Bulletin of the *Transilvania* University of Braşov • Vol 12(61), No. 2 - 2019
Series III: Mathematics, Informatics, Physics, 443-456
https://doi.org/10.31926/but.mif.2019.12.61.2.22

INNOVATIVE AIR QUALITY SYSTEM WITH EMERGENCY NOTIFICATIONS

Alexandra BĂICOIANU [1], Marius DEMETER [2] and Anca VASILESCU[*,3]

Abstract

This paper is organised around the problem of environmental monitoring and ambient environmental monitoring, as parts of social computing and the Internet of Things areas of research. In recent times, the use of sensing devices and technologies has had increasing significant interest in many fields, some remarkable results being achieved, particularly with evaluating, monitoring and analysing the air quality. The main purpose of the system developed in our project is to introduce a way to monitor the air quality in a given room, advancing the possibility of viewing history or even real-time data via a web service. Although the level of hardware is abstracted, the focus is on the problem that it addresses and on solving it, as well.

2000 *Mathematics Subject Classification:* 93C85, 91B76, 97M50, 68Q85, 97R50, 97P30.

2012 *ACM Subject Classification:* Computing Methodologies

Key words: air pollution, ambient environmental monitoring, smart home, sensors and actuators, Internet of things, web service.

1 Introduction

Environmental monitoring refers to "the identification and measurement of pollutants in the form of chemical, biological, microbiological, and radiological containments in water, soil, and air".[8] Additionally, ambient environmental monitoring takes into account parameters like temperature, humidity, noise or light levels. In the environmental monitoring context of health and especially well-being, many

[1]Department of Mathematics and Computer Science, *Transilvania* University of Braşov, Romania, e-mail: a.baicoianu@unitbv.ro

[2]Faculty of Mathematics and Computer Science, *Transilvania* University of Braşov, Romania, e-mail: marius.demeter96@yahoo.com

[3]*Corresponding author*, Department of Mathematics and Computer Science, *Transilvania* University of Braşov, Romania, e-mail: vasilex@unitbv.ro

factors could contribute to the precision of the results or the efficiency of the process, some of them discussed in [8, 13, 4].

The aim of this paper is to present an Internet of Things or IoT-based system which can monitor the air quality and send notifications in case of emergency. The main goal of the whole system is to provide a way to prevent emergencies inside the compartments containing this system. This system monitors the following properties of air: temperature, humidity, the level of carbon dioxide, gas and alcohol. After calibration, it will be able to provide notifications via SMS if the values of the variables mentioned above exceed a certain threshold.

The four main components of our system are a web platform, two servers and a data acquisition system. The web platform is the only component designed to interact directly with the customer, thus representing a control panel dedicated to managing the entire system. The first of the two servers developed here aims to achieve communication between the device and the web client. More specifically, all of the data from the monitoring system are sent to the server for processing, and then they are further directed to customers. The second server is used in its entirety by the function of linking and attaching a device to a given user account. It runs on a Raspberry board-based system, the connection being achievable only with other devices on the same network. As the acquisition system, a RaspberryPi 3 Model B development board is used, different modules being connected to it for data gathering and for displaying certain informational messages.

In [7], the authors reveal that "Raspberry Pi encourages experimenting with its hardware configuration", so that, using a Raspberry Pi board for our project was a market decision based on its well-known support for developing modern and educational IoT projects. Some other intelligent environmental monitoring systems were also presented in [2, 3] or in [10], where a part of an IoT environmental sound recording system is developed to monitor the birds' populations, with remarkable educational targets for kids.

2 System description

In the context of environmental monitoring, different approaches and solutions are available in literature, see [6, 11]. In contrast with these perspectives, we will consider a system consists of four components: the web page, two servers, and the data acquisition system. The proposed architecture of our project is diagrammatically represented in Figure 1 and described in details in the next sections.

