2025 IMEKO TC-6 International Conference on
Metrology and Digital Transformation - **M4DConf 2025**
Benevento, Italy, September 3-5, 2025

Setup of a distributed sensor network for acquiring environmental data

Alicja Wiora, Józef Wiora

Department of Measurements and Control Systems, Silesian University of Technology, ul. Akademicka 16, 44-100 Gliwice, Poland
alicja.wiora@polsl.pl; jozef.wiora@polsl.pl

Abstract **– This study presents the application of an Internet of Things (IoT)-based system for environmental data acquisition in a scientific research setting. The system comprises a network of 12 sensor nodes and a central server. Each node is built around a D1 Mini module, which collects data from an attached sensor and transmits the measurements to the server via web requests. A server-side script processes these requests and stores the data in structured text files. The collected data can be analysed either in real time during the experiment or retrospectively. To ensure durability and reliability in outdoor conditions, all sensor nodes are enclosed in protective housings. This work highlights the practicality, cost-effectiveness, and efficiency of a custom-designed, application-specific IoT measurement system, demonstrating its suitability for rapid deployment in environmental monitoring applications.**

Keywords **– IoT, ESP8266, particulate matter, CO2**

I. INTRODUCTION

Conducting scientific research on environmental parameters requires collecting data from many sensors, often located in different places. This necessitates the design and construction of a dedicated measurement system. Such a system should be low-cost (depending on available budget), scalable and quick to deploy. This concept has been applied to monitor particulate matter (PM) in the air. The resulting data formed the basis for the analysis described in [1]. The same concept is further continued in a study on monitoring carbon dioxide (CO_2) concentrations.

The Internet of Things (IoT) is a concept in which web technology allows devices to communicate with each other. Smart sensors can additionally perform data preprocessing [2]. This makes the measurement system more advanced and reliable, but also relatively expensive. Another important aspect in wireless sensor networks is the power consumption of the sensor, microcontroller and communication module as well. It is especially important with battery-powered systems [3]. Parallel to the advanced solutions, low-cost sensor networks are being developed. Their low cost drives popularity and contributes data, especially environmental, to society [4].

The ESP8266 microcontroller is currently often applied to collect data from digitally interfaced sensors, like DHT11, for humidity and temperature measurement. It is well known for its affordability for IoT solutions. Its integrated Wi-Fi module allows easy storage of data in the cloud [5]. Also in greenhouse effect monitoring, DHT11 and ESP8266 were used, supported by an MG135 CO_2 concentration sensor [6]. More advanced solutions for monitoring CO_2 and PM utilize the ESP32 microcontroller with FreeRTOS operating system, working with SCD30 infrared sensor and HM3301 scattered light sensor, respectively [7].

The solution used for designing the measurement system must always be adequate to the goals within the set requirements. We observe the increase in complexity, which results in efficiency improvement, cost savings of materials, equipment, and labour, as well as improved safety and quality [8]. It is, in general, true for mass production, especially when produced by teams of experts. Applying advanced solutions for simple and unit tasks unnecessarily increases the project costs, both in equipment and labour parts. When complicated or unpopular approaches are applied, the staff needed to design, run, maintain, and service them is much more expensive. On the other hand, off-the-shelf solutions can be supervised by engineers after short training, significantly reducing the project costs.

The approach of designing the system to allow gathering data for scientific applications differs from those dedicated to mass production. Whereas in the industrial approach, the prototypes are built to increase technological readiness levels and finally create a product ready to sell [9], in academia, the prototype is a target device or system. On the other hand, the academic prototype must still meet the requirements for operating in the target environment, like resistance to rain and dust.

This work focuses on the development and implementation of a sensor network designed for environmental monitoring within a research context. The primary objective is to demonstrate the realisation of a measurement system capable of acquiring environmental data through an IoT approach. A key consideration in the design process was minimising labour and setup time across mechanical, electronic, and software components. To achieve this, the system was built using widely available, cost-effective components. Popular and low-cost solutions were deliberately selected to ensure affordability, ease of deployment, and replicability.

II. ELEMENTS OF THE NETWORK

The research has been conducted in an area of about 50 m x 50 m covered by Wi-Fi with access to 230 V AC electricity. Installing cables with sockets proved more cost-effective and less time-consuming than designing and implementing battery or solar/wind power systems. Continuous measurements of changes of environmental parameters were required across different locations 24 hours a day, over several months. This necessity led to the development of a system (Fig. 1) with a typical architecture [3], containing nodes with sensors, microcontrollers, Wi-Fi modules and power adapters allowing powering from the 230 V network. The nodes connect to a server where the data is stored. Further data analysis was conducted on a personal computer.

