NanoSen-AQM: From Sensors to Users

https://doi.org/10.3991/ijoe.v16i04.11871

Pedro Lucas (✉), Jorge Silva, Filipe Araujo, Catarina Silva, Paulo Gil, Joel P. Arrais
University of Coimbra, Coimbra, Portugal
phlucas@student.dei.uc.pt

Daniel Coutinho, Pedro Salgueiro, Luís Rato, José Saias, Vítor Nogueira
University of Évora, Evora, Portugal

Alberto Cardoso, Bernardete Ribeiro
University of Coimbra, Coimbra, Portugal

Abstract—With the raising of environmental concerns regarding pollution, interest in monitoring air quality is increasing. However, air pollution data is mostly originated from a limited number of government-owned sensors, which can only capture a small fraction of reality. Improving air quality coverage involves reducing the cost of sensors and making data widely available to the public. To this end, the NanoSen-AQM project proposes the usage of low-cost nano-sensors as the basis for an air quality monitoring platform, capable of collecting, aggregating, processing, storing, and displaying air quality data. Being an end-to-end system, the platform allows sensor owners to manage their sensors, as well as define adjusting functions, that can improve data reliability. The public can visualize sensor data in a map, define specific clusters (groups of sensors) as favorites and set alerts in the event of bad air quality in certain sensors. The NanoSen-AQM platform provides easy access to air quality data, with the aim of improving public health.

Keywords—Air quality monitoring, low-cost nano-sensors, cloud platform

1 Introduction

Since the industrial revolution, human activity is responsible for multiple kinds of pollution that have negative consequences to the environment [1]. One of the most harmful kinds of pollution is air pollution (or atmospheric pollution), usually caused by the release of gases or particles into the atmosphere. Contact with air pollutants can be harmful to human health, being the cause for many pulmonary and cardiovascular diseases [2]. Access to air quality data would allow for people to prepare and avoid certain geographical areas with heavy pollution. Such data already exists, mostly provided by sensors from governmental entities. However, current coverage is very limited, as sensors used in stations are very expensive and deployed only to specific locations. Since air quality can change in short distances [3], a higher number of sensors would improve

the overall coverage of air pollution. While low-cost sensors could improve the coverage of air quality monitoring, their data is often unreliable. Performance of such sensors can be particularly degraded by external and internal factors: temperature, relative humidity, cross-sensitivity (i.e., other pollutants that affect the measurements of the one being measured) and sensor drift (i.e., hardware decay). To deal with these issues, we intend to develop adjusting functions for compensating low-cost sensor output data accuracy, according to historic comparisons against reference sensors (also known as an adjustment of a measuring system [4]).

The NanoSen-AQM project [5] proposes the creation of an end-to-end system that uses low-cost nano-sensors, to be developed in the project, integrated into a platform that stores, processes and makes air quality data available to the public. In addition, it also allows sensor owners to manage their sensors and respective adjusting functions. To increase the platform outreach, the project will also fetch air quality data from external air quality platforms, such as OpenAQ [6].

This article provides an overall description of the platform, which includes: the main requirements of the platform, which guided the development; a list of the technologies used by the NanoSen-AQM platform, along with a small description of each one; the architecture of the system, coupled with an explanation of each module; and finally, an overview of the client applications, which allow users to access air quality data and manage sensors.

2 Requirements

The NanoSen-AQM project proposes the development of a platform that offers:

- **Sensor integration and air quality data storage**: Sensors can be integrated in the platform, allowing them to store the collected air quality data in the system. This data can be accessed by users and adjusted by the system
- **Sensor data adjustment**: Sensors can be set up with an adjusting function, which allows incoming air quality data to be processed, handling issues related to low-cost sensors. The adjusting functions can be defined by the sensor owners
- **Air quality data collection from external platforms**: In addition to the data collected by the sensors integrated in the platform, the NanoSen-AQM system also fetches data from other air quality platforms. In this way, the project is not limited to its own sensors, which improves the overall air quality coverage of the platform
- **Sensor and adjusting function management**: Sensor owners can manage their sensors and clusters (group of sensors) in a CRUD (Create, Read, Update, Delete) manner. In addition, they can also manage the adjusting functions for each sensor, including the procedure that performs the adjustment
- **Data access and visualization**: Public users, as well as sensor owners, can access the air quality data stored in the system, allowing them to visualize such data in a geographic map, as well as downloading the data as a CSV file
- **User favorites, alerts and account management**: Users can have an account where they can manage their profile. This allows them to have their favorite sensors, for

easier and faster verification. They can also set up alerts to be notified when the air quality data in a sensor goes beyond a user-defined threshold.