2.1 Web platform

The main functions that the web platform performs concern: accounts management and authentication, attaching devices to the accounts, different formats for data viewing, settings and platform customization. These features are briefly presented below.

The web component provides the ability to create a new account for customers who interact with the system for the first time. The input data entered the related

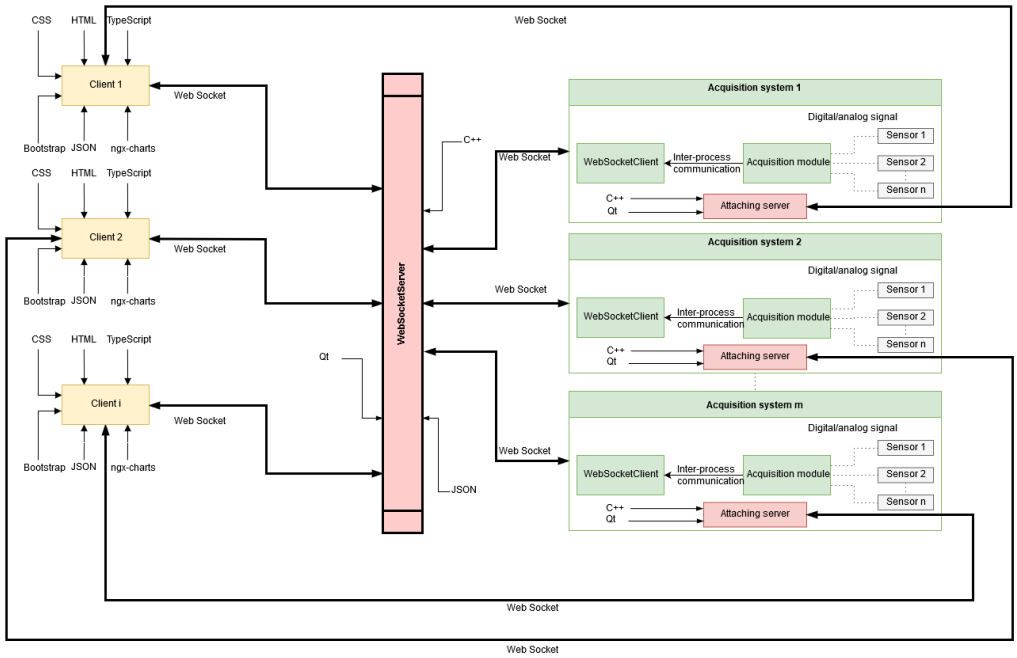

Figure 1: Proposed system architecture

fields are real-time verified, and the user is promptly alerted in case of an error. This way, when the data are sent for processing, many problems that might appear on the server-side are already anticipated and predicted. Authentication is also provided by the web component.

After the first login, the web component will automatically ask the user to attach a device to the account. In order to be able to attach a device to the account, the platform will look for the compatible devices on the same network by default. Although the platform can find a compatible device, it does not mean that the attachment to the account has been performed, which is actually part of the server component. At the same time, specific data packets are sent in order to locate the device for later authentication. Platform content may not be available until the device attached to the account will have been located.

Access to settings, whether related to system control, such as data sampling rate or even customization of the platform by selecting a preferred theme, is also offered through our system's web platform. Secure browsing from one page to another is assured by restricting access to certain pages for customers who are not authenticated. Showing pages in a user-friendly way was an interface priority. Both the pages and the platform components merge on different existing resolutions, so that they can be appropriately accessed both on a mobile device and on a computer screen.

2.2 The main server as communication server

In the context of this project, the main server is the basic architectural component since most of the communications between the other components have been made through it. The most important functions performed by the main server are: (i) providing communication between components, most of the data are first passed to the server, and then they are forwarded to the destination components; (ii) operating the customers management by creating links between customers for a more efficient communication; (iii) assuring the communication with the database, specifically where the interpretations of the data packets require this; (iv) quickly and efficiently interpreting the data packets using an event-based system; (v) saving the history of actions for evidence, recovering possible errors, or even security breaches.