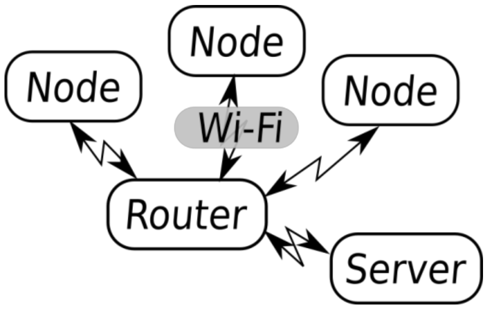

Fig. 1. Schematic overview of the measurement network

A. Nodes

As part of the project, we constructed 12 identical sensor nodes, each following the design shown in Fig. 2. It consists of (1) the sensor (Plantower PMS7003 for PM and Sensirion SCD30 for CO2), (2) a Wemos D1 mini module with ESP8266 microcontroller integrated with Wi-Fi and (3) a 230 V AC to 5 V DC power adapter. All components are enclosed in a 3D-printed PET-G housing.

The choice of sensor type is based not only on objective indicators such as measurement range, accuracy, long-term stability, communication standard, and price, but also on pragmatic ones such as local market availability and experience in use. Both sensors used had a UART serial interface that could be easily connected to a microcontroller. The price for the applied PMS7003 is about 20 EUR, while for SCD30 is 67 EUR on the Polish market.

The variety of microcontrollers is huge, so the choice is even more subjective. The D1 Mini module with ESP8266 is a low-cost (ca. 3 EUR) solution that integrates the appropriate computing power for the sensor application with the Wi-Fi communication stack. The possibility of programming using the Arduino IDE was also an advantage.

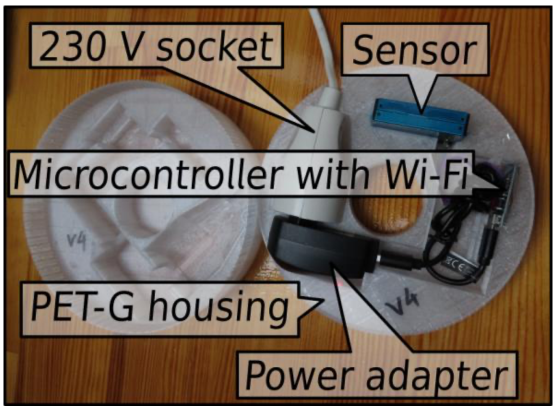

Fig. 2. Internal view of an node for PM measurements.

Owing to the availability of 230 V AC at the research plot, we abandoned a solution with battery power management. The 5 V power adapter selected was among the most affordable available, with a rated current of 1 A. The initial test showed it is suitable for this application. During the few months of experiments, it proved to be the least reliable part—some of these used in the nodes stopped working, terminating data acquisition. However, sourcing low-cost power adapters with reliable long-term performance remains a challenge.

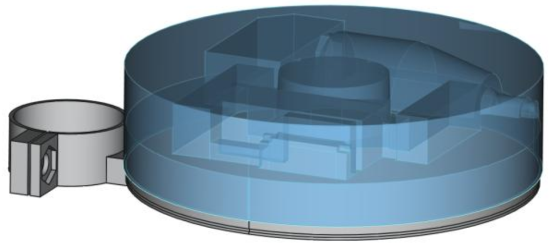

Fig. 3. The housing designed in FreeCAD.

The 3D-printed housing (Fig. 3) was designed in FreeCAD and printed on a Prusa MK3 3D printer using PET-G filament. The housing serves a mechanical construction for all node components. It consists of two parts. The elements can be placed in the niches of the bottom part. The upper part protects against rain and pollutants. An air inlet at the bottom ensures airflow to the sensor. The node can be mounted on a pole using an arm and an appropriate clamp. The housing performed reliably during tests that took place both in winter at freezing temperatures and in summer in full sunlight.

Prior to deployment, all nodes were co-located in a controlled indoor environment for a few days. During this time, the data were acquired in order to compare their values. After the period, offset and gain errors were established to standardize sensor readings. Due to the unavailability of a dedicated reference sensor for PM, the data were fitted to the data provided by state monitoring stations [1]. However, the primary objective was to observe the spatial and temporal variations rather than to measure the values accurately. For CO2, the reference meter was TESTO 440 with 0554 1111 probe.

