Escuela Técnica Superior de Ingeniería (ETSE-UV)

Doctoral Programme in Information Technologies, Communications and Computing

Universitat de València

Doctoral Thesis

Edge-Fog-Cloud Computing system analysis and implementations for IoT and Smart Cities

Author:
Rafael Fayos Jordán

Supervisors:

Santiago Felici Castell
Jaume Segura García

February, 2024

To my brother,
who has taught me so much,
who has given me so much,
to whom I owe so much.

Abstract

We live in an age where the internet is present in absolutely every aspect of daily life. This is where the potential of connected devices, i.e., the Internet of Things (IoT) lies, proving able to improve people's lives by facilitating everyday tasks and ensuring their health and safety.

This has opened the door to Smart Cities where interconnected IoT devices are employed using all types of communication networks and different computing paradigms, encompassing Cloud Computing, Fog Computing or Edge Computing. This helps provide full connectivity with the shortest possible response time and maximum efficiency to ensure the well-being of citizens.

This doctoral thesis presents several innovative solutions in the field of IoT and Smart Cities across various computational paradigms that contribute to the health, welfare and security of citizens in diverse areas and scopes, while also enhancing connectivity even in remote environments. All of this is approached from a perspective of sustainability, low cost and energy savings.

In order to achieve this goal, a number of advanced devices have been designed, built and tested. These range from airborne pollutant and allergen monitoring systems to indoor ventilation control devices in the context of the COVID-19 pandemic. In addition, devices for calculating noise annoyance, video surveillance systems for authorities, and platforms for processing and calculation in Fog Computing environments have been developed.

The whole process is carried out using a variety of communication technologies and their combination through an innovative multihomming system. The economic impact of implementing these solutions is minimised as far as possible by using low-cost devices or even recycling obsolete ones. In addition, special attention is paid to the energy impact, constantly seeking to reduce consumption and prioritising the use of batteries, thus facilitating the use of alternative and renewable power sources wherever feasible.

In this thesis, the importance, capabilities and usefulness of IoT devices, as well as their application in the context of Smart Cities, are highlighted.

Resumen

Vivimos en una era en la que internet está presente en absolutamente todos los aspectos de la vida cotidiana y es ahí donde radica el potencial de los dispositivos conectados, el internet de las cosas o IoT demostrando ser capaz de mejorar la vida de las personas facilitando tareas cotidianas y velando por su salud y seguridad.

Esto ha abierto las puertas a las ciudades inteligentes o Smart Cities en las que se emplean dispositivos IoT interconectados empleando todo tipo de redes de comunicaciones y distintos paradigmas de computación que engloban la computación en la nube (Cloud Computing), en la niebla (Fog Computing) o en el borde (Edge Computing) para ofrecer una conectividad total con el menor tiempo de respuesta posible y la máxima eficiencia para procurar el bienestar de sus ciudadanos.

En la presente tesis doctoral se presentan diversas soluciones innovadoras en el campo de IoT y Smart Cities en distintos paradigmas computacionales que contribuyen a la salud, el bienestar y la seguridad de la ciudadanía en diversos ámbitos y alcances así como mejoras en las conexiones de estos propiciando una alta disponibilidad incluso en entornos remotos. Todo ello bajo una perspectiva de sostenibilidad, bajo coste y ahorro energético.

Con el propósito de alcanzar este objetivo, se han concebido, construido y sometido a pruebas diversos dispositivos avanzados. Estos abarcan desde sistemas de monitorización de contaminantes y alérgenos en el aire, hasta dispositivos de control de ventilación en entornos interiores en el contexto de la pandemia de COVID-19. Además, se han desarrollado dispositivos para el cálculo de la molestia auditiva, sistemas de videovigilancia destinados a las autoridades, y plataformas para el procesamiento y cálculo en entornos de Fog Computing.

Todo este proceso se lleva a cabo utilizando una variedad de tecnologías de comunicación y su combinación mediante un innovador sistema de multihomming. Se busca minimizar en la medida de lo posible el impacto económico de la implementación de estas soluciones mediante el uso de dispositivos de bajo costo o incluso reciclando dispositivos obsoletos. Además, se presta especial atención al impacto energético, procurando reducir constantemente el consumo y priorizando el uso de baterías, facilitando así la utilización de fuentes de corriente alternativas y renovables siempre que ha sido factible.

En esta tesis, se subraya la importancia, capacidades y utilidad de los dispositivos IoT, así como su aplicación en el contexto de las Smart Cities.

Resum

Vivim en una era en la qual internet és present en absolutament tots els aspectes de la vida quotidiana i és ací on radica el potencial dels dispositius connectats, la internet de les coses o IoT demostrant ser capaç de millorar la vida de les persones facilitant tasques quotidianes i vetlant per la seua salut i seguretat.

Això ha obert les portes a les ciutats intel·ligents o Smart Cities en les quals s'empren dispositius IoT interconnectats emprant tot tipus de xarxes de comunicacions i diferents paradigmes de computació que engloben la computació en el núvol (Cloud Computing), en la boira (Fog Computing) o en la vora (Edge Computing) per a oferir una connectivitat total amb el menor temps de resposta possible i la màxima eficiència per a procurar el benestar dels seus ciutadans.

En la present tesi doctoral es presenten diverses solucions innovadores en el camp de IoT i Smart Cities en diferents paradigmes computacionals que contribuïxen a la salut, el benestar i la seguretat de la ciutadania en diversos àmbits i abastos així com millores en les connexions d'estos propiciant una alta disponibilitat fins i tot en entorns remots. Tot això sota una perspectiva de sostenibilitat, baix cost i estalvi energètic.

Amb el propòsit d'aconseguir este objectiu, s'han concebut, construït i sotmés a proves diversos dispositius avançats. Estos comprenen des de sistemes de monitoratge de contaminants i al·lergògens en l'aire, fins a dispositius de control de ventilació en entorns interiors en el context de la pandèmia de COVID-19. A més, s'han desenvolupat dispositius per al càlcul de la molèstia auditiva, sistemes de videovigilància destinats a les autoritats, i plataformes per al processament i càlcul en entorns de Fog Computing.

Tot este procés es duu a terme utilitzant una varietat de tecnologies de comunicació i la seua combinació mitjançant un innovador sistema de multihomming. Es busca minimitzar en la mesura que siga possible l'impacte econòmic de la implementació d'estes solucions mitjançant l'ús de dispositius de baix cost o fins i tot reciclant dispositius obsolets. A més, es presta especial atenció a l'impacte energètic, procurant reduir constantment el consum i prioritzant l'ús de bateries, facilitant així la utilització de fonts de corrent alternatives i renovables sempre que ha sigut factible.

En esta tesi, se subratlla la importància, capacitats i utilitat dels dispositius IoT, així com la seua aplicació en el context de les Smart Cities

Acknowledgements

During the years it has taken me to complete this thesis, there have been many people who, in one way or another, have contributed to my work, and have helped, supported, guided and motivated me, without whom it would not have been possible to reach this point. For this reason, I would like to thank them all:

Starting with my supervisors Santi and Jaume, whose guidance, support and work has been decisive. They were the first to believe in me and give me a chance in this wonderful world of research, and they were responsible for making me decide to take my studies to another level. Also, in the same section, I thank Jesús and Adolfo who were my first battle companions in that crowded and cosy laboratory of the ETSE, and of course, all the professors of the department with whom I have shared work, anxieties, laughs and coffees: Antonio, Juanjo, Máximo, Miguel G., Miguel A., Juan, Raúl, Alicia, and so many others who have helped me.

To my parents, who have suffered so much with me but have never thrown in the towel and whose sacrifice during these long years has brought me to this point. To my brother José Miguel, and my sister-in-law, Raquel, for always believing in me and supporting me unconditionally. Of course, my eternal love and gratitude to my 'bichos', my 'trasto' and my 'brujita', my nephews Iker and Laia, who condition every decision in my life, fill me with joy and happiness and give me the strength to go on. And to all my grandparents, those who are still here, those who are no longer here and the one I never got to meet.

To my lifelong friends, those who are always there, those who never fail: Jose, Visant, José María and Ximo for putting up with this friki and making this trip much more bearable. Also, to that select group that makes up the 'cafetico' every morning at the 'mesón' before going to work.

I don't want to forget my battle companions during my university degree, the great team of Utini Coms: Víctor, Guille, and the two Ismas, and, of course, my comrades in feats and outrages, Dani, Toni, Adri and Pedro, always conspiring and lurking.

Finally, to the great team that is the Beyond 5G Hub, who have welcomed me as one of them from day one and with whom I share work and fun.

Thank you all very much.

Contents

Chapter 1

Introduction

1.1 Introduction

As the number of connected IoT (Internet of Things) devices has dramatically increased to more than 15 billion[1] in recent years, and is expected to grow even more significantly in the coming years, doubling by 2030 (see Figure 1.1), one can infer the importance of and the wide possibilities that this technology encompasses. This not only opens up new opportunities in the context of Smart Cities, as mentioned above, but also poses new challenges, including the processing of the monitored data.

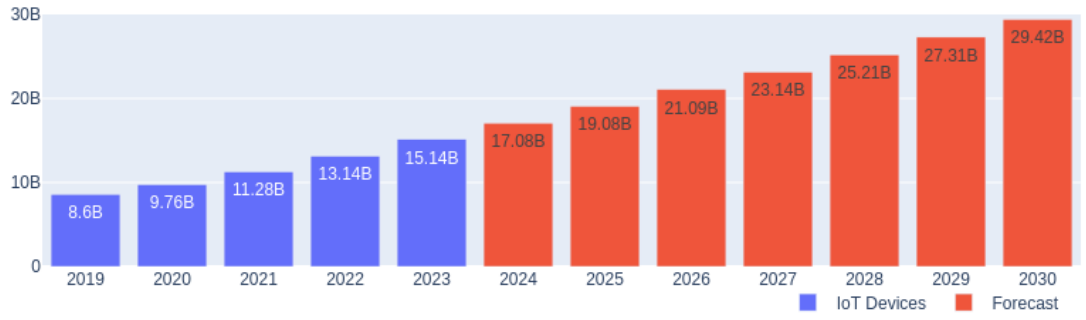


Figure 1.1: Connected IoT Devices and Forecast in Billions [1]

The proliferation of these new devices will result in an enormous amount of data traffic directed towards centralised servers. Consequently, it will be imperative for these servers to adapt or employ new paradigms, such as Fog Computing, which have the ability to mitigate this data flow and processing while simultaneously reducing response time and improving user experience.

The concepts of Edge, Fog and Cloud Computing have undergone various interpretations over the years from their initial coinage to the current era. [2]–[13]. With

[1]These amounts are expressed in billions respecting the original short numerical scale of the source by referring to billion as 10^9

the constant progress of computing and data transmission technologies, as well as the emergence of new devices, there is a pressing need to adapt certain concepts by introducing more complexity. This can be achieved by incorporating additional layers and restricting aspects related to location, distance, physical characteristics of the devices and their specific utility.

Until a few years ago, a trend wherein all data generated by devices was sent to a central server prevailed; this server assumed responsibility for processing, analysing and making appropriate decisions according to the circumstances. This approach is known as Cloud Computing, Figure 1.2 a), characterised by the centralisation of processing on servers hosted in datacenters or remote locations. However, this approach has proven ineffective as the number of devices and the amount of data collected has increased significantly.

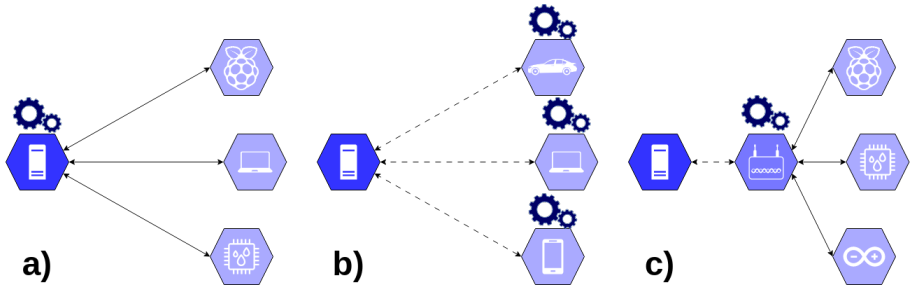


Figure 1.2: a) Cloud Computing, b) Edge Computing and c) Fog Computing

As an example, consider an autonomous vehicle equipped with numerous sensors capable of generating up to 40 Terabytes of data per day [14]. It would be unwise and illogical to transmit this volume of data to a central server for real-time analysis and decision making, followed by feedback to the vehicle with actions to be taken to, for example, avoid an accident.

Centralised servers, used for processing substantial amounts of data, require large-scale infrastructures that go beyond the mere acquisition of hardware [15]–[17]. It is clear that the ability to handle a huge amount of data in a short period of time demands extremely high-performing equipment with a large memory capacity, as well as the latest generation data connections. In addition, such equipment exhibits high power consumption and demands extreme availability, leading to the implementation of redundant power supply systems at considerable costs. As a direct effect of the operation of these systems, a significant amount of heat is generated which needs to be controlled by various cooling systems. This, in itself, is a substantial cost from both an energy and economic perspective.

These examples are just a small sample of the reasons behind the increasing decentralisation of certain types of services. Factors such as data volume, the imperative need for immediate response, energy efficiency considerations, economic impact, etc. converge to underline the need to bring data processing closer to the end devices, thus relieving the load on servers, optimising hardware investment and reducing energy consumption.

On the other hand, there is the Edge Computing modality, Figure 1.2 b), which is characterised by performing processing operations directly on the end devices, that is, the devices located at the edge of the network which do not function as gateways or access points for other devices. This set of devices boasts a wide variety, including IoT devices, smartphones, personal computers, autonomous vehicles and robots. In this modality, multiple challenges related to data processing arise. In some cases, computational capacity is insufficient to carry out the required processing, while in others, the data obtained either needs to be combined with data collected by other devices or be centralised for proper processing.

Finally, there is an intermediate solution between Cloud and Edge Computing, known as Fog Computing, Figure 1.2 c). This approach involves the use of devices capable of performing certain processing tasks strategically placed between end nodes and central servers. In general, these devices are equipped with higher-capacity hardware compared to those used in Edge Computing. However, they have lower capacities than those used in Cloud Computing. This feature allows them to execute more complex tasks compared to most Edge devices, while simultaneously functioning as intermediaries between nodes, gathering information from various devices and transmitting only the relevant data to the cloud computing environment, Figure 1.3.

These Fog Computing devices are generally located physically close to the end nodes, which results in shorter response times compared to central servers and makes them more appropriate in cases where speed of response is a crucial factor. Unless exceptional circumstances arise, the economic and energy costs associated with these devices are significantly lower than those of large-scale centralised servers, although they naturally exceed those of end nodes. In short, Fog Computing stands out as a highly versatile solution that can bring many benefits to the ecosystem of connected devices

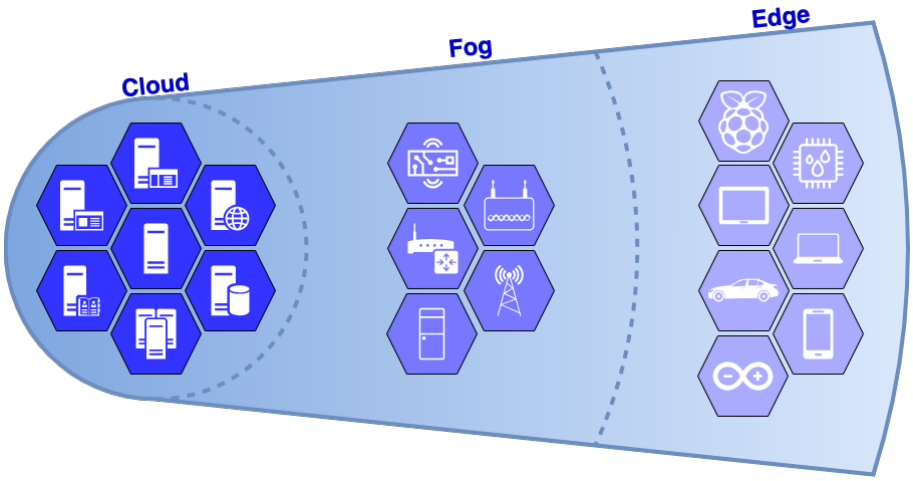


Figure 1.3: Cloud-Fog-Edge Diagram

None of the above-mentioned modalities can be categorised as superior or inferior

to the others. Rather, the choice between them depends on the particular circumstances of each case. An illustrative example is the field of autonomous vehicles as discussed above, which demands immediate responses and a high processing capacity. In this situation, the preferred option is to delegate the entire computational capacity to Edge Computing, that is, to the vehicle itself.

On the other hand, in cases such as sensor nodes for monitoring specific parameters (such as atmospheric data, telemetry or acoustic measurements), where a significant number of sensors are required to be deployed in a given region, the most appropriate solution is Fog Computing. These devices can collect data from all sensors, process it and make decisions in short time intervals, which relieves the load on the servers and reduces the costs associated with the sensors.

Finally, in contexts involving processes such as climate modelling, online services, data storage and other similar applications, Cloud Computing is often the most suitable choice due to its high processing capacity and constant availability.

Of course, another strategy is to combine these three modalities by segmenting the data processing into distinct phases and distributing each of them to the most suitable location, Figure 1.4.

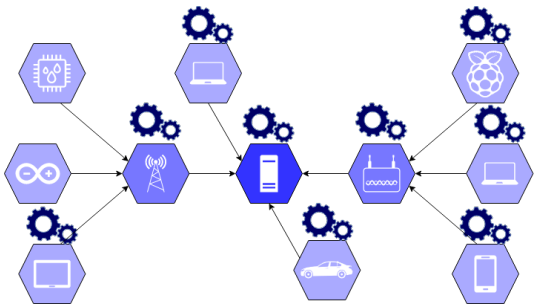


Figure 1.4: Mixed computing Diagram

As we can see, these concepts have undergone several interpretations and adaptations over time in accordance with the individual needs of each author or, in some cases, to broaden their scope. In this respect, some authors choose to conceive Edge Computing as any device beyond a telephone antenna or an Internet Service Provider (ISP) [18]. Some even segment this concept into categories such as Far Edge and Near Edge [19]. On the other hand, there are also those who define Fog Computing as devices located in 4G or 5G telephony signal towers [20]. Thus, multiple definitions can be identified depending on authors' perspective. In the context of this thesis, we have chosen to adopt the conventional interpretation presented above.

1.2 Objectives

This doctoral thesis aims to research and develop new tools and mechanisms applicable to the IoT, with a particular focus on Smart Cities. It seeks to achieve this through the implementation of Cloud, Fog and Edge Computing paradigms, as well as the incorporation of the most advanced data transmission technologies available. To this end, a number of specific objectives have been established:

- To develop systems and tools that contribute to the health and well-being of the population living in cities, with a specific focus on both atmospheric and noise pollutants. These factors can cause several ailments, such as respiratory and cardiovascular diseases and stress. The aim is to identify and monitor the presence and concentration of these pollutants in order to mitigate their negative effects on health.
- To contribute to the security of the population by developing video surveillance tools that enable authorities to strengthen security in exceptional situations.
- To analyse and improve current communication systems for IoT by using all available networks with the aim of maximising their availability.
- To propose and design Fog Computing-oriented systems, focusing on enhancing data processing, reducing response time and improving user experience.
- To achieve the above by promoting its implementation and reducing costs as far as possible in terms of hardware, energy consumption and data transmission costs, all from a sustainable perspective.

In addition, during the first half of 2020 and due to the COVID-19 pandemic, the decision was made to incorporate the development of IoT tools aimed at preventing infection in closed environments as one of the objectives. This approach specifically focused on the conduct of examinations in university classrooms.

1.3 Structure of the Doctoral Thesis

Following the regulations of the Doctoral School of the University of Valencia and the PhD programme in Information Technologies, Communication and Computing, as well as its established requirements, this doctoral thesis is presented in the form of a compendium of articles. The chapters it comprises are detailed below:

- Chapter 1 contains the introduction, which covers the description of the general subject matter of the thesis, the context of the problem to be addressed, the established objectives, the guidelines followed to approach its resolution, and the general structure of the thesis.
- Chapter 2 details how the various objectives have been addressed, describes the problems encountered, and explains how they have been handled.
- Chapter 3 summarises their characteristics, the specific problem addressed, and the approach followed in each case for their resolution. In total, eight

contributions are presented in five articles as first author. Additionally, a sixth article is presented as co-author and two contributions have been made to conferences.

- Chapter 4 presents the conclusions of this doctoral thesis as well as future work. Most of these lines of action are already being addressed at the time of writing.
- Finally, the annexes include the complete versions of the eight contributions made during the execution of this thesis, as well as the research projects in which the author has participated.

Chapter 2

IoT and Smart Cities

This PhD thesis focuses on Internet of Things (IoT) devices aimed at serving a specific purpose in the context of Smart Cities. The primary objective of these devices focuses on promoting the well-being and safety of citizens in their daily lives. One of the most pressing challenges faced by urban dwellers is related to pollution, both in terms of air quality and noise pollution. These are factors contributing to serious health problems, including lung conditions [21], [22]. These diseases affect hundreds of thousands of people anually and, in some cases, can have fatal consequences. [23], [24]. In addition, stress and mental health problems are associated with ill health [25], which can prompt many individuals to take extreme measures, such as taking time off work or even leaving the cities.

2.1 Edge Computing Approach

2.1.1 Soundscape Solutions

It is clear that improving the quality of life of these persons depends, to a large extent, on the ability to accurately monitor the external environmental factors that contribute to the deterioration of citizens' health. Therefore, one of the first research approaches was to design devices capable of recording and calculating the annoyance levels of sounds [26], [27] present in the everyday environment. Examples within an Edge Computing environment include traffic, public works, annoying noises, etc.

These devices were initially built using Raspberry Pi boards [28], and were successfully installed and tested in the cafeteria of the School of Engineering of the University of Valencia, allowing the calculation of annoyance levels [29].

However, it became apparent that the computations required for this purpose imposed a considerable load on the Raspberry Pi processor, and although these boards are versatile, their cost and power consumption did not make them the most suitable choice for implementation as IoT devices. Therefore, it was decided to explore alternatives based on more compact and less power-consuming devices, such as the

ESP [30] and the FiPy[1] [31], which can be installed in a wide variety of locations and run on solar power and batteries. These devices, while lacking the computing power of Raspberry Pi, have advantages in terms of energy efficiency, size and communication capabilities, thus making them fit for our purpose. These new devices, based on the ESP8266, are capable of recording audio at a price up to 90% cheaper than a Raspberry Pi, along with considerably lower power consumption and smaller physical dimensions; however, they could not carry out the processing required for annoyance calculations. This problem will be addressed later.

2.1.2 CO_2 Monitoring Solutions in Times of COVID-19

In this scenario, a new contribution involved monitoring air quality and identifying harmful agents that could have an impact on people's health in the short- and long-term. The beginning of this research coincided with the outbreak of the COVID-19 pandemic, which led to the rescheduling and cancellation of certain tests that were originally planned to be carried out in indoor and outdoor environments. However, this unexpected context provided an opportunity to contribute to research in the field of airborne diseases, such as the coronavirus itself, and their indoor influence.

As was widely known during the pandemic, especially during the period of confinement and in the subsequent transition to normality, infections occurred predominantly in poorly ventilated indoor spaces [32]. Governments around the world based their decontamination strategies on this principle [33], allowing outdoor activities, such as the opening of terraces in hospitality establishments, and restricting or banning indoor gatherings. This decision was based on the importance of ventilation in such areas, as insufficient or no ventilation significantly increased the risk of spreading the virus.

The dimensions of the SARS-CoV-2 virus are too small to be detected by conventional sensors and, even if this were not the case, it would be difficult to differentiate it from other particles in the environment once detected. Consequently, it became clear to many that indoor air quality, particularly the effectiveness of ventilation, could be measured by the concentration of carbon dioxide exhaled by people [34], [35]. An increase in carbon dioxide concentration suggested inadequate ventilation and, therefore, an increased risk of disease transmission.

This idea was closely aligned with the objectives of this thesis related to health and air quality monitoring. It was presented as a scalable solution for recording other parameters relevant to health in a personalised manner in accordance with the profile of each user.

For this purpose, a carbon dioxide (CO_2) sensor was designed to measure the concentration of the gas in real time and transmit this data to a server via a WiFi connection. The server, using the kriging technique, generated concentration maps within a delimited space and issued alerts if the concentration exceeded certain

[1]Pycom Ltd. went into bankruptcy in September 2022, but was acquired by Season Group which in turn created Pycom BV which took over the company to avoid end of life of the products.

predefined thresholds. It also provided continuous feedback to users via an RGB LED indicator light and an OLED display showing the concentration in real time. The device could be configured and calibrated via a mobile app using Bluetooth technology.

A total of nine sensors of this type were built and tested during the period of the final exams at the School of Engineering (ETSE), achieving highly satisfactory results. These sensors provided the ability to constantly and effectively monitor the quality of ventilation in the examination spaces, ensuring adequate conditions to safeguard the health of the people present.

The circuit designs, the printed circuit boards (PCBs) and the programming code for the ESP devices used, as well as the configuration application designed for Android devices, were publicly released in an open source format. This provided the opportunity for the community to use these resources in their own projects and designs. [36] (contribution 4.2.1, annex B).

2.1.3 Air Quality Monitoring Solutions

To continue in the field of health, this time related to air quality [21], [23], [24], and within the Eco4Rupa project (annex G), several sensor models designed to measure different parameters related to air quality were used. These sensors covered measurements of particles, formaldehyde levels, ozone concentration, presence of carbon monoxide, detection of volatile organic compounds, measurement of nitrogen dioxide, as well as monitoring of ambient temperature and humidity.

These sensor nodes, with the ability to transmit data via WiFi, Bluetooth or NB-IoT technologies, were integrated using the MQTT protocol to connect to a Broker server which stored and managed the data collected by all the nodes. This time, the main objective was to use the information generated by these sensors, together with the air quality data provided by the Valencia city council in a publicly available format, and combine them to analyse air quality in different areas. This assessment was intended to determine optimal routes from a health perspective for people wishing to walk from one point to another, allowing them to avoid areas with a higher concentration of health-damaging air pollutants.

This model underwent successful tests in an area of Burjassot, in which optimal routes specifically designed for people with conditions such as asthma [37] or pregnant women[38]were calculated, thus contributing to the promotion of health and safety in urban mobility [39] (contribution 4.2.2, annex C).

2.1.4 Video Surveillance Solutions

Having addressed the issues of noise annoyance and respiratory health in the previous examples, a concern arose about public safety and how IoT devices could contribute to its improvement. The answer lay in non-static video surveillance, as there are already cameras connected to the network that monitor specific areas and

traffic. Rather, a flexible, quickly deployable, highly reliable and easily implementable solution was sought for high-risk situations and occasional conflicts – such as those that may arise during festivities or protests as they could lead to vandalism – or for monitoring traffic in areas affected by accidents or ongoing works, where the risk of collisions is increased.

This solution consisted of a network of interconnected nodes, each of which was equipped with built-in surveillance cameras and designed for easy and rapid deployment. These nodes had the ability to stream real-time video using the Dynamic Adaptive Streaming over HTTP (DASH) protocol [40].

Given the computational demands required to encode and transmit the video fragments in different qualities, the choice of Raspberry Pi devices for carrying out this task was again justified. These devices proved to be appropriate for this application, facilitating the creation of a meshed network of mobile video surveillance nodes capable of addressing high-risk situations effectively and efficiently.

In our project, we developed nodes equipped with embedded cameras with the ability to encode video into multiple streams of different qualities. This allowed a customer to view the video in real time even when the quality of the network connection was poor. These nodes were interconnected via a layer 2 mesh network making use of the B.A.T.M.A.N (Better Approach To Mobile Adhoc Networking) protocol [41]. This infrastructure allowed large areas to be covered and made it possible to connect to the network from various points or as a stand-alone system [42].

A practical application example would be a scenario where a police officer can potentially connect to this network and access the display of any node at any time and from any location within the network. These nodes were designed to be versatile and quick to deploy, as they were intended to have a magnet mounting mechanism that allowed them to be temporarily attached to traffic signs or other metal street furniture. The implementation of these nodes was carried out without the need for manual configuration, as the mesh protocol automatically detected and established connections with neighbouring nodes, or for calculating the optimal routes for video transmission.

2.1.5 Analysis and Contribution in Multihomming

In all the previously mentioned scenarios, we were faced with a recurring challenge when choosing the best way to transmit information from the nodes to the server. Of course, when a WiFi network was available, as it was in several cases, this option was used due to its high speed and effective cost. However, the need to implement mobile or stationary nodes in remote areas that would not require additional configurations to send data to a server was always a concern. A node could be programmed to use a specific communication interface, or in case of a connection failure, another interface. However, the question arose: What if the node could automatically select the most convenient interface based on several criteria, such as the amount of data to send, the availability of networks, the cost associated with using those networks, the power consumption involved and the remaining battery level?

Currently, communication nodes have a variety of connectivity options; however, all of them are rarely used optimally. Based on this premise, we chose to use the FiPy device, known for its versatility in terms of communication options, to develop a system that would evaluate all these factors before initiating a data transmission. We implemented a weighting system that takes into account the actual transmission speeds through its interfaces and the power consumption associated with each of them.

Because of this, we were able to develop a fully customisable decision-making algorithm for each case. This algorithm automatically selects the most efficient way to transmit a given volume of data by taking into account all of the above-mentioned factors, user preferences and, in some cases, legal restrictions based on time of use. An example of this is LoRa technology in certain countries. This approach allowed for more effective and efficient communication in a wide variety of situations and contexts.

We carried out an exhaustive study of the capabilities of FiPy nodes by carefully analysing each of their interface aspects, including throughput, effective transmission distance and energy consumption per byte sent, among other parameters. Thanks to this study, we were able to successfully configure the interface selection algorithm so that it could automatically choose between WiFi, Bluetooth, LoRa and LTE-M according to the specific circumstances of each case [43] (contribution 4.1.1, annex A).

It is important to note that the algorithm was also originally intended to consider the signal level (RSSI) of each interface as a determining factor in calculating the potential transmission rate. However, during our study, we realised the hardware and software limitations of the FiPy nodes, which resulted in the loss of the connection before the transmission rate decreased due to the poor quality of signals. For this reason, we chose not to take RSSI into account as a parameter in the algorithm.

However, given the possibility of incorporating RSSI as an additional parameter, we are currently working on the design of a new node model with significantly higher capabilities than FiPy. In this new context, we will test this interface selection algorithm again.

This significant progress allowed us to successfully close the chapter related to IoT and Edge Computing nodes, providing a robust and adaptable solution for diverse monitoring and communication situations.

2.2 Fog Computing Approach

Due to the computational cost required, this section focuses exclusively on soundscape solutions. It is not necessary to extend this approach to the remaining solutions due to their reduced computational requirement and the need to limit the processing to Edge Computing.

As previously mentioned, the definitions of Edge and Fog Computing may vary according to different authors. In the context of this PhD thesis, we have chosen to define a dedicated Fog Computing device as one positioned between end devices, be it sensors, smartphones or computers, and cloud devices, such as servers and data centres. These devices should perform a data processing function as long and as comprehensively as their hardware capacity allows, thus relieving the workload of servers or data centres and reducing response time through their proximity to the Edge. Although they can function as gateways or access points, their primary function is data processing [44].

With this in mind, we set out to address several challenges in this field, specifically related to one of the ongoing research projects (annex F). This project aimed to measure the acoustic annoyance in an enclosure using sensor nodes that take into account multiple parameters and perform complex signal processing to obtain an accurate model of the annoyance. At an initial stage, Raspberry Pi was used to record and process the recorded audio; however, this approach presented several problems.

Despite being a more affordable device compared to other alternatives, deploying a large number of these devices could be costly. In addition, Raspberry Pi devices drained all their resources when processing audio, which often resulted in overheating and, consequently, possible crashes and unexpected reboots. The solution could not involve fans for cooling, as they would interfere with the quality of the audio recordings and produce undesirable results. On the other hand, cheaper devices such as Arduino or ESP8266 were not suitable due to their limited processing power, which made it impossible to perform real-time calculations and generated an incremental accumulation of work that would saturate them.

Moreover, from a legal perspective, it was not feasible to send the audio to a central server with greater processing capacity, as this could involve the recording of third party conversations. Due to privacy concerns, transmitting such data to another location would be questionable from a legal and ethical point of view regardless of intention.

The answer to these challenges was to adopt a Fog Computing approach. By moving the data processing away from the sensor nodes, it was possible to replace them with cheaper devices that simply recorded audio. By locating this processing node far enough away from the sensors, it could be cooled efficiently without affecting the recordings. Finally, since this node was located in the same enclosure as the sensors, there were no legal issues, as the audio would never leave the enclosure. The results related to the annoyance would be sent to the server once they were processed

internally.

Initially, the decision was made to use the same Raspberry Pi for this task due to its affordability and availability. However, a single Raspberry Pi would not be able to simultaneously process all the audio data sent by the sensor nodes, which was a limitation recognised during research. Therefore, the need to increase the computational capacity by implementing a cluster of Raspberry Pi working together was felt.

The cluster had to be fully scalable, which implied the possibility of adding any number of nodes as needed, depending on the number of sensors installed in the enclosure. To achieve this flexibility, a layer 2 (data link layer)-based mesh WiFi network was chosen. Most existing WiFi mesh protocols operated at layer 3 (network layer), which required each node to have a pre-configured IP address and did not allow the use of DHCP. By using the layer 2 protocol, DHCP could be used and did not require prior network configuration for each node to be added to the cluster. This allowed the interconnection of all nodes without requiring the use of physical switches, and facilitated the addition of additional nodes without the need to modify the existing network structure. In addition, one of the nodes was designated as the master, which would be connected directly to the Internet via an Ethernet cable to transmit all calculated results.

In previous studies, the B.A.T.M.A.N. protocol had been successfully tested on Raspberry Pi devices, achieving a data transmission rate of over 40 Mbps [45]. This transmission capacity was more than sufficient to meet the objectives of the project in question.

To optimise the processing capacity, we implemented containers and used an orchestrator to distribute the tasks among all available nodes. Given its popularity and broad support, Docker was chosen as the container system. In addition, the capabilities of Docker Swarm and Kubernetes as orchestrators were evaluated.

Specific containers were created for each of the tasks involved in calculating the annoyance level. The master node was configured to deploy multiple instances of these containers on the available slave nodes. The audio data was sent directly to the master node, which then distributed the tasks among the various containers deployed on the slave nodes. These containers performed the necessary calculations and sent the results directly to the server, where they were stored and used to generate real-time maps of the acoustic annoyance using kriging. [46] (contribution 4.2.3, annex D).

During the different tests conducted under equivalent conditions, it was observed that in resource-constrained environments, such as Raspberry Pi, Docker Swarm outperformed Kubernetes. This was because Docker Swarm required fewer resources to run, allowing it to allocate more processing time to assigned tasks.

After the testing concluded, further optimisation of the cluster was carried out. This was achieved by serially interconnecting the Raspberry Pi via GPIO pins and monitoring resource usage by allowing nodes to be switched on or off based on

resource demand. This resulted in a significant improvement in resource efficiency along with a reduction in overall system consumption.

With these optimisations, a fully scalable and low-cost Raspberry Pi cluster for data processing in the context of Fog Computing was created.

Once the development of this device was completed, the question arose as to what performance could be obtained if other low-cost devices were used instead of Raspberry Pi. The possibility of creating a similar but heterogeneous cluster using various embedded system development boards (SBC) or even an old PC that could be efficiently used as a processing device in Fog Computing was considered. Consequently, a heterogeneous cluster was created, including two Raspberry Pi 3Bs (one as a master node and one as a slave node), an Asus Tinker Board [47], a Udoo [48], a Nvidia Jetson Nano [49] and a PC with a dual-core processor that was no longer in use.

To ensure fair measurements in this heterogeneous environment, a meshed WiFi network was not used as each device would use a different network card, the capabilities of which could influence the speed of receiving and sending packets. Instead, a switch with 100 Mbps interfaces (a speed that all the devices more than supported) was used to interconnect the cluster nodes. However, it is worth mentioning that this configuration could easily be replaced by the mesh network used in the previous cluster in terms of scalability.

During the development of this project, it was discovered that not only was this cluster fully functional and scalable but it also allowed new nodes of virtually any type to be added. This is particularly relevant in Fog Computing, where one of the main goals is to alleviate the processing load in the cloud and reduce response time. With this configuration, it is possible to easily deploy such clusters (both homogeneous and heterogeneous) wherever needed without incurring high investments [50] (contribution 4.2.4, annex E).

Docker Swarm was also found to outperform Kubernetes in terms of resource usage, which affected the overall performance of the cluster. As machines with abundant memory and processing power are used, this increase in resources hardly has a significant impact on overall performance of cloud-hosted servers. For this reason, Kubernetes is often preferred, as it has been around longer and offers advanced features useful in large data centres. However, when hardware is limited, as in the case of SBCs and older PCs, Docker Swarm is more appropriate as it uses available resources more efficiently.

These various cluster implementations open up new possibilities in the field of Fog Computing, allowing their application in a wide variety of scenarios, as well as their integration in virtually any location. We have seen examples of its use in audio processing in a confined space where it could be implemented without incurring legal infringements and reducing the costs of sensor devices. Other possible applications where it could bring great advantages include the deployment of weather or atmospheric monitoring sensors, where multiple sensors can be interconnected using various communication technologies, such as WiFi, LoRa or Bluetooth or any other 'free' transmission technology with a single central fog computing element that could

process and transmit only the relevant data to a central server via 4G, 5G, NB-IoT or LTE-M technologies, thereby bypassing the sensor nodes, reducing hardware and data plan costs and being easily deployable in urban and rural environments.

Another example would be the implementation in road infrastructures, such as motorways, conventional roads and streets with surveillance cameras that transmit images directly to the cluster. By using AI, the system could detect traffic conditions, such as traffic jams, accidents or objects on the road, and communicate this information quickly to drivers through various technologies. It could simultaneously send the data to the traffic control centre for appropriate action. This information could reach drivers much faster than using cloud servers for data processing.

In summary, such Fog Computing devices have the potential to be a breakthrough in Smart Cities. They can monitor a wide variety of atmospheric, acoustic, visual and other parameters, as well as control various actuators, such as lighting, traffic lights and warning systems, thereby improving the quality of life of citizens in terms of health and safety without requiring large investments compared to other centralised systems.

Chapter 3

Scientific Outcomes

As a result of the work carried out during this research, a series of scientific outcomes have been published in different journals and congresses as a result of the multidisciplinary collaboration with different departments and universities. In all of them, the PhD student has contributed his knowledge acquired in the field of IoT and Smart Cities and the paradigms of Clod-Fog-Edge Computing in the development, design, experimentation, analysis, writing and proposal of new, relevant and challenging ideas.

Next it is shown the list of all the studies in which the PhD student has participated, divided into publications in journals and publications at conferences.

