monitoring system is presented in Fig. 6. It consists of the following blocks: ModBus RS_485 temperature & RH% sensors, GSM router with accessories, servers, and data storage. Other modules needed but not presented here are the ModBus RS_485 I/O digital/analogue device for breakers monitoring and the main power supply with battery backup. The presented workflow should be able to monitor 20 switching cells with 3 sensors per cell, which is a total of 60 sensors. In areas where the 2G/3G/4G signal coverage is poor, in the block diagram, a Yagi antenna was recommended. The choice of the antenna depends on the local geographical conditions. For weak mobile signal areas, performing specific measurements for mobile telephony signal to determine the gain of the Yagi antenna is recommended before installation.

In Fig. 6, a SIM card with private APN and fix IP must be used for the GSM 3G/4G provider to have remote full control of the GSM router for additional configuration changes or software update.

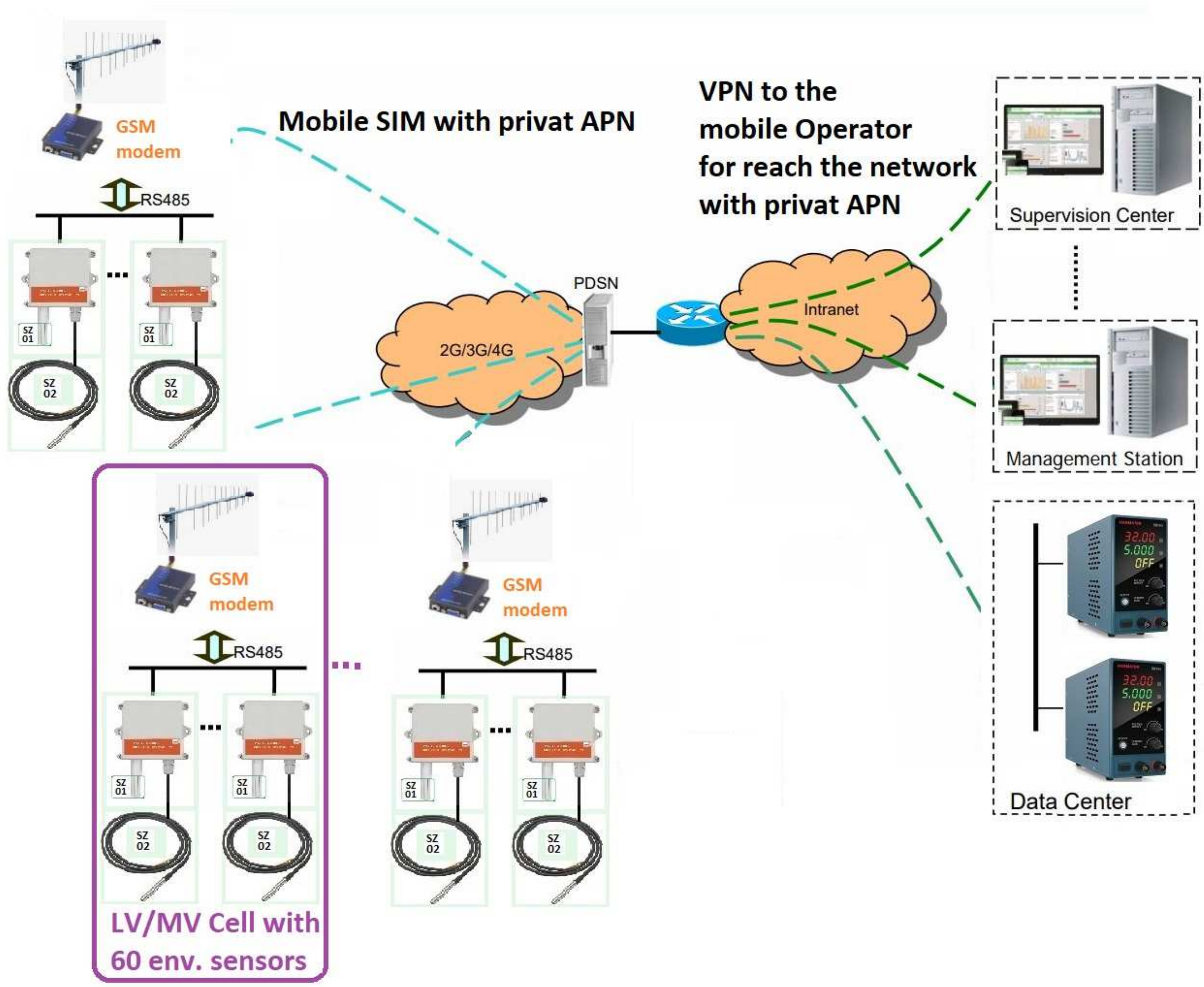


Fig. 6. LV/MV switchgear monitoring system proposal.