2.3 The second server dedicated to attaching the devices to the user account

The system got a second server, which is dedicated to attaching the devices to the user account. It is an invisible component to any user, the access to it is possible exclusively through other associated applications, such as the web component. This server mainly operates on: (i) permanent communication with the main server, whether or not the device has been attached to an account; (ii) sending identification packages to the web component as well as to the main server; (iii) sending package attachments to a specific account to the main server; (iv) the possibility of pairing with applications other than the web component of this system.

2.4 Data acquisition system

The acquisition system mainly deals with collecting and sending data to the customers. It is also an invisible component to customers, and it may be accessed only through project-related applications, such as the web component. We can refer to this component as a client since it is connected to the main server. The main functions performed by this procurement system refer to (i) collecting data from the connected modules and sending them to the main server; (ii) hosting the server dedicated to attaching the devices to the user account; (iii) assuring the communication between the connected modules and the acquisition system in an efficient way, with respect to the fact that the system permanently reads data from modules and not just on request; (iv) providing the ability to attach the device to a user account, or even to another linked applications, if applicable, via an interface button.

For the interest of developing this specific IoT system, some applicable sensors have been chosen, mainly a digital DHT11 temperature and humidity sensor and an SNS-MQ135 sensitive air quality sensor. The DTH11 sensor is an NTC-based one (*Negative Temperature Coefficient*) for measuring temperature and humidity. It interfaces with an 8-bit serial and bidirectional communicating microcontroller

to output the values of temperature and humidity as serial data. Moreover, it easily interfaces also with other microcontrollers, in our case, one provided by a Raspberry Pi single board micro-computer. An SNS-MQ135 sensor is a useful solution for measuring the air quality parameters, both in analog and digital format, by detecting ammonia, nitrogen oxides in combustion environments, alcohol, benzene, smoke, carbon dioxide, and other air pollutants. These sensors have fitted very well the scope of this project by classifying the output values in many risk classes, as follows: no risk, moderate air quality, sensitive people could be harmed, all people are affected, negative effects and emergency. National considerations and premises for an acceptable indoor air quality [12] are coming out as very important research topics also in crowded, digitalized and over-tech societies.

Together with these sensors, a communication button, an LCD for real-time data displaying and an analog to digital conversion module for the air quality data are the main hardware components held by our breadboard. When a user is near the system prototype, this LCD is a better and more convenient solution for reading the output values instead of connecting the web application.

Our data acquisition system is related to the device attaching server and to the client for ensuring the communication with the main server.

3 Air quality mathematical modelling

In order to investigate the environmental pollution impact on a given area for both prediction and prevention, two different approaches are recognised as air quality mathematical modelling solutions, namely physical modelling and mathematical-computational modelling. Two of the statistical models which are the most appropriate in this context are the MLR (Multiple Linear Regression) analysis together with the SPSS software and the ANN (Artificial Neural Network) model with MATLAB software support. The mathematical-computational approach considers the areal relationships represented as a mathematical model, ready to be solved analytically or, in case of a large system dimension, the solution emerges from an algorithmic and computed process. The specific mathematical support in this context could be, on the one hand, an empirical or statistical one - based on expressing the system behaviour in terms of statistical distributions and probability theory or, on the other hand, a deterministic one - based on expressing the dynamic model with partial differential equations. It follows that we have a mathematical simulation of how air pollutants disperse in the air by evaluating their downwind concentration in certain places and times.