3.1 Journal Publications

- S. Felici-Castell, J. Segura-Garcia, J. J. Perez-Solano, R. Fayos-Jordan, A. Soriano-Asensi, and J. M. Alcaraz-Calero, 'AI-IoT Low-Cost Pollution-Monitoring Sensor Network to Assist Citizens with Respiratory Problems', Sensors, vol. 23, no. 23, p. 9585, Dec. 2023
  Impact Factor: 3.9 (Q2)
  DOI: 10.3390/s23239585

- S. Felici-Castell, E. Fernandez-Vargas, J. Segura-Garcia, J. J. Perez-Solano, R. Fayos-Jordan, and J. Lopez-Ballester, 'Accurate Estimation of Air Pollution in Outdoor Routes for Citizens and Decision Making', Applied Sciences, vol. 13, no. 17, p. 9930, Sep. 2023
  Impact Factor: 2.7 (Q2)
  DOI: 10.3390/app13179930

- R. Fayos-Jordan, S. Felici-Castell, J. Segura-Garcia, J. M. Alcaraz-Calero, and A. Cervello-Duato, 'Exploiting Multihoming Capabilities in 5G-Enabled IoT Nodes', IEEE Access, vol. 11, pp. 134940–134950, 2023
  Impact Factor: 3.9 (Q2)
  DOI: 10.1109/ACCESS.2023.3338180

- R. Fayos-Jordan, R. Araiz-Chapa, S. Felici-Castell, J. Segura-Garcia, J. J. Perez-Solano, and J. M. Alcaraz-Calero, 'ECO4RUPA: 5G-IoT Inclusive and Intelligent Routing Ecosystem with Low-Cost Air Quality Monitoring', Information (Switzerland), vol. 14, no. 8, 2023
  Impact Factor: 3.1 (Q2)
  DOI: 10.3390/info14080445

- E. Díaz-Rubio, J. Segura-Garcia, R. Fayos-Jordan, S. Cerdá, R. M. Cibrián, and A. Giménez-Pérez, 'Soundscape Evaluation of a Heritage Event in an Open Environment: The Water Tribunal of the Plain of Valencia (Spain)', Applied Sciences 2022, Vol. 12, Page 4292, vol. 12, no. 9, p. 4292, Apr. 2022
  Impact Factor: 2.7 (Q2)
  DOI: 10.3390/APP12094292

- S. Felici-Castell, M. Garcia-Pineda, J. Segura-Garcia, R. Fayos-Jordan, and J. Lopez-Ballester, 'Adaptive live video streaming on low-cost wireless multihop networks for road traffic surveillance in smart cities', Future Generation Computer Systems, vol. 115, pp. 741–755, 2021
  Impact Factor: 7.307 (Q1)
  DOI: 10.1016/j.future.2020.10.010

- R. Fayos-Jordan, J. Segura-Garcia, A. Soriano-Asensi, S. Felici-Castell, J. M. Felisi, and J. M. Alcaraz-Calero, 'VentQsys: Low-cost open IoT system for CO2 monitoring in classrooms', Wireless Networks, vol. 27, no. 8, pp. 5313–5327, 2021
  Impact Factor: 2.701 (Q2)
  DOI: 10.1007/s11276-021-02799-5

- R. Fayos-Jordan, S. Felici-Castell, J. Segura-Garcia, J. Lopez-Ballester, and M. Cobos, 'Performance comparison of container orchestration platforms with low cost devices in the fog, assisting Internet of Things applications', Journal of Network and Computer Applications, vol. 169, 2020
  Impact Factor: 6.281 (Q1)
  DOI: 10.1016/j.jnca.2020.102788

- R. Fayos-Jordan, S. Felici-Castell, J. Segura-Garcia, A. Pastor-Aparicio, and J. Lopez-Ballester, 'Elastic computing in the fog on internet of things to improve the performance of low cost nodes', Electronics (Switzerland), vol. 8, no. 12, 2019
  Impact Factor: 2.412 (Q2)
  DOI: 10.3390/electronics8121489

3.2 Conferences

- A. Soriano-Asensi, J. J. Perez Solano, S. Felici-Castell, J. Segura-Garcia, R. Fayos-Jordan, and E. A. Navarro Camba, 'Comparison of statistical methods for time synchronization between network nodes equipped with UWB transceivers', in Proceedings of the 11th Euro American Conference on Telematics and Information Systems, Aveiro Portugal: ACM, Jun. 2022, pp. 1–4
  DOI: 10.1145/3544538.3544669
- A. Suarez et al., 'Evaluation of the impact of the use of a robotic platform as a teaching tool', in 15th International Conference of Technology, Learning and Teaching of Electronics, TAEE 2022 - Proceedings, 2022
  DOI: 10.1109/TAEE54169.2022.9840600
- J. Segura-Garcia et al., 'Soundscape monitoring of modified psychoacoustic annoyance with Next-Generation EDGE computing and IoT', in ACM International Conference Proceeding Series, 2022.
  DOI: 10.1145/3544538.3544666
- A. Suarez et al., 'Educational robotics platform based on PSOC technology for teaching electronic integrated systems', in 15th International Technology, Education and Development Conference - INTED2021 Proceedings, Valencia: IATED, 2021, pp. 6749–6757
  DOI: 10.21125/inted.2021.1344
- R. Fayos-Jordan et al., 'Air quality detection and alert system as preventing COVID-19 transmission method in educational centers', in 15th International Technology, Education and Development Conference - INTED2021 Proceedings, Valencia: IATED, 2021, pp. 6791–6799
  DOI: 10.21125/inted.2021.1354
- A. Suarez, D. Garcia-Costa, P. A. Martinez, R. Fayos-Jordan, A. Amaro, and J. Martos, 'A transversal educational robotic platform for teaching in different education levels', in 14th International Technology, Education and Development Conference - INTED2020 Proceedings, in 14th International Technology, Education and Development Conference. Valencia: IATED, 2020, pp. 6032–6040. DOI: 10.21125/inted.2020.1629
- E. Navarro-modesto, A. Daghestani, E. A. Navarro-Camba, S. Felici-Castell, R. Fayos-Jordan, and J. Segura-Garcia, 'Green Telecommunication Networks to Mitigate CO 2 Emissions', in EATIS '20: Proceedings of the 10th Euro-American Conference on Telematics and Information Systems, Aveiro: Association for Computing Machinery, 2020, pp. 1–8.
  DOI: 10.1145/3401895.3401929

- M. Montagud, J. Segura-Garcia, J. A. De Rus, and R. Fayos-Jordan, 'Towards an Immersive and Accessible Virtual Reconstruction of Theaters from the Early Modern: Bringing Back Cultural Heritage from the Past', in IMX 2020 - Proceedings of the 2020 ACM International Conference on Interactive Media Experiences, in IMX '20. New York, NY, USA: Association for Computing Machinery, 2020, pp. 143–147.
  DOI: 10.1145/3391614.3399390

- M. Montagud, J. A. De Rus, R. Fayos-Jordan, M. Garcia-Pineda, and J. Segura-Garcia, 'Open-Source Software Tools for Measuring Resources Consumption and DASH Metrics', in IMX 2020 - Proceedings of the 2020 ACM International Conference on Interactive Media Experiences, in MMSys '20. New York, NY, USA: Association for Computing Machinery, 2020, pp. 261–266.
  DOI: 10.1145/3339825.3394931

- I. Mialdea-Flor et al., 'Development of a low-cost IoT system to detect and locate lightning strikes', in EATIS '20: Proceedings of the 10th Euro-American Conference on Telematics and Information Systems, Aveiro: Association for Computing Machinery, 2020, pp. 1–8.
  DOI: 10.1145/3401895.3401916

- J. Lopez-Ballester et al., 'Análisis de inteligibilidad y aproximación a parámetros de sala mediante Internet de las Cosas', in Proceedings del XI Congreso Ibérico de Acústica, 51º Congreso Español de Acustica TECNIACÚSTICA 2020, Algarve: Sociedad Española de Acústica, 2020, pp. 1–8.
  ISBN: 978-989-33-1221-6

- D. Garcia-Costa, A. Suarez, P. A. Martinez, J. Martos, R. Fayos-Jordan, and E. Lopez-Iñesta, 'A transversal virtual remote laboratory for teaching in STEM disciplines using robotic platforms', in 14th International Technology, Education and Development Conference - INTED2020 Proceedings, in 14th International Technology, Education and Development Conference. Valencia: IATED, 2020, pp. 6069–6075.
  DOI: 10.21125/inted.2020.1643

- R. Fayos-Jordan et al., 'Thorough analysis of Raspberry Pi devices in outdoor/indoor communications in terms of QoS', in EATIS '20: Proceedings of the 10th Euro-American Conference on Telematics and Information Systems, Aveiro: Association for Computing Machinery, 2020, pp. 1–7.
  DOI: 10.1145/3401895.3401915

- E. Díaz et al., 'Simulación acústica de un evento singular en un entorno abierto: El Tribunal de las Aguas de la Huerta de Valencia', in Proceedings del XI Congreso Ibérico de Acústica, 51º Congreso Español de Acustica TECNIACÚSTICA 2020, Algarve: Sociedad Española de Acústica, 2020, pp. 1–10.
  ISBN: 978-989-33-1221-6

- A. Pastor-Aparicio et al., 'Zwicker's annoyance model implementation in a WASN node', in INTER-NOISE 2019 MADRID - 48th International Congress and Exhibition on Noise Control Engineering, 2019. ISBN: 978-848-79-8531-7
- A. Pastor-Aparicio et al., 'URBAURAMON: Herramientas inteligentes para la gestin y control del paisaje sonoro urbano', in Actas de las XIV Jornadas de Ingeniería Telemática (JITEL 2019), Zaragoza: Universidad de Zaragoza, 2019.
  DOI: 10.26754/uz.978-84-09-21112-8
- A. Pastor-Aparicio, J. Lopez-Ballester, S. Felici-Castell, J. Segura-Garcia, R. Fayos-Jordan, and M. Garcia-Pineda, 'Efficient implementation of an IoT deployment for sound-scape monitoring', in Actas de las XIV Jornadas de Ingeniería Telemática (JITEL 2019), Zaragoza: Universidad de Zaragoza, 2019, pp. 166–172.
  DOI: 10.26754/uz.978-84-09-21112-8
- R. Fayos-Jordan, D. Garcia-Costa, M. Garcia-Pineda, S. Felici-Castell, and J. Segura-Garcia, 'Herramienta web portable para la creación , distribución y reproducción de vídeos DASH. MediaDASH Tool', in Actas de las XIV Jornadas de Ingeniería Telemática (JITEL 2019), Zaragoza: Universidad de Zaragoza, 2019, pp. 150–153.
  DOI: 10.26754/uz.978-84-09-21112-8

Chapter 4

Compendium of Publications in JCR as first author

Throughout the execution of this doctoral thesis, several milestones have been achieved in the field of IoT and Smart Cities, which deserve to be highlighted:

1. Integrate the concepts of orchestration and container management in Fog Computing environments.

2. Efficiently extend the interconnection capabilities of nodes with different interfaces by applying multihoming techniques and algorithms.

3. Solve and provide solutions for the welfare society by making use of IoT technologies, particularly for the scenarios outlined. These are:

   (a) Preventing the transmission of COVID-19

   (b) Monitoring noise annoyance

   (c) Using route planner for assisting citizens with respiratory or risk problems

This thesis has been conceived as a compendium of publications, adopting an approach that integrates and synthesises the knowledge derived from five original scientific contributions that meet the above-mentioned milestones.

These contributions can be divided into two sections: those focused on IoT and restricted to the Edge Computing domain, and those oriented to the whole Cloud-Edge-Fog Computing paradigm, as illustrated in Figure 4.1.

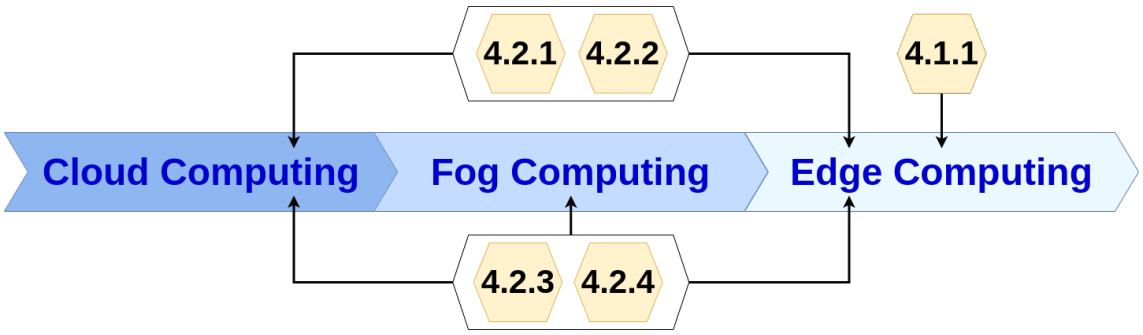


Figure 4.1: Contributions per Area Diagram

The selection of journals and conferences for the publication of the results of this thesis is based on rigorous criteria centred on the prestige and specific scope of each publication. This process guarantees the dissemination of contributions in environments where they are highly valued. In addition, careful consideration has been given to the thematic alignment between the research content and the editorial focus of each journal or conference, thus ensuring effective dissemination among experts and academics of the specific subject matter addressed in each contribution.

4.1 IoT and Edge Computing

This section describes the contributions focused on IoT and Edge Computing. These relate to research conducted and developments made for use in the Edge Computing paradigm only. However, like all IoT devices, they are capable of being incorporated into wider networks and can carry out some or all of the processing in other areas. In the context of the compendium of articles only one contribution has been included in this area, however, during the PhD more research has been carried out in this field as well as designed solutions relevant to this area as can be seen at [29], [42], [45]

4.1.1 Exploiting Multihoming Capabilities in 5G-Enabled IoT Nodes

Authors	Rafael Fayos-Jordan, Raquel Araiz-Chapa, Santiago Felici-Castell, Jaume Segura-Garcia, Juan J Perez-Solano, Jose M Alcaraz-Calero
Publication date	30 November 2023
Journal	IEEE Access
Impact Factor	3.9 (Q2)
Citations	n/a
DOI	`https://doi.org/10.1109/ACCESS.2023.3338180`

In the design phase during the research and development process of different IoT solutions, decisions are made regarding the mode of data transmission between nodes

or to a server. This choice is always subject to the availability of connections at any given time.

However, many devices make it possible to send data via several types of connections. WiFi is often chosen due to its simplicity, low cost and high transmission rate. However, in certain circumstances, this choice may not be feasible, making it necessary to consider alternatives such as LoRa, LTE-M, Bluetooth, NB-IoT, Sigfox, among others.

In response to this question, an idea was conceived to use all available forms of connections after taking into account factors such as availability, energy consumption, cost and transmission rate of each of them.

This paper presents an innovative solution in which, by calculating various factors, the device itself has the capacity to automatically select the most suitable form of communication for data transmission in each instance.

4.2 Cloud-Fog-Edge Computing Paradigm

4.2.1 VentQsys: Low-Cost Open IoT System for Monitoring in Classroom

Authors Rafael Fayos-Jordan, Jaume Segura-Garcia, Antonio Soriano-Asensi, Santiago Felici-Castell, Jose M Felisi, Jose M Alcaraz-Calero
Publication date 18 October 2021
Journal Wireless Networks
Impact Factor 2.701 (Q2)
Citations 4
DOI https://doi.org/10.1007/s11276-021-02799-5

Due to the COVID-19 pandemic that affected the world in the early 2020s, many countries, including Spain, implemented strict restrictions and containment measures to inhibit the spread of the virus. These measures limited the mobility of the population and had a significant impact on several areas, including university research.

In Spain, severe restrictions were in place between March and June 2020, resulting in the halting of most research, including the development and deployment of atmospheric conditions and pollution monitoring nodes that were part of the thesis.

However, with the easing of restrictions, the return of students to the university and the accumulation of information about the transmission of the virus, an opportunity to collaborate with the community on public health issues arose, which was aligned with one of the objectives of the thesis and contributed to society in times of crisis.

With the progressive publication of information on virus transmission, it became clear that enclosed spaces with inadequate ventilation presented an increased risk of infection. As the examination period at the School of Engineering was approaching,

there were concerns about the safety of students in poorly ventilated classrooms. Therefore, air quality monitoring became a crucial factor in ensuring student safety.

Accordingly, one of the objectives of this work included creating an indoor air quality monitoring system that combined Cloud and Edge capabilities, was easy to implement and deploy, had low cost and power consumption, could run on batteries, was autonomous in terms of updates and calibration, and whose electronic designs and code were open-hardware and open-software. This would allow anyone to replicate, improve or use these designs for their own projects.

4.2.2 ECO4RUPA: 5G-IoT Inclusive and Intelligent Routing Ecosystem with Low-Cost Air Quality Monitoring

Authors	Rafael Fayos-Jordan, Raquel Araiz-Chapa, Santiago Felici-Castell, Jaume Segura-Garcia, Juan J Perez-Solano, Jose M Alcaraz-Calero
Publication date	7 August 2023
Journal	Information
Impact Factor	3.1 (Q2)
Citations	1
DOI	`https://doi.org/10.3390/info14080445`

In all cases examined so far in relation to the health of citizens, the maximum scope that can be addressed is the monitoring of harmful atmospheric agents. The reduction of these agents is beyond the scope of this research, as it would require the implementation of regulations and legislation for their control.

However, it is feasible to assist the population in minimising their exposure to these agents as far as possible. The various factors harmful to health are not homogeneously distributed throughout the urban area; instead, they are conditioned by variables such as traffic at specific times or wind direction.

Against this background, advice can be given to a citizen travelling on foot to choose a less damaging route according to their personal circumstances. Such a choice may vary according to specific factors, making one route more advisable than another depending on the possible conditions he/she suffers from.

In this research, IoT nodes are used as sensors, along with kriging computational techniques and routing processes in the field of Cloud Computing. This approach aims to determine the healthiest route after considering specific conditions, such as if an individual suffers from asthma or the pregnancy status of a woman.

4.2.3 Elastic Computing in the Fog on Internet of Things to Improve the Performance of Low-Cost Nodes

Authors	Rafael Fayos-Jordan, Santiago Felici-Castell, Jaume Segura-Garcia, Adolfo Pastor-Aparicio, Jesus Lopez-Ballester
Publication date	6 December 2019
Journal	Electronics
Impact Factor	2.412 (Q2)
Citations	6
DOI	`https://doi.org/10.3390/electronics8121489`

The computational load used in the above-mentioned noise annoyance calculation is significantly high, which makes it very difficult to execute it in real time and rules out the feasibility of cost-effective nodes, such as ESPs, for this task.

In addition, as previously noted, due to legal restrictions, it is not feasible to transmit audio recordings made in public spaces to a remote location for processing (Cloud Computing).

The resolution of this problem is in line with one of the fundamental objectives of this doctoral thesis: the conception of systems oriented towards Fog Computing.

By using several Raspberry Pis configured in a cluster along with container orchestrators, it was possible to move the processing to a new, fully scalable and low-cost Fog Computing device.

This article outlines the research carried out to establish this cluster and details its operation through the implementation of Docker Swarm and Kubernetes.

4.2.4 Performance Comparison of Container Orchestration Platforms with Low-Cost Devices in the Fog, assisting Internet of Things Applications

Authors	Rafael Fayos-Jordan, Santiago Felici-Castell, Jaume Segura-Garcia, Jesus Lopez-Ballester, Maximo Cobos
Publication date	6 August 2020
Journal	Journal of Network and Computer Applications
Impact Factor	6.281 (Q1)
Citations	32
DOI	`https://doi.org/10.1016/j.jnca.2020.102788`

A novel device for Fog Computing which is closely related to the previous article has been proposed. However, on this occasion, it has been conceived with remarkably diverse hardware, making it possible to use virtually any set of hardware capable of running a complete operating system.

In this instance, the result was a heterogeneous cluster integrating diverse devices with markedly different capabilities, enabling the reuse of disused devices and thus

promoting sustainability.

This paper details the configuration of the aforementioned cluster, as well as its performance at the node level on both of the aforementioned orchestrators.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

During the course of this thesis, we have endeavoured to solve various problems within the framework of IoT, computing and communications by providing different solutions and contributions.

Solutions have been analysed for monitoring air quality, resulting from the increase in the number of vehicles, industrial facilities and other pollutants, as well as natural elements such as pollen or dust, together with noise generated by traffic or construction activities, and atmospheric factors such as temperature and humidity. These are all variables that directly affect the health of citizens, and although they are not always controllable, they can be relatively easily measured and monitored using Internet of Things (IoT) devices.

Also, in the field of public safety, these devices can be used for the monitoring and automatic detection of events such as traffic accidents, disturbances, riots, robberies and other factors that may affect the physical integrity of the population.

The integration of this information makes it possible to minimise the impact on citizens by optimising traffic lights, automatically issuing alerts to emergency services, recommending routes with less air pollution, notifying people affected by or susceptible to air quality, regulating traffic, alerting police authorities, and implementing a variety of improvements aimed at transforming today's cities into true Smart Cities.

The benefits of using IoT devices in healthcare have also been demonstrated in practical contexts, such as preventing infection during the COVID-19 pandemic and optimising walking routes for individuals with lung conditions or in at-risk situations.

It has also contributed to the area of public safety with an innovative video surveillance solution, distinguished by its high availability through the use of WiFi mesh networks and the application of DASH encryption. This proposal represents an extremely useful alternative in cases of emergency, such as accidents, or in situations of risk, such as disturbances.

To address these problems, different computational, communication and orchestration strategies have been proposed throughout this doctoral thesis, with several contributions conceived and implemented to tackle them. Although their ultimate implementation would hinge on large-scale deployment in a city, their correct functioning and usefulness have been demonstrated using different hardware in terms of operability and complexity. Yet, they constantly exhibit low cost, which makes their implementation possible without large investments. It has been demonstrated that these devices have the capacity to use different connections, such as WiFi, LoRa or Bluetooth. Moreover, they work in a complementary and automatic manner, which allows a single device to have a wide range of possibilities for data transmission in any situation.

Furthermore, in the context of this thesis, two low-cost solutions have been presented that have proven to be highly effective in addressing the aforementioned problems. On the one hand, a homogeneous Raspberry Pi cluster, fully scalable through the use of WiFi mesh networks, is proposed. This cluster stands out for its high efficiency given by its ability to connect or disconnect nodes on demand, and has been extensively tested in various working conditions, using container orchestrators such as Docker Swarm and Kubernetes. On the other hand, a cluster with similar characteristics, but heterogeneous in nature, has been developed using various low-cost hardware components. This approach has demonstrated the compatibility between hardware elements of different natures, allowing the reuse of disused components. These new Fog Computing devices are evidence of the viability and versatility that this technology can bring to both Smart Cities and the IoT ecosystem in general. This translates into reduced costs, minimised response times, energy savings, hardware reuse, mitigation of traffic and processing on centralised servers, and improved user experience.

The integration of these new IoT and Fog Computing devices, developed within the framework of this doctoral thesis, is evidence of the usefulness that these technologies can provide to cities and their inhabitants, significantly contributing to the improvement of the quality of life.

5.2 Future Work

IoT devices, Smart Cities, and Cloud, Fog, and Edge Computing paradigms are currently experiencing continuous expansion, with new technologies constantly emerging that can be integrated into this ecosystem. This makes it very challenging to define a set of future research or work in a static manner, as the landscape is constantly evolving, with new ideas emerging and old ones being discarded. However, in the context of this doctoral thesis, certain lines of action have been set aside for various reasons. It is relevant to highlight these as, at the time of writing, they remain areas of interest for future developments and more in-depth research.

- Use and Integration of Artificial Intelligence in a Variety of Environments

  While there have been significant advances in the application of Artificial In-

telligence (AI) in Cloud Computing environments, it is possible to bring this technology to Fog Computing environments in a very simple way. It is also feasible to integrate it with Edge Computing, where its implementation is not always possible due to the type of hardware used. In this context, the feasibility of using pre-trained AI models in environments that lack a complete operating system, such as ESP chips, has been demonstrated. This implementation can represent a significant advance, especially when combined with specific sensors.

- Evaluation, Proposal and Improvement of New Sensors for IoT Nodes

  Closely related to the previous point is ongoing research involving the use of e-noses, i.e., devices capable of detecting odours over a wide range. These sensors are used for detecting certain substances or their combinations. When integrated into IoT devices, they offer numerous advantages over conventional gas sensors. The incorporation of these sensors, together with the application of AI techniques, makes it possible to identify virtually any element in the environment. Substance detection, odour classification and quality control are just some of the many applications.

- Proposal and Design of new IoT Devices

  Most IoT devices currently available have one or two communication interfaces; these are largely determined by the boards on which they are based, such as Arduino, ESP, Raspberry Pi, etc. There are, however, some exceptions, such as the FiPy board despite its perception as being obsolete with a number of limitations. Due to these restrictions, the decision has been taken to develop a new board based on the ESP32 chip. This board would feature versatile connectivity, including WiFi, Bluetooth, LoRa, Sixfox, LTE-M and NB-IoT. In addition, it would incorporate features such as dual SIM, very low power consumption, the ability to connect any type of sensor and the integration of TensorFlow to enable the use of AI in Edge Computing. Presently, a first test version has been completed without a SIM card and the design for a new version with dual SIM capability is well advanced.

- Improved Multihomming Algorithm

  Due to the hardware and software restrictions present in the FiPy boards used in this research, it was necessary to disregard some factors, such as the received signal level (RSSI) when selecting the best communication path. Proposed improvements include the use of boards with less pronounced limitations in this aspect in addition to the one mentioned in the previous point. Similar tests are contemplated, incorporating RSSI as a factor in the equation, with the aim of optimising interface selection and developing a more efficient system.

- Deployments and Automatic Updates in Cloud-Fog-Edge

  One of the most significant challenges in large deployments of sensor nodes lies in the ability to perform upgrades without the need to manually modify the software at each deployed node, which could number in the hundreds or

even thousands. Construction of a comprehensive upgraded system using a variety of technologies is currently underway. The premise is that a software update in the cloud is propagated to all the devices involved, which is achieved using orchestrators – such as Docker Swarm or Kubernetes – and tags on the different nodes, especially on Linux-based devices (PCs, servers, Raspberry Pi). However, this approach excludes devices without a full operating system, such as FiPy or ESP. In an innovative approach, it has been possible to update both the software and the firmware of these nodes in an automatic and customised manner through the use of OTA (Over-The-Air) updates. This process is initiated from the Cloud and propagated through the Fog to the end nodes that require updates. This system is currently under development.

Anexxes I: Publications

A Exploiting Multihoming Capabilities in 5G-Enabled IoT Nodes

IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 2 November 2023, accepted 26 November 2023, date of publication 30 November 2023,
date of current version 5 December 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3338180

RESEARCH ARTICLE

Exploiting Multihoming Capabilities in 5G-Enabled IoT Nodes

RAFAEL FAYOS-JORDAN[1], SANTIAGO FELICI-CASTELL[2], JAUME SEGURA-GARCIA[2], JOSE M. ALCARAZ-CALERO[1], (Senior Member, IEEE), AND ANTONIO CERVELLO-DUATO[3]

[1]School of Computing, Engineering and Physical Sciences, University of the West of Scotland, Paisley, PA1 2BE Scotland, U.K.
[2]Department of Computer Science, Escola Tècnica Superior d'Enginyeria, Universitat de Valencia, Burjassot, 46100 Valencia, Spain
[3]Department of Electronic Engineering, Escola Tècnica Superior d'Enginyeria, Universitat de Valencia, Burjassot, 46100 Valencia, Spain

Corresponding authors: Santiago Felici-Castell (santiago.felici@uv.es) and Jaume Segura-Garcia (jaume.segura@uv.es)

This work was supported in part by the MCIN/AEI/10.13039/501100011033 under Grant PID2021-126823OB-I00, in part by the European Regional Development Fund (ERDF)—A Way of Making Europe funded by MCIN/AEI/10.13039/501100011033 and the European Union NextGenerationEU/Plan de Recuperación, Transformación y Resiliencia (PRTR) under Grant TED2021-131040B-C33, in part by the European Commission Horizon 2020 5G-PPP Program "6G BRAINS: Bringing Reinforcement learning Into Radio Light Network for Massive Connections" under Grant H2020-ICT-2020-2/101017226, in part by the Horizon Europe "INCODE: Programming Platform for Intelligent Collaborative Deployments Over Heterogeneous Edge-Internet of Things (IoT) Environments" under Grant HORIZON-CL4-2022-DATA-01-03/101093069. Also, to the Generalitat Valenciana with grant references CIAICO/2022/179 and CIAEST/2022/64, as well as the Research Vice-rectorship of Universitat de València for funding the grant with reference UV-INV-EPDI-2647726 for a research stay and the Spanish Ministry of Education in the call for Senior Professors and Researchers to stay in foreign centers for the grant PRX22/00503.

ABSTRACT 5G communications allow the integration of different wireless technologies to exploit the added value of multihoming scenarios. Furthermore, in connection with the Internet of Things (IoT), multi-homed nodes are very interesting, since it allows one to address new challenges and adapt the usage of communication technologies to new requirements imposed by the location of each deployment, the available energy resources on each node, and the IoT application itself. There are available commercial wireless nodes that embed and handle multiple interfaces. Terms such as seamless and transparent communications as well as technology integration fit well within the 5G standards. However, in practice, these wireless nodes are basically relayed in one of these wireless technologies when connected, and in the case of using other interfaces, their usage is not coordinated. In this context, we propose and implement a novel and efficient heterogeneous interface selection algorithm for multihoming and integrative communications using commercial products by performing local decisions on a per-packet basis, analysing different issues and approaches from a practical point of view, both hardware and software. We compare our proposal with other alternatives, highlighting the opportunities by exploiting these multihoming features in 5G-enabled IoT nodes.

INDEX TERMS 5G, heterogeneous networks, Internet of Things, multihoming, seamless/transparent communications.

I. INTRODUCTION

5G performance targets high data rate, reduced latency, energy savings, cost reduction, higher system capacity, and massive device connectivity [1]. These 5G communications allow the integration of different wireless technologies with the goal of supporting massive machine-type services, enhanced mobile broadband for multimedia distribution, and ultra-reliable low latency for emergency cases.

The associate editor coordinating the review of this manuscript and approving it for publication was Hosam El-Ocla.

5G is highly integrative, as one of its priorities is to provide a deep level of integration between cellular and short-range communications, tying any new 5G air interface and spectrum to provide universal high-rate coverage, seamless and transparent communications across radio access technologies. Thus, short- and long-range technologies would complement each other by using virtualization technologies and softwarization of the network and radio functions.

With regard to the Internet of Things (IoT), these multi-technology features provided by 5G technologies can add extremely new and interesting opportunities, since we

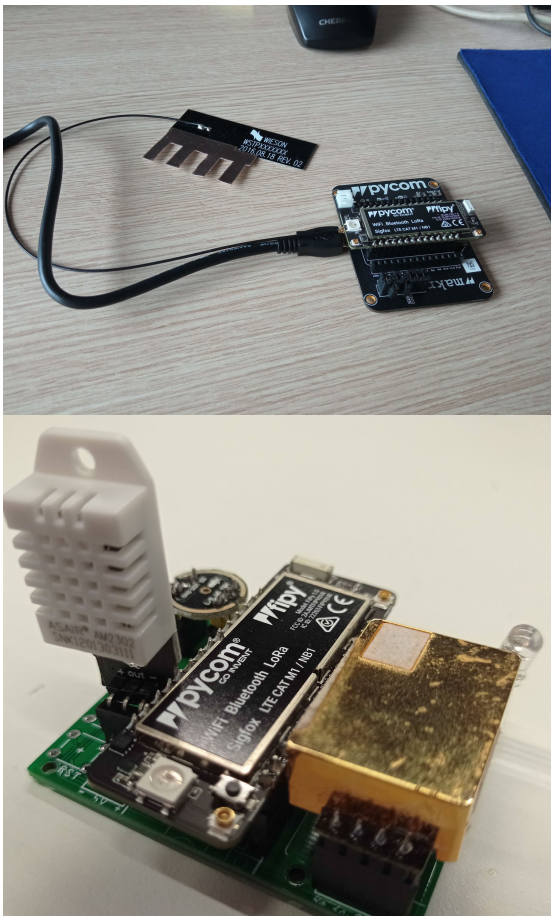


FIGURE 1. Above: a Pycom FiPy module. Below: an example of a 5G-enabled IoT node for air quality monitoring based on this module.

can address new challenges and adapt to new requirements imposed by the location of each deployment, the available energy resources on each node as well as the IoT application itself. Notice that the proliferation of wireless sensor networks requires a vast number of wireless nodes, where costs are a key factor, constraining drastically in general the performance of these deployed nodes, in terms of computation and communications capabilities. This limitation opens the doors to solutions based on fog and edge computing for smart distribution of computing load using 5G Mobile/Multi-access Edge Computing (MEC) architecture [2] to mitigate and overcome these issues. An overview of these challenges and enabling technologies for IoT within a 5G context are shown in [3], where it is discussed physically, MAC (medium access control), and network layers of wireless IoT networks, which all have significant impacts on latency and reliability.

In this context, several commercial products are embedding multiple wireless interfaces under the 5G paradigm within the same device. Anexample is the so-called FiPy module [4], [5] by Pycom Ltd,[1] that includes onboard technologies such as Low-Power Wide-Area Network (LPWAN) with LoRa/Sigfox,[2] short-range communications with WiFi and Bluetooth, classified as Low-power Wireless Personal Area Networks (LoWPAN) and cellular communications-based technologies, such as Long Term Evolution (LTE) for machines (LTE-M) and Narrow Band IoT (NB-IoT). In Figure 1, above it is shown this module with LTE-M communication and below, our own prototype for air quality IoT monitoring node.

Although these wireless technologies are rather different, under an abstraction layer these technologies could cooperate and work together, building scenarios known as heterogeneous networks. Nevertheless, in practice in most cases, as shown in the state-of-the-art section, these wireless nodes only use one of these wireless technologies at a time, and in the case of using other interfaces, their use is not coordinated.

In this paper, we focus on heterogeneous networks and multihomed 5G-enabled IoT nodes to exploit these features by using commercial products (in particular the FiPy module) in real scenarios, by analyzing different issues from a practical point of view, both hardware and software. For the interface selection, we propose and implement a novel algorithm by performing local decisions on a per-packet basis. We will compare our proposal with other alternatives, highlighting the opportunities by exploiting the multi-homing features.

The rest of the paper is structured as follows. Section II introduces the related work, highlighting the opportunities of our proposal. Section III describes the main features of commercial nodes with support for 5G communications on IoT missions, exploring the potential for multi-homing applications and their requirements. In Section IV, we propose an optimization algorithm to handle these requirements. In Section V, we present the experimental evaluation and discuss the results. Finally, in Section VI, we conclude the paper and show the future work.

II. RELATED WORK

In the context of heterogeneous networks within the 5G paradigm, we can find interesting contributions. In [6], it is proposed in a theoretical way a self-optimizing technique based on vertical handover (among different wireless technologies) for load balancing in heterogeneous wireless networks (2G/3G/4G/5G/Wi-Fi) where User Equipment's (UE) collect several Quality of Service (QoS) metrics and send them to the core (big-data repository) for further processing. These metrics are extended by the network provider's information under the perspective of a global environment as a global heterogeneous mobile network. The authors formalize the problem of access network selection to improve the quality of mobile services through the efficient use of heterogeneous wireless network resources and optimal horizontal-vertical handover procedures, proposing

[1]Pycom Ltd. went into administration on September 2022, but the newly created Pycom BV took over this company to prevent the products going end of life at the time of writing.

[2]In 2022, Sigfox went into bankruptcy proceedings and the firm Unabiz took its control at the time of writing.

a method for adaptive selection of wireless access based on big-data analysis over the centralized information gathered and shared. With their proposal, it is achieved a 16% increase in performance of the heterogeneous network using the statistical network resource reservation method, compared to homogeneous networks. In [7], the authors formulate the problem of distributed Radio Access Technologies (RAT) selection by considering multiple traffic classes over different heterogeneous technologies for serving the different applications running at the UEs, by applying a set of policies for ordering the preferences and the resource allocation of the network. Their framework is evaluated by simulation for a variety of traffic classes in terms of QoS requirements and provides results on capacity utilization and load distribution over available RATs.

With regard to seamless and transparent communications, we have found as well interesting initiatives. In [8], small cells are used to improve the performance in 5G networks. They increase the number of cell changes, handovers, and losses. Using distributed mobility-aware solutions in this scenario, as an alternative to a centralized mobility solution, scales better and is more efficient in terms of routing. However, there is a limitation in performance. Their results show 30% reduction in signaling, 53% in packet loss, and 90% reduced load on the core compared to the existing Locator and Identity Split Protocol (LISP) mobile node protocol. Other approaches based on device-to-device cooperation are shown in [9], [10], and [11]. In particular, in [9] it is shown, in a theoretical way, how a heterogeneous cellular network can provide traffic offloading that can be exploited to effectively improve network capacity by utilizing complementary network communication techniques (device-to-device) with a focus on massive connections for machine-type communications, determining network access, in order to improve overall network capacity and mitigate traffic congestion. In [10], in the same line, it is proposed theoretically an optimized caching strategy to share content distribution on top of device-to-device technology, based on information-centric networking principles in order to improve also network efficiency. In [11], we can find other alternatives to improve network performance in 5G, using overlay networks combined with device-to-device cooperation. In this case, it is proposed a smart base station-assisted partial-flow device-to-device offloading system that provides seamless video streaming services to clients by effectively offloading parts of the video traffic.

In [12], a novel paradigm for wireless network access is proposed in a 5G wireless Dynamic Network Architecture (5G-DNA), where certain classes of smart devices act as an Access Point (AP) temporarily, while connected to the Internet on 5G wireless DNA. Since these AP's are not stable, users need a seamless, efficient, and reliable schedule for switching between available AP's. The authors propose two different switching policies (taking into account preemptive issues) to improve network performance, concluding that in the low mobility scenarios, preemptive switching is preferred because it has acceptable signaling overhead and provides higher capacity, and brings higher profit for AP's. In the high mobility scenarios, non-preemptive switching is a better option because it incurs fewer switching and higher profit for AP's.

Also, we can find different applications where communications in this context are a key point and where the simultaneous use of different interfaces is extremely important. In [13], a use case for an ambulance service with medical video streaming to the hospital based on 5G small cell-based is shown. Using small cell networks, it can enhance the medical QoS. Besides, it studied the impact of small-cell heterogeneous networks on medical video streaming along with the system modeling and technical requirements, including the performance analysis for medical video sequences affected by packet losses. The results of the proposed scenario show through a simulation that in a mobile small cell-based ambulance scenario, outperforms the traditional macrocell network scenario. In [14], this critical application is better shown with a real deployment and a specific network slicing for this video streaming priorization.

About commercial 5G-enabled IoT platforms, in [15], using the mentioned FiPy module, it proposed a comparison between WiFi and LoRa technologies taking into account the constraints of each technology. The comparison done in a factory, includes transmission time, energy consumption, and coverage, highlighting that LoRa has better coverage but less Throughput. In a similar way, in [16] it was tested the same module for IoT monitoring using both LoRa and Sigfox technologies, but independently, achieving distances of 1.17 and 10 km respectively. It is worth mentioning other similar and interesting commercial products based on ESP32 system-on-chip [17] that could be used under certain 5G paradigms, such as LILYGO TTGO T-SIM7000G Module ESP32 [18], but their characteristics, functionality, and design do not fit in the same way as the FiPy module does.

The concept of multihoming is also linked to routing protocols. From this approach, different authors have proposed routing protocols for multi-homing nodes in an IoT context. In [19], the authors propose the Ant Colony Optimization On-demand Multi-path Distance Vector (ACO-MDV) routing algorithm to enhance power efficiency, considering multi-homing options on IoT. In [20], it is shown an energy-aware heuristic routing and resource allocation for a multi-hop network via smart edge device-to-device communications.

In terms of transport layer protocols, traditional Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) as well as Stream Control Transmission Protocol (SCTP) are still used in this context. In particular, SCTP manages several logic interfaces with the goal to use all of them to improve fault tolerance. Extensions to these established transport protocols are receiving considerable attention from

Internet Engineering Task Force (IETF). In [21] two main extensions for concurrent transmissions, the Multipath TCP (MP-TCP) extension for TCP and the Concurrent Multipath Transfer extension for SCTP (CMT-SCTP) are analyzed and compared. The authors highlight that the different path management strategies of both protocols have a significant impact on their performance in real scenarios. MP-TCP creates a full mesh of paths among the available interfaces that perform significantly better than CMT-SCTP, which only uses an alternative path for robustness. Nevertheless, the MP-TCP strategy suffers from scalability problems due to the full mesh among the multiple interfaces. Other solutions based on overlay networks, tunneling techniques, and Software Defined Networks (SDN) are also relevant. In [22], it is analyzed the switching time (handoff time) among heterogeneous interfaces (Ethernet, WiFi, Bluetooth) using an Open Virtual Switch (OvS) managed by an RYU SDN controller on a Raspberry Pi. In conclusion, the authors show that the handoff time varies greatly depending on the interface type and the switching direction. But notice that the switching decision is done manually for the experiments and benchmarking, not decided based on other external or internal requirements, such as application, battery status, etc.

Finally, in [23] are depicted the open challenges and a survey of the potentials of 5G technologies in terms of multihoming options considering QoS and Quality of Experience (QoE) issues. Besides, in [24], is done a review of the related work taking into account the utilization of 5G technologies in IoT applications, considering the limitations due to network optimization and interferences. In this line, in [25], it is proposed a multihomed network model for coexistence and diverse IoT applications in smart cities from a theoretical point of view. Also in [26], the use of machine learning techniques has been applied for traffic steering in scenarios with heterogeneous ultra dense 5G networks.

In summary, we see from this review that on one hand there are many theoretical approaches that can help us to analyze and to define different strategies, most of them based on offloading techniques based on device-to-device communications and using vertical handovers. However, although these scenarios are interesting for certain 5G applications, they do not fit well with our approach from a practical point of view. On the other hand, we have seen practical implementations using different wireless technologies on heterogeneous networks, but without a global vision of cooperation. Thus, based on this, we see that there is a need to coordinate and manage the different wireless technologies in the context of heterogeneous networks, focusing on 5G-enabled IoT nodes, in particular, based on Fipy modules. In this case, some constraints are internal to the nodes, given by the Fipy module itself and its implementation, the available energy resources on each node and other constraints are external to the nodes, given by the wireless technologies and the location of each deployment and of course by the IoT application. All these issues are discussed and explained in the following sections. Notice that at the end, in Section V, we have introduced Table 4, where we compare our proposal with the references from the related work with higher similarity, analyzing the pros and cons of each, along with some comments.

TABLE 1. Comparison of the different wireless technologies embedded in a FiPy module.

Technology	Consumption	Coverage	Bit-rate
Sigfox	very low	medium	very low
LoRa	low	medium	low
NB-IoT	medium	high	low
LTE-M	medium++	high	medium –
WiFi	high	medium-	high
Bluetooth	medium+	low	medium++

III. A COMMERCIAL 5G-ENABLED IOT NODE

One of the most relevant low-cost commercial platforms for the 5G paradigm is the FiPy module. The design of this module consists of an integration of different elements on a board with the goal to build a generic 5G-enabled IoT node with a set of heterogeneous wireless interfaces.

A. HARDWARE ISSUES

The FiPy module includes different wireless communication technologies, such as WiFi, Bluetooth, LoRa, Sigfox and dual LTE-M (CAT M1) and NB-IoT on one single tiny board. The main features of these onboard technologies are detailed next. Sigfox (as a class 0 device) and LORA (both class A and C device types) use 800/900 MHz band ISM. WiFi (IEEE 802.11b/g/n 16 Mbps) and Bluetooth (low energy and classic) use 2.4 GHZ band ISM. LTE-M/NB-IoT, based on 3GPP release 13 LTE Advanced Pro, supports narrow-band LTE UE categories M1/NB1 use 17 radio frequency bands supported from 699 MHz to 2170 MHz. In LTE-M/NB-IoT the transmission current is 420 mA peak at 1.5 W and a reception current of 330 mA peak at 1.2 W.