## III. THE FIRST SOLUTION-THE MODBUS SLAVE SENSOR SIMULATION USING ESP8266

The ESPDUINO IoT board featuring the ESP8266 microcontroller with pin-compatible Arduino board has been used for implementation, as shown in Fig. 7. The choice of this board offered a great variety of software support from the Arduino community, as well as the compatibility with various Arduino shields.

For the ModBus RTU slave part, a modified Arduino library was used to simulate the operation of 60 ModBus RTU slave sensors (from address 01 to 60), which can answer the query of 5 values from the Holding Register (40001-40005) at 9600 baud rate. The Holding Registers contain randomized data for 3 sensors as shown in TABLE I. For the RS_485 interface, a board (Fig. 7, board named Tx-Rx TTL to RS_485) was chosen, which that does not require a directionality pin because the ESP8266 does not have as many available pins as compared with ESP32more advanced version. The query addresses for 0 and 255 was not implemented. The practical implementation of this solution is presented in Fig. 8. To determine the reliability of the solution. an endurance test of at least 6 months must be continuously run and checked for errors.

Our results showed that another functionality of the Arduino board can be used next. Therefore, a web page inside ESPDUINO environment trusting the WiFi capability of the IoT chip is implemented to enable/disable different slave sensor simulated behaviour on the ModBus RS_485 bus.

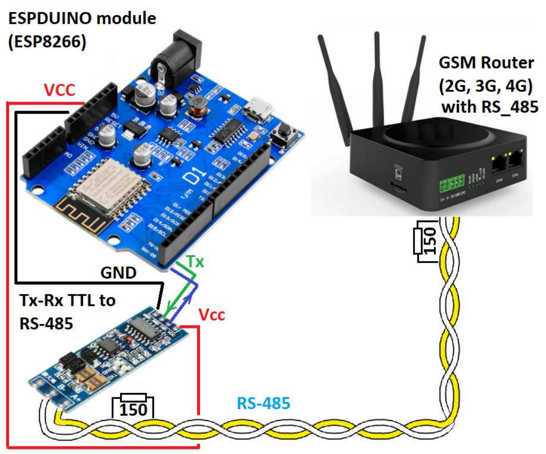


Fig. 7. Block diagram of ModBus RTU slave sensors simulation using IoT ESP8266. *(60 slave sensors)*

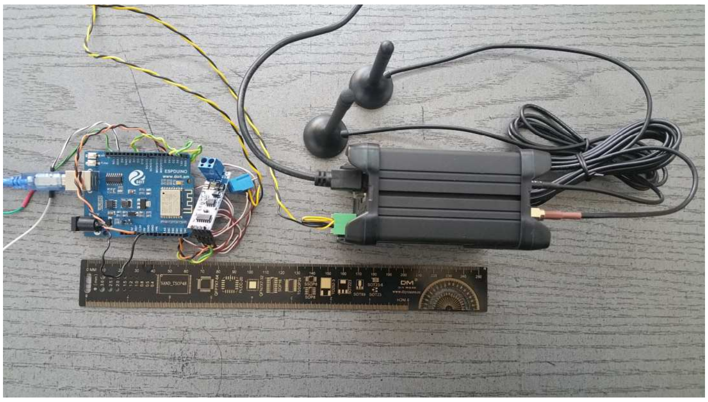

Fig. 8. Solution 1, practical implementation ModBus RTU slave sensors simulation using IoT ESP8266. *(60 slave sensors)*

The following implementation advantages can be highlighted:

- The low price due to the low cost of IoT microcontroller which comes with an open-source free Arduino IDE development software and cheap RS-485 interface.
- 4MB for the flash code - enough memory space to allow the creation of a web page inside the IoT board for setting various simulation configurations for the ModBus RTU slave protocol.
- The activation or deactivation of certain sensors on the web page.

The following disadvantages can be mentioned:

- The cable lengths between the sensors cannot be considered because the ModBus RTU serial answers of sensors are simulated.
- Less flexibility: if other slave behaviour is needed, the IoT board must be reprogramed.
- ModBus address 0 and 255 are not implemented.

## IV. The Second Solution-Modbus Sensor Simulation Using ModRSsim2 Application on x86 Industrial PC (Win7 or Win10)

Another interesting free ModBus RTU slave serial RS_485 simulator solution was identified in [12], as an improved version of the original that was created by Conrad Braam, in 2013. The new version ModRSsim2 (updated last in 2019-03-06 by Doug Lyons) is a fork of the original software which was called mod_RSsim.