Usually, the air quality is expressed by the value of two indicators: the air quality index and the concentration of each pollutant for given natural conditions. In addition to the thresholds already mentioned in the previous section from the human perspective, these calculated values are mathematical thresholds for the air quality classification in good, moderate, poor, very poor or severe pollutant level. The air quality index, AQI, is evaluated for the main criteria pollutants, such as: CO (Carbon Monoxide), SO_2 (Sulphur Dioxide), NO_x (Nitrogen Oxides), SPM

(Suspended Particles Matter), $RSPM$ (Respirable Suspended Particles Matter), or even TSP (Total Suspended Particles) to measure the content of Pb, Cd, Zn, Cu. The specific formula recommended by the standard EPA 1999 [14] for the AQI is as follows:

$$I_P = \left[\frac{(I_{Hi} - I_{Lo})}{(BP_{Hi} - BP_{Lo})}\right](C_P - BP_{Lo}) + I_{Lo} \tag{1}$$

where I_P is the AQI for pollutant P, and the other values which it depends on are: C_P for the current level of pollutant P in the air or atmosphere, BP_{Hi} for the specific breakpoint that is greater than or equal to C_P, BP_{Lo} for the specific breakpoint that is less than or equal to C_P, I_{Hi} for the subindex value corresponding to BP_{Hi} and I_{Lo} for the subindex value corresponding to BP_{Lo}.

The pollutant concentration, evaluated at the ground level, is successfully evaluated using the Gaussian plume equation [5]:

$$C = \frac{Q}{2\pi u\sigma_y\sigma_z} exp\left(-\frac{y^2}{2\sigma_y^2}\right)\left\{exp\left(\frac{(z-H)^2}{2\sigma_z^2}\right) + exp\left(-\frac{(z+H)^2}{2\sigma_z^2}\right)\right\} \tag{2}$$

where Q is the source emission rate, u is the wind speed, y is the crosswind distance from stack of point of interest, z is the vertical height of point of interest (0 for ground-level concentration), H is the effective stack height (includes plume rise), σ_y is the horizontal stability parameter (a function of downwind distance x, and stability), σ_z is the vertical stability parameter (a function of downwind distance x, and stability).

These kinds of values, calculated for specific pollutants and physical conditions, are operated as input values in our project to automatically be compared with the information gathered by the sensors, for an accurate decision upon the emergencies and prompt notifications sending to the customers.

4 System development

Considering the specific development topics, our main interests focused on: (i) implementing a web-socket server ready to provide the platform structure and real-time data for all its clients, (ii) creating an HTML pattern which has to encapsulate the web platform structure, (iii) assuring a real-time communication with the acquisition component, both for data receiving and for sending those data to each of the users' accounts and (iv) real-time viewing and processing data for each customer.

A sum of specific details about the modern software technologies and programming languages involved in these system components development is mentioned below. Both the main server and the device-attaching server are written in C++ with Qt as a modern inter-platform framework for software development. A controls-component library for creating both graphical and non-graphical user interfaces is widely used in this project. Other important Qt-based applications are the KDE desktop environment, the web browser Opera or Google Earth, Skype and Qtopia

as well. For the specific interest of the servers and the acquisition system in addressing the Raspberry Pi board pins using the GPIO support, the C-based library WiringPi is used.

4.1 System resources for communication

Our system's main server aims to achieve communication between the device and the web client. Specifically, all data interchanged between the device and its assigned applications must first pass through this server. The presence of this server is particularly justified by the necessity of monitoring the data displayed by the device modules, both inside and outside of a network without any delay between responses due to limited resources. Another reason could be further expansion possibilities, as we will mention as future work.

Considering the software aspects, in making this component we used the following support: C++ as a programming language, Qt as a framework for modern applications development, CMake to generate the solution valuing the advantages of a build system, Visual Studio for editing the code in a modern and up to date integrated development environment.

For the remaining part of this section, we will describe the contributions of these technologies to developing our software components as servers, interfaces or managers. As the system workflow, when the main server starts, the first step is to have all manager-applications successfully initialized. This stage is highlighted by messages displayed in the console or through log files. After all the managers have successfully started, the server expects and is waiting for customers to connect. Once there is at least one connected customer, a message is received from him/her, and it is forwarded to the package manager, where the polymorphism and reflection principles will be used to sound the desired behaviour. These interpretations of incoming packets may result in database modification, customer relationship creation, sending another package as a response to the sender, or sending another package to another customer.