This module is based on the Tensilica LX6 CPU Xtensa®dual-core 32-bit LX6 microprocessor(s) (the same as the ESP32 system-on-chip [17]), that allows up to 600 Dhrystone Million Instructions per Second (DMIPS), with an extra ultra-low power coprocessor that can monitor General Purpose Input/Outputs (GPIO), the ADC channels and controls most of the internal peripherals during deep-sleep mode, consuming only 25uA. The module is equipped with 520KB and 4MB RAM and 8MB flash memory.

Table 1 summarizes the main features of the different interfaces and technologies embedded in the FiPy module, highlighting consumption, coverage, and bit rate. Among these technologies, we must stress that Sigfox, LoRa and Bluetooth technologies work in Layers 1 and 2 and the other ones in Layer 3 of the Open System Interconnection (OSI) model. For Layer 1 and 2 technologies, we should add externally a gateway to include Layer 3 connectivity through a gateway, in the same way as LoRa and LoRaWAN specification do [27].

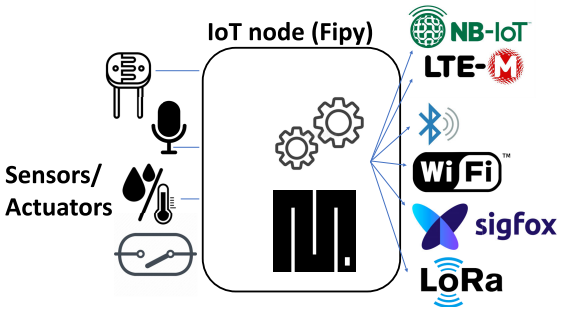


FIGURE 2. Scheme of a multi-homed IoT node based on the FiPy module with sensors and actuators.

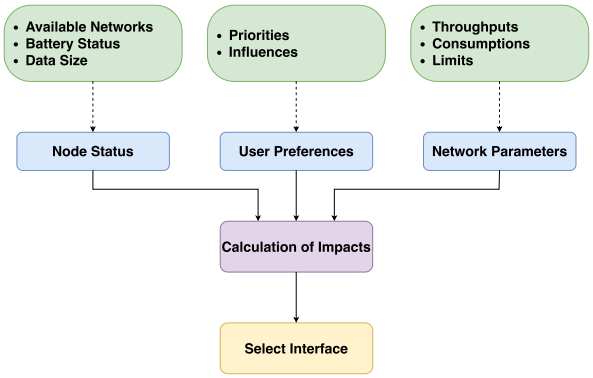


FIGURE 3. Flow chart of the proposed multihoming optimization algorithm to select the best wireless interface.

B. SOFTWARE ISSUES

In our case, the main program is running the IoT application, reading from sensors and writing to actuators, as it is shown in Figure 2. When it is necessary to send or receive information, the module will run locally a decision algorithm, explained in the next section, to choose at the time of transmission the most appropriate wireless interface.

The FiPy module by default runs a MicroPython interpreter (multi-threading enabled). MicroPython is a programming language designed specifically for constrained environments such as microcontrollers and embedded systems. It is a subset of the Python 3 programming language that has been heavily optimized for these specific platforms. MicroPython does not support some of the advanced features found in Python 3 such as meta-classes (at least yet) and dynamic typing. Additionally, the libraries and functions available in MicroPython are limited due to the constrained environment and the need for minimal memory and processor utilization.

Thus, the access to the wireless interfaces must take into account a set of rules and policies given by each wireless technology and others imposed by the FiPy module itself. Notice that FiPy modules have some limitations given by the quality of the built-in modems, as we will show in Section V.

We conclude that based on these constraints, the FiPy module fits well in a scenario with heterogeneous networks. However, for the IoT application, it is challenging to optimize the communication process. The IoT application will send and receive packets and when this is done, the wireless technology should be selected according to local criteria. This decision should be based on the location of each deployment, the available energy resources on each node as well as the IoT application itself. With regard to the transport protocols to manage the communication process, we discard heavy and standardized multihoming and multipath transport protocols, such as MP-TCP and CMT-SCTP, since they exceed the resources on these modules. In addition, for simplicity and to provide verifiable results, our approach to the communication process is based on the analysis of independent packets.

IV. PROPOSED MULTIHOMING OPTIMIZATION ALGORITHM

In this section, we propose the multihoming optimization algorithm and its implementation to select the appropriate wireless interface, according to a set of policies and preferences given at the transmission time, on a per-packet basis as discussed before. It must be stressed that the proposed algorithm can be easily generalized and adapted to any other similar multihoming node.

A. ALGORITHM DESIGN

Figure 3 shows the flow chart of the proposed algorithm. This algorithm is running on each FiPy module, and it selects locally the best technology (or interface) by weighting different metrics and parameters as it is explained next, according heuristically to the performance benchmarking of this module. As depicted in this figure, the selection is based on the calculation of the different impacts, by analyzing the status of the node, the user preferences, and the network parameters per technology.

Thus, this selection is performed using an optimization logic by processing parameters, values, and their relation (by given equations), defined in Tables 2 and 3. In particular, Table 2 defines the different parameters, metrics, and impacts, given by the communication requirements and the IoT application's needs. In this table, we distinguish three types of metrics: a) **measured metrics**, directly measured from the module, such as measured throughput, consumption in miliwatts per byte sent (mW/B), and price or monetary costs, b) **user-defined metrics**, that can be set as predefined user influences or imposed by the standards (for instance, in case of LoRa and Sigfox) for the selection criteria, determined by priority, time limit and impacts given by throughput, data limit, consumption, battery and price and c) **automatically detected metrics**, that depend on the hardware, such as the technology availability, battery levels, data to send or the required time for the transmission.

In Table 3, we define the proposed and underlying equations that determine the selection of the communication technology. To carry out the comparison, we normalize these values among the technologies, such as throughput, data limit, consumption, and price, denoted as NTp_x, NDl_x, NCo_x and NPr_x. Additionally, users can influence the result through personal preferences by defining the impact of each parameter defined, setting influence indexes

TABLE 2. Parameters defined per technology $x = [wifi, bt, lora, lte, nb]$ used in the proposed algorithm.

Measured			User Defined			Automatically Detected		
Name	**Abbreviation**	**Units**	**Name**	**Abbreviation**	**Units**	**Name**	**Abbreviation**	**Units**
Throughput(x)	Tp_x	Kbps	Priority(x)	P_x	[0..1]	Available(x)	A_x	[0, 1]
Consumption(x)	Co_x	mW/B	Time Limit(x)	Tl_x	%	Battery Level	Bl	[0..1]
Price(x)	Pr_x	-	Throughput *Influence*	I_{tp}	[0..1]	Data to Send	Ds	Bytes
			Data Limit *Influence*	I_{dl}	[0..1]	Time Used(x)	Tu_x	%
			Consumption *Influence*	I_{co}	[0..1]			
			Battery *Influence*	I_{bl}	[0..1]			
			Price *Influence*	I_{pr}	[0..1]			

TABLE 3. Set of equations used by the proposed Algorithm per technology $x = [wifi, bt, lora, lte, nb]$.

Normalized Throughput(x): $NTp_x = \frac{Tp_x}{Tp_{max}}$		**Throughput Impact(x)**: $ITp_x = NTp_x^{I_{tp}}$	
Data Limit(x):	Remaining Data:	Normalized Data Limit(x):	**Data Limit Impact(x)**:
$Dl_x = Tp_x \cdot 60 \cdot 60 \cdot 24 \cdot \frac{Tl_x}{100}$	$Rd_x = Tp_x \cdot 60 \cdot 60 \cdot 24 \cdot \frac{Tl_x - Tu_x}{100}$	$NDl_x = \frac{Dl_x}{Dl_{max}}$	$IDl_x = NDl_x^{I_{dl}}$
Normalized Consumption(x):	Inverse NCo_x:	**Consumption Impact(x)**:	
$NCo_x = \frac{Co_x}{Co_{max}}$	$INCo_x = -NCo_x + NCo_{min} + 1$	$ICo_x = INCo_x^{I_{co}}$	
Battery Level Impact: $IBl_x = (NCo_x^{-Bl+2})^{I_{bl}}$			
Normalized Price(x):	Inverse NPr_x:	**Price Impact(x)**:	
$NPr_x = \frac{Pr_x}{Pr_{max}}$	$INPr_x = (-NPr_x + NPr_{min}) \cdot 0.9 + 1$	$IPr_x = INPr_x^{I_{pr}}$	
Time Limit Impact:	$ITl_x = \begin{cases} 1, & Rd_x \geq 8Ds \\ 0, & Rd_x < 8Ds \end{cases}$		

from 0 (no impact) to 1 (full impact) and are calculated automatically based on measures, user preferences and status of different parameters. The data limit and remaining data expressions are related to the measured throughput (Tp_x) and the time limit (Tl_x). As a consequence, data limit impact is related to the data limit influence (I_{dl}) and the normalized data limit (NDl_x). Notice that these expressions will allow us to fulfill the LoRa and Sigfox standards. In relation to energy consumption, the consumption impact is proportional to the inverse normalized consumption ($INCo_x$) and the consumption influence (I_{Co}), together with the battery level impact (IBl_x) that depends on normalized consumption (NCo_x), the battery level (Bl) and the battery influence (I_{bl}). The price impact is related to the inverse normalized price ($INPr_x$) and the price influence (I_{Pr}). Finally, the impact of the time limit (ITl_x) is a binary function that considers eight times the data to send (D_s) as a threshold for the remaining data (Rd_x).

Then, given these parameters, the algorithm calculates a Final Score (Fs) for each technology (x) as follows:

$$Fs_x = P_x \cdot A_x \cdot ITp_x \cdot IDl_x \cdot ICo_x \cdot IBl_x \cdot IPr_x \cdot ITl_x \quad (1)$$

Thus, from these scores, the selected interface to transmit the packet is given by the one with the highest Fs_x.

B. ALGORITHM IMPLEMENTATION

The implementation of the algorithm in the FiPy module is as follows. The file structure of the MicroPython application is led by the *main* and the *boot* files. Initially, the *boot* file disables all the interfaces, in order to avoid energy consumption. Then, in the same folder where these two files are located, we add a file containing a class with the implementation of the selection algorithm, as described before. Thus, we call this class from the *main* file every time we want to send a packet through the selected interface. Besides, the parameters in Tables 2 and 3 are automatically updated internally to the module every time a packet is sent, except Ax that it should be updated periodically (in our case, every day), as a kind of self-auto calibration, to check if there are changes in availability for the onboard wireless technologies.

V. EVALUATION AND RESULTS

As was explained in the previous section, the selection algorithm relies on the evaluation and estimation of the parameters and metrics depicted in Tables 2 and 3 for each technology. Due to the limitations of the FiPy module in each of its interfaces, a prior benchmarking and profiling process is necessary to adjust the selection algorithm. For example,

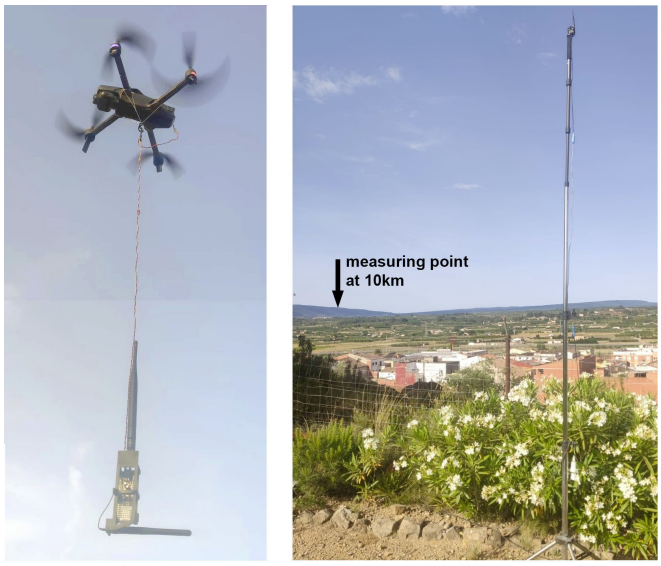


FIGURE 4. Example of pictures for the technology benchmarking and profiling process of the FiPy module for LoRa, using a drone as support: a) client node and b) server node.

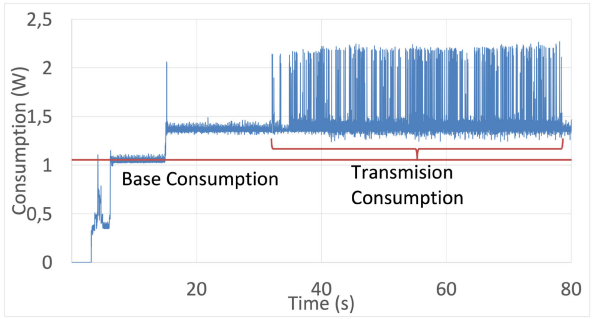


FIGURE 5. WiFi Consumption while transmitting a total of 10MB.

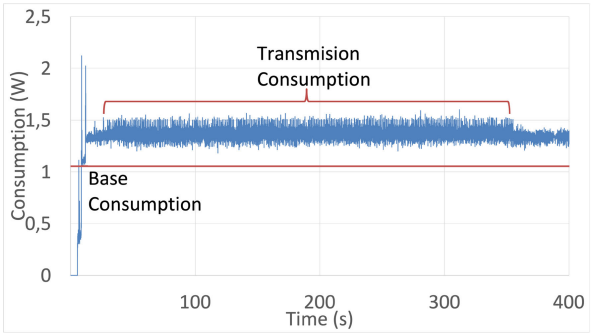


FIGURE 6. Bluetooth Consumption while transmitting a total of 1MB.

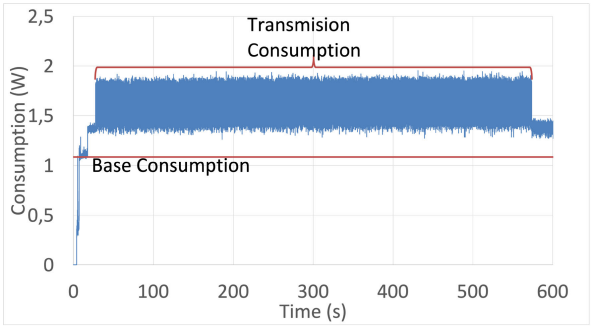


FIGURE 7. LoRa Consumption while transmitting a total of 1MB.

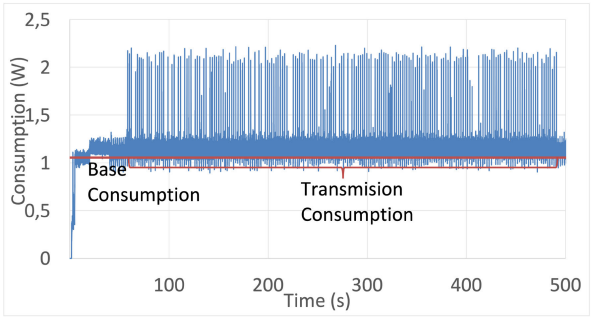


FIGURE 8. LTE-M Consumption while transmitting a total of 1MB.

it is not the same case if it is transmitted from one module to another directly than to an access point, as the performance depends on the hardware used in each case, affecting the throughput and the transmission distance. Besides, to ensure reliable performance measurements in the communication process, for these tests in particular, Fresnel zones were taken into account according to the carrier frequency used in each technology, the line of sight, and avoiding external interference zones. Thus, tripods over 4 m high were used when the distance did not exceed 300 m, as in the case of WiFi and Bluetooth, and for longer distances, a drone was used instead. Figure 4 shows the testbed for LoRa technology using this drone (as a client) and the server.

The tests carried out to estimate the parameters shown in Table 2, were as follows. For throughput analysis two FiPy modules were used, one as a server and the other as a client, and the transmission was done at different distances, using different packet sizes to determine the optimal for each distance.

For consumption analysis, both voltage and current were monitored all the time. The nodes were programmed to deactivate all interfaces and only activate the one to be tested. Initially, to measure the base consumption the nodes were inactive (idle) for a few seconds, and after that, started transmitting a total of 10MB in the case of WiFi and 1MB otherwise, to measure the transmission consumption. Notice that since the heterogeneous interfaces in the same module have different nominal throughputs, and because we are also measuring power consumption, we have analysed each interface under a similar time window, that is the reason we use different data sizes. With these results, it was possible to determine the base consumption of the FiPy module, the consumption when the interface is active, and when the data is being transmitted. Figures 5-8 show the consumption for the interfaces tested at these three different steps: base (idle), active, and transmitting. With these results, we are able to calculate and estimate the consumption per byte, by taking as a reference the consumption while sending data, subtracting

	Tp(x) [Kbps]	Co(x) [mw/B]	Pr(x)	Tl(x) [%]	Tu(x) [%]
WiFi	1800	0,000176	0	100	20
Bluetooth	50	0,010522	0	100	50
LoRa	18	0,016693	0	1	0
LTE-M	18	0,004087	20	100	23

Ds	Bl		Itp	Idl	Ico	Ibl	Ipr		
50	0,9		0,5	0,5	0,1	0,8	0,9		
	P(x)	A(x)	ITp(x)	IDl(x)	ICo(x)	IBl(x)	IPr(x)	ITl(x)	Fs(x)
WiFi	1	0	1	1	1	1	1	1	0
Bluetooth	1	0	0,166666667	0,166666667	0,969001655	0,972670124	1	1	0
LoRa	1	1	0,1	0,01	0,945130747	0,951552741	1	1	0,000899342
LTE-M	1	1	0,1	0,1	0,989286488	0,990566025	0,125892541	1	0,001233688

FIGURE 9. Example of interface selection using the proposed algorithm on the FiPy module.

Ds	Bl		Itp	Idl	Ico	Ibl	Ipr		
50	0,9		0,5	0,3	0,1	0,8	0,9		
	P(x)	A(x)	ITp(x)	IDl(x)	ICo(x)	IBl(x)	IPr(x)	ITl(x)	Fs(x)
WiFi	1	0	1	1	1	1	1	1	0
Bluetooth	1	0	0,166666667	0,341278752	0,969001655	0,972670124	1	1	0
LoRa	1	1	0,1	0,063095734	0,945130747	0,951552741	1	1	0,005674463
LTE-M	1	1	0,1	0,251188643	0,989286488	0,990566025	0,125892541	1	0,003098885

FIGURE 10. Another example of interface selection changing data limit influence.

TABLE 4. Comparison of the multi-homing proposal with the state-of-the-art alternatives with pros and cons.

Reference	Pros	Cons	Comments
[9]	traffic offloading by device to device to reduce congestion	theoretical	heterogeneous cellular network
[10]	optimized caching strategy	theoretical	share content distribution on top of device to device
[6]	self-optimising technique based on vertical handover	theoretical	load balancing in heterogeneous wireless networks based on big data learning
[12]	seamless and efficient schedule for switching between AP	theoretical	5G-DNA where smart devices act temporarily as AP
[11]	overlay networks combined with device to device cooperation	theoretical	smart device-to-device offloading to provide seamless video streaming
[7]	dynamic heterogeneous network selection algorithm based on network UE requirements	theoretical	UEs allocate each traffic class through each technology given certain policies
[15]	performance evaluation	WiFi and LoRa independently	comparison between WiFi and LoRa technologies using FiPy modules
[16]	performance evaluation	Sigfox and LoRa independently	comparison between Sigfox and LoRa technologies using FiPy modules
ours	multi-home communications in 5G-enabled node	constraints imposed by MicroPython, extra 2 *ms* delay	context-aware technology selection in heterogeneous wireless network based on FiPy module

the base consumption, and dividing by the total number of bytes sent.

Finally, the results of this technology benchmarking and profiling process are shown in Figures 11-14. In this case, the Throughput vs. Received Signal Strength Indicator (RSSI) is shown for each technology. We must stress that due to the limitations of the modems embedded in the FiPy module, the throughput seems to be unaffected by RSSI (it does not depend on the distance) and only slightly when the connection is going to be lost. Thus, according to this, RSSI is not relevant to this algorithm for these modules. It may seem that in the case of LTE-M the RSSI affects the throughput but in fact, measurements taken below −75dBm are very unstable and have continuous connection losses and are therefore not taken into account.

Figure 9 shows an example of these calculations and trade-off analysis based on Equation 1 to select the best interface. In this case, the parameters have been introduced as an example to simulate that the WiFi and Bluetooth interfaces were not available and only LoRa and LTE-M were available.

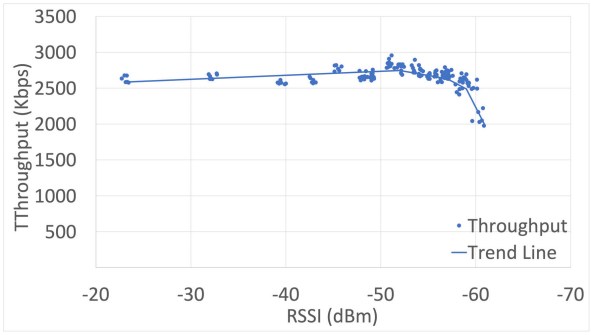


FIGURE 11. Throughput versus RSSI in WiFi transmission.

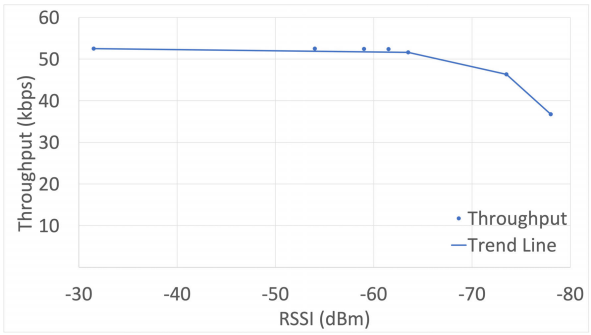


FIGURE 12. Throughput versus RSSI in bluetooth transmission.

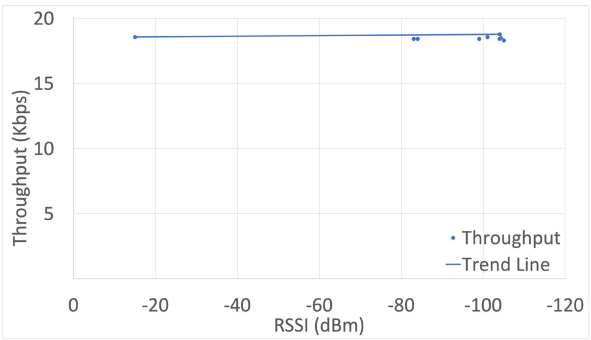


FIGURE 13. Throughput versus RSSI in LoRa transmission.

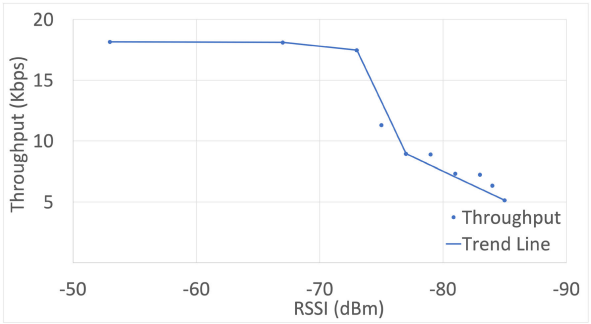


FIGURE 14. Throughput versus RSSI in LTE-M transmission.

In this example, LoRa was better in terms of price and had the same throughput score. However, LTE-M was better in consumption and data limit, and it was finally the interface selected by the algorithm. But, if we increase the price of LTE-M, the result changes to LoRa, as well as if we make available WiFi and Bluetooth, then the algorithm would choose WiFi. Another example can be seen in Figure 10, but reducing the influence of the data limit, where in this case, we can see that the algorithm selects LoRa as the interface to send data.

As a conclusion of these exhaustive tests, it is clear that the WiFi interface outperforms the others in all terms analysed (throughput, consumption, price and time limits). Thus, if this interface is available, the algorithm always selects WiFi as the transmission interface, but this may change at other nodes, with the availability of new transmission technologies, or if the user sets the WiFi Priority very low.

In Table 4, we compare our proposal with the state-of-the-art. In particular, we analysed and showed similar alternatives considered in Section II, the pros and cons of each, along with some relevant comments. As we can see, most of them are theoretical, except two ([15] and [16]), that have a practical approach using the same FiPy module, although considering each wireless technology independently and not concurrently.

VI. CONCLUSION

In this work, we have focused on transparent and integrative communications in a 5G and IoT context. We have shown an implementation of a multi-homed 5G-enabled IoT node, based on a novel and efficient heterogeneous interface selection algorithm running in a commercial multi-homed product, the Fipy module, by performing local decisions on a per-packet basis at the transmission or reception time. The selection criteria depend on the location of the node and available technologies, the available energy resources as well as the IoT application itself.

On the one hand, with regard to the advantages of our proposal, we see the benefits of managing simultaneously several heterogeneous wireless interfaces, in terms of resilience and fault tolerance, efficient communication management, flexibility to adapt new scenarios in a proactive way, as well as to adapt easily to new application requirements. However, on the other hand, we add additional burden and processing, which is translated into an extra delay of 2 ms. Although this extra delay is very small and should have a nearly negligible impact on most IoT applications, it could have some impact on the performance in high-speed and real-time applications. But we must stress that these applications are not the most common in the context of IoT.

Also from our results and performance evaluation, we realize that there are some limitations on the Fipy module due to a combination of the hardware selection (mainly the ESP32 MCU selection) and the firmware implementation (we have used the latest published version: 1.20.2 release 6 of this firmware[3]).

[3]This firmware has a release date of 28/10/2021 but has no major modifications since August 2019 and is based on a branch of ESP-IDF which has no updates since 11/01/2019 and its last major modification dates from 2017.

As future work, we are currently working on a new design and enhanced version of this module to overcome these limitations, taking into account important factors such as energy and costs. Besides, we are porting the algorithm to other IoT devices and boards, such as Raspberry Pi or Nvidia Jetson. In addition, we are currently working on a new board that implements the same communication technologies as FiPy but without so many hardware and software limitations and we plan to test the algorithm on it and carry out specific tests to include parameters such as RSSI or packet size in order to improve the algorithm. Also, in order to increase the alternatives and communications paths, we will consider the option to forward packets using the cooperation of neighbouring nodes in a kind of fog collaboration process.

ACKNOWLEDGMENT

The authors would like to thank Mohamed Khadmaoui-Bichouna for his contributions to the work of this article.

REFERENCES

[1] *5G; System Architecture for the 5G System*, document TS 23.501, ETSI, 3GPP, Version 15.3.0, Release 15, 2022, Accessed: Jun. 3, 2023. [Online]. Available: https://www.etsi.org/deliver/etsi_ts/123500_123599/123501/15.03.00_60/ts_123501v150300p.pdf

[2] (2022). *ETSI GS MEC 003: Mobile Edge Computing (MEC); 003 V3.1.1 (2022-03)*. Accessed: Jul. 6, 2023. [Online]. Available: https://www.etsi.org/deliver/etsi_gs/MEC/001_099/003/03.01.01_60/gs_MEC003v030101p.pdf

[3] Z. Ma, M. Xiao, Y. Xiao, Z. Pang, H. V. Poor, and B. Vucetic, "High-reliability and low-latency wireless communication for Internet of Things: Challenges, fundamentals, and enabling technologies," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 7946–7970, Oct. 2019.

[4] Pycom.io. (2022). *FiPy, Five Network Development Board for IoT*. [Online]. Available: Accessed: Dec. 28, 2022. [Online]. Available: https://pycom.io/product/fipy/

[5] Pycom. (2018). *Fipy Specifications*. Accessed: Apr. 29, 2022. [Online]. Available: https://docs.pycom.io/datasheets/development/fipy/

[6] M. Beshley, N. Kryvinska, O. Yaremko, and H. Beshley, "A self-optimizing technique based on vertical handover for load balancing in heterogeneous wireless networks using big data analytics," *Appl. Sci.*, vol. 11, no. 11, p. 4737, May 2021. [Online]. Available: https://www.mdpi.com/2076-3417/11/11/4737

[7] V. Passas, V. Miliotis, N. Makris, and T. Korakis, "Dynamic RAT selection and pricing for efficient traffic allocation in 5G HetNets," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–6.

[8] E. M. O. Fafolahan and S. Pierre, "A seamless mobility management protocol in 5G locator identificator split dense small cells," *IEEE Trans. Mobile Comput.*, vol. 19, no. 8, pp. 1745–1759, Aug. 2020.

[9] W. Cao, G. Feng, S. Qin, and M. Yan, "Cellular offloading in heterogeneous mobile networks with D2D communication assistance," *IEEE Trans. Veh. Technol.*, vol. 66, no. 5, pp. 4245–4255, May 2017.

[10] C. Xu, M. Wang, X. Chen, L. Zhong, and L. A. Grieco, "Optimal information centric caching in 5G device-to-device communications," *IEEE Trans. Mobile Comput.*, vol. 17, no. 9, pp. 2114–2126, Sep. 2018.

[11] G. S. Park, W. Kim, S. H. Jeong, and H. Song, "Smart base station-assisted partial-flow device-to-device offloading system for video streaming services," *IEEE Trans. Mobile Comput.*, vol. 16, no. 9, pp. 2639–2655, Sep. 2017.

[12] X. Huang, F. Ke, R. Yu, Y. Liu, and Y. Wu, "Preemptive and non preemptive switching protocols for 5G wireless dynamic network architecture," *IEEE Trans. Veh. Technol.*, vol. 68, no. 12, pp. 12256–12270, Dec. 2019.

[13] I. U. Rehman, M. M. Nasralla, A. Ali, and N. Philip, "Small cell-based ambulance scenario for medical video streaming: A 5G-health use case," in *Proc. 15th Int. Conf. Smart Cities, Improving Quality Life Using ICT IoT (HONET-ICT)*, Oct. 2018, pp. 29–32.

[14] I. Martinez-Alpiste, J. M. Alcaraz-Calero, Q. Wang, G. Golcarenarenji, E. Chirivella-Perez, and P. Salva-Garcia, "5G-based smart ambulance: The future of the emergency service in the global pandemic and beyond," *IEEE Commun. Soc. Technol. News*, pp. 1–7, Oct. 2020. [Online]. Available: https://www.comsoc.org/publications/ctn/5g-can-shape-mission-critical-healthcare-services

[15] P. Ferreira, R. N. Miranda, P. M. Cruz, and H. S. Mendonça, "Multi-protocol LoRaWAN/Wi-Fi sensor node performance assessment for industry 4.0 energy monitoring," in *Proc. IEEE-APS Topical Conf. Antennas Propag. Wireless Commun. (APWC)*, Sep. 2019, pp. 403–407.

[16] O. Elijah, S. K. A. Rahim, M. J. Musa, Y. O. Salihu, M. J. Bello, and M.-Y. Sani, "Development of LoRa-sigfox IoT device for long distance applications," in *Proc. IEEE Nigeria 4th Int. Conf. Disruptive Technol. Sustain. Develop. (NIGERCON)*, Apr. 2022, pp. 1–5.

[17] Espressif Systems. (2023). *ESP32 System-On-Chip*. Accessed: Jul. 28, 2023. [Online]. Available: https://www.espressif.com/en/products/socs/esp32

[18] LILYGO. (2023). *LILYGO TTGO T-SIM7000G Module ESP32-WROVER-B Chip WiFi Bluetooth 18560 Battery Holder Solar Charge Development Board*. Accessed: May 27, 2023. [Online]. Available: http://www.lilygo.cn/prod_view.aspx?TypeId=50033&Id=1246

[19] N. Krishnaraj and S. Smys, "A multihoming ACO-MDV routing for maximum power efficiency in an IoT environment," *Wireless Pers. Commun.*, vol. 109, no. 1, pp. 243–256, Nov. 2019, doi: 10.1007/s11277-019-06562-0.

[20] F. Al-Turjman, B. D. Deebak, and L. Mostarda, "Energy aware resource allocation in multi-hop multimedia routing via the smart edge device," *IEEE Access*, vol. 7, pp. 151203–151214, 2019.

[21] M. Becke, H. Adhari, E. P. Rathgeb, F. Fa, X. Yang, and X. Zhou, "Comparison of multipath TCP and CMT-SCTP based on intercontinental measurements," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2013, pp. 1360–1366.

[22] R. Nozaki, L. Guillen, S. Izumi, T. Abe, and T. Suganuma, "A study on heterogeneous network switching time in IoT environments using SDN," in *Proc. IEEE 10th Global Conf. Consum. Electron. (GCCE)*, Oct. 2021, pp. 859–860.

[23] S. Ibnalfakih, E. Sabir, and M. Sadik, "Multi-homing as an enabler for 5G networks: Survey and open challenges," in *Advances in Ubiquitous Networking 2*, R. El-Azouzi, D. S. Menasche, E. Sabir, F. De Pellegrini, and M. Benjillali, Eds. Singapore: Springer, 2017, pp. 347–356.

[24] M. Pons, E. Valenzuela, B. Rodríguez, J. A. Nolazco-Flores, and C. Del-Valle-Soto, "Utilization of 5G technologies in IoT applications: Current limitations by interference and network optimization difficulties—A review," *Sensors*, vol. 23, no. 8, p. 3876, Apr. 2023. [Online]. Available: https://www.mdpi.com/1424-8220/23/8/3876

[25] E. U. Ogbodo, A. M. Abu-Mahfouz, and A. M. Kurien, "Enabling LPWANs for coexistence and diverse IoT applications in smart cities using lightweight heterogenous multihomed network model," *J. Sensor Actuator Netw.*, vol. 11, no. 4, p. 87, Dec. 2022. [Online]. Available: https://www.mdpi.com/2224-2708/11/4/87

[26] I. Chatzistefanidis, N. Makris, V. Passas, and T. Korakis, "ML-based traffic steering for heterogeneous ultra-dense beyond-5G networks," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Mar. 2023, pp. 1–6.

[27] LoRa Alliance. (2022). *LoRaWAN Specification*. Accessed: May 5, 2023. [Online]. Available: https://tinyurl.com/lwspec11

RAFAEL FAYOS-JORDAN received the B.S. degree in telematics engineering from the University of Valencia, in 2018, where he is currently pursuing the Ph.D. degree. He is also a Researcher with the School of Computing Engineering and Physical Sciences, University of the West of Scotland. His research interests include the IoT platforms and node management, cloud computing, and network management.

SANTIAGO FELICI-CASTELL received the M.Sc. and Ph.D. degrees in telecommunication engineering from the Polytechnical University of Valencia, Valencia, Spain, in 1993 and 1998, respectively. He is currently an Associate Professor with the University of Valencia, Valencia. He is also a Cisco Systems Certificated Instructor and he has authored over 40 technical papers in international journals and conferences. His research interests include networking, communication systems, and multiresolution techniques for data transmission with the quality of service.

JAUME SEGURA-GARCIA received the M.Sc. and Ph.D. degrees in physics from the University of Valencia, Valencia, Spain, in 1998 and 2003, respectively. After completing his Ph.D. degree, he was with the Robotics Institute, University of Valencia, where he was involved in several projects related to intelligent transportation systems. Since 2008, he has been with the Department of Computer Science, University of Valencia, where he is currently an Associate Professor. He has been a Visiting Researcher with multiple European research centers. He has coauthored over 90 publications in national and international journals, book chapters, and conferences. He was on the Organizing Committee of several national and international conferences, including the International Workshop on Virtual Acoustics, in 2011. He is a member of the Spanish Acoustics Society and the European Acoustics Association.

JOSE M. ALCARAZ-CALERO (Senior Member, IEEE) is currently a Professor in networks with the School of Engineering and Computing, University of the West of Scotland (UWS). He is an IEEE Senior Member with experience in 5G networks, network slicing, monitoring, automation, and management. On the academic side, he has more than 150 publications in SCI-indexed international journals, conferences, and books. He has been involved in 20 editorial activities in the most prestigious journal in the field, he has served as the chair in 20 international flagship conferences, and contributed as a technical program committee member in more than 100 international conferences. On the industrial side, he has more than 50 patents and intellectual property rights. From the leadership perspective, he has significant experience as a Principal Investigator or as Co-Investigator in more than 20 research projects at local, national, and especially at European and international levels. He is the Co-Technical Manager for EU Horizon 2020 projects SELFNET and SliceNet.

ANTONIO CERVELLO-DUATO received the degree (Hons.) in telecommunication electronic engineering and the master's degree in electronic engineering from the University of Valencia, in 2016 and 2017, respectively, where he is currently pursuing the Ph.D. degree. His research interests include significant contributions to the data acquisition path for the tile calorimeter at the ATLAS experiment within the large hadron collider (LHC) with CERN. Moreover, he is involved in the development of wireless technologies for potential application in upcoming physics experiments. He possesses a specialized skill set in hardware and firmware design, which he leverages in his various academic and research endeavors.

B VentQsys: Low-cost open IoT system for monitoring in classroom

Wireless Networks (2021) 27:5313–5327
https://doi.org/10.1007/s11276-021-02799-5

ORIGINAL PAPER

VentQsys: Low-cost open IoT system for CO_2 monitoring in classrooms

Rafael Fayos-Jordan[1] · Jaume Segura-Garcia[1] · Antonio Soriano-Asensi[1] · Santiago Felici-Castell[1] · Jose M. Felisi[2] · Jose M. Alcaraz-Calero[3]

Accepted: 6 September 2021 / Published online: 18 October 2021

Abstract

In educational context, a source of nuisance for students is carbon dioxide (CO_2) concentration due to closed rooms and lack of ventilation or circulatory air. Also, in the pandemic context, ventilation in indoor environments has been proven as a good tool to control the COVID-19 infections. In this work, it is presented a low cost IoT-based open-hardware and open-software monitoring system to control ventilation, by measuring carbon dioxide (CO_2), temperature and relative humidity. This system provides also support for automatic updating, auto-self calibration and adds some Cloud and Edge offloading of computational features for mapping functionalities. From the tests carried out, it is observed a good performance in terms of functionality, battery durability, compared to other measuring devices, more expensive than our proposal.

Keywords IoT · WSN · Spatial statistics · CO_2 · Open-source · Sustainability

Abbreviations

TVOC	Total Volatile Organic Compounds
RH	Relative humidity
NDIR	Non-Dispersive InfraRed
UART	Universal asynchronous receiver-transmitter
ETSE	Escola Tècnica Superior d'Enginyeria (Engineering Technical School)
SARS-COV-2	Severe Acute Respiratory Syndrome COronaVirus 2
MCU	Micro-controller unit
EPROM	Erasable Programmable Read-Only Memory
IoT	Internet of Things
RPi	Raspberry Pi
Wi-Fi	Wireless Fidelity (IEEE 802.11)
OTA	Over the air
OLED	Organic light-emitting diode
RGB	Red-Green-Blue
COPD	Chronic Obstructive Pulmonary Disease
WSN	Wireless sensor network
LTE	Long term evolution
BLE	Bluetooth low energy

✉ Jaume Segura-Garcia
jsegura@uv.es

Rafael Fayos-Jordan
rafajor@uv.es

Antonio Soriano-Asensi
ansoras@uv.es

Santiago Felici-Castell
felici@uv.es

Jose M. Felisi
josemanuel@g-agua.com

Jose M. Alcaraz-Calero
jose.alcaraz-calero@uws.ac.uk

[1] Computer Science Department, Escola Tècnica Superior d'Enginyeria, Universitat de València, Burjassot 46100, Spain

[2] G-Agua (Tecnologia de la Gestion del Agua), SNLE, Riba-roja de Túria 46190, Spain

[3] School of Computing, Engineering and Physical Sciences, University of the West of Scotland, PA1 1LU Paisley, United Kingdom

1 Introduction

Carbon dioxide (CO_2) is a colorless and odorless gas. It is naturally found in ambient air in concentrations ranging from 300 ppm to 550 ppm, depending on whether we measure in rural or urban environments. It is produced by (human and animal) breathing and burning fossil fuels. In the atmosphere, this gas produces the displacement of oxygen and in high concentrations (over 30,000 ppm), it can produce rapid breathing, confusion and asphyxiation, by reducing the oxygen concentration below 20% [7].

Actually CO_2 is a great indicator of air quality, since it acts as a whistle blower for air renewal. It is known that from concentrations of more than 800 ppm in working environments, complaints due to odors begin to occur. However, the usual levels that we can find in an indoor environment will be related to different variables that affect this factor, such as outdoor air levels, indoor sources, occupancy levels and ventilation rates.

In addition, in the educational sector, high CO_2 concentrations are a source of nuisance for students, due to closed rooms, lack of ventilation or circulatory air in classrooms [24] and they can affect the performance of the students. The educational authorities show some concerns, because with this situation not only the performance is compromised but also their health. Several studies have shown some relationship between the environmental pollution levels, in particular those related to CO_2, and some breathing diseases: hypercapnia, Chronic Obstructive Pulmonary Disease (COPD), etc. [7, 11].

Although there are some sensors able to monitor CO_2 in indoor environments, their prices range from 100 to 300, and their networking abilities are limited to collect data (they only allow download data in csv or pdf format in the measurement device). In this work, our goal is to develop a configurable low-cost open-hardware and open-software IoT system (with nodes costing less than 70) to measure CO_2 concentration by using a Non-Dispersive Infrared Absorption Spectroscopy (NDIR) sensor, measuring T and RH as well. Also, we aim to improve the performance of the system by adding different functionalities for upgrading [20], auto-self networked calibration, as well as include some computional offloading capabilities to Edge and Cloud for advance mapping functions.