3 Technologies

As the project supports an end-to-end system, it requires multiple technologies to collect, transport, store and visualize air quality data from sensors to users. The following technologies were used in the development of the NanoSen-AQM platform:

- MQTT [7]. This publish-subscribe messaging protocol was chosen for the project for its simplicity and for being lightweight, which are desirable qualities when working with low-cost hardware, such as air quality sensors
- Apache Kafka [8]. Kafka is a distributed data streaming platform, ideal for data processing, transportation and storage. It uses a publish-subscribe pattern, where a producer publishes messages to a topic, which can be subscribed by consumers. By being distributed, Kafka also has fault-tolerance mechanisms, which can increase the availability of the system in case of failures. In addition, Kafka has proven to be scalable, able to handle high workloads [9]. This technology also provides additional components, such as Kafka Connect, which, as the name implies, connects the streaming platform to external systems (e.g., databases, file systems)
- TimescaleDB [10]. TimescaleDB is a relational scalable database, optimized for time series data, such as the measurements produced by air quality sensors. Since it is an extension for PostgreSQL, a popular and mature relational database, it is compatible with multiple tools and frameworks out of the box
- OpenWhisk [11]. OpenWhisk is a serverless platforms, which is used for managing and running code remotely. Users can upload code to the platform, which creates OpenWhisk Actions in response. These can be managed in a CRUD-like fashion and be invoked remotely. Users can also create Triggers and Rules to create a pipeline that fits their needs. For example, an incoming message to a Kafka Topic fires a Trigger, which in turn invokes an Action that gets the message as parameter. The NanoSen-AQM system uses OpenWhisk to adjust sensor measurements. This can be user-customized for each sensor
- Docker [12]. Docker is a platform that encapsulates source code and dependencies into a single unit, called “container”. By packaging code in this manner, the system can be deployed in multiple environments with minimal configuration. This allows the NanoSen-AQM platform to be deployed both on-premises and cloud environments
- Django Framework [13]. It is a Python framework used for building web applications. Additional modules can be included to provide extra features, such as, the support for RESTful APIs. Django also supports the creation of a dynamic administration interface automatically, by reading the internal database
- Ionic [14]. Ionic was the technology chosen for the client applications development. One of the main points for this decision was the importance of making the NanoSen-AQM platform available, not only as a website, but also as an application in mobile

stores. By being a hybrid application technology, Ionic makes simultaneous development of the website and mobile application much easier.