As mentioned before, an industrial PC is used to perform the experiment to extend the tests in the endurance part aiming to have a system that can work permanently for a longer time and low energy consumption. Therefore, it was used OS Win7-64b [13] operation system on an old low power industrial computer with the following configuration:

- Intel Atom CPU E3845 @1.92Ghz, 4GB RAM and 4 USB 2.0 ports (Fig. 9). To test the RS-485 communication, two USB to RS-485 adapters were used and two different software applications: one software as ModBus master (Fig. 10, on the left) and other software as ModBus Slaves Simulator (Fig. 10, on the right).

For each distribution cell, 3 sensors were assigned, therefore 60 sensors in a ModBus Slave were needed for 20 distribution cells. Around 1000 lines of scripted code were loaded in the application for ModBus Slave addresses starting from 1 to 60 to carry out the final form of the application. Thus, it was possible to individually simulate ModBus slave sensors holding registers with random data (five holding registers for each ModBus slave sensor), as presented in TABLE I.

As shown in Fig. 10, after 7 days of continuous testing, there were no errors on the RS-485 bus with than $10^6$ received ModBus packages.

The query of ModBus slave clients was realized at an interval of 400ms, showcasing that the ModRSsim2 application is reliable (no errors during over 7 days) with the hardware, software setup and windows drivers in stable condition.

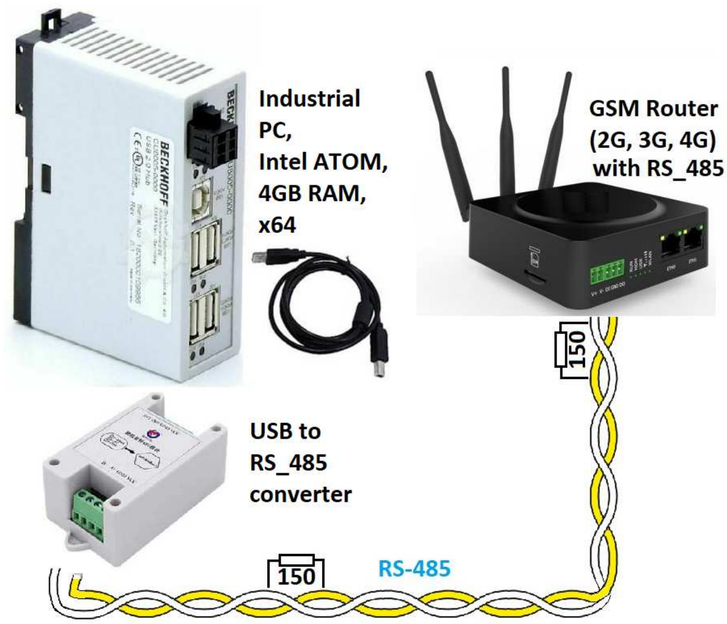


Fig. 9. Block diagram ModBus RTU slave sensors simulation using an industrial computer and software ModRSsim2. *(60 slave sensor)*