4.1.1 Signals and slots

The connections between the objects that Qt framework puts at work operate through a signals-and-slots mechanism. For a developer to be able to use this functionality, it only needs to use the specific interface provided by the framework.

The operation of this signals-and-slots based mechanism is as follows: the developer can define a specific behaviour that he wants to be executed in certain situations - this behaviour is defined in a slot, and then a triggering signal under certain circumstances is defined as well. When the signal triggers, the associated behaviour is launched. In other words, the slot defines the desired behaviour, while the signal represents the action that triggers this behaviour. Several slots can be connected to the same signal, meaning there may be more behaviours that trigger at the time of an event. Also, a slot can be connected to multiple signals, resulting in the same behaviour under different circumstances. Although it is possible to

connect a signal to another signal, this is not very common, the effect being to call up a specific behaviour when signal number 1 is triggered, but using signal number 2 as an intermediate. This functionality is useful for frameworks, where we may need to extend the hidden behaviour that occurs when a signal is triggered.

4.1.2 Interfaces

As we have already mentioned before, the main server uses only one interface, namely IPacket.

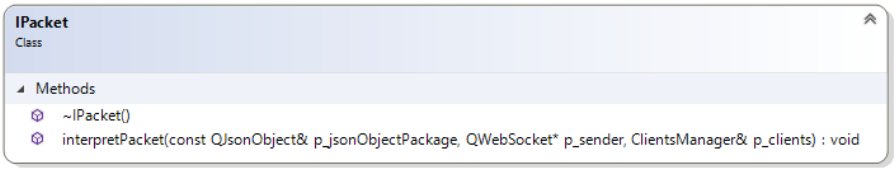

Figure 2: Interfaces

The meaning of the IPacket interface represented in Figure 2 is to define a template for all data packets that can be both received and sent from client to server and vice versa. Each of these packets of data must implement the Packet interpreter method, with the specific parameters: packet content, packet sender, and client manager. This solution is recommended, especially when the message resulting from interpreting the received data packet should not be sent back to the sender, the recipient being another.

Each class that inherits the IPacket interface must define a proper behaviour for interpreting the received package.

4.1.3 Managers

In order to be able to logically and efficiently separate the main functionalities, the manager-components have been defined and used here. Their purpose is to manage certain resources that can be categorized to a certain extent. For example, there is a manager who is strictly dealing with server-to-database communication or another manager who is strictly dealing with client-server communication. The member objects of these managers are usually hidden using the encapsulation principle, being exposed to specific methods only. Another goal is to submit a wrapper over the framework predefined objects, thus hiding the functionalities we do not need. At the same time, these managers contain methods that can also use several methods implicitly defined by the framework, thus enabling them to achieve specific behaviour depending on the case.

There are many manager-type components developed for this system, as follows: package manager, database manager, and clients manager.

The package manager contains the packets mapping, since the given package has been broken into data packets for being sent over the network. The Figure

3 points out that each such a data packet has a name and a function as a correspondent, being able to interpret that packet and consequently generates responses or actions on a case-by-case basis. At server startup, this manager initializes one instance for each type of data packet through its builder and inserts them into mapping.

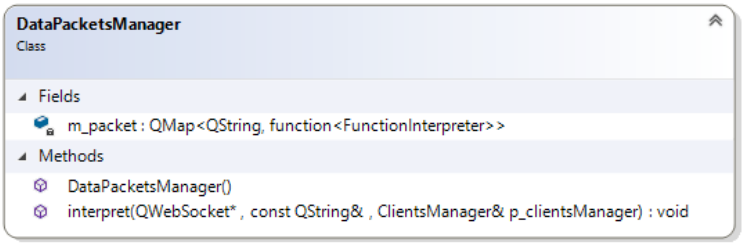

Figure 3: Package Manager

The goal of **the database manager** is strictly to deal with communication between the main server and the system database. As member data, one can see in the Figure 4 the path to the database and the QSqlDatabase object provided by the Qt framework together with all the necessary functionalities. The constructor also offers the facility to initialize the m_path variable.