4 Architecture

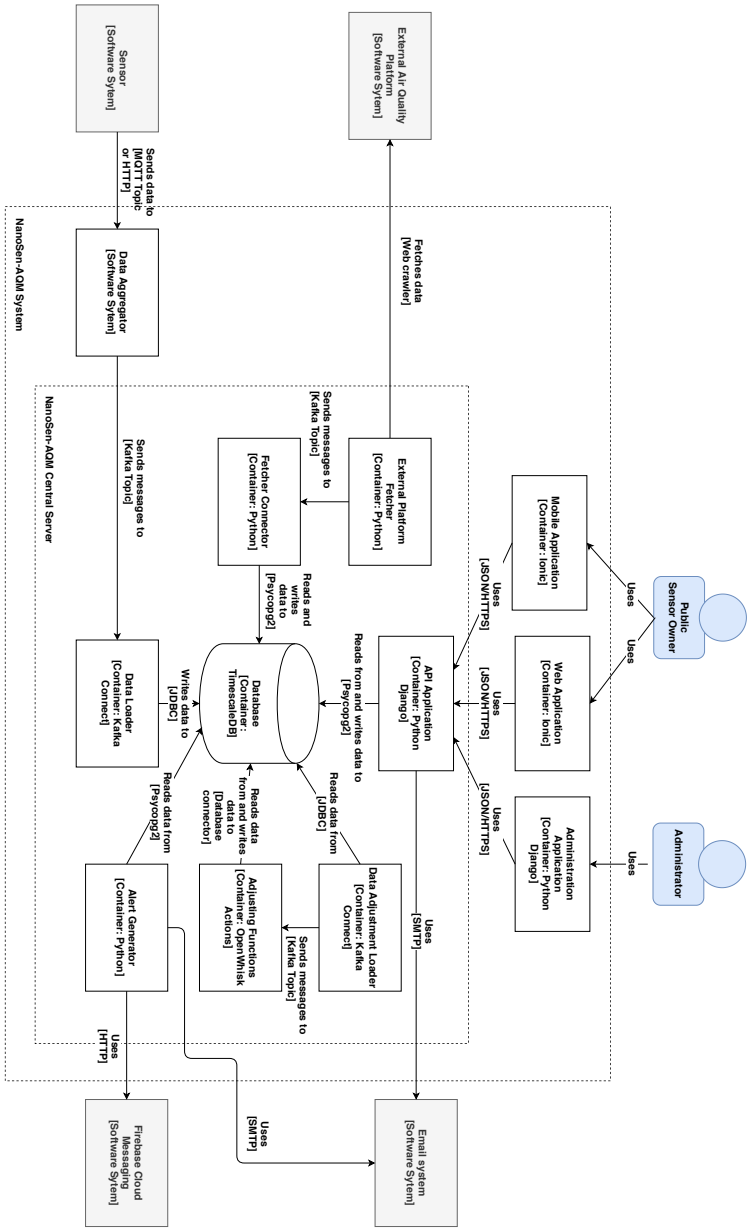

Fig. 1. Container diagram of the NanoSen-AQM platform

As depicted in Figure 1, the NanoSen-AQM system collects air pollution measurements from external nano-sensors through the "Data Aggregation Module". This module receives the data via MQTT or HTTP, validates it and sends it to the central server through a Kafka Topic. As the measurements arrive at the central server, they are received by the "Data Loader", a module that uses Kafka Connect to consume data from the Kafka Topic and inserts it into the database.

Since the data that arrives from the external nano-sensors may be prone to deviations from the real values, an adjustment procedure is performed. Raw measurements are collected by the "Data Adjustment Loader" module, also created using Kafka Connect, before entering another Kafka Topic. This topic is then consumed by the adjustment pipeline, composed by the "Data Adjustment Functions", OpenWhisk actions that process the incoming measurements and store them in the database as adjusted.

Additional air quality data is also fetched from external systems using "External Platform Fetchers", one for each external system. These modules periodically collect air quality data and insert them into a Kafka Topic, which is then consumed by the "Fetcher Connector". This module is responsible for inserting the data into the NanoSen-AQM system, along with necessary information related to the data (e.g., the country and the city of the incoming data).

Users can interact with the NanoSen-AQM platform through the client applications, web and mobile, both created with Ionic from a single codebase. They allow users to visualize the air quality and pollutant concentrations either on a map or from a list of sensor locations. In addition, they also allow sensor owners (users with additional privileges) to manage their sensors, clusters and adjusting functions. Administrators can also manage users, sensors and other aspects of the platform, via the "Administration Application", created using Django.

Finally, users can also receive notifications regarding the air quality in specific clusters. Alerts can be created by the user to get email and push notifications (if using a mobile device) whenever the air quality in a certain cluster crosses a user-defined threshold. The alert can take into account the air quality index of the cluster or be specific to a certain pollutant (e.g., NO2, O3, etc.). This task is performed by the "Alert Generator" module, which scans the database for new measurements and notifies the users via email or by using push notifications when the conditions match those set up by the user.