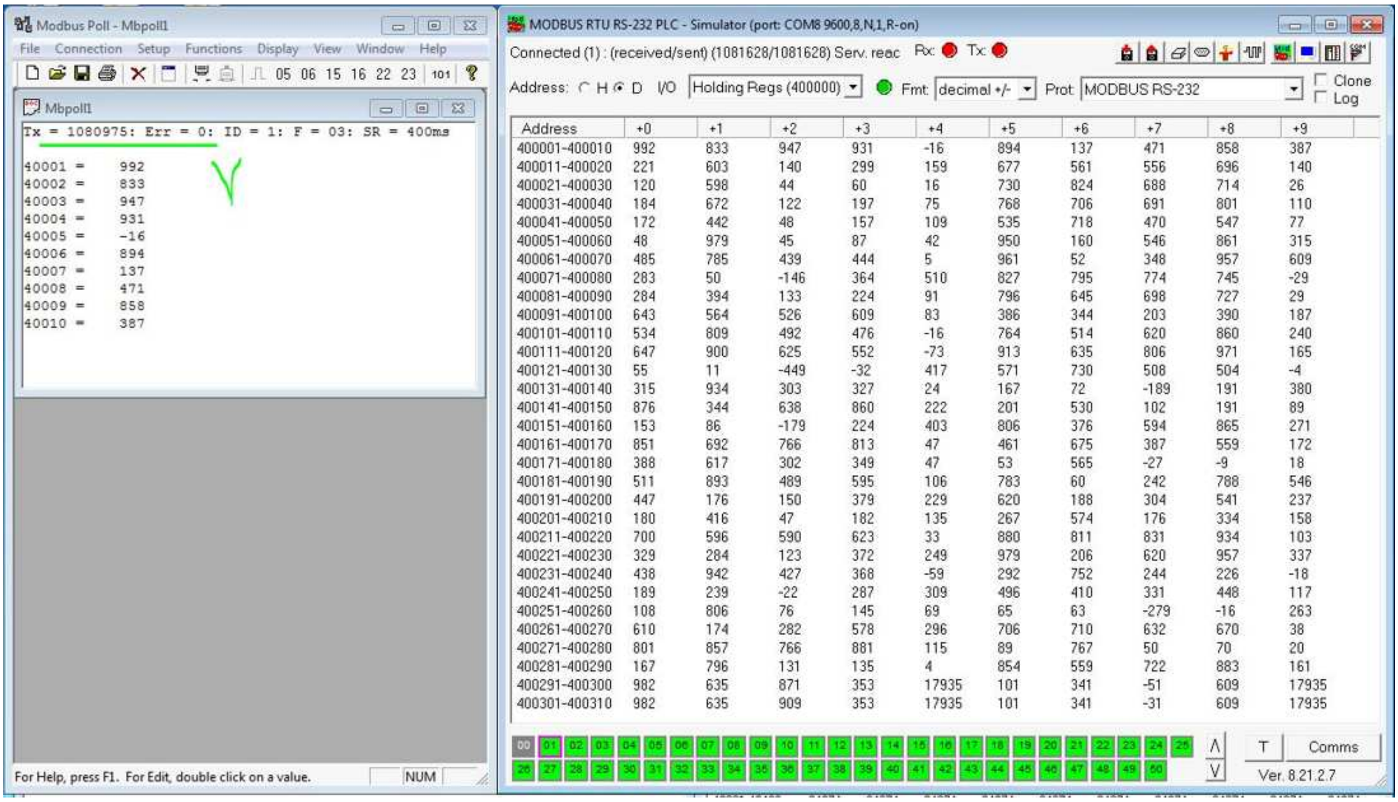


Fig. 10. ModBus slave sensors simulation using ModRSsim2 (vers. 8.21.2.7); randomize data for temperature & humidity and dew point calculation for each slave from 1 to 60.

Therefore, the following strong points of the implementation can be emphasized:

- similar data representation as in the case of real sensors with values scaled by 10, in the simulated ModBus Registers.

- flexibility (randomize-implemented, incremental-implemented by default, manual change-implemented by default). In the future, it is intended as follows:

a) Incremental by different steps

b) Brownian simulation (settings: start value, minimum value, maximum value, maximum change random differential)

c) Attractor implementation similar to SCADA systems (settings: maximum change value, volatility=random fluctuations around a point, attraction_point=sensor value)

- individual slave sensor randomizes data generator.
- can be extended to 247 ModBus Slaves if needed (only 60 Slaves used; ModBus address number 0 is reserved for broadcast).
- can be monitored 60 stations in one screen (3 x sensors for 20 MV switchgears).
- different Dew Point formulas for each sensor implemented (2 formulas in current case).
- for the second sensor, a randomized number around +-10C, was implemented like in the real environment.
- individual ModBus slave Enabled/Disabled to simulate the RS_485 communication loss.
- The possibility of implementation of different scenarios

Weak points:

- it works only on x86 architecture.
- no history of values written in the registers.
- uptime missing in the application.
- sensors cable length cannot be tested.
- cannot use the delays functions in the script coded procedures.
- Over 1000 lines for script code lines for 60 ModBus slaves and 5 registers for each slave with randomized numbers (60 slaves means 20 switchgears with 3 sensors for each MV switchgear).
- Fixed time of refreshing ModBus data in registers to 500ms.

## V. The Third Solution-Modbus Sensor Simulation Using the Node_RED Server Platform (Linux or Windows)

Node-RED was originally developed in early 2013 as a flow-based development tool with no help of programming coding lines. It has an open-source API which can be used to easily provide a development platform web-based for **Node-Rapid Event Developer**. As a basis ground, it uses JavaScript functions to create elements stored in JSON format which can communicate. The flows created in Node-RED can be used since version 0.14 with the protocol MQTT to create IoT nodes properly configured with TLS connections [15].

Node-RED can also work as a pre-installed package on some industrial devices, such as: FUJITSU IoT Solution INTELLIEDGE™ A700 Appliance (CentOS Linux, Intel Core I7) [16], Schneider Electric-Harmony HMIBSC Core Box (based on Linux Yocto, in addition with an SDK and Node-RED) [17], Siemens-SIMATIC IOT 2040 [18], etc. For this reason, the Node-RED platform was installed and used on our x86 industrial system based on Intel Atom which performed very well. Based on Node-RED site [14], we have identified several libraries for ModBus communication. The library *node-red-contrib-modbus*; does not support Modbus RTU as per the node descriptions, but instead, the *modbus-server* provided for testing a ModBus TCP server based on node-modbus (jsmodbus). The enhanced node *modbus-flex-server* provides a flexible Modbus TCP server based on modbus-serial, but not ModBus RTU serial (Fig. 11). Furthermore, the *node-red-contrib-arc-modbus-puls* provides a master TCP ModBus.