This paper is structured as follows. First this introductory section has introduced the problem, explaining the context, the goal and after a section with some related works. Then, the materials and methods section explains the design and implementation of the node, as well the architecture of the IoT system to interconnect these nodes with external tools for control and management. The following section explains the measurement sessions done in different environments and the energy consumption performance evaluation done with the sensing node, discussing also these results. Finally, the conclusions section explains the lessons learnt and summarizes the innovations introduced.

2 Related work

In [28], authors measure CO_2 concentration in different schools in Serbia. Their measurements exceed recommended concentrations, as over 800 ppm can generate sick building syndrome [27]. This syndrome is described associated to different symptoms as: headache; eye, nose, or throat irritation; dry cough; dry or itchy skin; dizziness and nausea; difficulty in concentrating; fatigue; and sensitivity to odors.

In [16], authors found a correlation between the SARS virus spread and the sick building syndrome. Also in the SARS-COV-2 pandemic context, an increasing concern is raising about the virus spread, related to the fact that the probability of infection increases in indoor environments. This probability is proportional to the CO_2 concentration [3, 15] and it is inversely related to the amount of ventilation. As ventilation is the air renovation, i.e. the exchange of potentially contaminated indoor air with outdoor air (theoretically free of virus), it allows the elimination of particles in suspension, which potentially can contain virus, and on its possible pathways [4]. This concern has evolved into recommendations, given by some Governments and research institutions [13, 18], for ventilation in school classrooms and indoor public places in order to reduce the probability of COVID-19 infections. This assertion is supported by [30], where the author concludes that indoor air quality control strategies can be integrated to reduce the risk of SARS-COV-2 infection.

The state of the art in environmental pollution issues is wide and many applications related to Wireless Sensor Networks (WSN) have been deployed. In [19], a survey on different applications of these networks is shown for real-time ambient air pollution monitoring and air quality in metropolitan areas. It must be noticed that these WSNs have been applied in different scenarios during the last two decades for different issues, such as lightning strike detection [17], or soundscape monitoring [21], etc.

It is worth mentioning that among the different aspects involved in this kind of systems for air quality monitoring, both the sensing part and the networking part are the predominant ones. On one hand, focusing on the sensing part, in [9], the authors carry out a performance evaluation of a number of low-cost consumer grade monitors and single-parameter sensors in detecting five indoor environmental parameters: particulate matter (PM or particle pollution), CO_2, Total Volatile Organic Compounds (TVOC), dry-bulb air (T) and Relative Humidity (RH). Their study shows that technological advancements have raised an opportunity for more effective indoor air quality control and management, suggesting that most of the tested monitors have the potential to be used to secure adequate indoor environments by triggering the right chain of actions.On the other hand, focusing on the networking part for environmental monitoring, in [5] are shown optimal WSN deployment models for air pollution monitoring. In [22], the authors describe a system for air quality monitoring in different cities. In this case, the authors, use ThingSpeak to collect data in the Cloud, using different communication

technologies in the WSN (i.e. Zigbee, Wifi, LTE and BLE). Also in [14], the authors develop a client-server system with LTE communicaton and with a set of sensors for measuring PM2.5 and PM10 (PM2007), VOCs, CO, CO_2, temperature/humidity.

In [23], the authors develop a system for air quality monitoring and temperature (T) in classrooms, based on Z-Wave technology sending information to a central gateway. They study the well-being of pupils as it depends on indoor environmental quality and thermal comfort. Finally, in [2], the authors focus on Cloud computing and artificial intelligence developments in order to assist in the air monitoring process. In Table 1, we have summarized some qualitative features comparing our solution with other solutions from the state-of-the-art.

3 Materials and methods

In this section, the materials and methods used in the development of this IoT system are explained, as well as the architecture used.

3.1 Node development

In the development of the sensing nodes, NodeMCU, a ESP8266-based platform, is used [20]. The NodeMCU is in charge of sampling and reading the CO_2 sensor outputs via UART communications with the selected CO_2 sensor, and of sending such values using WiFi-based communications. Two possible NodeMCUs models are supported in our prototype: Amica (with ESP-12E and chipset CP2102)[12] and Lolin (with ESP-12F and chipset CH340). The node is completed with a RGB led with common catode to show directly the different levels in the next scale: low or green for values up to 800ppm, medium or blue for values between 800 and 1500ppm, and finally high or red for values greater than 1500ppm. Besides an OLED (organic light-emitting diode) screen shows the values of the different parameters.

Our proposal is oriented to measure CO_2, T and RH. We have used the so-called NDIR sensor, which is the most popular tool for CO_2 monitoring, that does not require analytical grade concentration readings. A NDIR sensor is a simple spectroscopic sensor based on an infrared source (lamp) with a sample chamber (or light tube), a light filter and an infrared detector, providing high accuracy. For this reason, our selection has been made separately with a MH-Z19 [31] (or optionally, MG811 which is based on solid electrolyte cell principle) and a DHT22 [1] (with an accuracy of 0.5°C for T, ranging from −40°C to 80°C, and 2% for RH, ranging from 0 to 99.9%). In particular, the MH-Z19 sensor has a response time in less than 60 seconds with an accuracy ± (50 ppm + 5% value) in a range from 0 to 5000 ppm. Figure 1 shows the schematics of the whole node where the reader can see the interconnection between the key components. Also Fig. 2 shows both layers of the designed PCB.

Such schematics has been layout in a PCB as part of the prototyping of the device. Figure 3 shows a photo of the node using the both mentioned MCUs: Amica with CP2102 chipset and Lolin with CH340 chipset. These nodes are connected via WiFi and send the information to the collection system via REST API.

3.1.1 Calibration

NDIR sensors rely on an infrared light source and detector to measure the number of CO_2 molecules. But with aging, both the light source and the detector deteriorate, resulting in slightly lower CO_2 molecule counts, producing a drift in the readings. MH-Z19 sensor provides different options for calibration: a) auto-calibration according to the background, b) manual calibration referring to 400ppm, usually in a nitrogen environment, and c) digital zero-calibration (also referring to 400ppm), using fresh air. These calibration options have different pros and cons. In this case, the

Table 1 Qualitative comparison of the proposed solution with other solutions

Work	Sensors	Technology	OTA update	Openness
[5]	Air pollution	WiFi	No	No
[9]	Particulate matter, CO2, TVOC, Air Temp and Rel.Hum	WiFi	No	No
[14]	PM2.5/PM10, VOCs, CO, CO2	LTE	No	No
[19]	Electrochemical (CO2, NO2, SO2, PM, NH3 and toxic gases)	WiFi	No	No
[22]	Air pollution	Zigbee, WiFi, LTE and BLE	No	No
Our proposal	CO2, Temp, Rel. Hum	WiFi	Yes	Yes

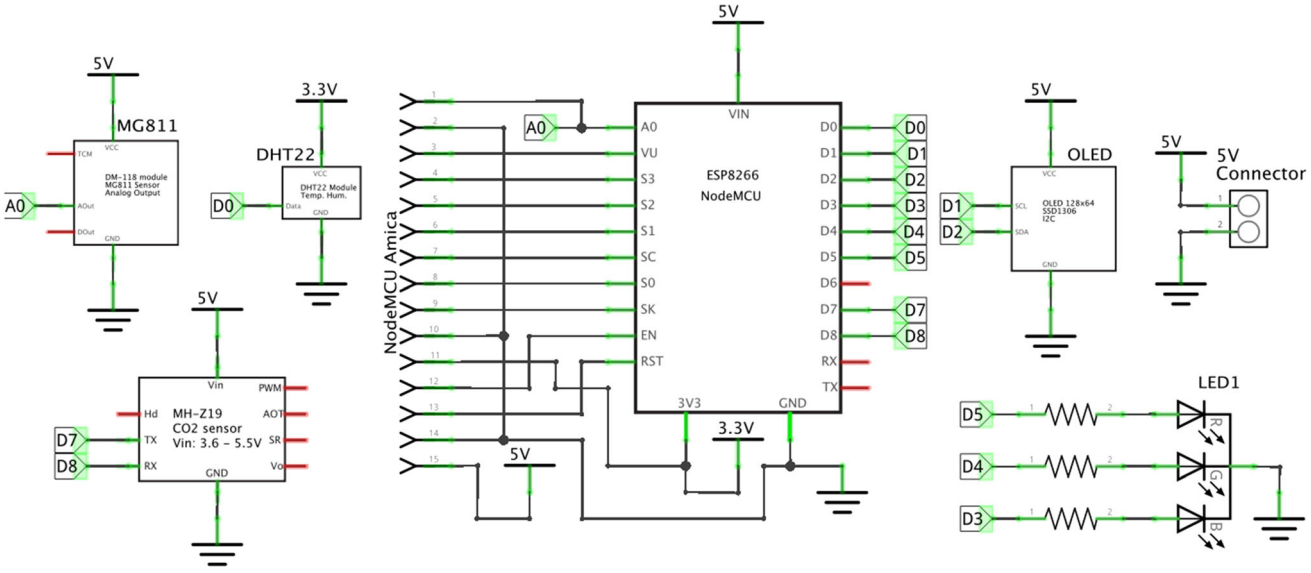
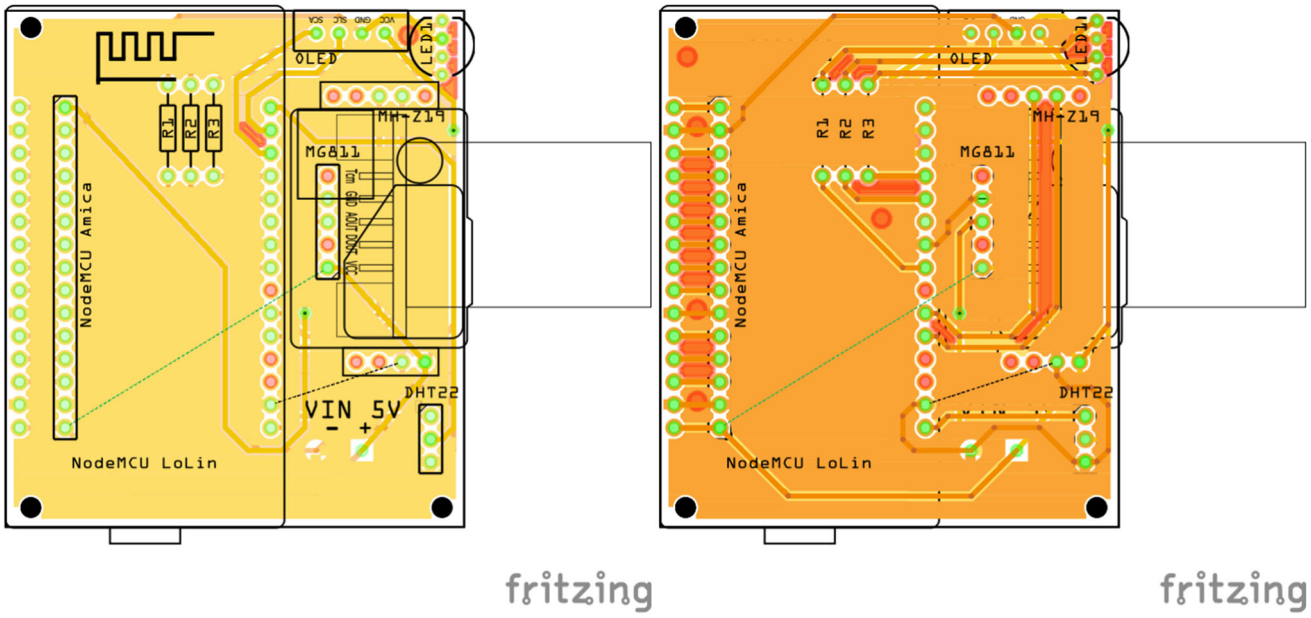


Fig. 1 Electronic schematic of the CO_2, T, RH node



(a) Front layer **(b)** Back layer

Fig. 2 Layers of the PCB

simplest one is the first one (a), based on the background, at the cost of a lower accuracy (around 50ppm plus 5% of the measurement value) that is a negligible amount in our case. This method is based on the fact that in a common environment, CO_2 levels come back to 400ppm when there is no CO_2 production for few hours, when the classroom is empty or during the night. That is because when there is nobody for a period of time of several hours, CO_2 levels drop to a minimum. Also, we have considered option (c) or manual calibration, in which the node is located in an environment with very low levels of CO_2, such as open spaces, far away from contamination, taking this reference as 400ppm. For this last option (c), optionally, the nodes have been designed to work with batteries, enabling the movement to suitable places to perform this calibration.

Finally it is worth mentioning that to combat the sensor drifts, during calibration of the sensor, multiple readings are taken. Then an average of these readings is calculated and the difference (or offset) between the new reading and the original reading when the sensor was originally

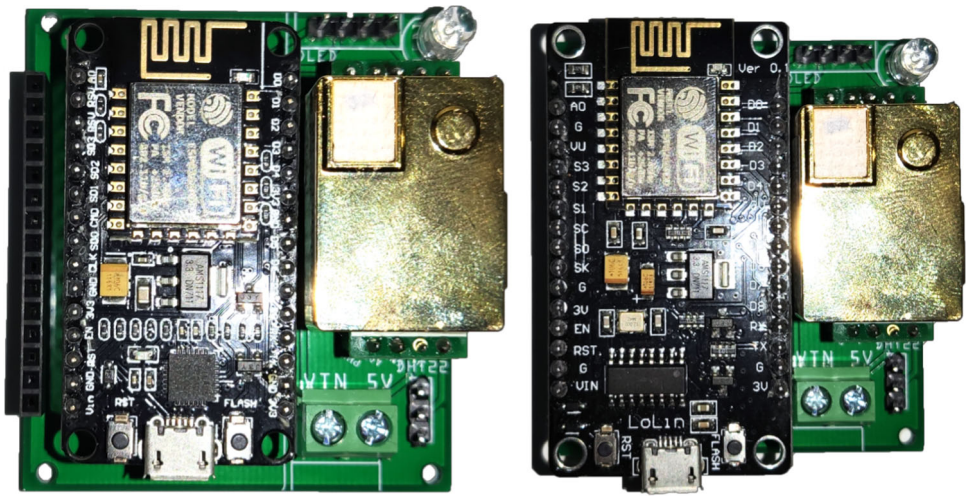


Fig. 3 Photograph of the CO_2, T and RH node with Amica and Lolin optional MCU

calibrated at the factory is stored in EPROM memory. Thus, this"offset"value is then automatically added or subtracted to later readings. This calibration is made via software.

In summary, our nodes combine a powering system, based on batteries (4xAA Energizer Max), and a software library to control the calibration options available for our NDIR CO_2 sensor. Figure 4 shows a photograph with a mobile App deployed with the calibration functions programmed for this purpose. This photo was done while calibrating one node. For the calibration procedure, we need to connect the APP to the node and then set the low level (p.e. 400 ppm).

3.2 Architecture of the system

Figure 5 shows the overall system architecture proposed. The IoT Network is the segment where all the IoT nodes are deployed, in our case the classroom or place being monitored. Each IoT node is exposing an HTTP Server Rest API, used to perform the calibration of the sensors using our mobile application previously described. The IoT node has also a REST HTTP Client used to perform a periodic submission of the monitored information. The IoT nodes are connecting to a RPi via WiFi, sending information of CO_2, T and RH. The RPis are deployed in the enterprise network segment acting as a gateway between the IoT network and the Internet. Notice that the RPis are considered optional devices in our infrastructure. They have been intentionally included to allow IoT devices to be deployed in the desired locations without the requirement to have direct Internet Wi-Fi coverage. However, if this limitation is acceptable for the concrete deployments (use case), the IoT devices are also capable to be directly connected to the Internet. The RPi are able to act as a gateway for a significant number of IoT devices allowing a high-dense deployment. Our experiments have successfully achieved 40 IoT devices simultaneously connected to just one RPi.

In the Internet, we are making use of a mobile edge computing architecture. Such architecture is composed by at least two different network segments worth to be explained. The Edge and Core Network segment. The Edge segment, located close to the Enterprise network segment is where we have decided to perform the deployment of our Monitoring Server. The server is exposing a Server Rest API used by all the IoT devices to upload the sensed information. Such API is in charge of storing the monitored information into a MySQL Database. The same Server Rest API can be used to retrieve the information from a PC or smart phones application. In addition, this information can be used as an input to a control application towards the automation of the ventilation system via machine-to-machine communications. In the most simple scenario, this monitoring server can be located in a stand-alone server, however to deal with scalability we recommend to shift such monitoring server to the cloud and if after this we need to deal with even large scalability, when then shift to edge computing replicating the Server API and keeping load balancing and high performance optimizations for the database.

Once the information is collected from all the IoT devices, we have also developed an application that retrieved such information, allowing to show on real-time the monitored metrics. This application is currently developed by Android and has been installed in our classrooms to allow inhabitants to see the current status. The system is open-source and as such, further visualization tools can be developed to adapt to different execution environments (web, Windows, Linux, Apple Watch, iOS, etc).

As an added-value service, our architecture has another compute-intensive optional component, the Spatial

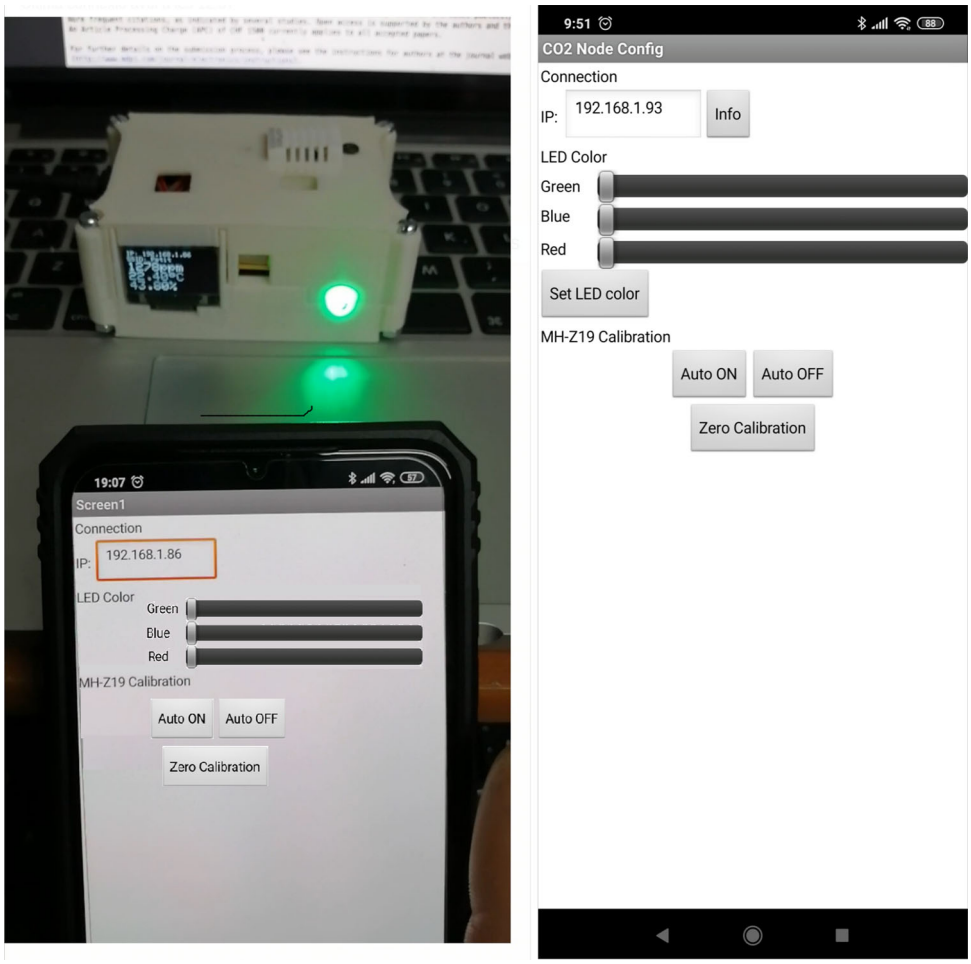


Fig. 4 Photograph with the APP for calibration and led control node

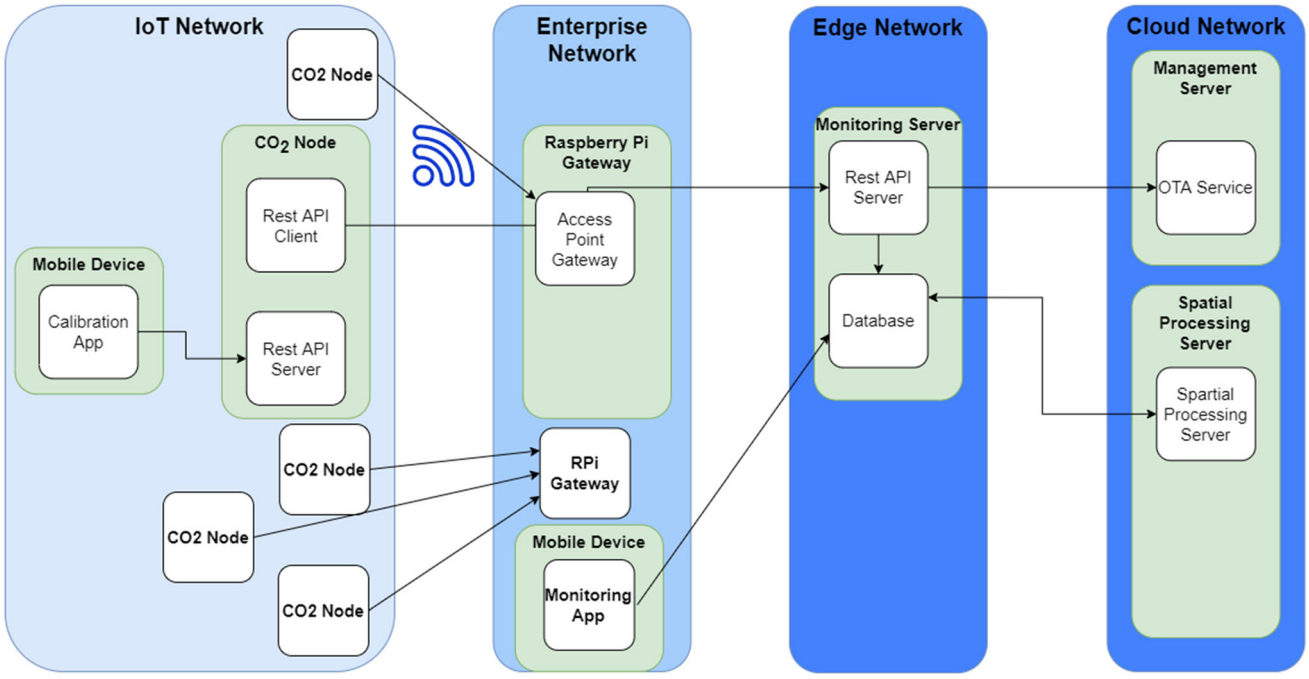


Fig. 5 Schema of the system architecture

Processing Server. This server allow us to perform spatial interpolation to allow us to estimate the level of concentration of CO_2 in every single point of the room by applying advance statistics, described in next subsection. This spatial processing server make use of the data gathered from all the IoT nodes and produces as an output the estimated interpolated value of the metrics for every of the positions of the room. Another optional added-value service compatible with our architecture is the usage of a Over-the-Air (OTA) firmware update server. This service allows us to perform the dynamic update of the firmware for all the IoT devices just in case more functionalities are pushed to the IoT devices without the need to re-deploy them. These additional compute-intensive added-value services are deployed in the Cloud to deal with scalability and compute-intensive requirements for large-scale deployments. It is worth to mention that the proposed architecture ranges from a very simple, client-server (2-layer) architecture until a 4-layer architecture able to deal with very large-scale deployments.

3.3 Spatial statistics

We have also studied the spatial statistical behavior of the information collected. To this end, we have chosen Kriging technique as a spatial interpolation method. The information from the CO_2 concentration samples establish a data set based on measurement from different and specific locations. By denoting the determined value of the CO_2 concentration measurements at a location x as $C(x)$, this data set is defined as $\{C(x), x \in \mathcal{D}\}$, where $\mathcal{D}$ are all the locations of the modelling sets, following the Kriging technique [8].

The proposed model aims to forecast the value $C(x_0)$ in any location x_0, specifically those in the validation set. The measurement reports contain information about the set of covariables included. Therefore in (1) , $C(x)$ is modeled as an average of each covariable involved in the process in the geographical area considered, plus some bounded spatial variability, which is explained by the short term process with spatial dependence.

$$C(x) = \mu(x) + \delta(x), \tag{1}$$

where $\mu(x) = \mathbb{E}[C(x)]$ and $\delta(x)$ is a stationary Gaussian process with zero mean, whose spatial dependence characterization is given by the variogram γ in (2). [10].

$$2\gamma(h) = \mathrm{Var}[C(x+h) - C(x)] = \mathrm{Var}[\delta(x+h) - \delta(x)], \tag{2}$$

where Var denotes the variance and h is an offset.

3.4 Public release

All the software and hardware presented in this paper has been released as open source available at https://github.com/ETSE-UV/VentQ. It includes, the hardware design and schematics together with the complete firmware of the IoT devices, the calibration application for the mobile application as well as the monitoring application to see the gathered metrics on real-time. The code of the monitoring server as well as the spatial processing server has been also released. For the OTA server, we are using an already existing ArduinoOTA open software[1]. The main intention is to make publicly accessible this low cost design to help controlling CO_2 concentrations. This is a true open-software and open-hardware design.

4 Results and discussion

The evaluation of the performance of our system has been done by measuring in different scenarios, with a particular interest in classrooms during examination periods. Besides, we have evaluated the energy consumption of the nodes, in order to guarantee the life of the batteries, at least during 7 or 8 hours, approximately the duration of an exam including time intervals both at the beginning and at the end.

4.1 Measurements in daily situations

Prior to perform the real tests in classrooms during exam period and in order to check the proper operation of the nodes, we decided to perform different tests during daily activities, such as monitoring CO_2 concentration in an office and in a bedroom. Notice that these two scenarios will remain with closed doors during the day.

Figure 6 a shows an office of approximately 19.8 m^3, used at least 10 hours a day by one person, turning on the heating system (by air conditioning) set to 25°C and turning it off at the end, keeping the room closed the whole time.The bedroom, shown in Fig. 6b, of 26.25 m^3, is only used for sleeping at night, by one person, turning on an electric radiator for a few minutes before going bed. Later, it remains closed the whole time, except early in the mornings when the window is opened 5 minutes for ventilation.

The results obtained in the office, as shown in Fig. 7, emphasize that the CO_2 levels are reduced to low levels after during several hours while the office is empty, even with closed doors and windows. This confirms that the self-calibration of the nodes is possible, allowing us to calibrate

[1] https://github.com/jandrassy/ArduinoOTA

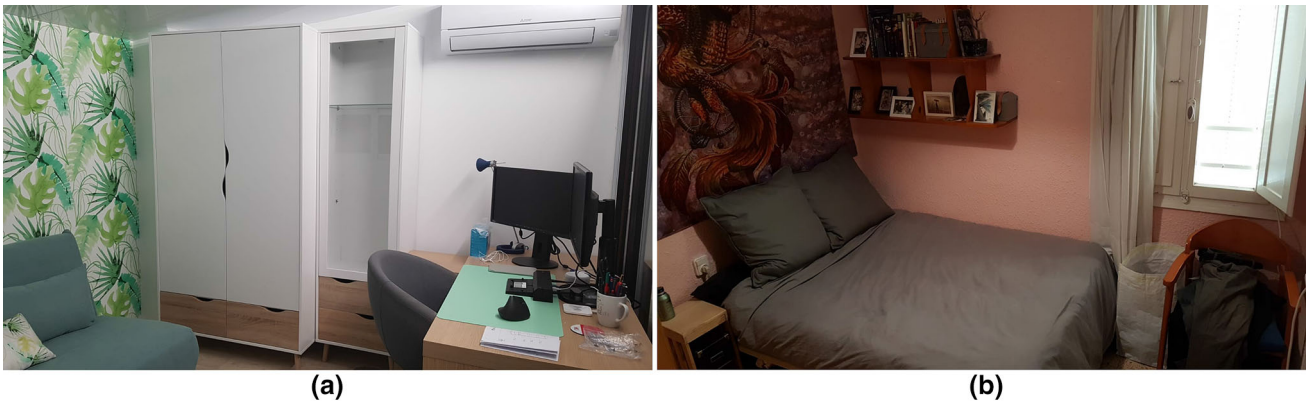


Fig. 6 Photograph of the daily scenarios analyzed

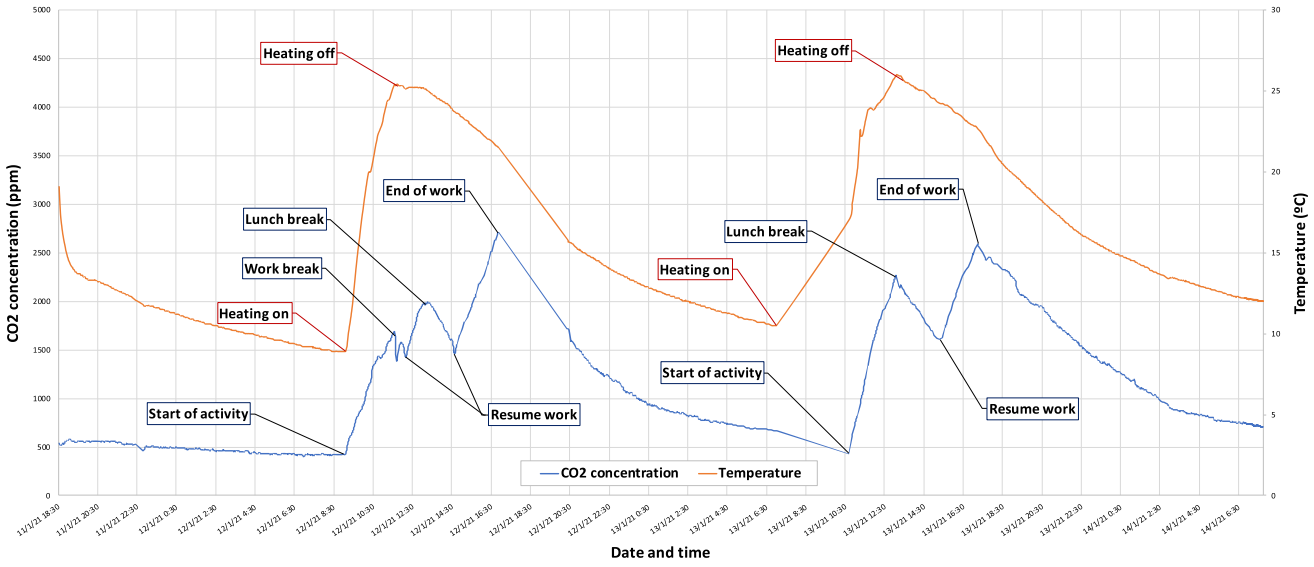


Fig. 7 CO_2 measurement in a office taken over several days

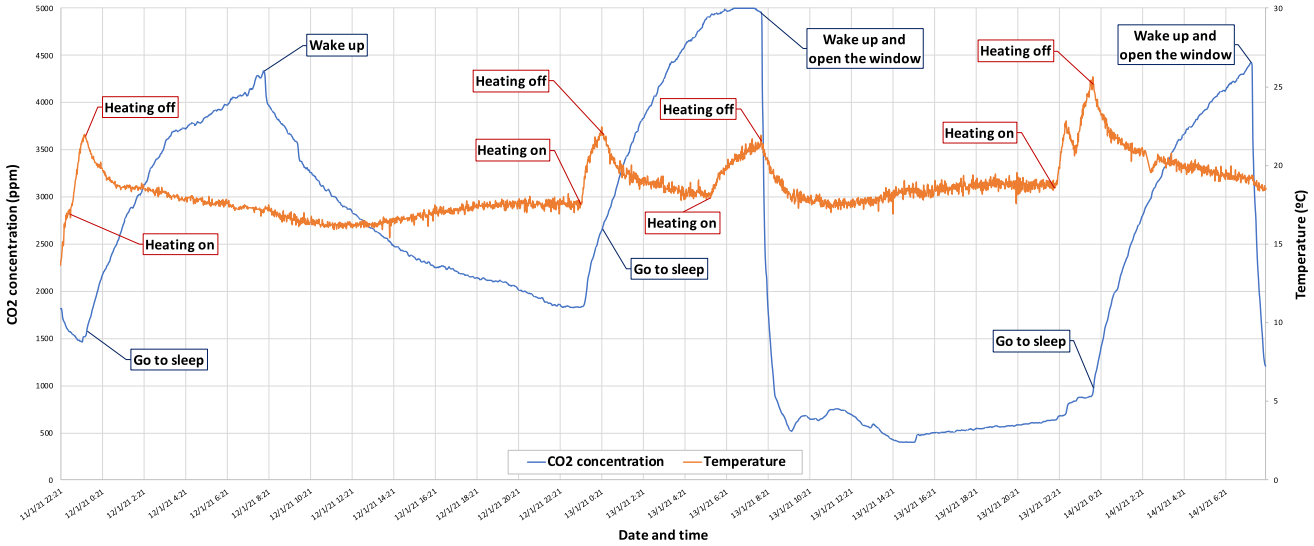


Fig. 8 CO_2 measurement in a bedroom taken over several days

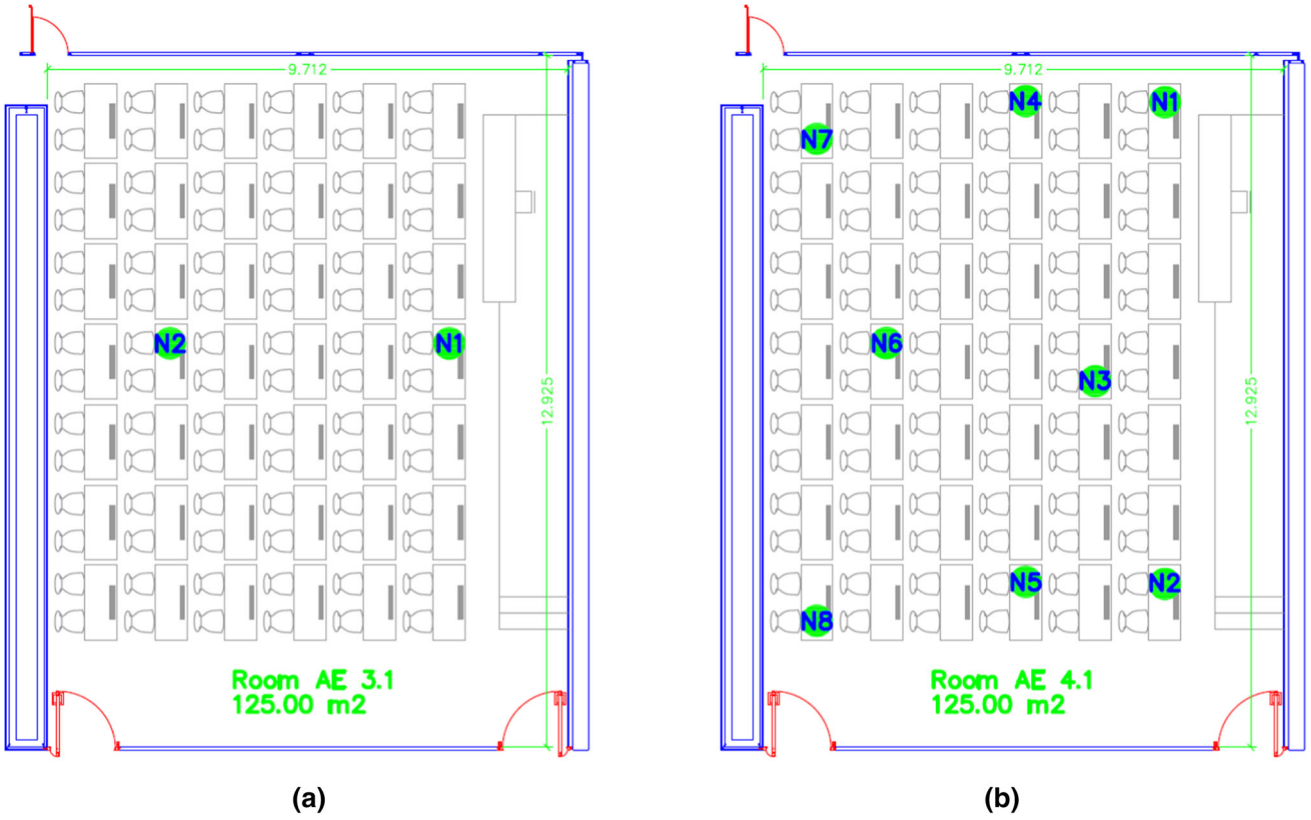


Fig. 9 Location of the nodes in classroom 3.1 (**a**) and classroom 4.1 (**b**) at the ETSE

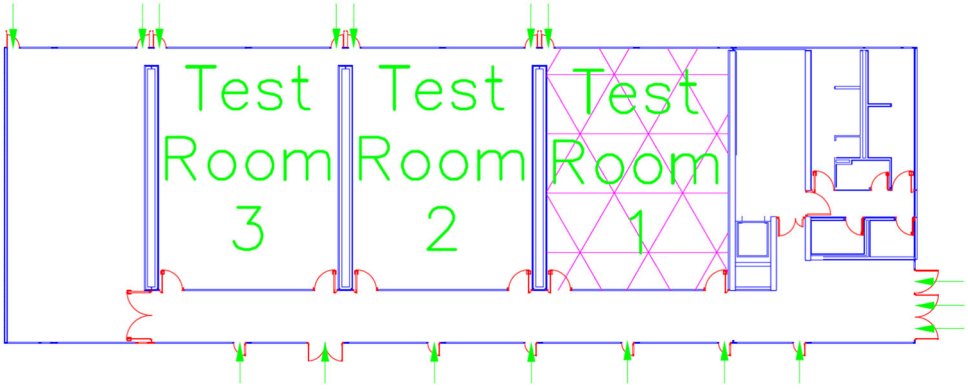


Fig. 10 External air intakes in the building at the ETSE (green arrows show the external air input to the block 4 of the building)

in the same way classrooms when they are empty. During the breaks, when the office was empty but with the heating on, CO_2 levels decreased and then increased again when the activity was resumed. This shows that this type of heating does not affect the measurements taken. Also notice that after lunch time, while being the heating system off, the increase in CO_2 levels is similar to the ones when the heating system was on.

In the case of the bedroom, it can be seen in Fig. 8 that CO_2 levels are much higher than those observed in the office. However these levels start to increase when the electric heating radiator is switched on, despite the fact that anybody remains in the room, which means that this type of heating system affects the concentration of CO_2 in the environment. After the first night, the window was not opened for ventilation, so it can be seen how CO_2 levels slowly dropped to their initial levels. However, after the second and third night, when the window was opened for ventilation, the levels dropped to a minimum in a very short time (in the order of minutes), which demonstrates how ease and quick a closed space can be adequately ventilated.

Fig. 11 Photograph of the classroom 4.1 during the test

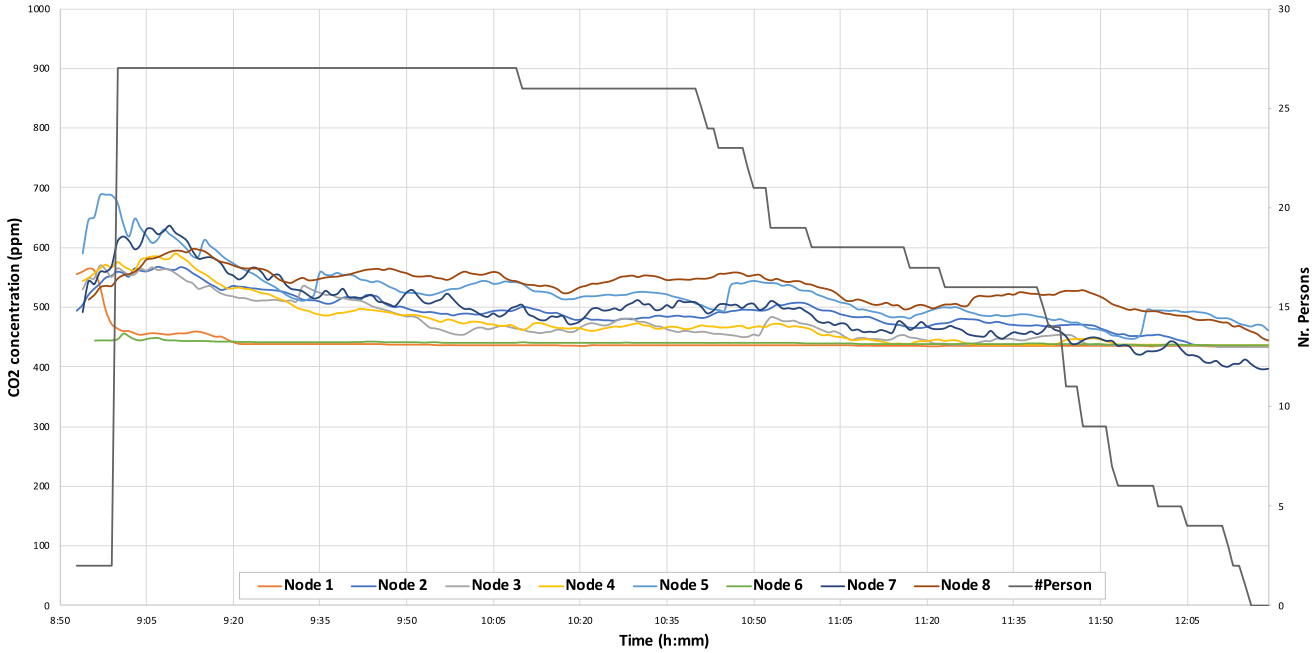


Fig. 12 CO_2 measurement in room 4.1 at ETSE during an exam in January with 8 nodes vs. number of persons in the room (doors and windows were open)

4.2 Measurements in the examination period

In order to test our system in a field evaluation, we have made some measurements during different exams in the examination period in December 2020 at room number 3.1 and January 2021 at room number 4.1 in the ETSE of the University of Valencia. Figure 9 shows the location of the nodes in the classroom 3.1 (Fig. 9a) and 4.1 (Fig. 9b).