4.1 Adjustment pipeline

The adjustment is performed in the Adjustment Pipeline, illustrated in Figure 2, which is a combination of system modules, Kafka Topics, and OpenWhisk Actions and Triggers. New raw measurements arrive at a Kafka topic dedicated to raw (non-adjusted) measurements. An OpenWhisk Trigger, configured previously to listen to the Kafka topic, detects when a new message (or group of messages) arrives and responds by calling an OpenWhisk Action, the *SwitchAction*. This action is responsible for receiving new sensor measurements and executing the respective adjusting functions, depending on the sensor's configuration. Following that, it collects the result of each adjustment and inserts it in the database as an adjusted measurement.

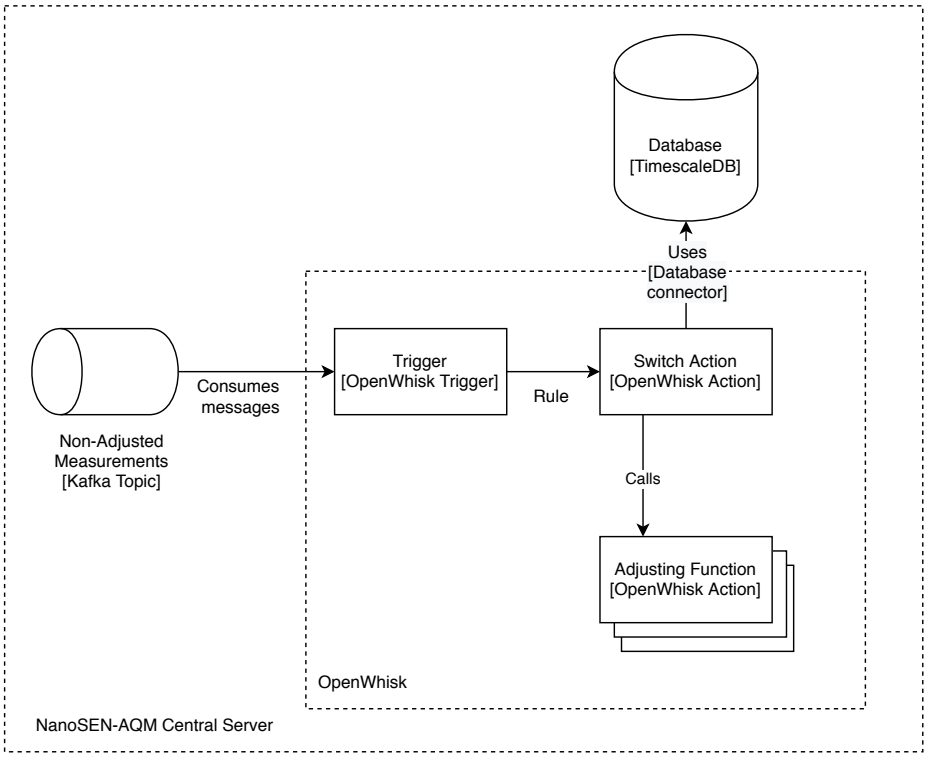

Fig. 2. Adjustment pipeline diagram

An Adjusting Function is a piece of code that transforms an incoming value into an outgoing one. What happens inside the function is up to the programmer to decide. The function must follow specific rules in order for it to work in the NanoSen-AQM system, which can change depending on the programming language. At the moment, only Python3 is supported. The function must consist of a file named *__main__.py* inside a zip file. This Python file must have a function called *main*, which receives and returns a dictionary. The following code exemplifies a simple Adjusting Function along with an example of an input:

```
Example input
{
   "measurement":{
      "time":"2019-09-15T22:00:00.000Z",
      "sensor_id":1,
      "value":45.2
   },
   "latest_measurements":[
      {
         "time":"2019-09-15T21:00:00.000Z",
         "sensor_id":2,
```

```
            "value":0.3
        },
        {
            "time":"2019-09-15T21:00:00.000Z",
            "sensor_id":3,
            "value":54.9
        },
        {
            "time":"2019-09-15T21:00:00.000Z",
            "sensor_id":4,
            "value":7
        }
    ]
}

def main(input_dict):
    desired_value = input_dict["measurement"]["value"]
    desired_value = desired_value + 1
    return {"value": desired_value}
```

The function will receive as input the measurement to be adjusted, along with the latest measurements of the sensors from the same cluster. For example, a measurement from sensor 1 is to be adjusted. It belongs to a cluster with another 4 sensors: sensor 2, sensor 3 and sensor 4. The adjusting function will receive the measurement from sensor 1, alongside the latest measurements from sensor 2, 3 and 4.