There may be other modules supporting RTU slaves, but no software module to implement ModBus RTU serial was identified. As such, the options are:

- To modify the existing node to add support for ModBus RTU serial.
- To use an external gateway. There may be a software that can do this, but nothing free was found yet.
- To use a serial-communication node and other function-node to build the ModBus RTU serial messages, which was adopted (*node serial in* and serial out in Fig. 11).

The flow implemented in Node-RED flows for the implementation of ModBus Slave RTU Serial is quite complex and is not currently the subject of this paper.

To be as close as possible to the actual behaviour of the ModBus sensors that is used, a ModBus RTU request of 5 Holding registers (40'0001-40'0005) must be defined. The answer provided by Node-RED consists of random 16-bit integers with sign in those registers, whose content is presented in TABLE I. Random numbers have been generated in the operating range of environmental sensors. The calculated dew point enabled running outside of the measuring range of the sensors.

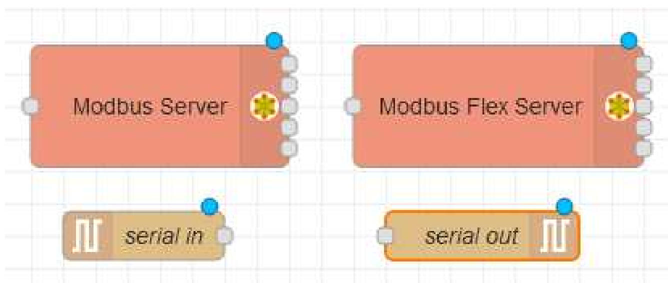


Fig. 11. Node-RED nodes from Node-RED software. ModBus node and Serial node for ModBus RTU Slave simulator.

Fig. 12 shows a small part of the total implementation and the basic part of the ModBus RTU slave client simulation. Other improvements and tests need to be made, especially in the detection and counting of errors in both reception and transmission on RS_485 line.

Using the node library *simpletime node*, the transition of the ModBus RTU serial packet from the entry into the Node-RED server (ModBus RTU serial Request) to the exit from the server (ModBus RTU serial Answer) varies around 10ms, which is considered satisfactory. If we consider the average propagation time of the 23 bytes at a speed of 9600 baud rate (Tx = 8 bytes, Rx = 15 bytes, a total of 23 bytes with the corresponding CRC), it turns out that the query of 60 sensors (20 LV-MV cells with 3 sensors each) would take about 2.4 seconds (60 * 40ms) if all sensors responded. It can be seen in Fig. 12 that there were no errors in communication after approximately 222,740 ModBus requests at a regular interval of 100 ms processed by Node-RED, which shows that the system is reliable.

Another aspect to consider is the *time-out* parameter for querying the sensors. A maximum of 60 seconds was queried considering the most unfavourable situation when only one sensor out of 60 responds in a time-out value for the ModBus RTU serial query less than 1 second (500ms is recommended).

Building a WEB page to allow the possibility to function as a ModBus slave both manually and automatically is intended in the future. In the manual mode, it will allow the setting of certain values for temperature and humidity, with the automatic calculation of the dew point and manual temperature calculation of the second sensor.