A photograph can be seen during the measurements in Fig. 11. As these nodes are equipped with batteries, they can be located anywhere. Also, as shown in this figure, doors and windows are opened, ensuring a good ventilation, following the COVID-19 instructions.

Both classrooms are located in different buildings with identical distribution. Also, they have multitude of air inlets from the outside, as we can see in Fig. 10, marked with green arrows. These inlets promote a good ventilation inside the classroom, ensuring low levels of CO_2. The location of the classrooms inside the building is marked as Test Room 1.

In Fig. 11, we can see a photograph during an exam in January 2021. Figure 12 shows the time evolution of CO_2 concentration measured using eight different nodes. In both

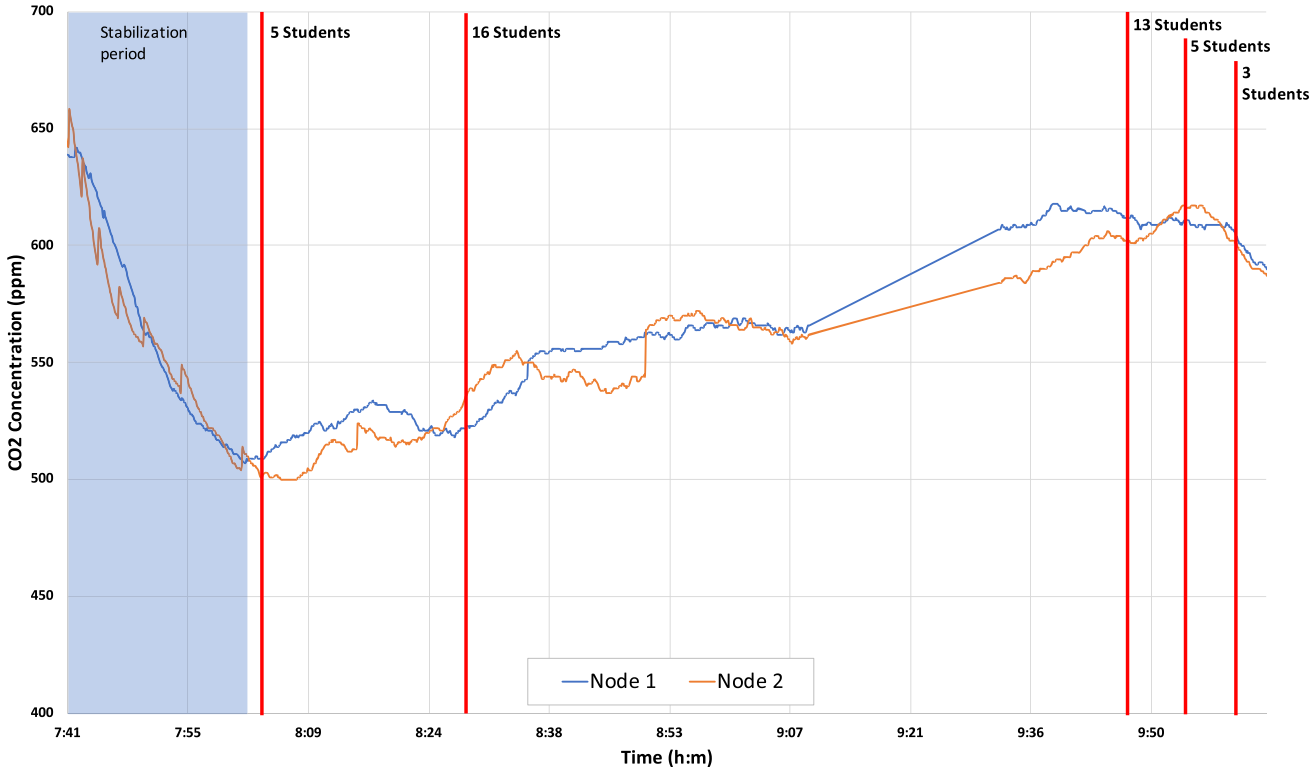


Fig. 13 CO_2 measurement in a classroom during an exam in December with 2 nodes

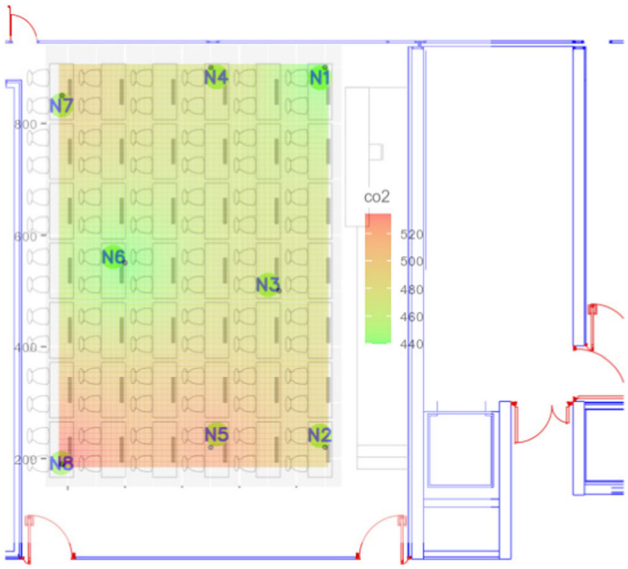


Fig. 14 Spatial statistic representation of mean values of CO_2 concentration in room 4.1 at ETSE during an exam in January with 8 nodes

cases, the natural ventilation (due to open doors and windows) allows to keep the CO_2 concentration controlled, reducing the risks of COVID-19 contagions. These results show the correct behavior of the proposed system. In addition, Fig. 13 shows a line of the evolution of CO_2 concentration from two nodes during an exam in December 2020. In this figure, we can see a slight rise of the concentration, while the number of students increases. As the room is well ventilated, the CO_2 levels are not too high and under control.

4.3 Spatial statistics

Figure 14 shows the spatial statistic study of the average of CO_2 concentration, using Kriging technique [8, 10, 21, 26] during the exam in January 2021. This method allows to predict (by means of statistical interpolation) the values inside the defined grid. In this case, we have used Ordinary Kriging [29] to compute the evolution of the spatial distribution of CO_2 concentration in the classroom. The video provided as supplementary material shows the time evolution per minute of the CO_2 concentration during the exam. It shows how well the class is ventilated during the exam

4.4 Energy consumption performance evaluation

Now, we evaluate the energy consumption of the proposed nodes with two different configurations: (a) by using the MCU with the MH-Z19 and DHT22 sensors, together with a RGB led and an OLED screen and (b) by using both

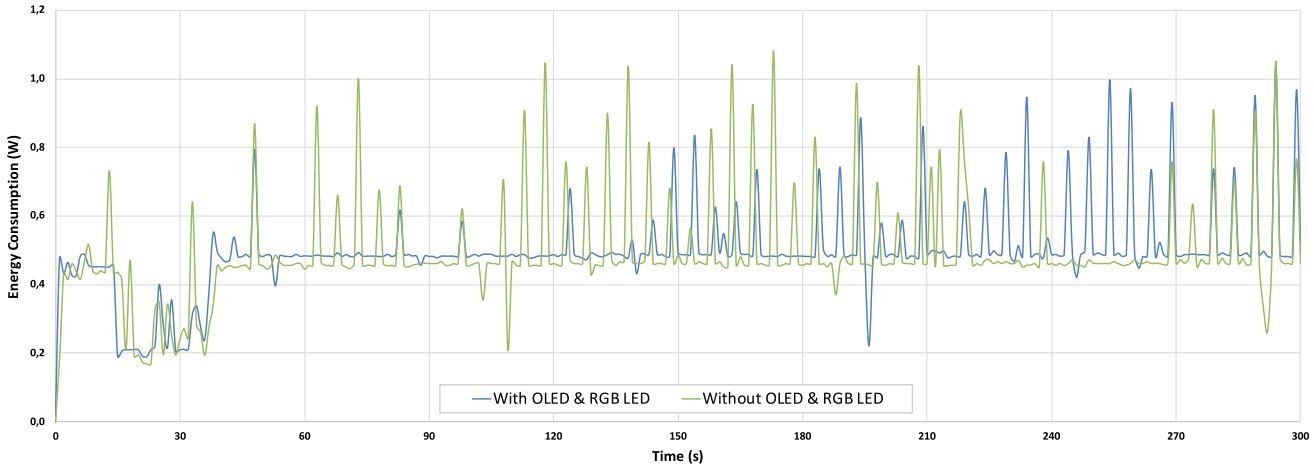


Fig. 15 Energy consumption of the measurement and communication process in the node

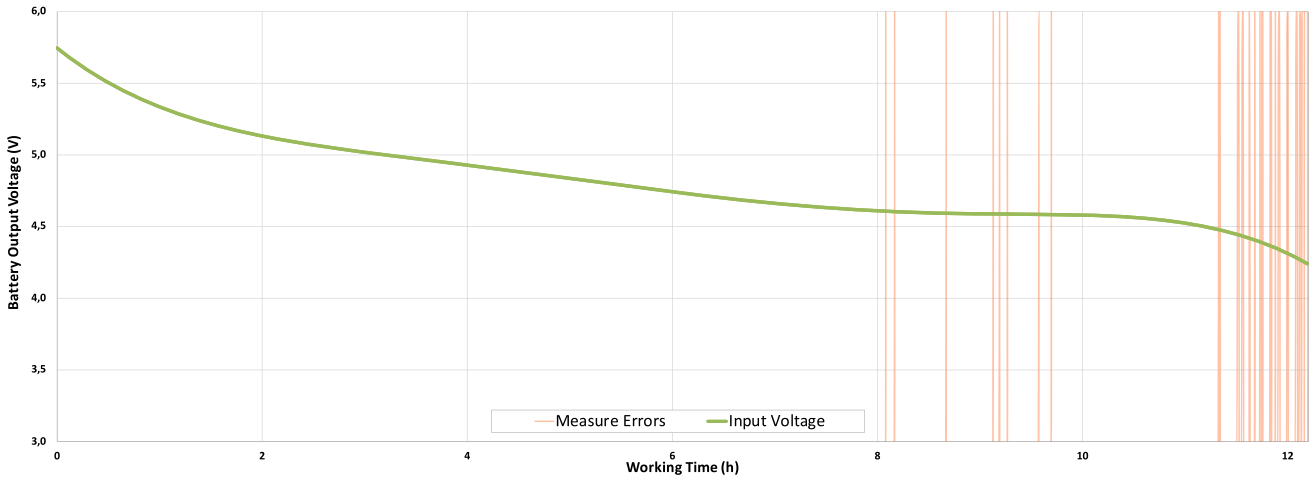


Fig. 16 Operation and duration of the node with batteries

sensors without the RGB led and the OLED screen. To this end, we use a USB power-meter UM34C [25].

Figure 15 shows the initialization of the node, with a starting sleep mode, and a measurement period in one node, with both configurations (a) in green color and (b) in blue color. We can observe from this figure that the consumption of the OLED screen and RGB led is very low, around 0.03W in average.

Also from Fig. 15, we can see different peaks corresponding to the measurement period of the MH-Z19 sensor, since the Infrared (IR) source of the NDIR switches on every 5 s during 400 ms. Therefore, we can conclude that these peaks correspond to the energy used to switch this IR lamp on. It must be noticed that these nodes have been designed to operate with 4 AA batteries for short-term measurement campaigns, in places where direct power supply is not possible. Therefore, these tests have been carried out to verify the duration of the nodes when measuring.

Energizer Max E91 type AA batteries [6] have been used and the node has been configured to take and send samples every 5 seconds with the display and RGB led on. Both the samples sent and the voltage supplied by the batteries have been monitored. Figure 16 shows the output voltage of the batteries and the errors in data readings. As it can be observed, errors start appearing sporadically after 8 hours of uninterrupted operation when the voltage drops to 4.6V and they continue working during 11 hours more, when the voltage drops to 4.5V. Finally the node stops working after more than 12 hours when the voltage drops below 4.2V.

5 Conclusions

In this work we have developed a fully operative open-hardware and open-software fully operational IoT system and architecture to measure CO_2 concentration,

temperature and relative humidity. This system is fully scalable and automatically upgradeable, thanks to the OTA updating function. The building cost of this system is low-cost, less than 100$.

This system has been tested in different environments: at work, at home and in class during different exams. The measurements taken shows full range operation (limited only by the sensor performance), allowing us to show perfectly the temporal evolution of the CO_2 concentration.

In order to study the spatio-temporal evolution of the CO_2 concentration measured in different situations, the architecture has been empowered with cloud-based processing capabilities to achieve Kriging technique to perform spatial interpolation of the metrics.

Finally, the energy consumption of the developed nodes has been also evaluated in each part of the circuit, lasting till 12 hours of continuous monitoring. As a future work, we are going to develop the node using a Fipy MCU which will allow 5G communication with these nodes. As this is a modular system that can be improved and upgraded (due to its openness). The use of other sensors such as PM3005 (for monitoring PM2.5 or PM10), TVOC and other gasses can be a matter of improvement by adding I2C, SPI or other interfaces to the sensing module. We also would like to explore ultra low power consumption real-time processors to extend the duration of the system with the same number of batteries.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s11276-021-02799-5.

Acknowledgements This work was partially funded by the European Commission Horizon 2020 5G-PPP Program under project H2020-ICT-2020-2/101016941: "5G INDUCE: OpenCooperative 5G Experimentation Platforms for The Industrial Sector NetApps" and by the Generalitat Valenciana under the grant number AEST/117/2020 and the Universitat de Valencia for the grant number UV-INV-AE-1544281.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

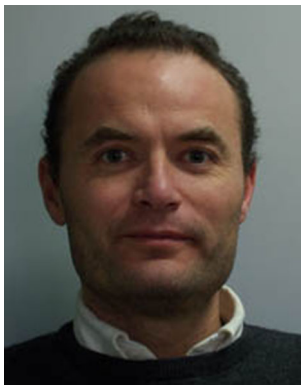
1. Adafruit Industries: DHT22 temperature & humidity sensor (2016). URL: http://www.adafruit.com/datasheets/DHT22.pdf. (Accessed: 10/01/2021).
2. Arroyo, P., Herrero, J. L., Suárez, J. I., & Lozano, J. (2019). Wireless sensor network combined with cloud computing for air quality monitoring. *Sensors (Switzerland)*. https://doi.org/10.3390/s19030691.
3. Azuma, K., Yanagi, U., Kagi, N., Kim, H., Ogata, M., & Hayashi, M. (2020). Environmental factors involved in SARS-CoV-2 transmission: Effect and role of indoor environmental quality in the strategy for COVID-19 infection control. *Environmental Health and Preventive Medicine*. https://doi.org/10.1186/s12199-020-00904-2.
4. Bhagat, R. K., Wykes, M. S. D., Dalziel, S. B., & Linden, P. F. (2020). Effects of ventilation on the indoor spread of COVID-19. *Journal of Fluid Mechanics*. https://doi.org/10.1017/jfm.2020.720.
5. Boubrima, A., Bechkit, W., & Rivano, H. (2017). Optimal WSN deployment models for air pollution monitoring. *IEEE Transactions on Wireless Communications*. https://doi.org/10.1109/TWC.2017.2658601.
6. Brands, E. Energizer e91 datasheet (2020). http://data.energizer.com/pdfs/e91.pdf (Accessed: 17/01/2021).
7. Chaabouni, S., & Saidi, K. (2017). The dynamic links between carbon dioxide (CO2) emissions, health spending and GDP growth: A case study for 51 countries. *Environmental Research*. https://doi.org/10.1016/j.envres.2017.05.041.
8. Cressie, N. (1993). *Statistics for spatial data*. New York: John Wiley.
9. Demanega, I., Mujan, I., Singer, B. C., Andelkovic, A. S., Babich, F., & Licina, D. (2021). Performance assessment of low-cost environmental monitors and single sensors under variable indoor air quality and thermal conditions. *Building and Environment*. https://doi.org/10.1016/j.buildenv.2020.107415.
10. Diblasi, A., & Bowman, A. (2001). On the use of the variogram in checking for independence in spatial data. *Biometrics, 57,* 211–218.
11. Duan, R. R., Hao, K., & Yang, T. (2020). Air pollution and chronic obstructive pulmonary disease. *Chronic Diseases and Translational Medicine*. https://doi.org/10.1016/j.cdtm.2020.05.004.
12. Espressif Systems: Nodemcu datasheet (2013). URL: https://www.elecrow.com/download/ESP8266_Specifications_English.pdf. (Accessed: 01/02/2021).
13. European Centre for Disease Prevention and Control: Heating, ventilation and air-conditioning systems in the context of COVID-19. Tech. rep., European Union, Stockholm (2020). https://www.ecdc.europa.eu/en/publications-data/heating-ventilation-air-conditioning-systems-covid-19.
14. Jo, J., Jo, B., Kim, J., Kim, S., & Han, W. (2020). Development of an IoT-based indoor air quality monitoring platform. *Journal of Sensors, 2020,* 8749764. https://doi.org/10.1155/2020/8749764.
15. Liu, Z., Ciais, P., Deng, Z., et al. (2020). Near-real-time monitoring of global CO2 emissions reveals the effects of the COVID-19 pandemic. *Nature Communications*. https://doi.org/10.1038/s41467-020-18922-7.

16. McKinney, K. R., Gong, Y. Y., & Lewis, T. G. (2006). Environmental transmission of SARS at amoy gardens. *Journal of Environmental Health, 68,* 26–52.
17. Mialdea-Flor, I., Segura-Garcia, J., Felici-Castell, S., Garcia-Pineda, M., Alcaraz-Calero, J. M., & Navarro-Camba, E. (2019). Development of a low-cost IoT system for lightning strike detection and location. *Electronics (Switzerland)*. https://doi.org/10.3390/electronics8121512.
18. Minguillón, M.C., Querol, X., Felisi, J.M.,&Garrido, T. Guía para ventilación de las aulas CSIC (2020). 10.20350/DIGITALCSIC/12677. https://digital.csic.es/handle/10261/221538.
19. Nagaraj, S., & Biradar, R.V. Applications of wireless sensor networks in the real-Time ambient air pollution monitoring and air quality in metropolitan cities-A survey. In: Proceedings of the 2017 international conference on smart technology for smart nation, SmartTechCon 2017 (2018). 10.1109/SmartTechCon.2017.8358594.
20. NodeMCU documentation: Over-the-air updating (2018). URL: https://nodemcu.readthedocs.io/en/release/build/. (Accessed: 10/01/2021).
21. Pastor-Aparicio, A., Segura-Garcia, J., Lopez-Ballester, J., Felici-Castell, S., Garcia-Pineda, M., & Perez-Solano, J. J. (2020). Psychoacoustic annoyance implementation with wireless acoustic sensor networks for monitoring in smart cities. *IEEE Internet of Things Journal*. https://doi.org/10.1109/JIOT.2019.2946971.
22. Patil, D., Thanuja, T. C., & Melinamath, B. C. (2018). Air pollution monitoring system using wireless sensor network (WSN). In V. Balas, N. Sharma, & A. Chakrabarti (Eds.), *Data management, analytics and innovation* (pp. 391–400). Singapore: Springer.
23. Perez, A. O., Bierer, B., Scholz, L., Wöllenstein, J., & Palzer, S. (2018). A wireless gas sensor network to monitor indoor environmental quality in schools. *Sensors (Switzerland)*. https://doi.org/10.3390/s18124345.
24. Petersen, S., Jensen, K. L., Pedersen, A. L., & Rasmussen, H. S. (2016). The effect of increased classroom ventilation rate indicated by reduced CO2 concentration on the performance of schoolwork by children. *Indoor Air*. https://doi.org/10.1111/ina.12210.
25. Ruideng. UM34C User Manual (2020). http://ruidengkeji.com/inst/UM34C.pdf (Accessed: 10/01/2021).
26. Segura-Garcia, J., Navarro-Ruiz, J., Perez-Solano, J., Montoya-Belmonte, J., Felici-Castell, S., Cobos, M., & Torres-Aranda, A. (2018). Spatio-temporal analysis of urban acoustic environments with binaural psycho-acoustical considerations for IoT-based applications. *Sensors, 18,* 690.
27. Seppänen, O. A., Fisk, W. J., & Mendell, M. J. (1999). Association of ventilation rates and CO2 concentrations with health and other responses in commercial and institutional buildings. *Indoor Air, 9*(4), 226–252. https://doi.org/10.1111/j.1600-0668.1999.00003.x.
28. Turanjanin, V., Vučeć, B., Jovanović, M., Mirkov, N., & Lazović, I. (2014). Indoor CO2 measurements in Serbian schools and ventilation rate calculation. *Energy, 77,* 290–296. https://doi.org/10.1016/j.energy.2014.10.028.
29. Wackernagel, H. (2003). *Ordinary kriging* (pp. 79–88). Berlin: Springer.
30. Zhang, J. (2020). Integrating IAQ control strategies to reduce the risk of asymptomatic SARS CoV-2 infections in classrooms and open plan offices. *Science and Technology for the Built Environment, 26*(8), 1013–1018. https://doi.org/10.1080/23744731.2020.1794499.
31. Zhengzhou Winsen Electronics Technology Co. Ltd. MH-Z19B NDIR CO2 sensor for indoor air quality monitoring–Winsen (2018). https://go.uv.es/dulGL49, (Accessed: 20/01/2021).

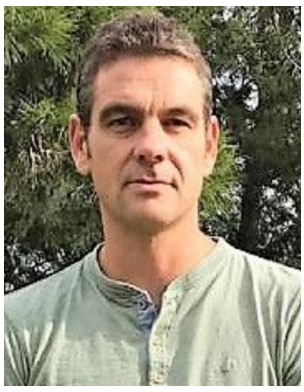
Rafael Fayos-Jordan received the B.S. Degree in Telematics Engineering from the University of Valencia in 2018. He is currently in the last year of master degree in Web Technology, Cloud Computing and Mobile Applications and working as researcher in Department of Computer Science, University of Valencia. His interest is in the IoT platforms and node management, cloud computing and network management.

Jaume Segura-Garcia received the M.Sc. and Ph.D. degrees in physics from the University of Valencia, Valencia, Spain, in 1998 and 2003, respectively. After completing his Ph.D. study, he was with the Robotics Institute, University of Valencia, where he was involved in several projects related to intelligent transportation systems. Since2008, he has been with the Department of Computer Science, University of Valencia, where he is currently an Associate Professor. He has been a Visiting Researcher with multiple European research centres. He has co-authored over 90 publications at national and international journals, book chapters, and conferences. He was on the Organizing Committee of several national and international conferences, including the International Workshop on Virtual Acoustics (2011). He is a member of the Spanish Acoustics Society and the European Acoustics Association.

Antonio Soriano-Asensi was born in Benaguasil, Valencia, Spain, in 1978. He received the Licenciado degree in physics and the Ph.D. degree from the University of Valencia, Valencia, in 2001 and 2007, respectively. He has worked as a Postdoctoral Research at the Spanish National Researh Council (CSIC), the Universitat Politecnica de Valencia, and the Universitat Jaume I. Since 2017, he has been an Associate Professor with the Department of Informatics, University of Valencia. His research interests include computer networks and robotics.



Santiago Felici-Castell received the M.Sc. and Ph.D. degrees in telecommunication engineering from the Polytechnical University of Valencia, Valencia, Spain, in 1993and 1998, respectively. He is currently an Associate Professor with the University of Valencia, Valencia. His research has focused on networking, communication systems, and multiresolution techniques for data transmission with quality of service. He is also a Cisco Systems Certificated Instructor, and has authored over 25 technical papers in international journals and conferences.



Jose M. Felisi was born in Villar, Valencia, in 1976. He received his Bachelor decgree in Chemistry Engineering in 2002. He is manager of the company G-Agua since its foundation in 2006. He is also leading the Mesura association, devoted to obtaining and processing data from many sources in different parts of the Valencia Community and applying open-science principles. His main interests are oriented to chemical processes, environmental monitoring, sensoring and IoT.

Jose M. Alcaraz-Calero (Senior Member, IEEE) received the Ph.D. degree in computer Science from the University of Murcia. He is currently a Full Professor in next-generation networks and security with the University of the West of Scotland. He is the Technical Co-Coordinator of the EU H2020 5G-PPP SELFNET and SliceNet projects and a Co-Principal Investigator of the EU H2020 5G INDUCE and 6G BRAINS projects. His professional interests include network cognition, management, security and control, service deployment, automation and orchestration, and 5G mobile networks.

C ECO4RUPA: 5G-IoT Inclusive and Intelligent Routing Ecosystem with Low-Cost Air Quality Monitoring

information

Article

ECO4RUPA: 5G-IoT Inclusive and Intelligent Routing Ecosystem with Low-Cost Air Quality Monitoring

Rafael Fayos-Jordan [1], Raquel Araiz-Chapa [2], Santiago Felici-Castell [2,*], Jaume Segura-Garcia [2], Juan J. Perez-Solano [2] and Jose M. Alcaraz-Calero [1]

[1] School of Computing, Engineering and Physical Sciences, University of the West of Scotland, Paisley PA1 2BE, UK; rafael.fayos@uws.ac.uk (R.F.-J.); jose.alcaraz-calero@uws.ac.uk (J.M.A.-C.)
[2] Computer Science Department, ETSE, Universitat de València, 46100 Burjassot, Spain; achara@alumni.uv.es (R.A.-C.); jaume.segura@uv.es (J.S.-G.); juan.j.perez@uv.es (J.J.P.-S.)
* Correspondence: felici@uv.es; Tel.: +34-96-3543-563

Abstract: The increase and diversity of low-cost air quality (AQ) sensors, as well as their flexibility and low power consumption, offers us the opportunity to integrate them into broad AQ wireless sensor networks, with the aim of enabling real-time monitoring and higher spatial sampling density of pollution in all parts of cities. Considering that the vast majority of the population lives in cities and the increase in respiratory/allergic problems in a large part of the population, it is of great interest to offer services and applications to improve their quality of life by avoiding pollution exposure in their movements in the open air. In the ECO4RUPA project, we focus on this kind of service, proposing an inclusive and intelligent routing ecosystem carried out using a network of low-cost AQ sensors with the support of 5G communications along with official AQ monitoring stations, using spatial interpolation techniques to enhance its spatial resolution. The goal of this service is to calculate healthy walking and/or cycling routes according to the particular citizen's profile and needs. We provide and analyse the results of the proposed route planner under different scenarios (different timetables, congestion road traffic, and routes) and different user profiles, with a special interest in citizens with asthma and pregnant women, since both have special needs. In summary, our approach can lead to an approximately average reduction in pollution exposure of 17.82% while experiencing an approximately average increase in distance travelled of 9.8%.

Keywords: air quality; low cost sensors; IoT network; WSN; deployment

Citation: Fayos-Jordan, R.; Araiz-Chapa, R.; Felici-Castell, S.; Segura-Garcia, J.; Perez-Solano, J.J.; Alcaraz-Calero, J.M. ECO4RUPA: 5G-IoT Inclusive and Intelligent Routing Ecosystem with Low-Cost Air Quality Monitoring. *Information* **2023**, *14*, 445. https://doi.org/10.3390/info14080445

Academic Editors: Maria Luisa Villani, Rita Zgheib and Antonio De Nicola

Received: 9 July 2023
Revised: 28 July 2023
Accepted: 31 July 2023
Published: 7 August 2023

1. Introduction

Urbanisation has meant that three-quarters of Europe's population now lives in cities. Citizens are constantly confronted with levels of air pollution that violate the safe thresholds for human health defined by the World Health Organization (WHO) [1], generally caused by the natural dynamics of the movement of people and the pollution associated with such transport.

According to Eurostat [2], in 2021, there were 369,000 deaths in the EU resulting from diseases of the respiratory system, equivalent to 7.9% of all deaths in the EU-28. The BBC reported [3] that "around 422,000 people died prematurely in European countries in 2018 due to exposure to harmful levels of fine particulate matter PM2.5". Moreover, the problem is worse if we consider that a large part of the population has or may have some kind of allergy, respiratory, and skin problem [4]. There is an increasing number of allergens that increase allergic problems, asthma, as well as other respiratory and skin problems [5].

In this scenario, wireless sensor networks (WSN) or Internet of Things (IoT) sensor networks for monitoring air quality (AQ) based on low-cost sensors and supported by 5G technologies, together with artificial intelligence (AI) techniques [6,7], along with official AQ monitoring stations, can help citizens in their day-to-day lives by means of a system that

looks after their health when they are on the move, especially when they have respiratory and/or allergy problems. These activities are carried out within the ECO4RUPA project.

To this end, the goal of this paper is to propose an inclusive and intelligent routing ecosystem with the objective of calculating healthy routes according to the profile and particular needs of each citizen (which include pathologies and clinical history) in their outdoor movements, assisted by a real-time AQ monitoring network within an IoT paradigm. In case we do not have a specific profile, as a user requirement, we will use a default one that will try to minimize global pollution exposure.

Notice that, at the international level, the air quality is ruled by ISO 11771:2010 [8] and ISO 37122:2019 [9] according to the European Regulation Directive 2008/50/EC [10], which states that cities with more than 2 million inhabitants must have at least one monitoring station for AQ. Thus, this monitoring network is supported by publicly available data from official AQ monitoring stations for polluting gases (the network of stations of the Generalitat Valenciana [11]) as well as other stations managed by the local councils, such as in Valencia city [12]. In Figure 1a, we show an example of an official AQ monitoring station, in particular from Burjassot (outskirts of Valencia, Spain). All this gathered information is also improved with statistical techniques of spatial inference to enhance the spatial resolution of these pollutants over the city maps. In areas with poor official AQ coverage, we deploy additional ECO4RUPA AQ monitoring nodes, as shown in Figure 1b,c with outdoor and indoor versions, respectively.

(**a**) Official AQ monitoring station

(**b**) Outdoor ECO4RUPA AQ IoT node

(**c**) Indoor ECO4RUPA AQ IoT node

Figure 1. Example of (**a**): official air quality (AQ) monitoring station in Burjassot (outskirts of Valencia, Spain) [11], (**b**): ECO4RUPA outdoor AQ IoT node, and (**c**): ECO4RUPA indoor AQ IoT node.

The AQ index scale is based on the US-EPA 2016 standard and is classified into six categories, given by different ranges and colours, as follows: range [0–50] as good (green), [51–100] as moderate (yellow), [101–150] as unhealthy for sensitive groups (orange), [151–200] as unhealthy (red), [201–300] as very unhealthy (purple), and more than 300 as hazardous (dark red). In Figure 2, we show a map with these official AQ monitoring stations, along with an indicator of the AQ index. In this case, all the different stations were reasonably good by the time they were queried. In Figure 3, we show, at a higher scale, the AQ index for the Valencia community region [13]. It must be stressed that in Figure 3, the pollution is mainly due to ozone, O_3, which is a secondary pollutant derived from the combustion of fossil fuels.

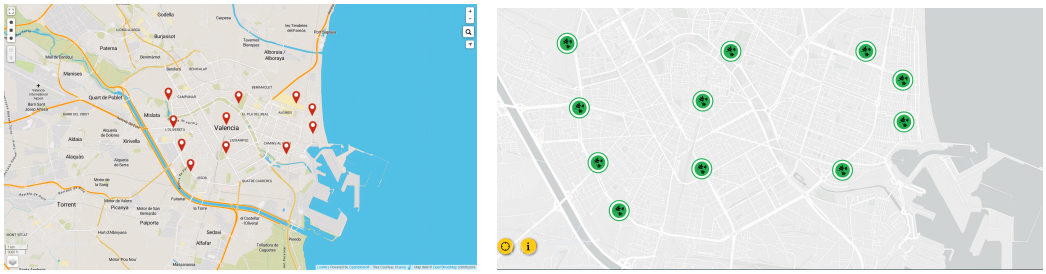


Figure 2. Official air quality (AQ) monitoring network in Valencia city [12] and surroundings; (**a**) location of the official AQ monitoring stations; (**b**) AQ colour index based on US-EPA 2016 standard.

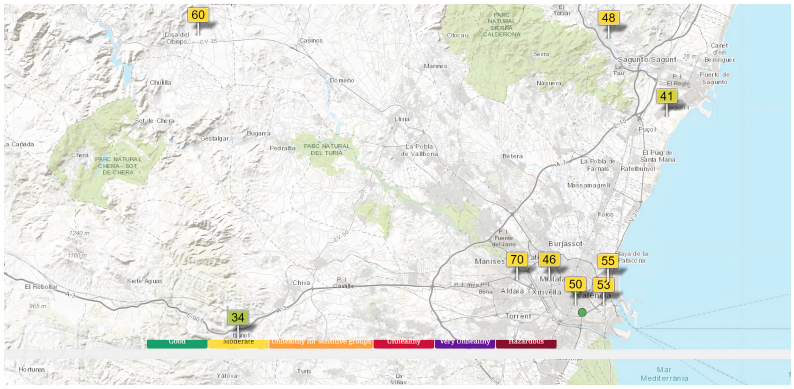


Figure 3. Map of the air quality for the Valencia community region [13].

The rest of the paper is structured as follows. In Section 2, we show available AQ sensors and the related work. In Section 3, we analyse the different design alternatives to be used in the broad monitoring network and its architecture. In Section 4, we consider the options to integrate and merge the information obtained for the route planner. In Section 5, we present and discuss the results with different users' profiles. Finally, in Section 6, we summarise the main conclusions and future work.

2. State of the Art

With regard to AQ monitoring, it should be noted that there are three distinct types of monitoring, at least, based on the different types of gases. That is, greenhouse gases (under control by emissions monitoring), chlorofluorinated gases (analysed in the upper layers of the atmosphere), and pollutants, which include nitrogen dioxide (NO_2), sulfur dioxide (SO_2), carbon monoxide (CO), ozone (O_3), as well as benzols and heavy metals (lead (Pb), arsenic (As), and cadmium (Cd)). From these areas, the most relevant for citizens is the last one, the pollutants, most of which come from the combustion of fossil fuels in the city and for which there are regulations and standards for their control, such as Directive 2008/50/EC.

With concern to pollutants, the recent boom in low-cost AQ sensors, due to their ease of installation and low power consumption, makes them increasingly used and interesting to integrate into WSN. These sensors can measure pollutants, such as the ones noted before, as well as volatile organic compounds (VOC, usually measured in totals, TVOC), particulate matter (PM) concentration or particle size distribution, along with temperature (T), atmospheric pressure (AP), and relative humidity (RH). Depending on their operating principle, these sensors are available in different technologies to react to the presence of the pollutant, such as electrochemical, metal oxide semiconductors, photo ionisation detectors, non-dispersive infrared, and light scattering, among others.

Manufacturers also integrate different sensors in the same module, which makes them easier to be used and more attractive. A list of these types of sensors (or sensor modules) and their main characteristics, in particular the type of gases measured as well as the type of data connection, are shown in Table 1. From all of them, the one we consider to have the best performance, the largest number of gases, and the best quality/price ratio is ZPHS01B [14]. In addition, we can highlight different commercial initiatives [15,16] for AQ monitoring, also considered as low cost, based on a network system that auto-calibrates the AQ measurements.

Table 1. Examples of different low-cost AQ sensors.

Module	Gas Sensors	Connection Type
SDS011 [17]	PM, T, HR, PA	UART
DL-LP8P [18]	CO_2, T, HR, PA	LoRAWAN
MiCS-6814 [19]	CO, NO_2, C_2H_5OH, NH_3, CH_4	I2C, SPI
ZPHS01B [14]	PM, CO, O_3, NO_2, TVOC, T, HR	UART

However, with reference to the measuring ranges and measurement quality of low-cost AQ sensors, the recent CEN/TS 17660-1:2021 [20] standard has set the criteria established by Directive 2008/50/EC for the equivalence of sensor systems used outdoors with the instruments for indicative measurements and objective estimations. In this scenario, these sensors have many limitations as they do not provide a reliable absolute measurement and therefore cannot be used as a substitute for a reliable absolute measurement, nor as a substitute for a reliable reference [21]. In practice, these sensors can be used to provide an order of magnitude and/or awareness of AQ and to allow the identification of pollution hot spots. Nevertheless, to increase the reliability of the readings, the measurements of these sensors can be used as input to the modeling procedure, assisted with AI techniques [6,7] and together with other data, typically measurements of other pollutants and ambient conditions (T and RH).

Furthermore, if we take into account the pollution information in a city, we can plan and influence the calculation of routes for citizens, also known as a route planner, according to the particular citizen's profile and needs. A route planner is a specialised search algorithm designed to find the optimal way to travel between two or more specific locations, which tries to minimize a determined cost function. In [22], a routing application is introduced that calculates the least polluted route through the streets. The authors employ a modified version of the popular *Ant Based Control* routing algorithm as the basis for their routing algorithm. To incorporate pollution data and minimize travel time, the authors tackle a multi-parameter problem.

Similarly, in [23], the significant health risks associated with air pollution are emphasized, with AQ being influenced by factors such as time of day, location within the city, and traffic intensity. To forecast AQ over time, the authors devise a meteorological model integrated into the Healthy Urban Route Planner (HURP), specifically designed for cyclists and pedestrians in Amsterdam (Netherlands). HURP enables users to select and plan a route that promotes a healthier environment, utilizing information gathered from various systems. Traffic emissions are computed based on observed traffic intensities and emission factors. The authors utilize the *WRF-Chem* atmosphere and AQ model, which generates

daily forecasts within a 48-h span, providing temperature and pollutant concentration forecast maps. These maps are then transformed into a unique metric that combines both factors. Hourly data of this metric are incorporated into the route planner, which employs the open source routing library *pgRouting* (*pgRouting* extends the PostGIS/PostgreSQL geospatial database to provide geospatial routing functionality) to identify healthier routes. Researchers from the National Institute for Public Health and the Environment in the Netherlands (RIVM) have developed the Atlas Living Environment [24]. Utilizing location-specific parameters, they generate maps displaying the local environment, particularly focusing on $PM10$, $PM2.5$, NO_2, and O_3 densities. These maps are derived from real-time measurements and prediction models. Additionally, the authors have developed an application that forecasts the AQ index for the next 48 h.

Along these lines, in [25], a monitoring system is introduced that utilizes a mobile network implemented on Android devices to provide real-time air pollution information to users. The pollution data collected from various sources are stored on a cloud-based server, facilitating real-time analysis and the development of an air pollution model. To measure air pollution levels, eco-sensors are deployed on public transport systems or bicycles. However, low-end sensors often suffer from reduced accuracy compared to more advanced sensors, as noted above. In [26], a system for air pollution monitoring in Mauritius Island is shown, featuring a novel data aggregation algorithm specifically designed for air pollution monitoring systems. In [27], a dynamic routing was carried out using data from a set of pollutant particles of particulate pollutants considered PM 10. The researchers used an open source routing machine (OSRM) to perform the routing. Finally, in [28], the authors also explore the integration of air pollution data with route planning. However, they propose alternative planning algorithms that aim to distribute traffic more evenly across urban areas. The authors demonstrate that such algorithms not only help alleviate traffic congestion but also contribute to reducing overall air pollution levels in urban environments.

Other alternatives to find out trajectories and routes are shown in [29,30]. However, our goal is slightly different, as we focus on minimizing pollution exposure against other criteria.

We can highlight several commercial applications known as route planners, such as Google maps [31], Ants Route [32], and Here [33], to name a few. Nevertheless, we must stress that these applications are focused mainly on driving and based on the shortest distance.

In summary from the related work, we can see that there are several initiatives to improve mobility and route planners with different strategies, but these are not focused on the user's profiles and his/her needs for healthy routes. Thus, this is the goal of this paper.

3. Design Alternatives and Techniques for a Broad AQ Monitoring Network and Its Architecture

For this purpose, the first step is to design and build the AQ monitoring network based on low-cost elements, adjusted with official AQ monitoring data. This network will be set up with ECO4RUPA IoT nodes based on a microcontroller that connects to different low-cost AQ sensors, seen in Section 2, with the option of different communication alternatives, as shown in Figure 4.

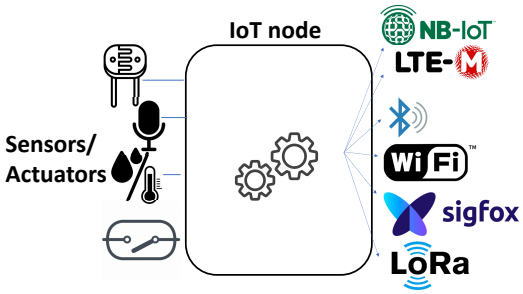


Figure 4. Generic IoT node for air quality monitoring and communications schema.

In case of failure, each IoT node incorporates a real-time clock, a memory card, and a watchdog mechanism for its recovery.