Internally, an Adjusting Function is an OpenWhisk Action. For each action that is invoked, a Docker container is created (or reused), the function code is injected and then executed. The default container already includes common Python libraries, but it can be modified to include additional ones. The NanoSen-AQM Python3 Docker container includes additional machine-learning libraries, such as, TensorFlow [15] and Keras [16].

5 Client Applications Overview

To better visualize the features of the NanoSen-AQM platform, we present some screenshots from the web application [17]. The first image, Figure 3, illustrates a map in which all the clusters (group of sensors) of the platform can be found. The map might be zoomed in or out from the street to the world level. Each cluster has a color and a number, which represent the Air Quality Index (AQI) in that location1. In addition, users can choose to view the pollutant concentrations, instead of an AQI value, by pressing the "Sensor" filter and select the appropriate pollutant. Since the platform also fetches data from external platforms, some of the clusters on the map can be external

[1]An air quality index is a value that takes into account multiple pollutants.

to the NanoSen-AQM project. If the user only wants to see the groups of sensors belonging to the NanoSen-AQM project, a checkbox in the lower-left corner of the map can be checked, to hide the external clusters.

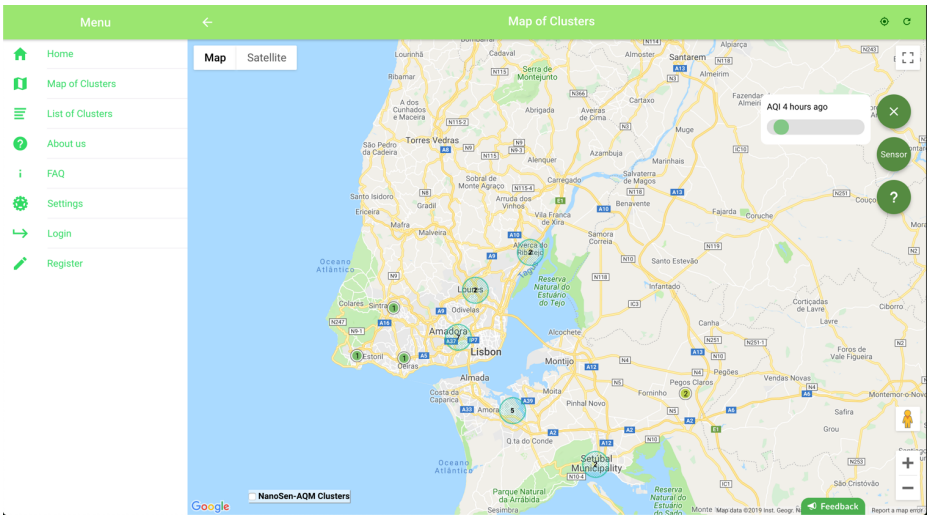

Fig. 3. Map view of the NanoSen-AQM platform

Additional information on a specific cluster can be found, by pressing on it in the map. A page showing statistics and graphs related to the cluster is presented to the user. This includes a pie chart with the percentage of times the air quality has been of a certain value in the past 100 days, as wells as plots that illustrate the evolution of air pollutants over time. Figure 4 shows a plot with the evolution of each pollutant over time, allowing users to select the time span they prefer to see and also allowing them to hide certain pollutants, for easier visualization. Similarly, Figure 5 shows the hourly evolution of the air quality in the form of AQI values, also allowing users to select a specific time period. Finally, users can also download air quality data of the cluster as a CSV file.