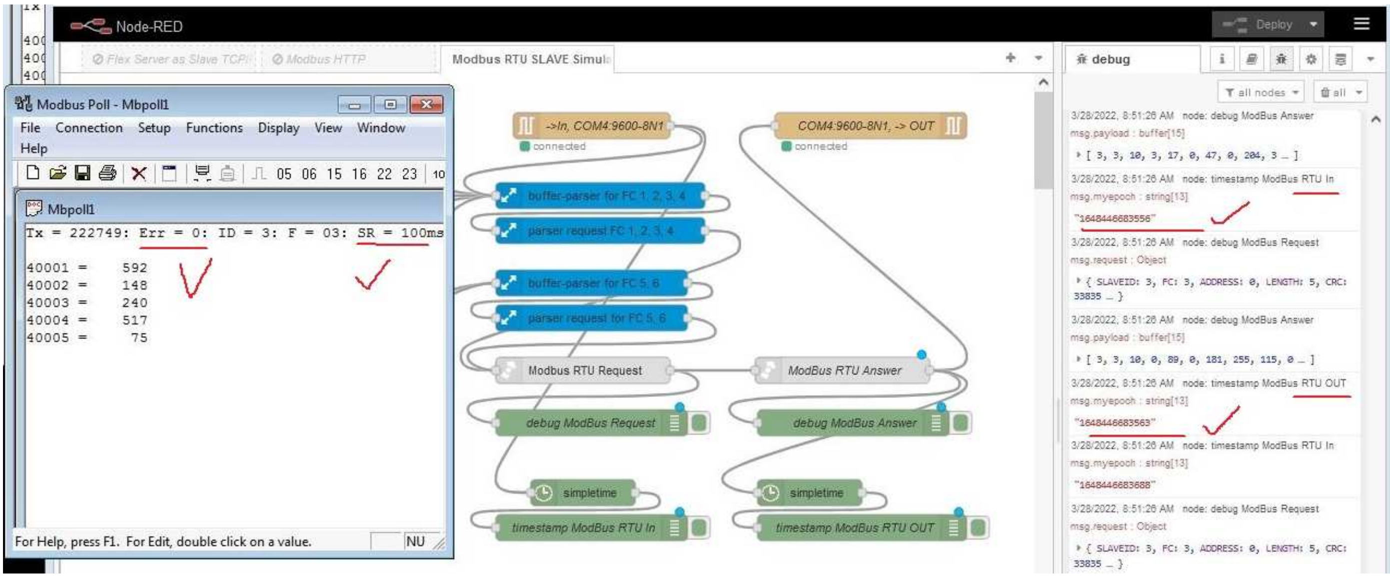


Fig. 12. Node-RED flows implementation for simulation of ModBus serial slave sensors (RS_485 at 100ms); register with randomized data for temperature, humidity and dew point calculation.

Our x86 industrial system based on Intel Atom with OS Win7-64b has the role of simulating the ModBus RTU serial slave protocol for 60 sensors using Node-RED (Fig. 14).

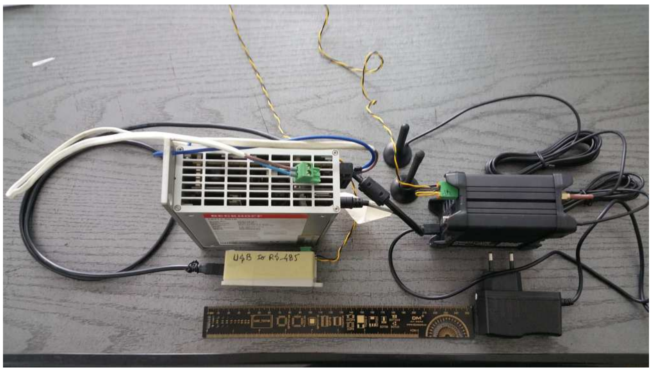

Fig. 13. Practical implementation of ModBus sensors simulation using Node-RED on an industrial PC with Intel Atom, 4GB RAM and OS Win7-64b.

In Fig. 13, a part of the dashboard web page for the simulation of ModBus RTU slave is presented. It can be seen the graphical representation of the sensors profile along with and a small history of the values sent at 60 seconds.

The great advantage of the Node-RED server is that it allows the definition of different graphical representations (buttons, sliders, measured sizes, communication error counters, etc.), history, automatic saves in various formats, offering an increased flexibility in developing a medium size application. The information shown in Fig. 12 and Fig. 14 was taken at different times and it is not identical. The values in the ModBus registers in Fig. 12 are scaled by 10, showing the information for the used sensors.