B. ESP8266 D1 Mini firmware

The ESP8266 firmware for all nodes is the same. No configuration is required. It uses the web technology approach typical for IoT. It consists of (1) a connection to the local Wi-Fi network, (2) reading the Media Access Control (MAC) address of the integrated Wi-Fi module and (3) an infinite loop. Within the loop, data from the sensor is read via the serial port and stored in the string variable DATA, which contains hexadecimal values encoded in ASCII format. For the PMS7003 sensor, DATA represents the raw data frame directly output by the sensor. In the case of the SCD30 sensor, DATA contains the content of the Modbus response frame, received after issuing the 'Read Measurement' command. Then, the firmware sends a GET request to the server using the following address and parameters:

server/index.php?mac=*MAC*&data=*DATA*

The request is sent immediately after new data is delivered from the sensor. The loop executes without intentional delays, minimizing latency. There is also no preliminary data analysis or correction of the known errors typical for smart sensors, to avoid the need for individual node configuration.

The *server* name or IP address had to be decided when the network was designed. Additionally, the firmware includes a simple web server. Whenever the node is requested, the current sensor readings are provided. This simple solution allowed for checking the sensor state and the operational status of the node. This part of the web server is not mandatory for the correct system operation.

The Arduino IDE environment was used to code and program the microcontrollers. It possesses libraries for the sensors and supports the D1 Mini module.

C. PHP server code

The PHP server is hosted on a Raspberry Pi 3. Only one short file is needed, named index.php. Its code is run if a node makes the request. This PHP script extracts the MAC and DATA from the request, then checks if the DATA contains the correct checksum. If no, such a line is not processed as it is considered unreliable. Otherwise, the hexadecimal numbers placed in the string are converted to decimal equivalents.

After each execution of this script, a text file is opened. The filename is constructed by concatenating the MAC with the current date. This way, each sensor has its data stored in a separate file, keeping individual files manageable in size. In the file, a line is appended to the end. The line starts with the current server date and time. A minor latency between measurement and processing on the server does not affect the analysis due to the low-frequency nature of the signal.

The remainder of the line contains the data obtained from the sensor stored in human-readable decimal format, separated by the space character. The data is stored in a consistent order, matching the format provided by the sensor. All values are natural numbers, and the units of measurement are defined by the sensor manufacturer.

The text files are accessible in real-time facilitating monitoring and debugging during data acquisition. Further processing with mathematical tools like Excel, MATLAB or Python is not complicated as well.

The firmware written for the microcontrollers, together with this PHP script, enables scalability. The number of nodes can be increased without any modifications. Each node has a unique identifier, which is the MAC address of the Wi-Fi module. The identifier serves as the name of the file where the environmental data is stored. The system works well for 12 nodes, making it suitable for the assumed scientific task.

Using a Raspberry Pi as a PHP server seems proved to be an efficient and compact solution for monitoring collected data. It works as a standalone device connected only to the power supply, without a monitor or keyboard. Easy access via SSH or VNC allows us to use the advantages of the Linux operating system. Despite its limited processing power, no impact of VNC operation on data acquisition was observed. Simple Python scripts generate plots visualising the data. File transfer to a Windows computer was done using the SFTP protocol with WinSCP. Application of a desktop computer with a PHP server working under Linux or Windows is also possible but is unlikely to offer significant advantages in this context. Because the server has to work constantly, the power consumption of such a computer is, however, higher. Cloud-based storage is a viable alternative, though it introduces additional cybersecurity aspects and firmware complexity considerations.

D. Router

The router operates with the default Dynamic Host Configuration Protocol (DHCP) settings, which automatically assign IP addresses to connected devices within the network. This default behaviour simplifies the initial setup and is sufficient for general operation. However, manual configuration becomes necessary when monitoring the operation of individual nodes. In such cases, the user must know the IP addresses of the nodes, which requires reserving IP addresses based on the MAC addresses of Wi-Fi modules. If node monitoring (e.g., via ping) is not performed, no additional configuration is necessary.

E. Python script for data analysis

Data analysis was conducted using Python, specifically the Pandas and Matplotlib libraries, within the Anaconda Navigator environment and Jupyter Notebook. This setup provided a flexible and interactive platform for processing and visualizing the collected data. The Pandas library facilitated efficient data import from structured text files and enabled convenient manipulation of large datasets. Matplotlib was used to generate visualizations, allowing for the extraction of valuable insights and the creation of plots such as those published in [1] or presented in Fig. 4. Offset and gain errors were also corrected during this phase. Performing these corrections at the data analysis stage allowed the same microcontroller firmware to be used across all nodes without requiring individual configuration.