A specific menu is available for sensor owners to manage their sensors. Figure 6 illustrates this view, which presents a list of clusters to the user. The sensor owner can choose to manage each cluster as needed (access, edit or delete it), as well as create new clusters. Sensor owners can also create individual sensors and assign them to a cluster. The tab next to clusters and sensors is dedicated to the adjusting functions that can also be managed in a CRUD-like fashion. Users can create new adjusting functions, by uploading code that will be used in the adjustment procedure of air quality measurements. Sensor owners can also make the data of their sensors completely private or public past a user-defined date.

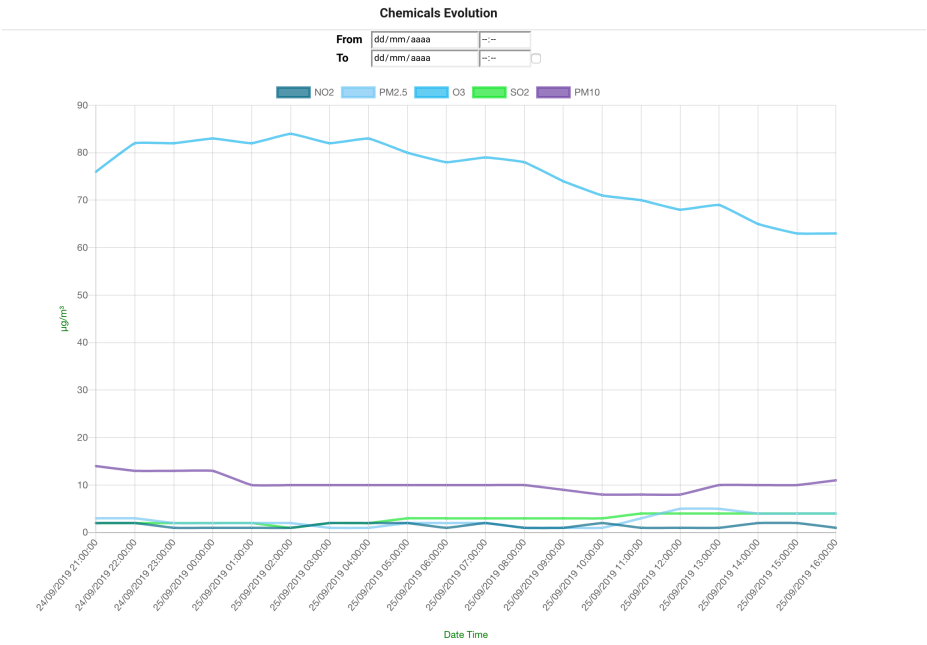

Fig. 4. Pollutants evolution graph

Fig. 5. Hourly AQI evolution graph

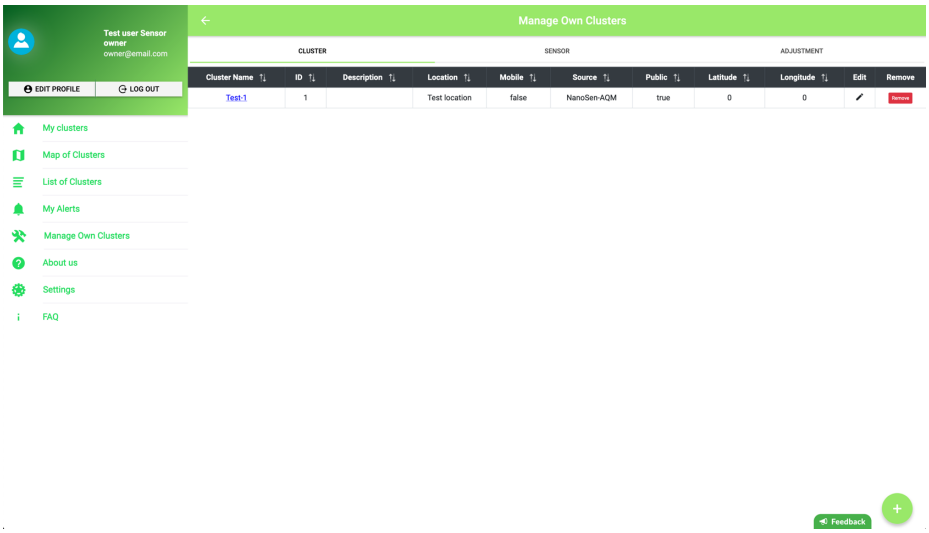

Fig. 6. Sensor management view of the NanoSen-AQM platform

6 Conclusion and Future Work

The current state of the NanoSen-AQM project already allows users to access air quality data via the platform, which can be publicly accessed using a web browser [17]. It can prove useful for users that want to know the air quality at a specific location or for researchers who wish to understand the evolution of an air pollutant over time. Accounts can be created on the platform, allowing users to mark specific clusters as favorites and set up alerts for cases where air quality data exceeds a given threshold. Special users with "sensor owner" privileges, can manage their sensors and clusters and upload adjusting functions. These last are used automatically by the system to transform incoming sensor measurements, providing that the user has set up the sensor to use the desired function.

However, the NanoSen-AQM project is not yet finished. Air quality nano-sensors are still under development, and additional external air quality sources could be added to the platform, to increase its coverage. At the current time, the only external source we use is OpenAQ, which has many clusters of sensors spread around the world. However, these can be very distant from each other. Additional sensors would help users detect areas where pollution is more substantial and plan accordingly. Another feature still under development is air quality forecasting. By using historical values for training, machine learning models could help in predicting future air quality values. Such a feature will allow users to plan ahead, instead of consulting the platform every time to check if the current air quality is healthy for outdoor activities.