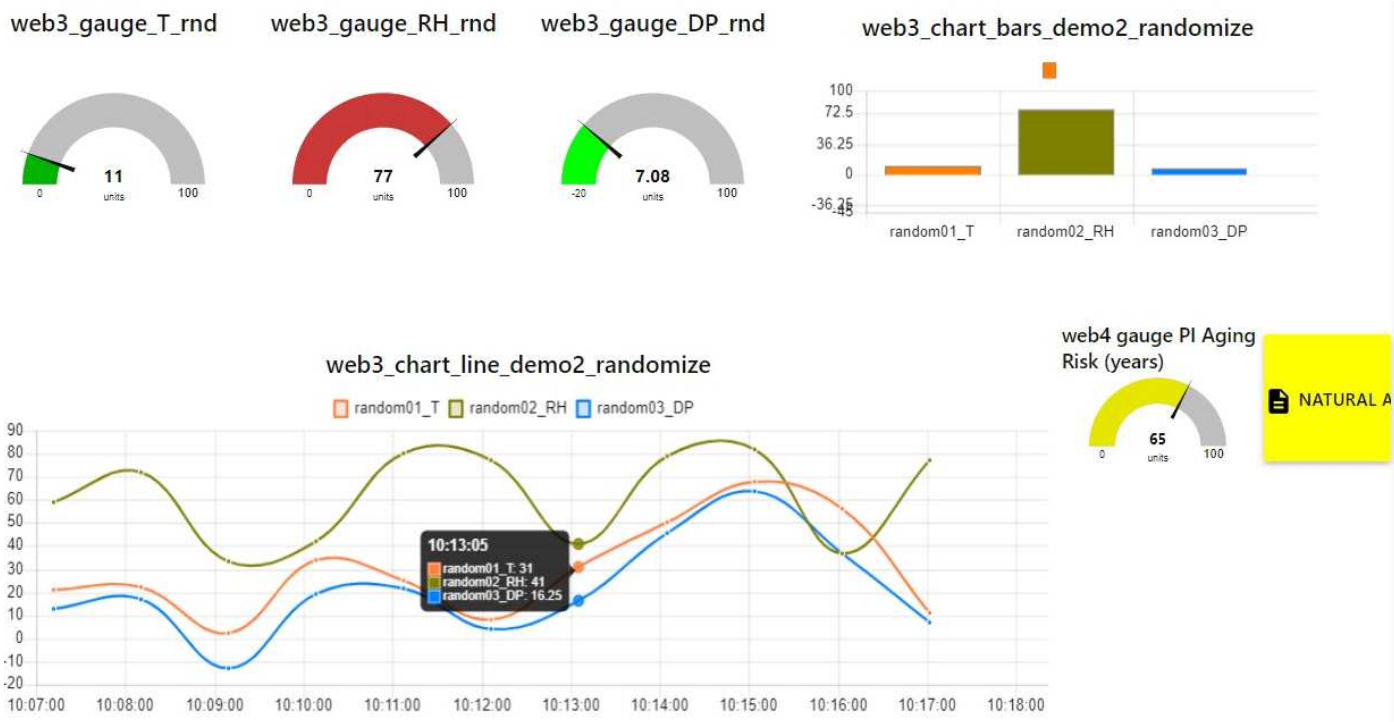


Fig. 14. Example of simulation of ModBus serial Slave sensors (RS-485) using Node-RED server; randomize data for temperature, humidity and dew point calculation.

To perform the endurance tests both in terms of software and hardware, the implemented system works permanently with a consumption of approx. of 24W (24Vdc / 1A).

The advantages of solution using Node-RED would be as follows:

- Flexibility of implementation; solution can be extended to any register and needs.
- Programmable of refreshing time of ModBus data in registers starting from 100ms.
- It can be used on any platform x86 or arm (i.e., Raspberry Pi in case of arm).
- It has a small Web history and the possibility of Excel exports with the history of generated numbers.
- An easy implementation of different behaviour for ModBus registers (incremental, randomized, Brownian, white noise, etc.).
- The processing time of an ModBus serial request is approx. 10ms.
- The possibility of introducing errors in the communication line.
- Medium number of programming lines in function-node to simulate 60 sensors (less than 300 lines of code).
- Uptime of simulated sensors and number of requests processed can be implemented in the web page.

Some disadvantages of this solution would be as follows:

- lack of a ModBus-node for a RS_485 RTU serial slave, implementation using a normal serial-node.
- Cable length cannot be considered.

## VI. Conclusions

Thus, the present paper presents technical solutions, both software and hardware, affordable financially and aimed to monitor the distribution cells. Compared to the variants offered by manufacturers such as ABB, Siemens or Schneider, stability over time and the GSM router can be emphasized. The loading and temperature of the GSM router processor, free memory and other system parameters were maintained within normal limits. The future goals refer to the security issues, which can be improved by encrypting data on the communication channels and data transfer protocol improvement. Using the SIM card with private APN, Cloud monitoring can also be considered to achieve a good solution at a low price. Current findings show that:

Solution 1 with IoT ESP8266 has the advantage of being very cheap, easy to implement and, as a next step, a web page can be created inside the IoT board to allow the setting of various configurations. The main disadvantages of this solution would are as follows:

- the cable lengths between the sensors cannot be considered.

- for the simulation of different behaviours, the IoT microcontroller should be reprogrammed.

Solution 2 using software application ModRSsim2 is reliable and has the following main advantages:

- it is free, open source and reliable for RS-485 communication.
- can be simulated up to 254 ModBus devices with desired values in different registers.
- easy to change the behaviour of simulated sensors data by loading specific scripts.
- easy to generate communication faults by deactivating desired ModBus sensors.

The disadvantages of Solution 2 would be as follows:

- the cable lengths between the sensors cannot be considered.
- lack of historical data displayed.

Solution 3 with Node-RED has a lot of benefits, but still needs improvements to meet research special requirements. The possibility to have a small history in a web page on the simulated sensors represents a great advantage in the development of the project due to the visual comparison of the data arrived on the monitoring server. Using the web page, the way the simulated data is generated can be changed. It seems that the solution to simulate ModBus sensors with Node-RED is a reliable solution and more testing for at least 6 months of continuous operation is planned.