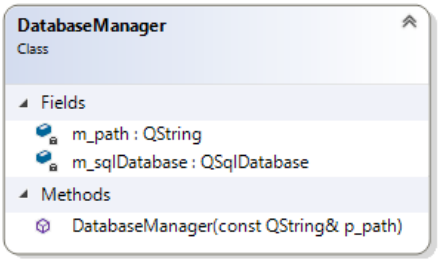

Figure 4: Database Manager

The clients' manager provides the appropriate support for connecting a device to a client, but also a client to a client, as the Figure 5 covers up. Specifically, it allows the association of a device-type client with a user-type client, so that when certain data packets are interpreted as sending a response to a client who is other than the sender, the scenario is still viable.

4.2 Server dedicated to attaching the device

The specific server for involving the devices into the system is running entirely on a Raspberry board, as compared to the central communication server which has to run on a specialized computer. In addition to its responsibilities already presented in the previous section, this server can also handle for attaching a device

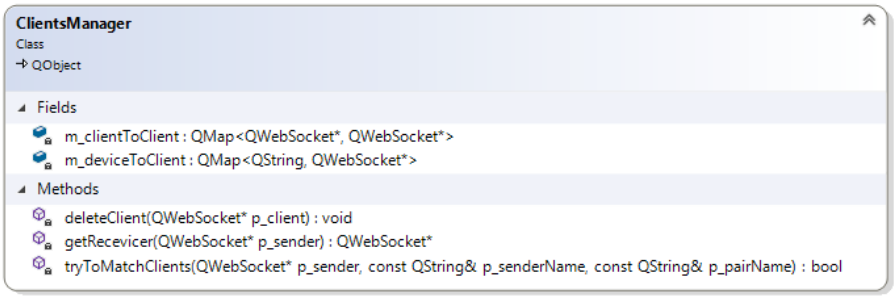

Figure 5: Clients Manager

to an affiliate application. For the current state of our system, the web application can be considered as being affiliated, meaning that once the user has logged in, (s)he needs to attach a device. For this to be feasible, the dedicated attachment server must communicate directly with an operational button connected to the Raspberry hardware.

In the step of attaching a device to a user account, the web application searches for and sends a package to the Raspberry board every second for one minute. The answer from it, if there is a stable connection, varies depending on the status of the button. When the button is pressed, the board sends a packet of data to the main server, and this inserts into the database a specific link between the account and the device. The web application will also be notified via another data packet and the interface will change accordingly, thus exposing the information provided by the modules connected to the Raspberry device.

4.3 The web application

The web application is the only component that can interact directly with a potential user. The three stages in which a user can be, following the web application scenarios, are: unauthenticated user, authenticated user without any attached device or authenticated user with an attached device. The unauthenticated user has access only to the registration and authentication functions. The authenticated user without an attached device has access only to the button that starts the process of attaching the device.

The web platform of this project represents the component that is accessible to any potential client. Users that do not own this system will only have access to registration and login functions, the remainder of the shares being blocked by a message which asks for attaching a device to the user account. Customers who own the system have full privileges, meaning access to viewing real-time data sent by the system as well as history. Moreover, this type of users have access to various settings, be they related to the system itself or customization.

Concerning data management, our web platform provides the ability to view real-time data. This functionality is present in two forms, namely as (1) quick

data viewing, meaning that the latest data received from each category can be viewed on any page or (2) detailed data viewing, which is available only on the specially designed page for this functionality by giving access to many of the data received from each category as a chart. Besides, there is the option to also have a data history based on a data viewing in tabular formats, like the capture from the Figure 6 which highlights the contrast between certain time moments.