7 Acknowledgement

This work acknowledges the Program Interreg-Sudoe of the European Union under grant agreement SOE2/P1/E0569 (NanoSen-AQM) and funding from the FCT, I.P., within CISUC Project UID/CEC/00326/2019 and CTS-UID/EEA/00066/2019.

8 References

[1] Ritchie H., Roser M. 2017. Air Pollution - Our World in Data. [Online]. Available: https://ourworldindata.org/air-pollution#historical-perspective. [Accessed 10 September 2019].

[2] World Health Organization. 2014. 7 million premature deaths annually linked to air pollution. [Online]. Available: https://www.who.int/mediacentre/news/releases/2014/air-pollution/en/. [Accessed 10 September 2019].

[3] Joshua S. Apte, Kyle P. Messier, Shahzad Gani, Michael Brauer, Thomas W. Kirchstetter, Melissa M. Lunden, Julian D. Marshall, Christopher J. Portier, Roel C.H. Vermeulen, and Steven P. Hamburg, "High-Resolution Air Pollution Mapping with Google Street View Cars: Exploiting Big Data" Environmental Science & Technology 2017 51 (12), 6999-7008. https://doi.org/10.1021/acs.est.7b00891

[4] BIPM, IEC, IFCC, ILAC, IUPAC, IUPAP, ISO, OIML (2012) The international vocabulary of metrology—basic and general concepts and associated terms (VIM), 3rd edn. JCGM 200:2012.

[5] NanoSen-AQM. 2019. Development and field validation of a low-cost nano-sensor system for real-time monitoring of air quality. [Online]. Available: https://nanosenaqm.eu. [Accessed 25 September 2019]

[6] OpenAQ. 2019. We empower communities to end air inequality through open data. [Online]. Available: https://openaq.org/. [Accessed 25 September 2019].

[7] MQTT. 2019. MQTT. [Online]. Available: http://mqtt.org. [Accessed 5 August 2019].

[8] Apache Kafka. 2019. Apache Kafka. [Online]. Available: https://kafka.apache.org. [Accessed 5 August 2019]. https://doi.org/10.1007/978-3-319-77525-8_196

[9] Jay Kreps. 2014. Benchmarking Apache Kafka: 2 Million Writes Per Second (On Three Cheap Machines). [Online]. Available: https://engineering.linkedin.com/kafka/benchmarking-apache-kafka-2-million-writes-second-three-cheap-machines [Accessed 5 August 2019].

[10] TimescaleDB. 2019. Time-series data simplified. [Online]. Available: https://www.timescale.com. [Accessed 05 August 2019].

[11] Apache. 2019. Apache Open Whisk is a serverless, open source cloud platform. [Online]. Available: https://openwhisk.apache.org. [Accessed 5 August 2019]. https://doi.org/10.1145/3284014.3284016

[12] Docker. 2019. Enterprise Container Platform | Docker. [Online]. Available: https://www.docker.com. [Accessed 5 August 2019].