We have initially selected the ESP32 microcontroller [34], due to its performance and quality/price, as it offers in each model the possibility of having different antennas, as well as the possibility to implement different communication standards. The ESP32 is a series of low-cost, low-power system-on-chip microcontrollers that embeds several communication modules. Based on this microcontroller, it is worth noting the Pycom (Pycom Ltd. went into bankruptcy in September 2022, but the newly created Pycom BV took over the company) FiPy module [35] includes technologies such as Lora/Sigfox, WiFi, and Bluetooth, and cellular technologies such as long term evolution (LTE) for machines (LTE-M) and narrow band IoT (NB-IoT). Notice that this FiPy module is flexible enough and permits the building of this type of IoT node.

Thus, since our goal is to cover different scenarios in urbanized areas, these ECO4RUPA low-cost AQ monitoring nodes have been designed to be flexible and can integrate all these technologies, as shown in Figure 4. In particular, we use one technology or another based on the available wireless services at the sampling point in the deployment using direct connection, without multihop.

In particular, Figure 5 depicts a hardware prototype of the implemented AQ IoT monitoring node, with the connection of the ESP32 microcontroller to the ZPHS01B AQ sensor module. Figure 1b,c show the indoor case (for tunnels and indoor environments) for this node that includes a tube and a fan to make air flow pass through the sensor board and also the outdoor version, where the air intake is at the bottom of the tube that sucks it in with a small fan. As noted before, these IoT nodes are used as a coarse reference of the AQ, compared with the measurements provided by the official AQ monitoring stations. The direct measurements taken from these low-cost sensors, in order to be considered valid, are processed by AI-based algorithms to correct and adapt the measurements to reliable values [6]. This process is out of the scope of this paper, as we focus only on how to calculate healthy routes according to the particular citizen's profile and needs.

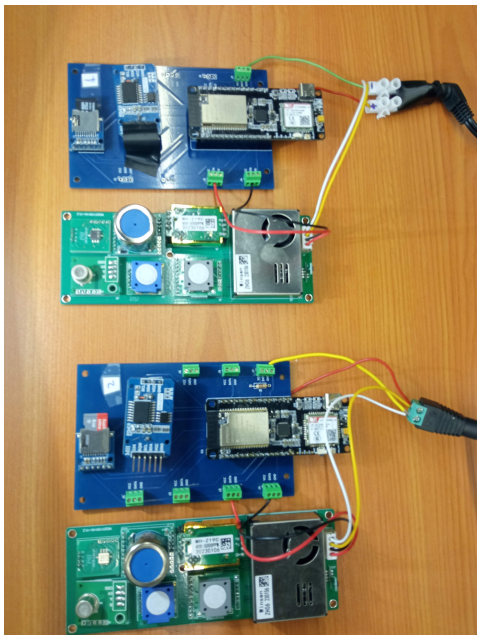


Figure 5. Example of two ECO4RUPA IoT nodes with ESP32 microcontroller connected to module ZPHS01B with different air quality sensors [14].

The communication scheme of the IoT node with the infrastructure is detailed in Figure 6. It is based on the IoT Message Queue Telemetry Transport (MQTT) protocol, which transmits information via messages between the nodes and the MQTT broker. It should be noted that MQTT allows three levels of quality of service (QoS) to verify the

delivery of messages and also several security mechanisms regarding the transmitted data. We have chosen the highest QoS level, QoS-2, which guarantees the delivery of messages only once, without loss or duplication. In terms of security, we use username and password-based login, both at the broker and at the clients, and SSL-certified encryption for transmitted data. The data received are stored locally in a database. For the publishing process, nodes can create a new topic by simply publishing to it, so that more nodes can be added to the IoT system, which greatly facilitates the scalability of the system. These data can also be stored in the cloud, providing additional backup and security against data loss. To graphically visualise the data, the geographical positions of the nodes are indicated.

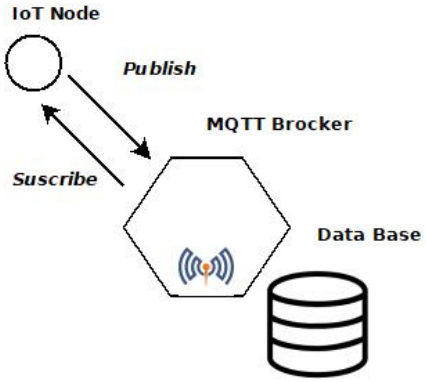


Figure 6. Communication scheme of the ECO4RUPA AQ IoT node connection and communications protocol.

Notice that the placement of these ECO4RUPA low-cost AQ monitoring nodes will improve the coverage given by the official AQ monitoring stations as noted before, following criteria explained in Section 4.2.

4. Data Fusion, Spatial Interpolation, and Route Planner Application

This section describes the core of the healthy router planner. The goal is to calculate healthy walking and/or cycling routes according to the particular citizen's profile and needs. For the development of this service, its flowchart is shown in Figure 7. In this case, initially, the user launches a request for a route calculation. With this action, his/her user profile is analysed, and based on it, the appropriate variables (specific pollutants) will be considered, and a complete interpolation in the area of interest defined by the search using Kriging technique is performed. Subsequently, these values are superimposed on the geographical map and define the metric to be minimised in the route search.

Notice that the minimum and recommended time frequency used for AQ monitoring is 10 min, according to ISO 11771:2010 [8], ISO 37122:2019 [9], and European Regulation Directive 2008/50/EC. In practice, this time interval is enough to provide accurate pollution information for the route planner, and this time scale is sufficient for this purpose.

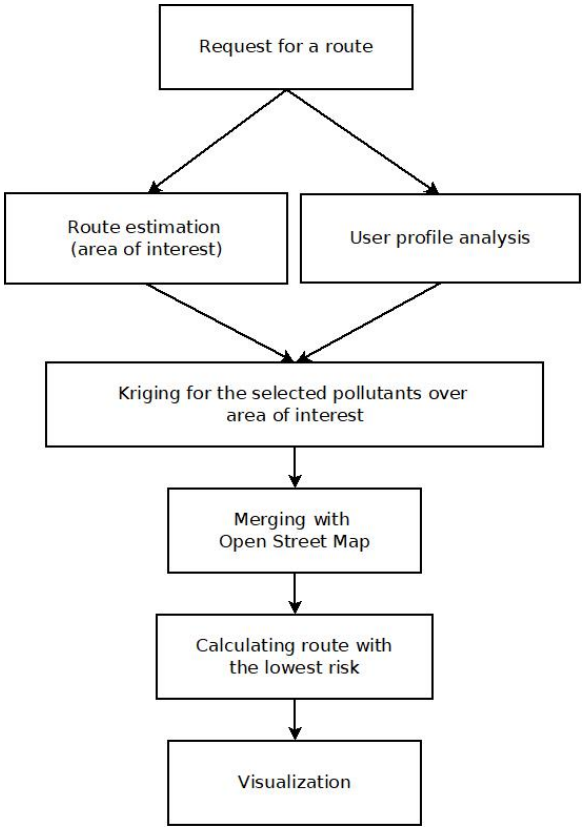


Figure 7. Flowchart of the user application to request healthy routes.

4.1. Analysis of the User's Profile: Weighting Pollutants

Based on the users' requirements (specified within his/her profile), we estimate the pollution according to it, as a combination of the different parameters (pollutants) by weighting their different measurements in the area of user mobility. In particular, as a proof of concept, we have considered citizens with asthma and pregnant women without a lack of generality. In this case, we will use the following weights for the different pollutants according to the literature.

In the case of asthma, we assign 40% for ozone, O_3, ($\mu g/m^3$); 10% for NO_2 ($\mu g/m^3$); and 50% for $PM2.5$. These weights are assigned because $PM2.5$ is considered one of the most dangerous pollutants, as it can penetrate the lungs and cause various health problems [36]. Furthermore, O_3 is an oxidant known to irritate the airways and has a clearly defined effect on asthma exacerbation [37]. In addition, according to [37,38], the results of a meta-analysis, there is evidence to support the link between increased ozone concentration and NO_2, which worsens asthma.

In the case of pregnant women, we assign 5% for O_3 ($\mu g/m^3$), 35% for NO_2 ($\mu g/m^3$), 20% for $PM2.5$, 30% for CO mg/m^3, and 10% for $PM10$. These gases are chosen because, according to [39], exposure to O_3, NO_2, and CO is associated with a reduction in neonatal weight, and exposure to $PM2.5$ and $PM10$ is related to an increased risk of premature birth. In addition, NO_2 adds delay in the development of children's attention span, according to a study conducted by [40]. We assign 30% for CO because exposure to this gas can directly affect the fetus through oxygen deficiency [39], which can cause brain damage, developmental delay, and complications during pregnancy [41]. For $PM2.5$ and $PM10$, we assign a weight of 20% for $PM2.5$ and 10% for $PM10$, because they have been associated with complications during pregnancy, such as premature birth, low birth weight, and respiratory problems in the fetus. Finally, we assign only 5% for O_3, because we have found studies that say that O_3 exposure increases the risk of premature birth, low birth weight, and respiratory problems in the fetus, but others do not confirm this relation [39].

Notice that, in practice, these weights will be personalized based on the end user's requirements, and they could even be saved within his/her profile. In case we do not have a specific profile, as a user requirement, we will use a default one that will try to minimize global pollution exposure using an equal distribution of weights as noted in Section 1.

4.2. Kriging for Spatial Interpolation of Pollution

Since the spatial sampling is still limited to the spots where the IoT nodes are deployed and/or the official AQ monitoring stations are installed for a real-time map of the pollutants, a spatial interpolation technique is required, because it is necessary for accurate pollution measurements at the different points over the city map in order to analyse the different paths for the routes.

Kriging [42] is a spatial statistical technique that allows the analysis of geolocated information and is based on spatial autocorrelation, unlike other techniques such as inverse distance weighting (IDW) and splines [42,43]. The main idea with Kriging is that the estimated variable is given by a deterministic (without spatial influence) part and a random (with spatial influence) part. In this case, Kriging employs the spatial function from the random section in order to deliver the best linear unbiased estimator. Thus, the information gathered from the IoT nodes establishes a dataset associated to different locations with their coordinates, longitude, and latitude, as a first step to applying the Kriging technique.

Based on this approach, we are interested in estimating a variable z within a region D in a 2-dimensional map ($D \subset \mathbb{R}^2$). For this, we obtain measurements of the variable z at a finite number of points (n points) within the region, denoted as $z(s_1), z(s_2), ..., z(s_n)$.

The covariance between measurements $z(s_1)$ and $z(s_2)$, denoted as $cov[z(s_1), z(s_2)]$, depends only on the difference in locations (distance and direction) between these two points.

In order to characterize the random behavior of the variable being estimated in Kriging, we use the variogram function, which measures how the variable changes with respect to distance. The variogram is defined as $2\gamma(s_1 - s_2) = E[(z(s_1) - z(s_2))^2]$ and the semivariogram is half of the variogram, which can be represented as $\gamma(s_1 - s_2) = \sigma^2 - cov[s_1 - s_2]$, assuming that the mean μ and the variance σ^2 are constant across the region, an assumption usually made for geostatistical data. In case of an isotropic process, the variogram can be expressed in terms of an auto-correlation function ρ as $2\gamma(h) = 2\sigma^2(1 - \rho(h))$, where h is the separation between two points.

The variogram is a key point in the Kriging technique and it requires three attributes, as depicted in Figure 8. First, the *sill*, which is the maximum height of the variogram curve, where, at large distances, the correlation between measured values becomes independent. Second, the *range*, representing the distance beyond which pairs of points are negligibly correlated. Finally, the *nugget effect*, indicating a non-zero value at zero distance due to measurement error and micro-scale variation.

Thus, based on the measurements (samples) with the n points within the region D, we will first estimate the *empirical variogram* and then fit a model to it. For this process, we can locate and improve the placement of the samples (even extra samples) using the ECO4RUPA AQ monitoring nodes, in order to enhance this variogram if it were necessary. Common models include spherical, exponential, Gaussian, and power models [43]. The variogram model will allow us to estimate values at unmeasured locations.

Next, linear interpolation is used to estimate the value $\hat{z}(s_0)$ at a location s_0 based on these n measurements $z = [z(s_1), ..., z(s_n)]^T = [z_1, ..., z_n]^T$ given by $\hat{z}_0 = w_0 + \sum_1^N w_i z_i = w_0 + w^T z$. For this estimation, we calculate the different weights w_i that minimize the estimation variance, which is derived from the input variogram model. There are different types of Kriging, such as simple Kriging (SK) and ordinary Kriging (OK), differing mainly in the treatment of the mean value of the stochastic field. Simple Kriging assumes the mean value μ is known and constant, whereas ordinary Kriging considers it unknown and constant, providing a more realistic approach to the estimation process.

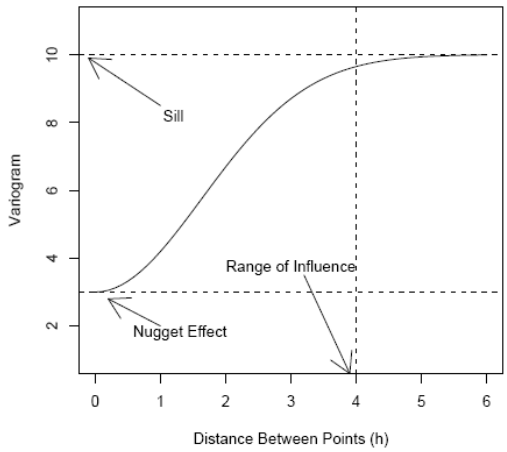


Figure 8. A theoretical Gaussian model of variogram for the Kriging technique.

4.3. Mapping of Pollution over the Grid on the City

Once we define the pollution for each user and it has been interpolated with OK over the city map on the area of the user mobility, we need to map this pollution over the grid of the city.

For this, the street network within a city is represented as a bidirectional graph, where nodes correspond to intersections and edges represent street segments. Each edge is associated with a parameter that signifies the cost of travelling along that particular segment. Common routing algorithms aim to minimize the total cost of a route by considering the cumulative costs along the edges, known as the cost function, in this case, given by the exposure to pollution for each user.

Notice that to do this, we have to assign the pollution at each point over the city map by using a grid of 0.0001 decimal degrees that each correspond to 11.5 m (since 360º corresponds to the whole perimeter, 40,075 km, of Earth). It is necessary to find the match between each graph node and its corresponding grid point and then assign the pollution value of the grid point to a new attribute of the graph node. In particular, the grid dataframe index is defined as a combination of the longitude and latitude coordinates, and a search over the grid for the corresponding point uses these coordinates as indices.

For this, we have used OpenStreetMap (OSM) [44], which is a collaborative project for the creation of editable and free maps. We can find different libraries and tools such as *OSMnx* [45] for Python, which will allow us to analyse these maps in a coherent way.

It is worth emphasizing that Krigring could be seen as a way to gather an accurate measurement of a concrete position in the map, but it is also true that it is not considering the altitude of the available buildings and objects available in the urban landscape. Thus, even if it is technically possible, the authors have decided to limit the concrete geographical positions where the Kriging algorithm is applied to the geographical locations of the nodes of the graph associated to the map of the city. Such nodes are, by definition, intersections/joints, i.e., roads. Thus, the fact that we are only using the technique in such nodes makes the calculation really accurate, as the altitude along the edges and nodes of the road is very constant and linearly incremental, where the change in altitude per every 100 m rarely is higher than 5% in urban areas.

Once the pollution values have been assigned to the nodes of the graph, the next step is to assign weights to the edges of the graph to convert it into a weighted graph. To determine the weights of the edges, the criterion based on the weighted average between the nodes that connect each edge will be used, considering both the pollution values registered at that specific time and the distance between the nodes. This means that the weight of an edge will be the average of the pollution measurement values of the nodes connecting that edge, multiplied by the distance between the two nodes. This is because the travel time between two nodes is assumed to be directly proportional to the distance. In addition, the pollution

to which people are exposed is proportional to the exposure time. Therefore, the pollution value and the length of the street segment are multiplied to obtain the cost function.

4.4. Healthy Route Planner

With the weighted graph in the area of user mobility with the pollution weights of the edges, we can proceed to find the path that minimizes overall exposure to pollution.

To do this, we identify the node closest to our departure location and the node closest to our arrival location. The widely used Dijkstra algorithm (or shortest path first) utilises the edge lengths as the cost function. Then, using the functions provided by the *OSMnx* library, we construct the optimal route to reach the destination.

5. Results and Discussion

In this section, we provide and analyse the results of the proposed route planner under different scenarios (different timetables, congestion road traffic, and routes) and different user profiles, with a special interest on citizens with asthma and pregnant women, as both have special needs based on the weights defined in Section 4.1.

In particular, the area under test where we have carried out our experiments is Burjassot city (Valencia, Spain), where we have six official AQ monitoring stations nearby (as shown in Figure 9), within a distance less than 5 km. According to the rules and standards noted in Section 1, it was not necessary to install any extra AQ monitoring node. Specifically, our target for the router planner is given by a minimum threshold of 15% of pollution reduction (PR) with an increase in distance (ID) traveled of less than or equal to 10%.

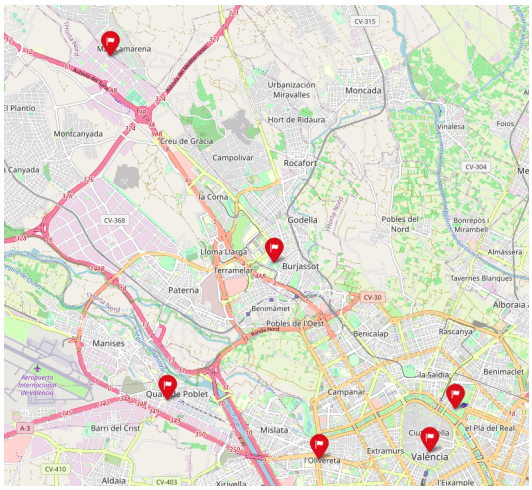
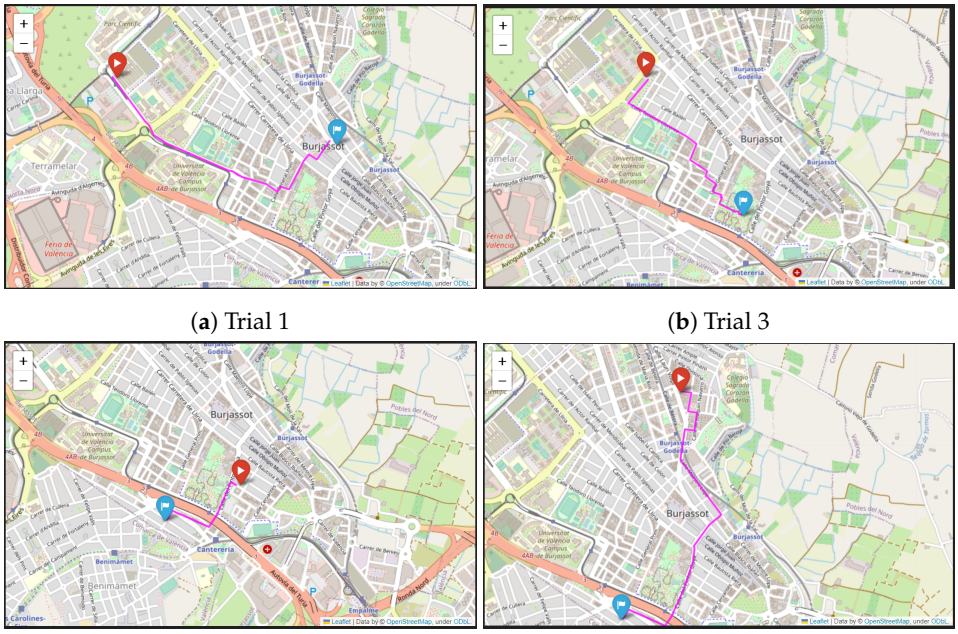


Figure 9. Official AQ monitoring stations near Burjassot city (Valencia, Spain).

5.1. Analysis of the Healthy Route Planner with Different Scenarios

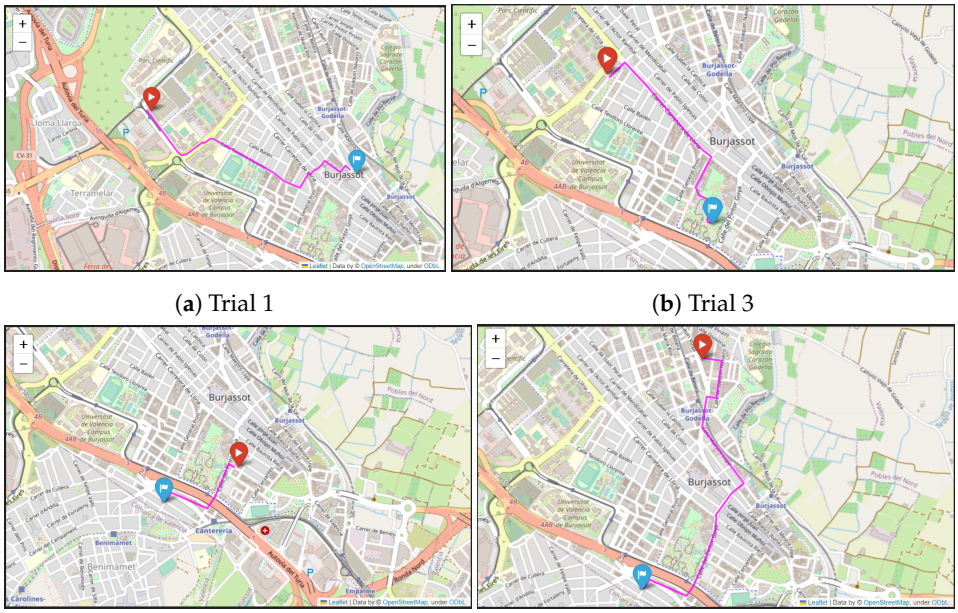
Figures 10–12 show the resulting routes for users with asthma, pregnant women, and shortest path first, respectively. Table 2 depicts the detail for each trial, in bold are the ones plotted in Figures 10–12. In this table, each column identifies the trial number, source (src.), destination (dto.), day (DD), hour (HH), total cost with asthma (C. Asthma), total distance with asthma (D. Asthma), total cost for pregnancy (C. Preg.), total distance for pregnancy (D. Preg.), cost with shortest path first (SPF) assuming asthma (C. SPF Asthma), cost with shortest path first assuming pregnancy (C. SPF Preg.), and, finally, distance with SPF (D. SPF).



(**a**) Trial 1 (**b**) Trial 3

(**c**) Trial 7 (**d**) Trial 8

Figure 10. Examples of several trials with asthma. See route details in Table 2.



(**a**) Trial 1 (**b**) Trial 3

(**c**) Trial 7 (**d**) Trial 8

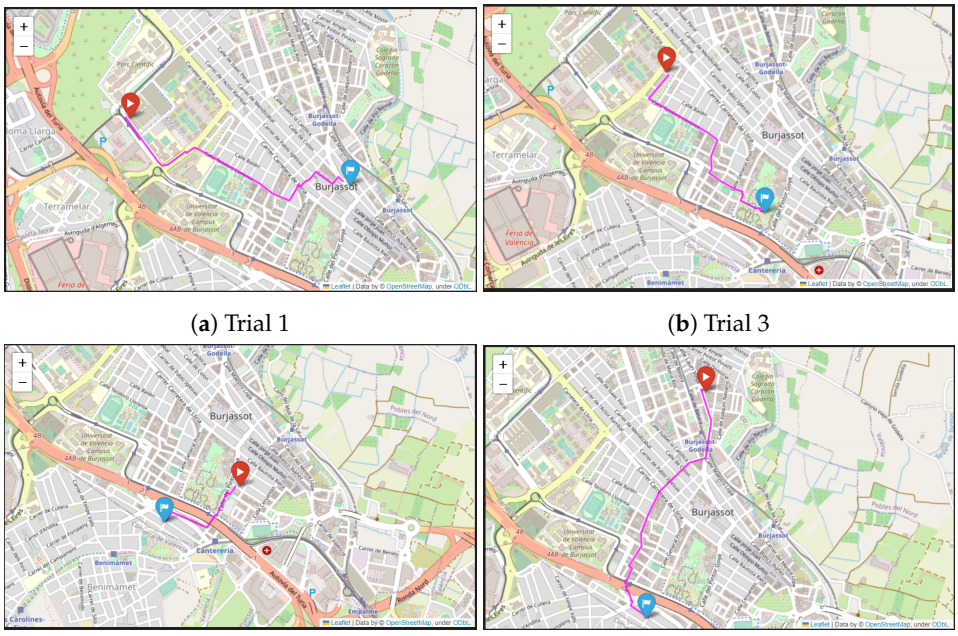
Figure 11. Examples of several trials with pregnant women. See route details in Table 2.

Table 2. Trials 1–20: detail. In bold are the ones plotted in Figures 10–12.

Trial	Src.	Dto.	DD	HH	C. Asthma	D. Asthma	C. Preg.	D. Preg.	C. SPF Asthma	C. SPF Preg.	D. SPF
1	**ETSE**	**Oficina de correos de Burjassot**	**Friday 16 June 2023**	**10:13**	**42,848.7**	**1728.1**	**17,927.14**	**1671.05**	**43,609.68**	**17,927.14**	**1671.05**
2	ETSE	Residencia micampus Burjassot	Friday 16 June 2023	12:21	28,349.79	816.02	7679.55	816.02	28,349.79	7679.55	816.02
3	**Residencia micampus Burjassot**	**Parque de la Granja**	**Friday 16 June 2023**	**19:32**	**46,232.2**	**1295.79**	**9822.41**	**1279.08**	**47,273.17**	**10,000.21**	**1278.33**
4	C/Vista Alegre 2	Polideportivo de Burjassot	Saturday 17 June 2023	9:21	432.97	850.25	191.17	850.25	492.98	216.3	740.34
5	C/Vista Alegre 2	Hospital IMED	Saturday 17 June 2023	20:47	1290.39	2385.04	322.92	2385.04	1716.59	424.17	2124.88
6	C/Maestro Giner 32	Restaurante Colonial buffet	Sunday 18 June 2023	14:09	954.03	1215.45	323.4	1178.19	1058.74	361.32	1134.74
7	**C/Maestro Giner 32**	**Restaurante Quitin**	**Sunday 18 June 2023**	**14:28**	**413.58**	**640.03**	**86.99**	**640.03**	**524.55**	**111.02**	**627.55**
8	**C/Vista Alegre 2**	**Restaurante Quitin**	**Sunday 18 June 2023**	**14:31**	**1161.19**	**1996.87**	**249.01**	**1997.02**	**1502.93**	**327.74**	**1702.75**
9	C/Vista Alegre 2	ETSE	Monday 19 June 2023	10:35	415.82	2281.57	893.67	2281.57	778.73	1675.34	1648.43
10	ETSE	Residencia micampus Burjassot	Friday 16 June 2023	12:21	28,349.79	816.02	7679.55	816.02	28,349.79	7679.55	816.02
11	ETSE	C/Maestro Giner 32	Monday 19 June 2023	14:31	546.89	1928.15	156.95	1928.15	624.33	175.94	1928.15

Table 2. *Cont.*

Trial	Src.	Dto.	DD	HH	C. Asthma	D. Asthma	C. Preg.	D. Preg.	C. SPF Asthma	C. SPF Preg.	D. SPF
12	C/Vista Alegre 2	Consum 1	Wednesday 21 June 2023	9:32	433.32	850.26	139.79	850.41	493.63	157.11	740.34
13	C/Vista Alegre 2	Consum 2	Wednesday 21 June 2023	9:41	513.38	663.54	159.03	923.98	671.82	159.03	575.77
14	C/Vista Alegre 2	Mercadona	Wednesday 21 June 2023	9:47	341.57	473.96	104.97	494.65	501.38	144.4	470.17
15	C/Lauri Volpi 12	Druni	Wednesday 21 June 2023	11:08	522.42	867.41	154.86	867.41	589.62	168.88	793.06
16	C/Lauri Volpi 15	Mercadona	Wednesday 21 June 2023	12:47	412.1	379.06	110.94	420.58	541.36	134.31	351.59
17	C/Vista Alegre 2	Mercadona	Thursday 22 June 2023	12:32	376.14	473.8	97.69	473.96	570.79	134.76	470.17
18	Parada Burjassot-Godella	Centro de salud	Thursday 22 June 2023	12:48	811.98	1380.04	196.48	1503.31	961.48	233.37	1273.13
19	C/Vista Alegre 2	Parque de la granja	Sunday 25 June 2023	17:19	373.1	528.61	65.95	528.77	551.04	96	502.75
20	C/Maestro Giner 32	Mercado	Monday 26 June 2023	11:46	291.5	666.91	69.12	666.91	381.59	90.15	653.92



(**a**) Trial 1 (**b**) Trial 3

(**c**) Trial 7 (**d**) Trial 8

Figure 12. Examples of several trials with shortest path first. See route details in Table 2.

It is important to highlight that the difference between the optimal healthy route and the shortest route is highly dependent on the selected origin and destination. In some cases, the difference between the shortest route and the optimal healthy route is minimal, for example, in trial number 3 in Table 2. In other cases, there may be a significant difference, for example, in trial number 9 in the same table. Furthermore, the results suggest that there is a trade-off between reduced pollution exposure and journey length.

5.2. Statistical Analysis for the Different Scenarios

We will estimate the average pollution reduction (PR) percentages for each of the trials. Table 3 shows % of PR and increased distance (ID) for each scenario, and the average (avg.) is shown in the last row. In bold are the trials plotted in Figures 10–12.

Our main goal was to recommend optimal routes that ensure low exposure to air pollution during the journey. First, we have evaluated whether the algorithm recommends routes that have low exposure to air pollution compared to routes that recommend the shortest path (which is the one usually chosen). We are interested in determining whether our proposal excels at providing routes that minimise exposure to these types of pollutants. We have also examined whether routes with low exposure to pollution turn out to be longer compared to shorter routes. We are interested in whether, by prioritising health and reducing pollution exposure, there is a trade-off in terms of distance travelled.

With this evaluation, we validate the effectiveness of our algorithm.

By analysing the trials performed, we can confirm that our algorithm succeeds in reducing exposure to polluted air by 18.72% in cases of asthma, which implies an increase of 9.27% in distance travelled. For pregnant women, our algorithm succeeds in reducing exposure to air pollution by 16.91%, increasing the distance travelled by 10.32%. On average, our approach can lead to an approximately average reduction in pollution exposure in Burjassot (Valencia) of 17.82% while experiencing an approximately average increase in distance travelled of 9.8%. Therefore, it is shown that the route planning system achieves the goal of reducing exposure to high air pollution. These results support the effectiveness of the method in providing healthier navigation options among users, without imposing a significant increase in distance. In addition, there is evidence that the optimal route for health is consistent for both the pregnancy and asthma trials.

Table 3. Percentage of pollution reduction (PR) and increased distance (ID) for each scenario. In bold are the trials plotted in Figures 10–12 and the average.

Trial	% PR with Asthma	% ID with Asthma	% PR with Pregnancy	% ID with Pregnancy
1	**1.75%**	**3.3%**	**0%**	**0%**
2	0%	0%	0%	0%
3	**2.20%**	**1.35%**	**1.78%**	**0.06%**
4	12.17%	12.93%	11.62%	12.93%
5	24.83%	10.91%	23.87%	10.91%
6	9.89%	6.64%	10.49%	3.69%
7	**21.15%**	**1.95%**	**21.65%**	**1.95%**
8	**22.74%**	**14.73%**	**24.02%**	**14.74%**
9	46.6%	27.75%	46.66%	27.75%
10	12.15%	24.13%	6.39%	24.13%
11	12.4%	15.25%	10.8%	15.25%
12	12.22%	12.93%	11.03%	12.94%
13	23.58%	13.23%	18.6%	37.68%
14	31.84%	0.8%	27.32%	4.95%
15	11.4%	8.57%	8.3%	8.57%
16	23.88%	7.25%	17.4%	16.4%
17	34.1%	0.77%	27.51%	0.8%
18	15.55%	7.75%	15.81%	15.31%
19	32.29%	4.89%	31.31%	4.92%
20	23.61%	1.95%	23.33%	1.95%
Avg.	**18.72%**	**9.27%**	**16.91%**	**10.32%**

6. Conclusions and Future Work

In this paper, we have described a healthy route planner application within the activities carried out in the ECO4RUPA project. The goal of this application is to calculate healthy walking and/or cycling routes according to the particular citizen's profile and needs, in particular when they require specific care in case of respiratory diseases, trying to reduce the exposure to specific air pollutants based on these profiles. We have shown several profiles as use cases, such as citizens with asthma and pregnant women, where each profile defines a set of weights for the different air pollutants that determine the hazardousness of their exposure.

In order to estimate the distribution of the pollutants over the city map grid, we have used ordinary Kriging. Weighting the different pollutants given by the user's profile over the map, we define a complex metric that is used as a cost function in order to run the shortest path algorithm. From our results, we have achieved on average a reduction in pollution exposure of 17.82% while experiencing an approximately average increase in distance travelled of 9.8%.

Notice that this healthy route planner application uses data from both official static AQ stations (mainly) and the low-cost ECO4RUPA AQ nodes when it is necessary. These ECO4RUPA nodes are simpler and easier to be deployed, but they are less accurate and their raw measurements must be further processed (including calibration) in order to provide accurate AQ monitoring data. In addition, the lifetime of the sensors equipped on these low-cost nodes is shorter, between 6 and 12 months. However, due to their

cost, we can replace them when needed, compared with the official stations that require weekly maintenance.

In future work, along the same lines, although it is not a pollutant gas, we could add pollen to this list. Pollen is of great concern to people with allergic pathologies. However, notice that, unlike the polluting gases, pollen is controlled in a more complex way through health institutions using traditional and even legacy systems that require a posteriori analysis of the collection filters, analyzing the different particles one by one. In addition, it must be noticed that there are other alternative and complementary metrics, such as subjective noise annoyance, which can be part of this route planning algorithm [46].

Author Contributions: Conceptualization, J.S.-G., S.F.-C. and J.M.A.-C.; methodology, R.F.-J., J.M.A.-C., J.S.-G. and S.F.-C.; software, R.A.-C., R.F.-J. and J.S.-G.; validation, R.A.-C., J.J.P.-S. and S.F.-C.; investigation, J.J.P.-S., S.F.-C. and J.S.-G.; resources, J.S.-G. and S.F.-C.; writing—original draft preparation, R.F.-J., S.F.-C. and J.S.-G.; writing—review and editing, J.J.P.-S., R.A.-C. and J.M.A.-C. All authors have read and agreed to the published version of the manuscript.

Funding: The authors would like to thank the Spanish State Research Agency (AEI) and the European Regional Development Fund (ERDF) for partially funding this research within the projects with grant references PID2021-126823OB-I00 ("ECO4RUPA project") funded by MCIN/AEI/10.13039/501100011033 and by the "European Union NextGenerationEU/PRTR". Thanks to the Research vice-rectorship of Universitat de València for funding the grant with reference UV-INV-EPDI-2647726 for a research stay for S.F.-C., and the Spanish Ministry of Education in the call for Senior Professors and Researchers to stay in foreign centers for the grant PRX22/00503 for J.S.-G.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Adair-Rohani, H. Air Pollution Responsible for 6.7 Million Deaths Every Year. 2023. Available online: https://www.who.int/teams/environment-climate-change-and-health/air-quality-and-health/health-impacts/types-of-pollutants (accessed on 27 February 2023).
2. Eurostat: Statistics Explained. Respiratory Diseases Statistics. 2022. Available online: https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Respiratory_diseases_statistics#Deaths_from_diseases_of_the_respiratory_system (accessed on 22 May 2023).
3. BBC News. Air Pollution News. 2018 Available online: https://www.bbc.co.uk/news/world-europe-46017339 (accessed on 23 May 2023).
4. Molinari, G.; Colombo, G.C.C. Respiratory allergies: A general overview of remedies, delivery systems, and the need to progress. *Int. Sch. Res. Allergy* **2014**, *1*, 1–16. [CrossRef] [PubMed]
5. González-Díaz, S.; Arias-Cruz, A.; Macouzet-Sánchez, C.; Partida-Ortega, A. Impact of air pollution in respiratory allergic diseases. *Med. Univ.* **2016**, *18*, 212–215. [CrossRef]
6. Zimmerman, N.; Presto, A.A.; Kumar, S.P.N.; Gu, J.; Hauryliuk, A.; Robinson, E.S.; Robinson, A.L.; Subramanian, R. A machine learning calibration model using random forests to improve sensor performance for lower-cost air quality monitoring. *Atmos. Meas. Tech.* **2018**, *11*, 291–313. [CrossRef]
7. Yadav, K.; Arora, V.; Kumar, M.; Tripathi, S.N.; Motghare, V.M.; Rajput, K.A. Few-Shot Calibration of Low-Cost Air Pollution ($PM_{2.5}$) Sensors Using Meta Learning. *IEEE Sens. Lett.* **2022**, *6*, 1–4. [CrossRef]
8. *ISO 11771:2010*; Air Quality -Determination of Time-Averaged Mass Emissions and Emission Factors—General Approach. Technical Report. ISO: Geneva, Switzerland , 2010.
9. *ISO 37122:2019*; Sustainable Cities and Communities—Indicators for Smart Cities. TC268 Sustainable Cities and Communities. Technical Report. ISO: Geneva, Switzerland, 2019.
10. FAO . Directive 2008/50/EC of the European Parliament and of the Council of the European Parliament and of the Council of 21 May 2008 on Ambient Air Quality and Cleaner Air for Europe. *Off. J. Eur. Commun.* **2008**, *L 152*, 1–44.