Figure 6: Temperature history

Here, the web pages are developed using JavaScript programming language and Angular as a development framework, as well as other derivative technologies: Bootstrap, JSON, Angular Material, ngx-charts et al.

4.4 Implementing the data acquisition component

From the operational point of view, our data acquisition system has efficiently accomplished at least these three functions: (1) converting the physical behaviour into a measurable signal, usually an electric one; (2) measuring the sensors' signals and managing the actuators' actions for an entire information overview; (3) data analysing and viewing in a useful and user-friendly interface.

Data acquisition systems are continually increasing their popularity in modern industry and even research, based on the interest in accurately collecting data as a premise of processed-data reliability. Despite this, the acquisition and maintenance costs to this type of systems are rather high. Our present project aims to find a solution for this problem by addressing a low-cost and high-performance data acquisition and processing system.

A sample of how our data acquisition component is working is in Figure 7. During the data acquisition time, the specific threshold for the input data is tested and, in case of exceeding values, the system launches an emergency notification. Hence, the connected customers receive SMS notifications in case of abnormal values purchased from modules. A third-party server has been used for the SMS notifications, that is *Twilio Notification System* (*Twilio Notify*). This service allows the application to send notification messages to multiple users, over different communication channels, all from the same unified API. With a single API request,

one can connect with contacts using SMS, mobile apps, Facebook Messenger, and more. The user can even specify preferred channels for reaching certain users, and tag them for more granular sending capabilities [15].

Figure 7: Sensors data view

5 Conclusions

Different studies are available in literature, most of them revealing smart environment monitoring systems [9, 1]. In comparison with these, this study aimed to understand the environmental monitoring issues and to develop an innovative, scalable and generic solution to this actual problem of monitoring. The problems of health, wellness, and environmental sensing as well, will continue to keep people attention at high levels for maintaining the parameters of our personal life well supported by smart technologies. In many research domains, we already notice the emergence of multi-sensor techniques that concern the impact of environmental qualifiers on our physical-physiological and mental-psychological well-being. The market for environmental sensing and monitoring is estimated to be growing by 6.5% per year [8].

This project-based article presents both hardware and software considerations for developing an original system for analysing the air quality and sending notifications in case of emergency. The main target of this system was to offer a low expensive way to facilitate monitoring real-time information about the quality of the air from a certain room and also to provide a history of the collected data. As a necessary step forward from a home to a smart home, this project is also a real solution for an industry monitoring system to have a comfortable climate or even to prevent working accidents.

The leading features of our system are presented below.

The inner web approach is provided by *a high-performance web system*. Taking into account that there is a limited set of web resources, we can say that an

application running on a web browser cannot be compared with a natively running application. In our context, processes that are considered to be resource consumers are running independently of the web application. Consequently, we can say that the performance of the system is at its best. *Highly efficient servers* have been provided, whereas our servers are written in C++ by controlling the resource management to a very high level. As a positive result, the servers run at a very high speed, even on less equipped devices such as Raspberry board. Both the communication server and the attaching server consume between 3MB and 5MB memory during execution.

The web platform was conceived in a manner that allows access to it from most modern devices and beyond, thus increasing step by step *the portability* of our system. Information is always displayed in an easy-to-read format, regardless of the host-screen resolution, encouraging the user to utilise this application. The project was conceived in a manner that allows the user to configure both the account and the device in the most accessible possible conditions, providing a low level of difficulty in use and a very user-friendly interface overall.

Since the connected modules are independent, our system is *modular and flexible*. We may count on the possibility of easily replacing the hardware modules in case of a defect or for achieving performance testing. Our system needs *cheap components*, more than sufficient for a small room, although we can assume that the current configuration is used for demonstration purposes assigned to the general theme.