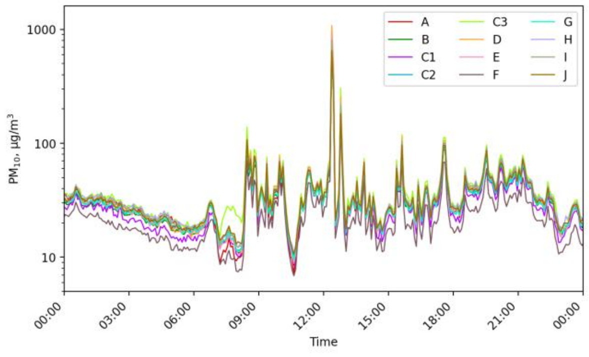

Fig. 4. Example of PM10 changes registed by 12 sensors visualised in Python.

F. Weather monitoring

Additional monitoring of the weather conditions was carried out independently of the main measurement system and was based on a setup adopted from a previous project. For this purpose, a Renkforce WH2600 weather station was used. This station is capable of measuring temperature, pressure, humidity, wind speed and direction, and solar radiation. Data is available through a built-in web interface. A Python script automatically retrieves the data and saves it into a text file every minute. The correlation of weather conditions to PM and CO2 indications is subject to further investigation.

III. CONCLUSION

In contrast to prevailing trends that favour complex and costly solutions, this study presents a straightforward and economical approach to measuring environmental parameters. The measurement network was deployed on a 2 500 m^2 plot with access to electricity and Wi-Fi as part of a research project. By selecting widely available components, we were able to keep the system inexpensive. While the sensor cost could not be reduced, other components were selected for their low cost.

We used an ESP8266 microcontroller to interface the sensor with the network. A Raspberry Pi 3 served as the data server. We implemented communication using HTTP GET requests, transmitting measurement data as URL parameters. This approach allowed us to keep both the microcontroller firmware and the PHP script concise and efficient.

The system operated reliably for several months, initially measuring PM and later acquiring carbon dioxide (CO2) concentrations. During this period, the power adapter was the most unreliable component, as we had not considered its reliability parameters during the design phase. This experience highlights the need to evaluate all system components—not just sensors and processors—when aiming to build robust, long-term monitoring solutions.

ACKNOWLEDGMENTS

This research was funded by EU Funds for Silesia 2021-2027, MONCO2 project No. FESL.10.25- IZ.01-06C9/23-00 “The modern methods of the monitoring of the level and isotopic composition of atmospheric CO2”. (AW); and by the Polish Ministry of Science and Higher Education 02/050/BK-25/0047 (JW).

REFERENCES

[1] A.Wiora, J.Wiora, J.Kasprzyk, “Indication Variability of the Particulate Matter Sensors Dependent on Their Location”, Sensors, vol.24, 2024, article 1683.

[2] S.Eichstädt, A.P.Vedurmudi, M.Gruber, D.Hutzschenreuter, “Fundamental aspects in sensor network metrology” ACTA IMEKO, vol.12, 2023, pp.1–6.

[3] K.Gulati, R.Boddu, D.Kapila, S.Bangare, N.Chandnani, G.Saravanan, “A review paper on wireless sensor network techniques in Internet of Things (IoT)” Materials Today: Proceedings, vol.51, part 1, 2022, pp.161–165.

[4] F.Mao, K.Khamis, S.Krause, J.Clark, D.M.Hannah, “Low-Cost Environmental Sensor Networks: Recent Advances and Future Directions”. Frontiers in Earth Science, vol.7, 2019, article 22.

[5] N.Mitu, V.Vassilev, M.Tabany, “Low Cost, Easy-to-Use, IoT and Cloud-Based Real-Time Environment Monitoring System Using ESP8266 Microcontroller”, International Journal of Internet of Things and Web Services, vol.6, 2021, pp.30–44.

[6] Z.Wan; Y.Song; Z.Cao, “Environment Dynamic

Monitoring and Remote Control of Greenhouse with ESP8266 NodeMCU", IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), 2019, pp.377-382.

[7] D.Pineda-Tobón, A.Espinosa-Bedoya, J.Branch-Bedoya, "Aquality32: A low-cost, open-source air quality monitoring device leveraging the ESP32 and google platform", HardwareX, vol.20, 2024, article e00607.

[8] L.Zhang, Y.Li, Y.Pan, L.Ding, „Advanced informatic technologies for intelligent construction: A review", Engineering Applications of Artiɘcial Intelligence vol. 137, 2024, article 109104.

[9] S.Yfanti, N.Sakkas, „Technology Readiness Levels (TRLs) in the Era of Co-Creation", Applied System Innovation, vol.7, 2024, article 32.