11. Conselleria d'Agricultura, Desenvolupament Rural, Emergència Climàtica i Transició Ecològica. Red Valenciana de Vigilancia y Control de la Contaminación Atmosférica. 2023. Available online: https://agroambient.gva.es/va/web/calidad-ambiental/datos-on-line (accessed on 27 May 2023).
12. Ajuntament de Valencia, Minut a Minut. Estaciones Contaminación Atmosféricas. 2023. Available online: https://valencia.opendatasoft.com/explore/dataset/estacions-contaminacio-atmosferiques-estaciones-contaminacion-atmosfericas/table/ (accessed on 21 May 2023).
13. aqicn.org Project. World Air Quality Index Project. 2023. Available online: https://aqicn.org/ (accessed on 27 February 2023).
14. Winsen, Ltd. Air Quality Sensor zphs01b. 2023. Available online: https://www.winsen-sensor.com/d/files/zphs01b-english-version1_1-20200713.pdf (accessed on 20 March 2023).
15. Kunak Tech., Calidad del Aire Urbano: Información Ambiental y Parámetros meteorológicos en Entornos Urbanos. 2023. Available online: https://www.kunak.es/ (accessed on 28 February 2023).
16. Oizom , Ltd. Accurate and Affordable Air Quality Monitoring Solutions. 2023. Available online: https://oizom.com (accessed on 28 February 2023).
17. Nova Fitness Co., Ltd. Air Quality Sensor SDS011. 2023. Available online: https://cdn-reichelt.de/documents/datenblatt/X200/SDS011-DATASHEET.pdf (accessed on 27 April 2023).
18. DecentLab, Ltd. Air Quality Sensor DL-LP8P. 2023. Available online: https://www.catsensors.com/media/Decentlab/Productos/Decentlab-DL-LP8P-datasheet.pdf (accessed on 27 April 2023).
19. SGX, SensorTech. Air Quality Sensor MiCS-6814. 2023. Available online: https://www.sgxsensortech.com/content/uploads/2015/02/1143_Datasheet-MiCS-6814-rev-8.pdf (accessed on 21 May 2023).
20. CEN, CEN/TS 17660-1 Air Quality—Performance Evaluation of Air Quality Sensor Systems—Part 1: Gaseous Pollutants in Ambient Air. 2021. Available online: https://www.boutique.afnor.org/engb/standard/din-cen-ts-176601/air-quality-performance-evaluation-of-air-quality-sensor-systemspart-1-gas/eu174644/322334 (accessed on 2 August 2023).
21. García, M.R.; Spinazzé, A.; Branco, P.T.; Borghi, F.; Villena, G.; Cattaneo, A.; Gilio, A.D.; Mihucz, V.G.; Álvarez, E.G.; Lopes, S.I.; et al. Review of low-cost sensors for indoor air quality: Features and applications. *Appl. Spectrosc. Rev.* **2022**, *57*, 747–779. [CrossRef]
22. Rothkrantz, L. Multi parameter routing in air polluted urban areas. In Proceedings of the 2020 Smart City Symposium Prague (SCSP), Prague, Czech Republic, 25 June 2020 ; pp. 1–6. [CrossRef]
23. Steeneveld, G.; Vreugdenhil, L.; van der Molen, M.; Ligtenberg, A. Towards a Healthy Urban Route Planner for cyclists and pedestrians in Amsterdam. In Proceedings of the Annual Meeting European Meteorological Society, Dublin, Ireland, 4–8 September 2017; EMS2017-305.
24. RIVM Institute. Environmental Health Atlas—Explore and Discover Your Living Environment. 2023. Available online: https://www.atlasleefomgeving.nl/en (accessed on 27 May 2023).
25. Alvear, O.; Zamora, W.; Calafate, C.T.; Cano, J.C.; Manzoni, P. EcoSensor: Monitoring environmental pollution using mobile sensors. In Proceedings of the 2016 IEEE 17th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM), Coimbra, Portugal, 21–24 June 2016; pp. 1–6. [CrossRef]
26. Khedo, K.; Rajiv, P.; Avinash, M. A Wireless Sensor Network Air Pollution Monitoring System. *Int. J. Wirel. Mob. Netw.* **2010**, *2*, 31–45. [CrossRef]
27. Müller, S.; Voisard, A. Air Quality Adjusted Routing for Cyclists and Pedestrians. In Proceedings of the 1st ACM SIGSPATIAL International Workshop on the Use of GIS in Emergency Management (EM-GIS '15), Bellevue WA, USA, 3–6 November 2015 ; Association for Computing Machinery: New York, NY, USA, 2015; EM-GIS '15. [CrossRef]
28. Vamshi, B.; Prasad, R.V. Dynamic route planning framework for minimal air pollution exposure in urban road transportation systems. In Proceedings of the 2018 IEEE 4th World Forum on Internet of Things (WF-IoT), Singapore, 5–8 February 2018; pp. 540–545. [CrossRef]
29. Wang, C.; Li, C.; Qin, C.; Wang, W.; Li, X. Maximizing spatial–temporal coverage in mobile crowd-sensing based on public transports with predictable trajectory. *Int. J. Distrib. Sens. Netw.* **2018**, *14*, 1–10. [CrossRef]
30. Wu, C.; Liu, Z.; Liu, F.; Yoshinaga, T.; Ji, Y.; Li, J. Collaborative Learning of Communication Routes in Edge-Enabled Multi-Access Vehicular Environment. *IEEE Trans. Cogn. Commun. Netw.* **2020**, *6*, 1155–1165. [CrossRef]
31. Google. Google Maps. 2023. Available online: https://google.maps/ (accessed on 27 February 2023).
32. AntsRoute. The Solution for Planning Last-Mile Route of Field Workforce. 2023. Available online: https://antsroute.com/en (accessed on 27 February 2023).
33. Here. Our Mission Is to Enable a Digital Representation of Reality to Radically Improve the Way the World Moves, Lives and Interacts. 2023. Available online: https://www.here.com/ (accessed on 27 February 2023).
34. Espressif Systems. ESP32 System-on-Chip. 2023. Available online: https://www.espressif.com/en/products/socs/esp32 (accessed on 28 January 2023).
35. Pycom.io. Fipy, Five Network Development Board for IoT. 2022. Available online: https://pycom.io/product/fipy/ (accessed on 28 December 2022).
36. Kiesewetter, G.; Schoepp, W.; Heyes, C.; Amann, M. Modelling PM2.5 impact indicators in Europe: Health effects and legal compliance. *Environ. Model. Softw.* **2015**, *74*, 201–211. [CrossRef]

37. Ubilla, C.; Yohannessen, K. Contaminación atmosférica efectos en la salud respiratoria en el niño. *Rev. Méd. Clín. Condes* **2017**, *28*, 111–118. [CrossRef]
38. Orellano, P.; Quaranta, N.; Reynoso, J.; Balbi, B.; Vasquez, J. Effect of outdoor air pollution on asthma exacerbations in children and adults: Systematic review and multilevel meta-analysis. *PLoS ONE* **2017**, *12*, e0174050. [CrossRef] [PubMed]
39. Vargas, S.; Onatra, W.; Osorno, L.; Páez, E.; Sáenz, O. Contaminación atmosférica y efectos respiratorios en niños, en mujeres embarazadas y en adultos mayores. *Rev. UDCA Actual. Divulg. Científica* **2008**, *11*, 31–45.
40. Bienestar, O. La Exposición Al Dióxido de Nitrógeno Durante El Embarazo perjudica La Capacidad de Atención de los Niños. 2017. Available online: https://www.atresmedia.com/objetivo-bienestar/actualidad/exposicion-dioxido-nitrogeno-embarazo-perjudica-capacidad-atencion-ninos_20170803598433970cf2c0f4136d712c.html (accessed on 15 June 2023).
41. ISGlobal. La Exposición a la Contaminación Atmosférica Durante el Embarazo También Perjudica a la Capacidad de Atención en la Infancia. 2017. Available online: http://bit.ly/exposicion-a-contaminacion-atmosferica-durante-embarazo (accessed on 15 June 2023).
42. Isaaks, E.H.; Srivastava, R.M. *An Introduction to Applied Geostatistics*; Oxford University Press: New York, NY, USA, 1989.
43. Cressie, N. *Statistics for Spatial Data*; John Wiley: New York, NY, USA, 1993.
44. OSM Contributors. Open Street Map. 2023. Available online: https://www.openstreetmap.org/ (accessed on 27 May 2023).
45. Boeing, G. OSMnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks. *Comput. Environ. Urban Syst.* **2017**, *65*, 126–139. [CrossRef]
46. Segura-Garcia, J.; Calero, J.M.A.; Pastor-Aparicio, A.; Marco-Alaez, R.; Felici-Castell, S.; Wang, Q. 5G IoT System for Real-Time Psycho-Acoustic Soundscape Monitoring in Smart Cities With Dynamic Computational Offloading to the Edge. *IEEE Internet Things J.* **2021**, *8*, 12467–12475. [CrossRef]

D Elastic computing in the fog on internet of things to improve the performance of low cost nodes

electronics

MDPI

Article

Elastic Computing in the Fog on Internet of Things to Improve the Performance of Low Cost Nodes

Rafael Fayos-Jordan †, Santiago Felici-Castell *,†, Jaume Segura-Garcia †, Adolfo Pastor-Aparicio † and Jesus Lopez-Ballester †

Departament de Informàtica, ETSE, Universitat de València, Avd. de la Universitat S/N, 46100 Burjassot, Spain; rafael.fayos@uv.es (R.F.-J.); jsegura@uv.es (J.S.-G.); adolfo.pastor@uv.es (A.P.-A.); jesus.lopez-ballester@uv.es (J.L.-B.)

* Correspondence: felici@uv.es

† These authors contributed equally to this work.

Received: 27 September 2019; Accepted: 2 December 2019; Published: 6 December 2019

Abstract: The Internet of Things (IoT) is a network widely used with the purpose of connecting almost everything, everywhere to the Internet. To cope with this goal, low cost nodes are being used; otherwise, it would be very expensive to expand so fast. These networks are set up with small distributed devices (nodes) that have a power supply, processing unit, memory, sensors, and wireless communications. In the market, we can find different alternatives for these devices, such as small board computers (SBCs), e.g., Raspberry Pi (RPi)), with different features. Usually these devices run a coarse version of a Linux operating system. Nevertheless, there are many scenarios that require enhanced computational power that these nodes alone are unable to provide. In this context, we need to introduce a kind of collaboration among the devices to overcome their constraints. We based our solution in a combination of clustering techniques (building a mesh network using their wireless capabilities); at the same time we try to orchestrate the resources in order to improve their processing capabilities in an elastic computing fashion. This paradigm is called fog computing on IoT. We propose in this paper the use of cloud computing technologies, such as Linux containers, based on Docker, and a container orchestration platform (COP) to run on the top of a cluster of these nodes, but adapted to the fog computing paradigm. Notice that these technologies are open source and developed for Linux operating system. As an example, in our results we show an IoT application for soundscape monitoring as a proof of concept that it will allow us to compare different alternatives in its design and implementation; in particular, with regard to the COP selection, between Docker Swarm and Kubernetes. We conclude that using and combining these techniques, we can improve the overall computation capabilities of these IoT nodes within a fog computing paradigm.

Keywords: fog computing; Kubernetes; Docker Swarm; containers; container orchestration platforms; elastic computing

1. Introduction

The Internet of Things (IoT) has become a widely used network to integrate almost everything, everywhere to the Internet, with different applications which include environmental monitoring, etc. [1–4], and with many open challenges [5,6]. These networks consist of distributed devices that have their own power supply, processing unit, memory, sensors and wireless communications. In addition, low cost devices are being used to make cheaper deployments. There is a wide range of these devices, such as TelosB motes [7] and small board computers (SBCs), such as Raspberry Pi (RPi) [8], with different technical features. Normally, the operating systems of such a device is based on a Linux system.

However, new trends on IoT and their applications make them to require a reconfigurable sensor architecture that can span multiple scenarios, requiring storage, networking and computational resources to be efficiently used at the edge of the network [2]. Thus, there are many scenarios that require enhanced computational power that these nodes alone are unable to provide, showing constraints in terms of limited energy, bandwidth, memory size and computational capabilities. In this context, we must make these devices cooperate to overcome these constraints by using orchestration techniques. In addition, it is necessary to implement a wireless mesh network within the cluster.

In addition, we need to orchestrate resources within the cluster to manage and improve overall processing capabilities in an elastic manner. This paradigm is in the context of fog computing on IoT. It is worth mentioning, that we can find slight differences from different documentary sources for the concept of fog computing, since it is a term not consolidated yet. In this context, we assume that fog computing is the collaboration of neighboring devices in order to improve their overall performance. Additionally, we can find similar definitions, such as, "Fog computing is an architecture that uses edge devices to carry out a substantial amount of computation, storage and communication locally".

It must be stressed that fog computing is one of the research fields that is gaining more importance and relevance. It has emerged to support the requirements of IoT applications that cannot be met by today's nodes [9]. Nevertheless, as discussed before, fog computing is not mature yet and there are not many solutions and available alternatives. We can find different applications and challenges of this technology in others areas [10]; for instance, security issues in vehicular networks [11].

Thus, we propose in this paper, a lightweight virtualization approach enabling flexibility and scalability within the cluster, by using Linux containers and a container orchestration platform (COP), but adapting them to a fog computing paradigm. All these technologies have been developed for a Linux operating system and they will be described in the following sections. The novelty of our proposal and our contribution is the use and combination of these technologies in this paradigm, since these technologies (containers and COPs) were initially defined for cloud computing, where both computers and interconnection networks have nothing to do with the SBC devices and wireless mesh network that we can find in an IoT deployment in terms of CPU, memory and network speed. In this context, the use and analysis of the mentioned technologies is new and there are not (to the best of our knowledge) any studies using them before, combining IoT, low cost devices, and fog computing. In addition, in order to exploit the results for this new approach we compared the different options in the design.

With more detail, the COP will manage the resources on the top of a cluster of these nodes, in the master. Linux container is a technology that allows an application to be broken into different containers that can be run on a single node or on a cluster of multiple nodes. Docker is currently the most used commodity container framework [12]. The reason behind of that orchestration is when the number of containers and nodes are high, we need to schedule and distribute the different tasks. Some COPs are widely used for container-based cluster management in cloud computing environments (but not so extended in fog computing), such as Docker Swarm [13], Marathon-Mesos [14] and Kubernetes [15]. All theses technologies are open source and designed for Linux operating systems. That is the reason we use SBC devices' running operating systems based on Linux.

Finally, we will show an implementation under these constraints applied to noise pollution and soundscape monitoring. The scenario is a real example of the aforementioned new trends of the IoT, requiring high computing without loss of generality. In addition, it will be used as a proof of concept, allowing us to compare different alternatives in its implementation; in particular, with regard to the COP selection between Docker Swarm and Kubernetes. From our results, we conclude that by using and combining the proposed techniques, we can improve the overall computation capabilities of these IoT nodes.

Eventually, with this paper we will try to answer the following question: *how can I increase the overall computational power of a set of SBC devices in an IoT deployment?*

The rest of the paper is structured as follows. Section 2 shows the related work. In Section 3, we define the requirements for the proposed architecture, the clustering options to coordinate the nodes in the fog, and an analysis of containers and orchestration platforms. In Section 4 we explain the design and implementation details. In Section 5, we analyze the test bed for psycho-acoustic annoyance/noise pollution monitoring, in order to highlight the enhancements introduced compared to traditional systems, showing the results from a performance evaluation with the different alternatives. In Section 6 we discuss the results obtained, and finally, in Section 7, we conclude the paper.

2. Related Work

Regarding cloud and fog computing on IoT, we found relevant references. In [9], a unified architectural model and taxonomy was presented by comparing a large number of solutions on fog computing for the IoT, taking into account several aspects, such as communications, security, management, and cloudification. The authors analyzed the main IoT applications requirements, and using the proposed model, they were able to compare different solutions and how they are applied. For instance, in the study presented in [1], the authors explore alternative deployments for a "smart warehouse" IoT application, both based on cloud and based on fog (fog-like) computing. The goal of their exercise was to determine if a cloud-based approach is able to meet the latency requirements of the application, given that low-latency is usually considered an essential requirement for many IoT applications. The authors were able to compare the event latency performance for both cloud and fog deployments, showing as it could be expected, that latency shows better results when the application is deployed according the fog-based approach.

Another approach of fog computing on IoT is shown in [2]. Their authors focus on the use of available gateways in order to enable IoT application deployments. The authors mention that there are a number of platforms and gateway architectures that have been proposed to manage these components. However, these platforms and gateways lack horizontal integration among multiple providers, and other functionalities like load balancing and clustering. The authors state that is partly due to the strongly coupled nature of the deployed applications, and a lack of abstraction of device communication layers as well as a lock-in for communication protocols. This limitation is a major obstacle for the development of a protocol agnostic application environment that allows for single application to be migrated and to work with multiple peripheral devices with varying protocols from different local gateways. Then, the authors propose a messaging-based modular gateway platform that enables clustering of gateways and the abstraction of peripheral communication protocol details. This proposal allows applications to send and receive messages regardless of their deployment location and destination device protocol, creating a uniform development environment.

Regarding management and orchestration within fog computing, we found interesting contributions. In [16], an efficient automated resource management in cloud computing was shown to improve important tasks such as launching, terminating, and maintaining computing instances rapidly, with a minimum overhead. The authors ran a performance analysis over Kubernetes using a Petri network based performance model. The authors suggested that the proposal could be used for supporting capacity planning and designing Kubernetes-based elastic applications.

Finally, it is worth mentioning similar works focused on the evaluation of the aforementioned COPs. In particular, in [17], the authors analyzed Kubernetes as a COP in order to design an on-demand model for renting computing resources and easy-to-use elastic infrastructure in cloud computing environments. The authors considered the choice of a reactive autoscaling method to adapt this demand. Kubernetes already embeds and autoscaling method but it significantly affects both response time and resource utilization. Then the authors discuss and suggest the use of different factors that should be taken into account under different types of traffic to develop new autoscaling methods. They conclude that the default autoscaling method in Kubernetes can be improved by considering the suggested influencing factors. These factors, which should be taken into consideration to handle different workload patterns, consist of (i) a conservative constant (α), (ii) an adaptation interval or

control loop time period (CLTP), and (iii) stopping at most one container instance in each adaptation interval. It must be stressed that the authors used as a testbed, a cluster of computers with four core CPUs at 2397 MHz, with 4GB of RAM and 1 Gbps network interfaces. Additionally, using Kubernetes, in [18] the authors focused on a new feature of this COP to support a federation function of multiple Docker container clusters, called Kubernetes Federation. It allows one to increase the responsiveness and reliability of cloud computing applications by distributing and federating container clusters to multiple service areas of cloud service providers. But the management required is high and complex. Thus, the authors proposed an interesting method and a tool to automatically form and monitor Kubernetes Federation.

To conclude this section, it is worth mentioning that fog-computing on IoT is not yet a mature research line and interesting tools developed for cloud computing could be considered and adapted to this new context. Although big efforts have been made, we cannot find many solutions and available alternatives, or a detailed architecture in order to follow some steps. Thus, with this paper we try to clarify different issues bound to this technology and its deployment through a case study.

3. Analysis of the System

We require an adaptive, reconfigurable, wireless, and scalable architecture in the fog, able to perform both simple and difficult tasks for IoT applications, that can span multiple scenarios, requiring storage, networking, and computational resources to be efficiently used at the edge of the network [2]. For this purpose, in this section we will analyze the clustering options to allow cooperation among the nodes and the Linux container technology to allow load balancing distribution.

3.1. Clustering Options for Nodes in the Fog

Since the IoT nodes will be close one each other in the fog within an area range lower than 50 m, we want a wireless solution, without requiring any additional external device except the nodes themselves. In particular, we will focus on RPi. RPi nodes are well-known and commonly used as SBCs in IoT applications. They fit in almost any coarse and initial IoT deployment. Thus, without loss of generality we consider them in order to make a real cluster, as a proof of concept.

There are several ways to interconnect these nodes in a cluster, as shown in Figure 1.

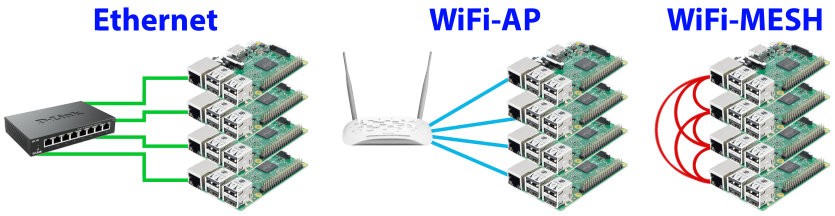


Figure 1. Different options for RPi clustering.

In terms of speed and performance, the wired option is the first and the most common interconnection method, using Ethernet interfaces and getting transmission speeds around 100 Mbps. But this requires adding network cables and an Ethernet switch. In this case, scalability would depend on this switch. Alternatively, using wireless interfaces, the nodes can connect by an access point (AP). This would give us a speed of approximately 40 Mbps in real scenarios [19]. But the limitations, apart from speed, are given by the AP itself, that can act as a bottleneck and requires a lot of energy. Besides, there are not direct connections among the nodes, since all traffic must necessarily go through the AP.

Thus, the last alternative shown in Figure 1 is a wireless mesh network. Since SBCs are equipped with WiFi interfaces, this option is feasible and even more interesting than the previous ones. WiFi mesh networks represent a simple and smart option for node clustering. However there is a constraint in these topologies due to the short communication range of these nodes, but in our case and in the fog we will not consider this a problem.

3.2. Linux Containers and Orchestration

The Linux container technology is a lightweight virtualization technology at the operating system level. Containers are an excellent option and can be booted up in few seconds; they are an efficient use of hardware resources. In this scenario, an application consists of numerous containers that can be run in one node or in a cluster of nodes. Thus, it is important for an orchestrator to keep track of containers belong to the same application and to deal with network connection. The orchestrator can manage hundreds or thousands of containers in a cluster.

Docker is an open source project to automate the deployment of applications within containers. Additionally, Docker is currently the most used commodity container framework; thus, we used this type of container.

When the number of containers in a cluster is high, new tools are needed for orchestration. Some orchestrators are widely used for container-based cluster management, such as Docker Swarm [13] and Kubernetes [15]. They are called container orchestration platforms (COPs). Thus, COPs are used to orchestrate the execution of containers in a cluster. The user describes the container and the COP selects which of the physical hosts or nodes are going to perform the execution of the container. It must be noticed that one could use the interfaces provided by a COP to directly deploy containers on a set of computing resources. Nevertheless, this approach would be disruptive since usage patterns would change. Next, we describe the most important available COPs and their features.

Kubernetes [15] is the most prominent open source orchestration system for Docker containers supported by Google and later donated to the Cloud Native Computing Foundation. It performs the scheduling in a computing cluster (Kubernetes cluster) made up with different nodes and actively manages workloads. In Kubernetes, there is a master, called Kubernetes master, to manage and orchestrate the cluster resources. It provides interesting features, such as reliable container restarting, load balancing, autoscaling, and self-healing. The scheduling in Kubernetes is based on *Pods*, that are groups of containers deployed and scheduled together. These *Pods* can be distributed among different nodes. Additionally, one single node can run several *Pods*. They form the atomic unit of scheduling as opposed to single containers in other systems. Containers within a *Pod* share an IP address and different labels can be used to identify each group of containers. These labeling features allow Kubernetes to work on a heterogeneous cluster, where different nodes are specialized to run specific *Pods*.

Docker Swarm [13], or simply Swarm, represents the native clustering approach proposed by Docker, and takes advantages of the standard Docker API. Swarm manages a set of resources to distribute Docker workloads, managing internal overlay networks within the containers. That way, a container-based virtual cluster can be easily created on top of virtual or physical resources. The architecture of Swarm consists of hosts running a Swarm agent (working nodes) and one host running a Swarm manager. The Swarm agent will run several Docker stacks and each one will accept containers. The concept of Docker stacks is equivalent in this paper to the *Pod* in Kubernetes. The manager is responsible for the orchestration and scheduling of containers on the agent nodes. While the connection between Swarm manager and agent is established by opening the port for the Docker daemon, the Swarm manager can access all existing containers and Docker images in the agent nodes. Additionally, it must be noticed that Swarm can be run in a high-availability mode.

4. Design and Implementation

In this section we describe the design and implementation of the cluster to perform elastic computing in the fog.

4.1. Hardware, Operating System, and Network Configuration of the Nodes

We used four RPi version 3B [8] as the hardware base of the cluster, with its respective power supplies and microSD cards. RPi version 3B uses an ARM Cortex-A53 @ 1.2GHz and 1GB LPDDR2 of

RAM. Figure 2 shows an example of a cluster using four RPis. In this figure, we see also that the cluster is performing a soundscape monitoring and this information is visualized on a map. This scenario is described later in Section 5. In the cluster, the assigned names for the nodes in the cluster are RPiC1 as master, and as slaves—RPiC2, RPiC3, and RPiC4. We used the default operating system provided by the manufacturer, called Raspbian [20]. Raspbian was developed by Raspberry Pi and based on a Linux Debian distribution. We used version 9, named "Stretch".

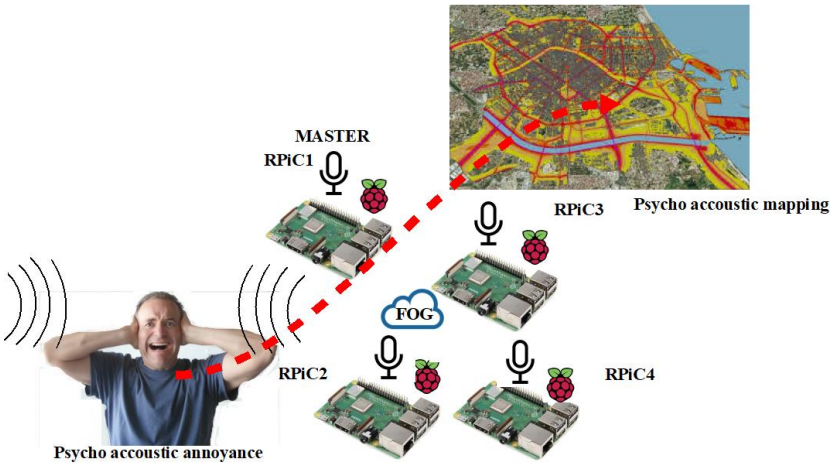


Figure 2. Soundscape monitoring scenario using a fog computing approach based on a cluster of Raspberry Pi 3B within a wireless mesh network, one acting as master (RPiC1, with external access) and the others as slaves (RPiC2-4). The information we gathered is visualized on a map.

For simplicity and scalability within the whole architecture, we relied on the wireless mesh infrastructure placing the nodes within a range lower than 50 m. In this scenario, a multihop routing protocol is required to interconnect the different nodes within the network and/or the cluster.

There are many multihop routing protocols. From all of them, based on our experience and our requirements, better approach to mobile ad hoc networking (BATMAN) [21] is very reliable, stable, well-known, and has little in the way of an overhead.

BATMAN is a proactive routing protocol based on distance-vector algorithm, that builds a loop free network. BATMAN does not try to estimate the whole path to each destination, but only the best next-hop to a given destination, avoiding unnecessary routing-information exchanges among nodes. If the origin and destination nodes are close, it creates direct routes. This is especially relevant in a cluster, since it allows direct communication among the nodes, without any additional intermediate node (relying nodes), speeding up the transmission of packets and minimizing errors. It must be stressed that BATMAN works in Layer 2; thus, the whole multihop network is a LAN and each node is identified by its own MAC address, simplifying the nodal configuration.

Finally, to allow Internet connection for the whole system, the master acts as a router (default gateway) for the nodes. The master must use another interface for this purpose, such as an Ethernet card, a second wireless card, or any telco adapter.

4.2. Cluster Configuration to Manage Containers and Their Orchestration

In order to perform the load distribution of the different tasks for an IoT application within the cluster, we use Docker containers. Docker works by downloading images from a repository (with authentication), that at the same time is in a container too, customizing and executing them in the system. This local repository is created at the master node where the slaves have access and all the images that the slaves will run at the master's request are available.

For the COP selection, as discussed before for the orchestration, we focus on Kubernetes and Swarm as COPs, since they are common.

Regarding Kubernetes configuration, first we will start up the node that will act as orchestrator and we will specify the address where the Kubernetes service will be available for the slaves, as well as the IP range where the *Pods* will be executed. In addition, we have to share a token between the master and the slaves in order to authenticate the process. It must be stressed that we will not enable the autoscaling, since the number of *Pods* will be assigned before hand. It does not mean that Kubernetes will not adapt to the node status but it will not autoscale using its default mechanism. In this case, we will schedule from the master the load distribution by using port forwarding to the slaves. Additionally, we have not included the default monitoring process embedded in Kubernetes because in the RPi, this process increases in excess the CPU use, around a 60% extra.

Regarding Swarm configuration, things are easier compared to Kubernetes since it is the native orchestrator of Docker containers and it is embedded in it. Following the same steps that in Kubernetes, we publish the address at the master where the service is available for the slaves. This was to provide a token to bind the slaves.

5. Test Bed for Soundscape Monitoring and Its Performance Analysis

As a proof of concept and in order to highlight the advantages of the proposed fog computing scheme, in this section we analyze a deployment that requires high computational costs applied to psycho-acoustic annoyance (PA) monitoring, or soundscape monitoring, as shown in Figure 2. We used this scenario as a real example to performance elastic computing on a cluster of RPi, without loss of generality in our results. This IoT application for soundscape monitoring is an example of the new trends on IoT, as mentioned before. Thus, in this case we see the challenges imposed by these low cost devices (RPi) when requiring enhanced computational power that these devices alone are unable to provide, and that they need to collaborate in a fog computing paradigm.

Soundscape monitoring is characterized by requiring tough signal processing algorithms. These algorithms are explained and analyzed in [22,23]. Behind this monitoring process, there are several rules and standards such as Environmental Noise Directives (ENDs) 2002/49/EC and ISO 12913 (soundscape) [24,25]. In particular, END 2002/49/EC requires main cities (with more than 250,000 inhabitants) to gather real data on noise exposure in order to produce local action plans and to provide accurate real time mappings of noise pollution levels.

The evaluation of this annoyance (or PA) is mainly based on the work by Zwicker and Fastl [26], defining a set of parameters such as loudness (L), sharpness (S), fluctuation strength (F), and roughness (R) that will let us measure PA. In other words, PA is given by a function knowing L, S, F, and R, and we need all of them before calculating PA. The flow diagram of these algorithms is shown in Figure 3. Due to the complexity of their implementation, we cannot perform this monitoring process with conventional hardware (such as RPi) in a simple IoT deployment, and less so in real time. We must highlight that we refer to real time as when the time required to process an audio chunk is shorter than the chunk itself, which is by default 1 second.

It must be stressed in these scenarios that due to stringent laws related to personal privacy, it is essential that PA calculations must be performed in real-time and as close as possible to the source within the communication range, because the data are not allowed be to sent the recorded audio out.

First, we carry out the performance analysis in two steps. First, we analyze parallelization and granularity issues of the containers, and secondly, we analyze its throughput.

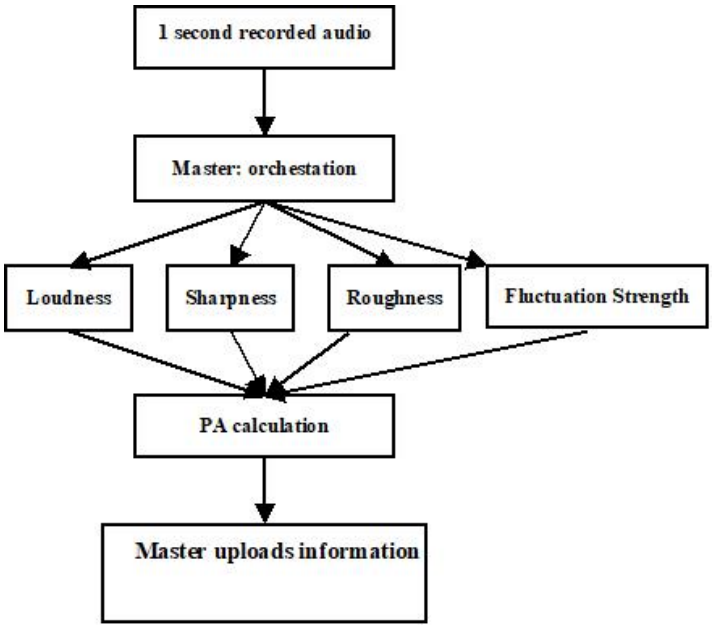


Figure 3. Flow diagram to measure and calculate psycho-acoustic annoyance (PA) on the proposed novel, fog computing architecture based on L, S, R, and F.

5.1. Analyzing Parallelization and Granularity

Thus, first we characterize the processing time of the acoustic parameters. We created as many containers as psycho-acoustic parameters to allow their parallelization and to speed up the PA calculation. In this case, our aim was to increase the granularity of the containers (to make them small and one per parameter) in order to distribute them easily and lightly, among the different slaves in the cluster, to reduce the processing time.

In this scenario, there is a client node (external node) that records audio chunks and sends them at a certain rate to the system under test (SUT). We used 100 audio chunks (of daily sounds of one second) randomly selected from 60,150 chunks, recorded beforehand using a USB microphone. This process was repeated several times till we achieved a confidence interval of 95%. The client timestamped each audio chunk, and then it sent it to the SUT by WiFi. Once each audio chunk was processed, the SUT sent the results back to the client. Once the client received them, it measured the total computation time, as the elapsed time between the audio chunk was sent (and time stamped) and the results were received. The SUT ran a REST-API and for a comparison; we used different approaches to compute these parameters: (a) in a computer as a base line, (b) in a single RPi, and (c) in a cluster of four RPis using the proposed fog computing.

In the case of the computer, we ran the parameters on one computer using both Matlab and C++/Python. The computer was an i7-7700HQ at 3.5 GHz and 16GB DDR4 of RAM with eight cores.

In the case of single node (RPi) and fog computing (cluster of RPi), we used RPi models 3B and 3B+, using only C++/Python code. In the single node (RPi) approach, only one RPi was computing all the parameters. This is the worst case, the slowest approach. In the cluster (fog approach), we had four RPi3B, one as master and three as slaves. In this case, we were running the parameters (in different containers), based on Swarm. All of the RPis were close one each other. The master in the cluster was in charge of orchestrating the different slaves. RPi 3B used an ARM Cortex-A53 at 1.2 GHz and 1 GB LPDDR2 of RAM and RPi 3B+ used an ARM Cortex-A53 at 1.4 GHz and 1 GB LPDDR2 of RAM, all of them with four cores. It must be noted that with RPis, the main program was based on Python, and we used C++ in order to implement a Python library that performed all the tough processing from each psycho-acoustic parameter in an efficient way by using the linear algebra library called Armadillo [27]. This code has not been programmed using threads.

In Table 1, we summarize the computational times for L, S, F, R, and PA on average per each second of recorded audio, using different devices and programming languages and specifying these approaches, specifying both the computer, single devices (one RPi), and a cluster of four RPis. This table only shows average values in seconds and the standard deviation is at least three orders of magnitude smaller; the highest is given by RPi working as a single node. The computation time is nearly constant for all of them independently of the audio chunk (all of them of one second). The communication times are included in each approach. In addition, as it could be expected, we saw that the computer outperformed any combination with RPis. Additionally, we saw the effect of parallelization in the cluster, where the total time was approximately the longest time of the parameters (R is the toughest), compared with the total time required on a single RPi. In the cluster, because each psycho acoustic parameter (L, S, F, and R) can be processed independently, we could parallelize them, making the final computation of the PA (the parameter that includes all of them) faster. In particular, we saw that when using a single and isolated RPi, the results (total time for PA estimation) were the worst ones, 1.479 and 1.406 seconds for RPi3B and RPi3B+ respectively. These times are far from a real time execution, which should be less than one second (the duration of the audio chunk). However, when using fog computing in the cluster, we took advantage of their parallelization and those times were reduced to 0.875 and 0.824 seconds for RPi3B and RPi3B+ respectively. These results initially validate our approach.

Table 1. Time comparison in seconds to calculate the pyscho-acoustic parameters (loudness (L), sharpness (S), fluctuation strength (F), roughness (R), and PA (total time)), among different devices and programming languages using different types of approach: a computer (base line), a single node (RPi), and a cluster with four RPis (fog computing approach).

	Type	**L**	**S**	**R**	**F**	**PA (Total)**
Matlab	*Computer*	0.058	0.000	0.288	0.404	0.699
C++/Python	*Computer*	0.003	0.000	0.128	0.235	0.238
RPi3B	*Single node*	0.018	0.000	0.849	0.742	1.479
RPi3B+	*Single node*	0.017	0.000	0.794	0.694	1.406
RPi3B	*Cluster*	0.022	0.000	0.853	0.754	0.875
RPi3B+	*Cluster*	0.021	0.000	0.802	0.703	0.824

5.2. Performance Evaluation of the Cluster in the Fog

Now, we will focus on the cluster and its capabilities. We performed the same calculations as before, but embedding all the parameters within the same container, in order to measure the total throughput of the whole cluster, comparing Swarm and Kubernetes, with the aim of measuring its performance. We will refer to the concept of Docker stacks from Swarm as *Pod* as in Kubernetes, since both are equivalent.

We used the same testbed as before, with one client node recording audio chunks and the cluster (the SUT) processing all of them. The cluster was set up with four RPi 3B, one node acting as a master and three slaves. We embedded all the parameters within the same container and the master distributed this container among the slaves and their different *Pods*.

In the testbed, the external client timestamped each audio chunk and sent it to the master by WiFi on the mesh network at a certain rate (λ or audio chunks per second). The master distributed the audio chunks among the slaves by WiFi too. The slaves were running a REST-API and received requests from the master. The slaves kept the audio chunks assigned temporarily, if they were busy (while working with other audio chunks), in a FIFO queue implemented internally in the REST-API implementation. If the audio chunks had to wait for processing, the queuing time depended on the number of pending audio chunks. Each slave processed their assigned audio chunks one by one and sent the results (acoustic parameters and PA) back to the master, every time an audio chunk was processed. The master forwarded them to the client. Once the client received each result, it measured

the total computation time, as before. It must be stressed that each computation time per audio chunk includes: the sending time from the client to the master and from the master to the slave.

As seen in Table 1 in a RPi3B, each audio chunk takes an average of 1479 ms to calculate PA, with a standard deviation of 1.405×10^{-4} (it is near a deterministic time). If we consider 1479 ms, as the average service time, equal to $\frac{1}{\mu}$, the service rate (μ) is 0.676 chunks/second. Besides, we will call the number of available *Pods* c. The number of *Pods* is defined by configuration before hand at each slave in the cluster. In this case, we can model this approach as $D/D/c$ according to the Kendall notation. That is, a deterministic arrival process (since audio chunks are recorded one per second) and a deterministic time service with c cores (number of *Pods*). Because each RPi had four cores, while in the cluster we had one, two, or three slaves, we got four, eight, or 12 cores, maximum. Each core can run a *Pod*. Thus, with one slave c is in the range $[1, ..., 4]$; with two slaves c is within $[2, ..., 8]$; and with three slaves c is within $[3, ..., 12]$.

In particular, we evaluated the performance of the cluster with and without congestion.

On one hand, if $\lambda \geq c \cdot \mu$, we have congestion. In this case, each slave running a REST-API will keep the audio chunks in RAM memory. The slaves have memory enough to keep the audio chunks in RAM. The global queue will grow as $\lambda - c \cdot \mu$ and it will be distributed among the slaves. We test λ in steps of 0.1 chunks/second from values greater than $c \cdot \mu$, with a maximum of 10 chunks/second. In all the scenarios, we sent the same workload, 100 audio chunks at the given rate several times till we met the confidence interval. Notice that we always create congestion at each slave once its cores are busy with their first audio chunk. The idea is to observe how the system behaves under congestion. Additionally, it must be stressed that the client is configured without time outs in the application (REST-API) because it would produce duplicated chunks. Then, the cluster is behaving as a conservative system.

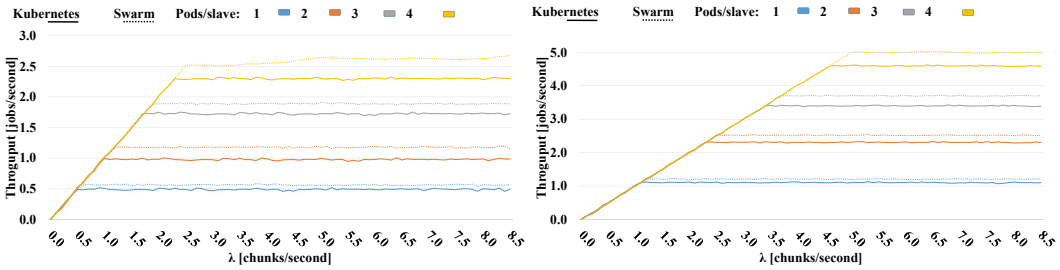
On the other hand, without congestion, the input workload would be lower than the output workload processed, avoiding any queue. In the scenario without congestion, $\lambda < c \cdot \mu$, we tested λ from 0.1 chunks/second till $c \cdot \mu$. Figure 4a–c shows the input workload (λ in audio chunks per second) compared to throughput (audio chunks processed (or jobs) per second), comparing Kubernetes to Swarm with *Pods*/slave and different slaves ((Figure 4a) one slave, (Figure 4b) two slaves, and (Figure 4c) three slaves). In this case, in all the scenarios when the total number of *Pods* were busy, as could be expected, the scenario became congested, although the systems showed a constant throughput since each node kept their tasks in RAM in a FIFO fashion. It must be noticed that Swarm shows a greater throughput according to the previous results, around 10% more, and this enhancement increases with the number of *Pods*.

In addition, we compare in Figure 5a–c, the average resource uses in terms of percentage of CPU, percentage of RAM memory, and CPU temperature, respectively.

In terms of percentage of CPU (Figure 5a) both COPs have a similar behavior, increasing with the number of *Pods*. They show a minimum of 35% approximately, adding 20% approximately when each core is activated. In theory, because the RPi has four cores, every time a core is completely busy the CPU will increase 25%. From these results, Swarm shows a slightly greater use of CPU, because the cores have less idle time providing higher utilization. The maximum was reached by Swarm with a 95% (with four cores working) and Kuberentes had 10% less. Notice that we should take into account the initial resource use without any service (by default), with both COPs.In this case, Kubernetes used in the master, 17% CPU and 74% RAM, and in the slaves, 6% CPU and 33% RAM. In the same conditions, Swarm used in the master, 0.7% CPU and 22.48% RAM, and in the slaves, 0.53% CPU and 16.16% RAM.

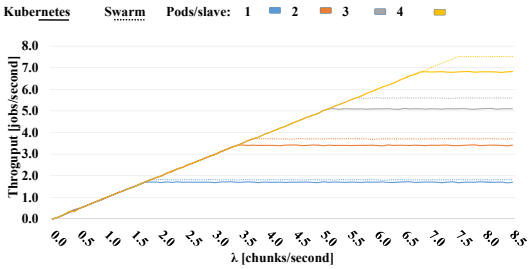
In terms of percentage of RAM memory (Figure 5b), things are different because Kubernetes usually uses 15% approximately more memory. In part, this is due to a lower throughput, as we saw in Figure 4a–c, and then it has to keep the audio chunks in memory in the meantime.

Finally, for the CPU temperature (Figure 5c), according to the CPU use, Swarm gets between 2 and 3 degrees higher than Kubernetes in all the combinations.



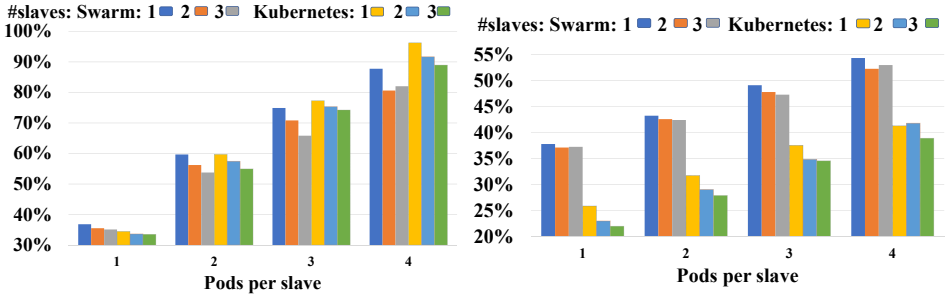
(**a**) 1 slave

(**b**) 2 slaves



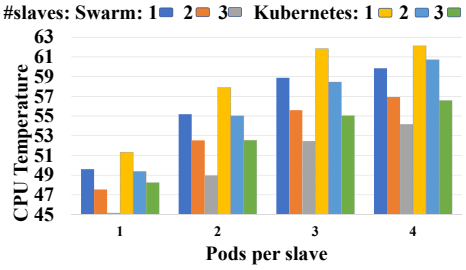
(**c**) 3 slaves

Figure 4. Throughput comparing Kubernetes vs Swarm with different slaves ((**a**) 1, (**b**) 2 and (**c**) 3) and *Pods*/slave.



(**a**) % CPU

(**b**) % RAM



(**c**) Temperature

Figure 5. Resource use comparing Kubernetes to Swarm with different slaves and *Pods*/slaves: (**a**) percentage of CPU, (**b**) percentage of RAM memory, and (**c**) temperature.