Based on the hardware and software structures already defined, the expansion possibilities for our system are in the direction of creating a multi-device neural network or attaching many devices to a given user account or even developing a supplemental application for restricting certain aspects but providing a stable set of developer capabilities. A neural network like that could be trained to predict and alert upon some dangerous situations, based on a set of given patterns. All the attached devices and the authenticated users would be part of this learning process, transparently and cooperatively, with successful results in terms of increasing the human condition and decreasing the total cost at the end.

Further research in the field aims to use knowledge discovery techniques for supporting the decision-making process in the problem of air quality, by developing a monitoring multi-agent system ready to assess the air quality dependent variables based on previous data measurements.

References

[1] Al-Dahoud, A., Fezari, M., Jannoud, I., AL-Rawashdeh, T., *Monitoring Metropolitan City Air-quality Using Wireless Sensor Nodes based on ARDUINO and XBEE*, Proceedings of the International Conference on Circuits, Systems, Signal Processing, Communications and Computers (CSSCC), Vienna, Austria, March 1517, 2015, Series: Recent Advances in Electrical Engineering Series, 2015.

[2] Athanasiadis, I. N., Mitkas, P. A., *An Agent-Based Intelligent Environmental Monitoring System*, Management of Environmental Quality An International Journal 15(3), 2004.

[3] Athanasiadis, I. N., Mitkas, P. A., *Knowledge Discovery for Operational Decision Support in Air Quality Management*, Journal of Environmental Informatics 9(2), 100-107, 2007.

[4] Calvert, S., *Air Pollution Research Problems*, Journal of the Air Pollution Control Association, November 1971 Volume 21, No. 11, 694-701, on-line 2012.

[5] Cooper, C. D., Alley, F. C., *Air Pollution Control. A Design Approach*, Waveland Press, 4th ed., 2011.

[6] Kaur, N., Mahajan, R., Bagai, D., *Air Quality Monitoring System based on Arduino Microcontroller*, International Journal of Innovative Research in Science, Engineering and Technology, 9635-9646, 2016.

[7] Kurkovski, S., Williams, C., *Raspberry Pi as a Platform for the Internet of Things Projects: Experiences and Lessons*, Proceeding of The 22nd Annual Conference on Innovation and Technology in Computer Science Education, Bologna, Italy, 3-5 July, 64-69, 2017.

[8] McGrath, M. J., Ni Scanaill, C., *Sensor Technologies - Healthcare, Wellness, and Environmental Applications*, SpringerLink, 2014.

[9] Mujawar, T., Deshmukh, L., *Atmospheric Air Pollution Monitoring - Smart Environment Monitoring System Using Wired and Wireless Network: A Comparative Study*, Atmospheric Air Pollution Monitoring, 2019.

[10] Soro, A., Brereton, M., Dema, T., Oliver, J.L., Chai, M.Z., Ambe, A.M.H., *The Ambient Birdhouse: An IoT device to Discover Birds and Engage with Nature*, Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, April 2126, Montreal, QC, Canada, Paper 397, 1-13, 2018.

[11] Taştan, M., Gokozan, H., *Real-Time Monitoring of Indoor Air Quality with Internet of Things-Based E-Nose*, Applied Science, Volume 9, Issue 16, 2019.

[12] Wang, J., Zhang, X., *Recommended concentration limits of indoor air pollution indicators for requirement of acceptable indoor air quality*, International Journal of Energy and Environment (IJEE), Volume 1, Issue 4, 697-704, 2010.

[13] Zimmer, C. E., Nehls, G. J., *The Impact of Computers Upon Air Pollution Research*, Journal of the Air Pollution Control Association, Volume 18, 1968, Issue 6, 383-386, on-line 2012.

[14] ***, EPA 1999, Air quality index Reporting Final Rule 1999 - Federal Register, Part III, CFR Part 58, 1999.

[15] ***, Programmable SMS on Twilio Platform, https://www.twilio.com/sms