[13] Django. 2019. The Web framework for perfectionists with deadlines. [Online]. Available: https://www.djangoproject.com. [Accessed 5 August 2019].

[14] Ionic. 2019. Ionic - Cross-Platform Mobile App Development. [Online]. Available: https://ionicframework.com [Accessed 5 August 2019].

[15] TensorFlow. 2019. An end-to-end open source machine learning platform. [Online]. Available: https://www.tensorflow.org. [Accessed 25 September 2019].

[16] Keras. 2019. Keras: The Python Deep Learning library. [Online]. Availble:https://keras.io/. [Accessed 25 September 2019]. https://doi.org/10.1007/978-1-4842-4240-7_2

[17] NanoSen-AQM. 2019. NanoSen-AQM Platform. [Online]. Available: https://nanosenaqm.dei.uc.pt. [Accessed 25 September 2019]

9 Authors

Pedro Lucas is a Researcher at CISUC, Dep. of Informatics Engineering of the University of Coimbra, DEI, Polo II – UC, 3030-290 Coimbra, Portugal (email: phlucas@student.dei.uc.pt).

Jorge Silva is a Researcher at CISUC, Dep. of Informatics Engineering of the University of Coimbra, DEI, Polo II – UC, 3030-290 Coimbra, Portugal (email: jadsilva@student.dei.uc.pt).

Filipe Araujo is an Assistant Professor at the Dep. of Informatics Engineering of the University of Coimbra, DEI, Polo II – UC, 3030-290 Coimbra, Portugal (email: filipius@uc.pt).

Catarina Silva is an Assistant Professor at the Dep. of Informatics Engineering of the University of Coimbra, DEI, Polo II – UC, 3030-290 Coimbra, Portugal (email: catarina@dei.uc.pt).

Paulo Gil, Electrical and Computer Engineering Department/CTS-UNINOVA, Faculty of Science and Technology, University NOVA of Lisboa, 2829-516 Caparica, Portugal (email: psg@fct.unl.pt) and CISUC, Dep. of Informatics Engineering, University of Coimbra, 3030-290 Coimbra, Portugal (email: pgil@dei.uc.pt).

Joel P. Arrais is an Assistant Professor at the Dep. of Informatics Engineering of the University of Coimbra, DEI, Polo II – UC, 3030-290 Coimbra, Portugal (email: jpa@dei.uc.pt).

Daniel Coutinho is a Researcher at NOVA LINCS, Computer Science Department of the University of Évora, Colégio Luís António Verney, 7000-671 Évora, Portugal (email: m43354@alunos.uevora.pt).

Pedro Salgueiro is an Assistant Professor at the Computer Science Department of the University of Évora, Colégio Luís António Verney, 7000-671 Évora, Portugal (email: pds@uevora.pt).

Luís Rato is an Assistant Professor at the Computer Science Department of the University of Évora, Colégio Luís António Verney, 7000-671 Évora, Portugal (email: lmr@uevora.pt).

José Saias is an Assistant Professor at the Computer Science Department of the University of Évora, Colégio Luís António Verney, 7000-671 Évora, Portugal (email: jsaias@uevora.pt).

Vítor Nogueira is an Assistant Professor at the Computer Science Department of the University of Évora, Colégio Luís António Verney, 7000-671 Évora, Portugal (email: vbn@uevora.pt).

Alberto Cardoso is Assistant Professor at the Dep. of Informatics Engineering of the University of Coimbra, DEI, Polo II – UC, 3030-290 Coimbra, Portugal (email:alberto@dei.uc.pt).

Bernardete Ribeiro is Full Professor at the Dep. of Informatics Engineering of the University of Coimbra, DEI, Polo II – UC, 3030-290 Coimbra, Portugal and the Director of CISUC (email: bribeiro@dei.uc.pt).

Article submitted 2019-10-15. Resubmitted 2020-01-17. Final acceptance 2020-01-11. Final version published as submitted by the authors.