The endurance tests performed have shown that the environmental simulators presented can work without problems for long periods (months) even in more hostile environments such as the industrial environment of a factory. The errors on the Modbus RS_485 communication line were very few compared to the number of packets transited (millions of packets) and were generally generated by the operating system rather than by the environmental simulators. The use of these environmental simulators in the factory that builds the distribution cells is useful for the overall testing of the entire monitoring system even if the environmental sensors are not physically present.

## ACKNOWLEDGMENT

This work was supported by:

1. the project "Integrated Center for Research, Development and Innovation in Advanced Materials, Nanotechnologies, and Distributed Systems for Fabrication and Control (MANSiD)".

2. the project "Supporting the research of excellence within the Ștefan cel Mare University of Suceava" financed by the Romanian National Council for Higher Education Funding, CNFIS, project number CNFIS-FDI-2020-0615.

3. research contract number 172/8.02.2021 between "Ștefan cel Mare" University of Suceava and the ElectroAlfa International Botosani company (https://electroalfa.ro/).

4. other sponsor: EGGER Radauti, (https://www.egger.com).

## REFERENCES

[1] Peng Kern (2012). Equipment Management in the Post-Maintenance Era: A New Alternative to Total Productive Maintenance (TPM). CRC Press. pp. 132–136. ISBN 9781466501942. Retrieved 18 May 2018.

[2] Amruthnath, Nagdev; Gupta, Tarun (2018). "Fault Class Prediction in Unsupervised Learning using Model-Based Clustering Approach", 2018, doi:10.13140/rg.2.2.22085.14563.

[3] A Rykov, M. The Top 10 Industrial AI use cases. Available online: https://iot-analytics.com/the-top-10-industrial-ai-use-cases (accessed on 28 January 2020).

[4] https://library.e.abb.com/public/3c3e5714dae64f78b4fe525be5377569/9AKK107992A1474_PDCOM%20partial%20discharge%20monitoring%20of%20SWICOM%20system_Presentation.pdf, slide 6.

[5] Hassan Javed, Kang LI, Guoqiang Zhang and Adrian Traian Plesca, Experimental Study on Air Decomposition By- Product Under Creepage Discharge Fault and Their Impact on Insulating Materials, J Electr Eng Technol.2018; 13(6): 2392-2401, http://doi.org/10.5370/JEET.2018.13.6.2392, ISSN(Online) 2093-7423, https://www.researchgate.net/publication/328262433

[6] Kang Li, Hassan javed, Guoqiang Zhang, "Experimental Study on Ozone Production under 50 Hz Corona Discharge Used for Fault Diagnostic," Proc. 2nd International conference on Machinery, Materials Engineering, Chemical Engineering and Biotechnology, pp. 456-460, Chongqing, China, Nov 2015. DOI:10.2991/mmeceb-15.2016.90.

[7] Hassan JAVED, Kang LI, Guo-qiang ZHANG, and Adrian Traian PLESCA, "Online Monitoring of Partial Discharge by Measuring Air Decomposition by products under Low and High Humidity", International Conference on Energy, Power and Environmental Engineering (ICEPEE2017), pp. 186-191, Shanghai, China, April 2017, DOI10. 12783/dteees/icepe2017/11837, https://www.researchgate.net/publication/316472592, april 2017.

[8] R. Schwarz, T. Judendorfer, M. Muhr, "Review of Partial Discharge Monitoring techniques used in High Voltage Equipments," 2008 Annual Report Conference on Electrical Insulation Dielectric Phenomena, pp. 400-403, 2008. DOI:10.1109/CEIDP. 2008.4772825.

[9] IEEE Standard 493-1997

[10] Simon Goldthorpe, Robert Ferris, Simon Hodgson, 2009, "Use of web based partial discharge monitoring to extend asset life", Cired 20th International Conference on Electricity Distribution, Paper 0530

[11] Martin Wanielista, Robert Kersten and Ron Eaglin. 1997. Hydrology Water Quantity and Quality Control. John Wiley & Sons. 2nd ed.

[12] (available at https://sourceforge.net/projects/modrssim2/)

[13] (available at https://www.microsoft.com/en-us/)

[14] (available at https://nodered.org/)

[15] Community staff writer (June 14, 2016). "Version 0.14 released". nodered.org/blog. Node-RED. p. 1. Retrieved July 6, 2016. MQTT with TLS support

[16] (available at https://kastl-gmbh.de/datenblatt/S26381-K410-L420_Fujitsu_de.pdf)

[17] (available at https://download.schneider-electric.com/files?p_enDocType=User+guide&p_File_Name=EIO0000003374_04.pdf&p_Doc_Ref=EIO0000003374)

[18] (available at https://www.automation.siemens.com/sce-static/learning-training-documents/tia-portal/advanced-communication/sce-094-100-node-red-iot2000-en.pdf)