6. Discussion

On one hand, as we can see from Section 5.1, by reducing the granularity of the containers we decrease the processing time, as could be expected, in the proposed fog computing architecture. But in this case, because the different running containers require sharing the same audio chunk, it increases the input data overhead that makes it inefficient in terms of throughput. The bottleneck is given by the mesh network and the communication processes within the cluster. For that reason, on the other hand in Section 5.2, we also evaluated an approach where one container included all the different PA parameters. In this scenario, the overhead was reduced, allowing a higher throughput, and the limits were imposed by the nodes and the number of cores. It must be noted that in this case, the cluster has one master and three slaves based on RPi 3B (with four cores each), that will limit the number of *Pods* running efficiently at each slave.

Additionally, we compared Swarm to Kubernetes. We have seen that Swarm always outperforms Kubernetes on RPi with the different metrics used (memory and CPU use, throughput, etc.). Swarm is more efficient (10% approximately) and faster on RPi3B nodes in all the scenarios and for the different evaluated metrics.

7. Conclusions

Nowadays IoT is requiring a flexible and scalable network design but built on low cost nodes to access and connect almost everything at everywhere. In particular in this paper, we focused on nodes such as RPi. But with the new trends, the IoT is facing many scenarios that require high computation capabilities beyond the possibilities of these nodes alone. For this goal, we proposed an architecture leveraging fog computing, based on Linux containers and an orchestration platform, to run on the top of a cluster of these nodes in order to cooperate and schedule different tasks in an efficient way.

As a proof of concept, we showed a scenario that requires high computing requirements, such as soundscape monitoring, and compared different alternatives in its implementation. With these results, we conclude that using and combining clustering techniques, Linux containers, and an orchestrator, we can improve the overall computation capabilities of these IoT nodes. We have used Linux Docker containers and compared two different COP as orchestrators, such as Docker Swarm and Kubernetes. The experimental results showed the improved performance in terms of execution time and throughput in a cluster of four RPis. We have seen that Docker Swarm always outperforms Kubernetes in this scenario.

Thus, finally, an educated answer for "how can I increase the overall computational power of a set of SBC devices in an IoT deployment?" is: We can overcome the constraints imposed by single SBC devices using a cluster of interconnected nodes in a wireless mesh networks, since these devices have wireless capabilities. In addition, in order to improve the overall computational power, the use of Linux Docker containers add flexibility, adaptability, and responsiveness. However, it is necessary to orchestrate the resources among the nodes in the cluster by using a COP; in particular, Docker Swarm performs best.

Author Contributions: conceptualization, methodology, and validation by R.F.-J. and S.F.-C.; software, A.P.-A.; resources, J.S.-G.; writing—original draft preparation, S.F.-C.; writing—review and editing, J.L.-B.

Funding: This work was supported by the Ministry of Economy under the project Urbauramon BIA2016-76957-C3-1-R and a grant by University of Valencia UV-INV-EPDI19-995284.

Conflicts of Interest: The authors declare no conflict of interest

References

1. Gomes, M.; Pardal, M.L. Cloud vs. Fog: Assessment of alternative deployments for a latency-sensitive IoT application. *Procedia Comput. Sci.* **2018**, *130*, 488–495. [CrossRef]
2. Verba, N.; Chao, K.M.; James, A.; Goldsmith, D.; Fei, X.; Stan, S.D. Platform as a service gateway for the Fog of Things. *Adv. Eng. Inform.* **2017**, *33*, 243–257. [CrossRef]

3. Pastor-Aparicio, A.; Segura-Garcia, J.; Lopez-Ballester, J.; Felici-Castell, S.; Garcia-Pineda, M.; Pérez-Solano, J.J. Psycho-Acoustic Annoyance Implementation with Wireless Acoustic Sensor Networks for Monitoring in Smart Cities. *IEEE Internet Things J.* **2019**. [CrossRef]
4. Segura-Garcia, J.; Perez-Solano, J.J.; Cobos, M.; Navarro, E.; Felici-Castell, S.; Soriano, A.; Montes, F. Spatial Statistical Analysis of Urban Noise Data from a WASN Gathered by an IoT System: Application to a Small City. *Appl. Sci.* **2016**, *6*, 380. [CrossRef]
5. Pérez-Solano, J.J.; Felici-Castell, S. Improving time synchronization in Wireless Sensor Networks using Bayesian Inference. *J. Netw. Comput. Appl.* **2017**, *82*, 47–55. [CrossRef]
6. Cobos, M.; Perez-Solano, J.J.; Felici-Castell, S.; Segura, J.; Navarro, J.M. Cumulative-Sum-Based Localization of Sound Events in Low-Cost Wireless Acoustic Sensor Networks. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2014**, *22*, 1792–1802. [CrossRef]
7. Polastre, J.; Szewczyk, R.; Culler, D. Telos: Enabling Ultra-low Power Wireless Research. In Proceedings of the 4th International Symposium on Information Processing in Sensor Networks, Los Angeles, CA, USA, 24–27 April 2005; IEEE Press: Piscataway, NJ, USA, 2005; pp. 364–369.
8. Raspberry Pi 3B+. 2018. Available online: https://www.raspberrypi.org/documentation/ (accessed on 2 January 2019).
9. Bellavista, P.; Berrocal, J.; Corradi, A.; Das, S.K.; Foschini, L.; Zanni, A. A survey on fog computing for the Internet of Things. *Pervasive Mob. Comput.* **2019**, *52*, 71–99. [CrossRef]
10. Mukherjee, M.; Shu, L.; Wang, D. Survey of Fog Computing: Fundamental, Network Applications, and Research Challenges. *IEEE Commun. Surv. Tutor.* **2018**, *20*, 1826–1857. [CrossRef]
11. Erskine, S.K.; Elleithy, K.M. Secure Intelligent Vehicular Network Using Fog Computing. *Electronics* **2019**, *8*. [CrossRef]
12. Docker Containers. 2018. Available online: https://docs.docker.com/engine/examples (accessed on 2 March 2019).
13. Docker Swarm. 2017. Available online: https://docs.docker.com/swarm (accessed on 11 November 2018).
14. Apache Mesos. 2018. Available online: http://mesos.apache.org (accessed on 11 January 2019).
15. Kubernetes (K8s). 2018. Available online: https://kubernetes.io (accessed on 11 October 2018).
16. Medel, V.; Tolosana-Calasanz, R.; Ángel Bañares, J.; Arronategui, U.; Rana, O.F. Characterising resource management performance in Kubernetes. *Comput. Electr. Eng.* **2018**, *68*, 286–297. [CrossRef]
17. Taherizadeh, S.; Grobelnik, M. Key influencing factors of the Kubernetes auto-scaler for computing-intensive microservice-native cloud-based applications. *Adv. Eng. Softw.* **2020**, *140*, 102734. [CrossRef]
18. Kim, D.; Muhammad, H.; Kim, E.; Helal, S.; Lee, C. TOSCA-Based and Federation-Aware Cloud Orchestration for Kubernetes Container Platform. *Appl. Sci.* **2019**, *9*, 191. [CrossRef]
19. Strazdins, G.; Elsts, A.; Nesenbergs, K.; Selavo, L. Wireless Sensor Network Operating System Design Rules Based on Real-World Deployment Survey. *J. Sens. Actuator Netw.* **2013**, *2*, 509–556. [CrossRef]
20. Foundation, R.P. Download Raspbian for Raspberry Pi, 2019. Available online: https://www.raspberrypi.org/downloads/raspbian/ (accessed on 3 December 2019).
21. Neumann, A.; Aichele, C.; Lindner, M.; Wunderlich, S. Better Approach To Mobile Ad-hoc Networking (B.A.T.M.A.N.). IETF Draft. Available online: https://tools.ietf.org/html/draft-wunderlich-openmesh-manet-routing-00 (accessed on 3 December 2019).
22. Noriega-Linares, J.E.; Rodriguez-Mayol, A.; Cobos, M.; Segura-García, J.; Felici-Castell, S.; Navarro, J.M. A Wireless Acoustic Array System for Binaural Loudness Evaluation in Cities. *IEEE Sens. J.* **2017**, *17*, 7043–7052. [CrossRef]
23. Pastor-Aparicio, A.; Lopez-Ballester, J.; Segura-Garcia, J.; Felici-Castell, S.; Cobos, M.; Fayos-Jordan, R.; Perez-Solano, J. Zwicker's annoyance model implementation in a WASN node. In Proceedings of the INTER-NOISE and NOISE-CON Congress and Conference Proceedings, Inter-Noise 2019, Madrid, Spain, 16–19 June 2019; Volume 1, pp. 1–11.
24. ISO. *Acoustics—Soundscape—Part 1: Definition and Conceptual Framework*; ISO 12913-1:2014; International Organization for Standardization: Geneva, Switzerland, 2014.
25. ISO. *Acoustics—Soundscape—Part 2: Data Collection and Reporting Requirements*; ISO 12913-2:2018; International Organization for Standardization: Geneva, Switzerland, 2018.

26. Fastl, H.; Zwicker, E. *Psychoacoustics: Facts and Models*; Springer Series in Information Sciences; Springer-Verlag: Berlin/Heidelberg, Germany, 2007; Volume 22.
27. Sanderson, C.; Curtin, R. A User-Friendly Hybrid Sparse Matrix Class in C++. *Lect. Notes Comput. Sci. (LNCS)* **2018**, *10931*, 422–430.

E Performance comparison of container orchestration platforms with low cost devices in the fog, assisting Internet of Things applications

Journal of Network and Computer Applications 169 (2020) 102788

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

Performance comparison of container orchestration platforms with low cost devices in the fog, assisting Internet of Things applications

Rafael Fayos-Jordan, Santiago Felici-Castell *, Jaume Segura-Garcia, Jesus Lopez-Ballester, Maximo Cobos

Departament de Informàtica, ETSE, Universitat de València, Avd. de la Universitat S/N, 46100 Burjassot, (Valencia), Spain

ARTICLE INFO

Keywords:
Fog computing
Docker swarm
Kubernetes
Containers
Container orchestration platforms
Performance comparison

ABSTRACT

In the last decade there has been an increasing interest and demand on the Internet of Things (IoT) and its applications. But, when a high level of computing and/or real time processing is required for these applications, different problems arise due to their requirements. In this context, low cost autonomous and distributed Small Board Computers (SBC) devices, with processing, storage capabilities and wireless communications can assist these IoT networks. Usually, these SBC devices run an operating system based on Linux. In this scenario, container-based technologies and fog computing are an interesting approach and both have led to a new paradigm in how devices cooperate, improving overall capacity in a cluster of these SBC devices. The use of containers is considered a lightweight virtualization, allowing an application to be broken into small tasks as services, enabling load balancing, flexibility and scalability. Nevertheless when the number of devices and containers increases in the cluster, it is required an orchestration layer. There are not many solutions and available alternatives using these technologies applied on these networks, and less an assessment of their performances. This paper focuses on these technologies when we use fog computing with low cost SBC devices in a context of IoT. We use Linux containers and different available orchestration platforms (in particular Docker Swarm and Kubernetes), to run on the top of the cluster of commercial SBC devices. Thus, we carry out a thorough functional and performance comparison with different real topologies (wired and wireless) and using both homogeneous and heterogeneous clusters of SBC devices, showing their results. We conclude that with the collected experimental results, Docker Swarm orchestration platform outperforms its counterparts in the scenarios shown.

1. Introduction

In the last decade there has been a proliferation of Wireless Sensor Networks (WSN) and their applications, that later were integrated with Internet and re-named as Internet of Things (IoT). These networks are used to monitor everything almost everywhere, and have been widely used, focused on smart cities for environment and acoustic monitoring, as seen in Gomes and Pardal (2018), Cobos et al. (2014).

Nevertheless, new tendencies on these networks are requiring improved as well as more complex applications, making them to require a flexible architecture that can cover different scenarios, requiring more storage, computational resources and/or enhanced communications (Verba et al., 2017), to name a few. In addition, these requirements are needed near to the deployment itself to avoid extra communication delays, which it is translated into more complexity, because many times we are not able to increase the computing capacity and keep pace with these new requirements on demand at these locations.

In this context, low cost autonomous and distributed Small Board Computers (SBC) devices, with processing, storage capabilities and wireless communications can assist these IoT networks and their applications, allowing us to meet these new requirements (Bellavista et al., 2019). For this purpose, container-based technologies and fog computing are an interesting approach and both have led to a new paradigm in how devices cooperate and can improve overall capacity in a cluster of these SBC devices. All these things are driving the design of new computing paradigms by offloading computing complexity onto this cluster.

In the market, we can find different alternatives for these SBC devices such as Raspberry Pi (RPi) (2018), Tinker (2019), Udoo (2019)

* Corresponding author.
E-mail addresses: rafael.fayos@uv.es (R. Fayos-Jordan), felici@uv.es (S. Felici-Castell), jsegura@uv.es (J. Segura-Garcia), jesus.lopez-ballester@uv.es (J. Lopez-Ballester), maximo.cobos@uv.es (M. Cobos).

https://doi.org/10.1016/j.jnca.2020.102788
Received 20 December 2019; Received in revised form 14 July 2020; Accepted 27 July 2020
Available online 6 August 2020

and Jetson (2018) boards, just to cite some of the most commonly used. Usually, these devices run an operating system based on Linux.

Besides, container-based technologies have an important key to enable quickly and easily services at low overhead, enabling load balancing, flexibility and fine-grained scalability. The use of containers is considered a lightweight virtualization, allowing an application to be broken into small tasks, each task as a service in a different container. Then an application consists of different containers that can be run in one device or in the cluster. Docker is currently the most used commodity container framework (Docker Containers, 2018).

However, the number of containers and slaves in the cluster could difficult their deployment and would require further orchestration capabilities. It is important for an orchestrator to keep a record of containers belonging to the same application. The orchestrator can manage hundreds or thousands of containers in a cluster. But with this approach in a fog computing paradigm, much more research is needed, as it is not yet ready and there are neither many solutions nor alternatives.

Our study is focused on a cluster of commercial SBC devices, where there is a master that will schedule and plan the different activities using a Container Orchestration Platform (COP) and the slaves, that will carry out the different tasks assigned. For this purpose and due to the constraints of these SBC devices, the master will be in charge of monitoring periodically the status of the devices, measuring their memory and CPU availability.

Some COPs are widely used for container-based cluster management, such as Docker Swarm (2017) or simply Swarm, Kubernetes (2018), Marathon-Mesos (Apache Mesos, 2018), etc. just to cite some of the most diffused ones. To the best of our knowledge, focusing on container orchestrators there are not assessments of their performances, and less in the context of fog computing. In this paper, we carry out a thorough functional and performance comparison with these container orchestrators under different real topologies in the cluster, using wired and wireless infrastructures, with multi-brand SBC devices. In addition, the performance evaluation of these orchestrators is done using an IoT application (described in Section 5), applied to psycho-acoustic annoyance/noise pollution monitoring. This study and analysis serve as a proof of concept of these technologies in the fog computing context as well. It must be noticed that for a fair comparison and clear results, since these SBC devices are quite simple, we avoid excessive congestion on them as well as we avoid bottlenecks in the network infrastructure. These issues are out of the scope of this paper.

The rest of the paper is structured as follows. Section 2 shows the related work. In Section 3, we analyze containers and orchestration platforms. In Section 4, we describe the commercial SBC devices used and their specifications, as well as the different options to build a cluster with these devices. In Section 5, we show a soundscape application (psycho-acoustic annoyance/noise pollution monitoring) and explain its requirements and constraints in terms of high level of computing. In Section 6 we specify the testbed for the evaluation and in Section 7 we present and discuss the results obtained. Finally in Section 8, we conclude the paper.

2. Related work

Cloud and fog computing have brought a new era in how devices interact with the Internet, allowing the design and deployment of many IoT applications. However, the increasing communication and processing demands that these systems face to, have required different approaches and solutions. In Zao et al. (2018), it is defined an OpenFog Reference Architecture where fog computing provides the missing link in the cloud-to-thing continuum, by moving compute, storage, communication, control, and decision making closer to the network edge (far from the core) where data is being generated, in order to alleviate the limitations at the edge in current infrastructures. A unified model and taxonomy from a fog computing point of view is presented in Bellavista et al. (2019), considering aspects such as communications, security, privacy, management and cloudification. The authors analyze the main IoT applications requirements and using the proposed model, they are able to compare different solutions and how they are applied to specific domains.

Using this taxonomy, in Gomes and Pardal (2018) the authors describe a "smart warehouse" application using different technologies to compare their performance, using both a cloud and fog (fog-like) computing approach. The goal is to analyze how both can meet in this case the latency requirements, given that lower values of latency are usually necessary. The authors show and conclude as it could be expected, that latency gets lower values when the application is designed in a fog-based approach. In a similar way comparing both technologies, the authors of Alam et al. (2019) use computation offloading as a protuberant elucidation for the resource-constrained mobile devices. In this case, mobile cloud is used as offloading platform in a far-end network, to leverage computation of the resource-constrained mobile devices. But because of this far-end network solution, the user devices experience higher delays, which negatively impacts real-time IoT applications. Thus, the authors propose a near-end network solution of computation offloading in mobile edge/fog. For handling the computation resource demand from the devices, they propose an algorithm using deep Q-learning through Markov decision process.

Another interesting case of fog computing is introduced in Verba et al. (2017), where the authors discuss the use of available gateways as a solution to allow enhanced IoT applications. It is described in this article that there are many platforms and gateways proposed, but they lack horizontal integration as well as other functionalities such as clustering, load balancing, etc. One of the main reasons for this problem is due to the strongly coupled nature of the deployments and applications. It is highlighted that there is a lack of abstraction of communication layers and a lock-in for their protocols. Thus, in this case the authors propose a messaging based modular gateway platform, by using Open Service Gateway Interface, that enables clustering of gateways and the abstraction of communication protocols. This proposal will facilitate the exchange of messages, independently of the IoT deployment location and destination device protocol, creating a uniform environment.

In reference to orchestration tools and platforms, we can also find interesting contributions. The authors in Medel et al. (2018) explain an automated resource management for a cloud computing environment based on Kubernetes, to enhance traditional tasks such as launch, maintain and terminate containers with reduced overhead. For this issue, the authors carry out a performance analysis using Petri networks, suggesting that this could be interesting for sharing and scheduling capacity plan while designing elastic applications. In the same way, to enhance overall computer performance, another example is shown in de Alfonso et al. (2017). In this case, Docker containers are used to support the creation of virtual elastic computer clusters for the execution of scientific applications. The authors use the open-source EC4 Docker tool to support the deployment of such clusters managed by Docker Swarm. Their authors demonstrate the feasibility of adopting containers to execute scientific applications, compared with traditional virtual machines, with a lower delay and overhead. Using the same COP, in Perles et al. (2018), the authors show an IoT solution based on LoRa and Sigfox to supervise the artwork, achieving a lifespan of more than 10 years.

In addition, a comparison of orchestrations tools is shown in Luong et al. (2018). The authors analyze the performance of different COPs with the goal to meet the requirements of 5G networks. In these networks, the use of containers seems to be the adequate solution for a rapid deployment and scalability. The authors discuss the cloudification of mobile network functions using containerization technology. An implementation is done using Docker and Docker Swarm. They chose this technology because the autoscaling mechanisms included in Kubernetes and Marathon, add an extra delay, that it is incompatible with real time

required by the telco applications. Similarly, as we do in this article but in a different scenario, in Jawarneh et al. (2019) is performed a thorough functional and performance evaluation with different orchestration platforms, such as Docker Swarm, Kubernetes, Apache Mesos and Cattle, focused on cloud environments. Their authors conclude that Kubernetes outperforms its counterparts for very complex application deployments. In this case, the authors carried out the study only over 8 physical servers Hewlett Packard Enterprise (HPE) Proliant Gen 8, with 2x Intel (R) Celeron(R) CPU G16610T @ 2.30 GHz processors with 12 GB of RAM. Notice that this study differs from ours since we want to analyze the performance of low cost devices in an IoT and not in a virtualized data center.

Finally, it must be pointed out a preliminary study in Fayos-Jordan et al. (2019), carried out by the same authors in order to evaluate both Kubernetes and Docker Swarm in a fog computing paradigm. In this case, we use a cluster of RPis interconnected by using their wireless capabilities and taking into account the number of cores in order to assign the load to each RPi slave in the cluster. This study is done assuming a queuing system with as many servers as cores in each RPi, assuming scenarios with congestion and without congestion. In practice, as we will see later in this paper, the number of cores is not a constraint. From these experiments and results shown in Fayos-Jordan et al. (2019), we initially conclude that Swarm is better in terms of execution time and throughput compared with Kubernetes.

In Table 1 we summarize the related work presented in this section, based on: case of use (or application), approach given for the computing scenario and goal.

To conclude this section and based on Table 1, we can state that fog-computing on IoT is not yet a mature research line. Although big efforts have been done, we cannot find many solutions and available alternatives, in particular if we take into account the new IoT requirements. To the best of our knowledge, there is not a thorough assessment of container orchestrators platforms and their performances, and less in the context of fog computing using low cost devices. In particular from our previous work (Fayos-Jordan et al., 2019), our results were limited since they were based on RPi using wireless communications and taking into account the number of cores, but in practice this is not a constraint and in addition, it is more interesting and accurate to compare these COPs while running in multibrand environments and using different communication infrastructures.

3. Analysis of linux containers and orchestration platforms

Linux containers are considered a lightweight virtualization in an operating system context. Containers can include both data and code, isolated from the host system in which they run on. All these containers are accessed via specific Application Programming Interfaces (APIs), usually a Representational State Transfer-API (REST-API). Besides, containers in terms of efficiency can boot up in the order of milliseconds. Nowadays, Docker is currently the most used commodity container framework (Docker Containers, 2018), from an open source project, and now on we will use it in this paper.

However, it must be pointed out that there are other container options, in particular LinuX Containers (LXC), that at the beginning Docker was based with (nowadays, they are independent and Docker no longer depends on it). Compared with LXC, Docker focuses on ephemeral, stateless, minimal containers, trimmed down to a single application environment, while LXC was designed for hosting virtual environments based on a clean distribution image (EDUCBA (Corporate Bridge Consultancy Pvt Ltd) , 2020). Docker containers are then lighter weight for handling applications in order to support fast pacing, achieving higher scalability. Besides, we can add that LXC is not so extended and compatible (Rodriguez and Buyya, 2019). Finally, there is not any advantage in terms of performance when comparing the different type of containers (including LXC) as it is concluded from Beserra et al. (2017) and Kovács (2017), except that Docker is more efficient in terms of energy (Morabito, 2015). This argument is critical in our proposal due to the low performance of SBC and the environmental conditions they work with.

Nevertheless, if the number of containers is high, it is required a kind of orchestration used to manage the execution of the containers in a cluster. There are several COPs, such as Docker Swarm (2017), Kubernetes (2018), Marathon-Mesos (Apache Mesos, 2018), etc. Usually, the application designer describes the containers (with its code and data) and the COP will select in execution time what available devices within the cluster will execute them. It must be stressed that application designer could, through interfaces provided by the COP, assign directly the containers among the devices. But, this approach lacks of flexibility since use patterns, may change in the cluster.

Next we describe main features from the most diffused COPs. Kubernetes (2018) is an open source orchestration system introduced by Google in 2014 for managing containerized applications within a cluster of devices. It is based on a master/slave architecture where the master controls and performs the scheduling or orchestration in the cluster (Kubernetes cluster) made up with several devices or slaves. The slaves execute the containers assigned. The master actively manages workloads for load balancing, and carries out management task such as autoscaling, self-healing (since it can be replicated to guarantee fault-tolerance) and reliable container restarting. *Pod* is a basic, virtual and independent processing unit and represents a group of containers deployed and scheduled together, managed by Kubernetes. They are distributed (orchestrated) among the devices (slaves in the cluster) and one single device can run several *pods*. Containers in a *pod* have a unique IP address, but different labels can be used to identify each group of containers. These labels allow Kubernetes to work on an heterogeneous cluster, with devices specialized on specific *pods*. By default Kubernetes initially checks the devices' performance to estimate weights for load balancing and later, schedules based on these weights. It is worth mentioning that one of the requirements for its installation is a minimum of 2 GB of RAM in the devices. With regard to the cluster size, Kubernetes can handle 5000 slaves (nodes) in a cluster, with 150 000 pods, with 300 000 containers and a maximum of 100 pods per node.

Docker Swarm (2017), or simply Swarm, is the native clustering and scheduling tool for Docker containers. In the same way as in Kubernetes, Swarm manages a cluster of devices, where one node acts as a master (manager or orchestrator) responsible for scheduling containers and several slaves (agents) that run the assigned containers, allowing load balancing and fail-over mechanisms. The master monitors all slaves, called discovery process, in order to determine if a Docker daemon on a remote host is on. Swarm agents (slaves) run several Docker stacks and each Docker stack accepts containers. For analogy, a Docker stack is similar to *pod*, as explained in Kubernetes. We will refer to the concept of Docker stacks from Swarm as *pods* as in Kubernetes, since both are equivalent. Also Swarm supports several mechanisms ("spread", "binpack" and "random") to deploy containers in the cluster and for load balancing. With regard to the cluster size, Swarm can use until 7 different masters as backup and 3000 slaves approximately.

Finally, Apache Mesos (2018) and Hindman et al. (2011) is an open source project maintained by Mesosphere and developed by the University of California (Berkeley) to define framework to build scalable and efficient systems based on containers. It is based on a master/slave architecture. The architecture of Mesos consists of four elements: Mesos Master, Mesos slaves, an application-level management Marathon (or Chronos) and Zookeeper. Marathon is the management container orchestrator and interacts with the Mesos master (via its address provided by Zookeeper) and schedules the containers. The master sends the containers from Marathon to the selected slaves based on their available resources. In the case of failures, Marathon starts a new instance to guarantee fault-tolerance. In addition, the architecture of Mesos is designed with high-availability by replicating master node, providing

Table 1
Summary of the related work based on: case of use (or application), approach and goal.

Reference	Case of use	Approach	Goal
Gomes and Pardal (2018)	Smart warehouse	Cloud, fog	Reduce latency
Alam et al. (2019)	Mobile computing	Cloud, fog	Reduce latency, offloading
Medel et al. (2018)	Container management	Cloud	Kubernetes evaluation
de Alfonso et al. (2017)	Computing	Cloud	COPs evaluation
Luong et al. (2018)	5G, containers	Cloud, edge	Reduce latency, COPs evaluation
Jawarneh et al. (2019)	Data centers	Cloud	COPs evaluation
Fayos-Jordan et al. (2019)	IoT application	Fog	COPs evaluation on RPi
Verba et al. (2017)	IoT, gateways	Gateways management	Load balancing
Perles et al. (2018)	IoT application	Communication technologies	Network lifespan

fail-over mechanisms if it fails. This task is done by Zookeeper. With regard to the cluster size, Apache-Mesos allows up to 10 000 nodes in the cluster.

For these COPs, it is interesting to analyze the underlying mechanisms provided in terms of scalability as well as their support and research community behind of them. Regarding the scalability, it must be noticed that the implemented autoscaling feature is not natively in Swarm. In the same way, Marathon does not share this feature neither. Nevertheless, Mesosphere provides a simple autoscaling feature (using a Python application) to automatically scale the Marathon tasks. This application periodically monitors CPU and memory use and when the monitored resources achieve predefined thresholds, the autoscaling application reacts. Lastly, Kubernetes provides a built-in autoscaling mechanism called Horizontal *pod* Autoscaling or autoscaler. It is integrated in the manager, that periodically retrieves the use of CPU of all *pods* and calculates the mean values. These values are compared with predefined thresholds, allowing to orchestrate the number of *pods*. The autoscaler is designed for working in a conservative way and uses a reactive mechanism. That is, if the use of CPU increases, Kubernetes quickly increases the number of *pods*, but on the other hand, it will delay for scaling down. Finally, if we compare the size of the research community behind them based on the number of scientific publications, querying the main scientific databases at the time of writing, we get that Swarm, Kubernetes and Marathon-Mesos have 23%, 64% and 13% respectively.

From this analysis, we will select Swarm and Kubernetes as the selected COP candidates. While Swarm is the native orchestrator of Docker containers and simple, Kubernetes from Google is considered as the most feature-rich orchestrator (and complex).

4. Specification of the SBC devices used and their clustering options

We will describe the specifications of the commercial SBC devices used and will show the different options (wired and wireless) to implement a cluster with them.

4.1. SBC Devices

We will use to carry out the performance analysis within a fog paradigm the following SBC devices, since they are the most commonly used: Raspberry Pi (RPi) (2018), Tinker (2019), Udoo (2019) and Jetson (2018) boards, as well as a computer (PC) for baseline. In Table 2 we detail the main features of the SBC devices. Notice that most of these SBC devices have a GPU, but in our study we did not use it, skipping this detail from their specifications.

4.2. Clustering based on wireless infrastructure (mesh network)

One of the simplest ways to build a cluster with these devices is by using their WiFi cards. In this case, we use four RPi (version 3B) with their power supply and micro SD cards, setting up an homogeneous cluster. The names of the devices are RPiC1 (master) and RPiC2, RPiC3 and RPiC4 (slaves), as shown in Fig. 1. We use the default operating

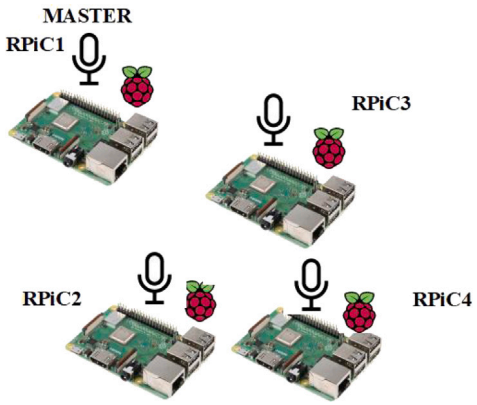


Fig. 1. Homogeneous cluster based on wireless infrastructure with 4 RPis: one master (RPiC1) and three slaves (RPiC2,3,4).

system provided by the manufacturer, Foundation (2019) version 9 (named "Stretch") based on a Linux Debian distribution.

Thus to implement a cluster based on wireless infrastructure, we will rely on a wireless mesh network, running a multihop routing protocol. It must be pointed out that this routing protocol initially is not necessary in a direct communication between the master and the slaves within the cluster, if they are within the communication range (in our case, less than 10 m away). But in practice and in a real fog computing scenario, the slaves could be spread around an area larger than the communication range and in this case, they should use multihop routing to reach the master. There are many multihop routing protocols available, but based on our experience, we will use Better Approach To Mobile Ad Hoc Networking (BATMAN) (Neumann et al., 2008; Open Mesh, 2019). This routing protocol is reliable, proactive and with little overhead as confirmed in Liu et al. (2018) and it works in Layer 2.

4.3. Clustering based on wired infrastructure

In this approach, we include all the different SBC devices described before. The master of the cluster is a RPi and there are five slaves: RPi, Tinker, Udoo, Jetson boards as well as a computer (PC). For the interconnection, we have used the Ethernet interface cards of the devices. Fig. 2 shows a scheme of this cluster and in Fig. 3 is shown a picture of the SBCs connected in our lab, using a Cisco Catalyst 2950 switch with 100 Mbps interfaces.

5. Soundscape monitoring: an IoT application with fog computing

In this section, we describe the IoT application used for soundscape monitoring in order to carry out the performance evaluation of the different computing alternatives. It must be pointed out that the use of this application does not remove lack of generality in our results.

Table 2
Main features of commercial SBC devices used in the cluster: RPi, Tinker, Udoo, Jetson boards and computer (PC).

Feature	RPi	Tinker	Udoo	Jetson	PC
Model	3B Rev 1.2	Rockchip RK3288	UDOO x86 (Sku)	Jetson-nano	Gigabyte G31M-ES2
Architecture	Armv7l	armv7l	x86–64	arch64	x86-64
#CPUs	4	4	4	4	2
freq.min/max [MHz]	600/1200	126/1800	480/2560	102/1428	1600/2800
Memory size [GB]	1	2	8	4	4
OS	Raspbian GNU Linux 10 (buster)	Ubuntu 18.04.2 LTS	Ubuntu 16.04.6 LTS	Ubuntu 18.04.2 LTS	Ubuntu 19.04

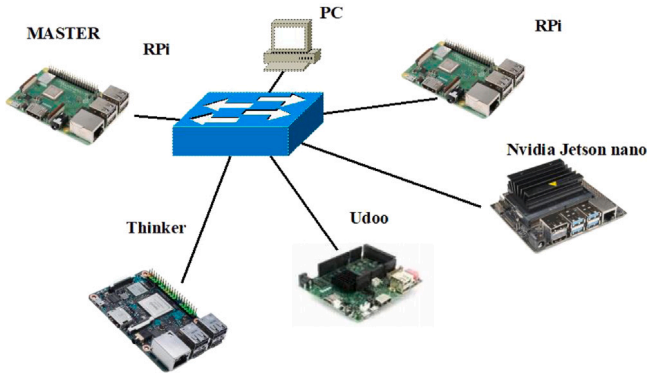


Fig. 2. Heterogeneous cluster based on wired infrastructure: one RPi as master and a multibrand set of slaves: RPi, Tinker, Udoo, Jetson boards and computer (PC).

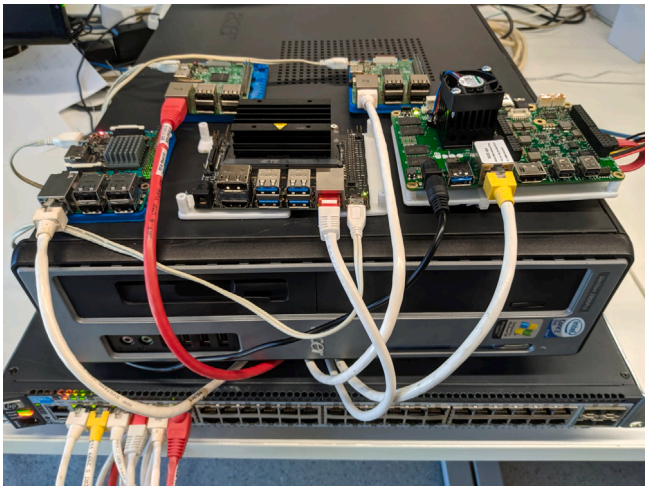


Fig. 3. Picture of the heterogeneous cluster based on wired infrastructure, one RPi as master with five slaves: RPi, Tinker, Udoo, Jetson boards and the computer (with black case) connected by a switch (at the bottom).

5.1. Soundscape: a containerized application

Soundscape monitoring is an extension of acoustic contamination or simply noise pollution monitoring. Different rules and standards have been issued to cope with it in the last decades, such as ISO 1996-1, Environmental Noise Directive (END) 2002/49/EC and ISO 12913:1. Initially these standards were based on the monitoring of traditional Sound Pressure Levels. However, lastly these standards (such as ISO 12913:2 (Soundscape)) are focusing more on subjective nuisance or Psycho-Acoustic Annoyance (PA), that is based on four psycho-acoustic parameters named: Loudness (L), Sharpness (S), Roughness (R) and Fluctuation strength (F). These parameters are usually calculated over

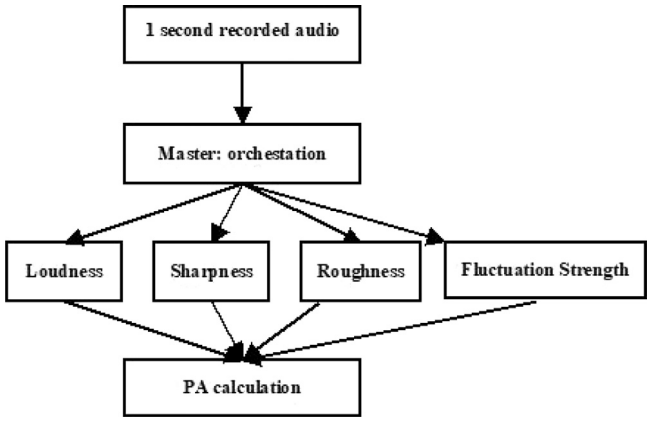


Fig. 4. Flow diagram to measure and calculate Psycho-acoustic Annoyance (PA) based on a containerized approach based on L, S, R and F.

Table 3
Time comparison in seconds to calculate L, S, R and F and PA, with the different devices: RPi, Tinker, Udoo and Jetson boards and a computer (PC).

	L	S	R	F	PA
RPi3B	0.018	0.000	0.849	0.742	1.479
Tinker	0.015	0.000	0.793	0.711	1.243
Udoo	0.014	0.000	0.759	0.698	1.133
Jetson	0.005	0.000	0.193	0.334	0.368
Computer	0.003	0.000	0.128	0.235	0.238

recorded audio samples (chunks) of one second. The properties and definitions of these parameters are found in the Zwicker model (Fastl and Zwicker, 2007), and their implementation details in Pastor-Aparicio et al. (2019) and Noriega-Linares et al. (2017). From these rules, the monitoring process must be done in a dense mode, geographically speaking.

The containerized application is depicted in Fig. 4, where one second of audio is recorded and sent to the cluster to be processed. Then, L, S, R and F are computed in order to calculate PA. Notice that it is necessary to perform the PA estimation as close as possible to the location, without sending out the raw audio for privacy reasons.

In Table 3, we characterize the processing time of these acoustic parameters, to calculate L, S, R and F and PA, with the different devices (RPi, Tinker, Udoo and Jetson boards and a computer). For this time analysis, we have created as many containers as parameters, as well as a container embedding all of them, as shown in the last column of the table. Notice that this table shows mean values in seconds. The standard deviation of these values is at least three orders of magnitude smaller than the mean value, that is, the computation time is approximately constant for all of them independently of the audio chunk.

Since one of the requirements for real time monitoring is given by processing each audio chunk with a lower time than the chunk itself (one second), as we can see in Table 3, the computational complexity

of these algorithms exceeds the computing performance of the SBC devices alone (except with the computer and Jetson), and this becomes worse in practice when sampling in dense mode. If we compare the total time required to calculate PA, the computer and Jetson are the fastest, while RPi3B is the slowest one.

Now on, we will use only the container embedding all the parameters (allowing directly the calculation of PA) and will orchestrate them within a cluster through the different *pods* in order to avoid excessive overhead in the cluster. Then, we can perform load balancing at a low overhead. The containerized application will be orchestrated in the cluster by the selected COPs, Kubernetes and Docker Swarm, to carry out the performance comparison of both.

5.2. Containers and COPs configuration details

The configuration details for the use of containers and their orchestration for the Soundscape application are given next.

As said before, we use Docker containers to perform the load distribution of the different tasks within the cluster. Docker works downloading images from a repository (with authentication), customizing and executing them in the system. This local repository is created at the master node.

Regarding Kubernetes and its configuration, first we will start up the master that will act as orchestrator and we will specify the address where the Kubernetes service will be available for the slaves, as well as the IP range where the *pods* will be executed. In addition, we have to share a token between the master and the slaves in order to authenticate the process. Also, we will require *weave* and *flannel* services, that will be responsible for managing network processes (DNS and DHCP). Notice that we will not enable the autoscaling, since the number of *pods* will be assigned before hand. It does not mean that Kubernetes will not adapt to the node status but it will not autoscale using its default mechanism. In this case, we will schedule the load distribution from the master by using port forwarding to the slaves. Besides we have not included the default monitoring process embedded in Kubernetes because, this process increases in excess the CPU use, around a 60% extra on these low cost SBC devices.

Regarding Swarm configuration, things are easier compared to Kubernetes since it is the native orchestrator of Docker containers. Following the same steps that in Kubernetes, we will publish the address at the master where the service is available for the slaves.

Notice that with both COPs, the election of the master is done initially by configuration, and we could include several masters for fault tolerance and load balancing. In our experiments, we did not use these features in the master.

It must be pointed out that both COPs manage natively an internal overlay network to connect containers hosted on different devices under the same network using VXLAN (Virtual eXtensible LAN) tunnels. This type of tunneling is explained later in Section 6 and Fig. 6. In addition in both COPs natively, the master provides a token in order to allow the slaves accessing the master in a secure way.

6. Testbed

In order to extract the merits from the different COPs analyzed in the cluster, we will use timestamps within the containers. We will analyze the COPs independently. The testbed is made up of an external client device (out of the cluster) that will gather audio samples (chunks) of one second and will offload them (sending them within IP packets) using a REST-API to the cluster, at a certain rate to compute PA.

A scheme for this testbed and the detail of timestamps are depicted in Fig. 5, with the external client, the master and slave x. These time stamps are related to the packets exchanged, as shown in Fig. 6 and explained later.

In Fig. 5 at t_1, the client will timestamp the outgoing packet (request) with the audio chunk, that it is managed and forwarded by the master of the cluster to the slaves (slave x) without delay. This packet is received by the slave and timestamped at the container at t_2. Once the request is processed and the PA calculated (as well L, S, F, R), the response is sent back at t_3 to the client through the master, and this response is received at the client at t_4.

In order to simplify the whole process and the management of the audio chunks, the client has access to a data base with 60150 audio chunks of one second. To record the audio chunks, we used a single-channel audio input with a sampling frequency of 16 kHz with 16 bit per sample, to fulfill the standards. These audio chunks are encoded with base 64 and have a size of 171 Kbytes. Once the client receives the processed packet back, we can estimate the total time to answer the request as $t_4 - t_1$ (including computing, queuing and communications) and the processing time for each audio chunk at each container as $t_3 - t_2$.

The master offers the services (access to the slaves) at a specific address and port. The slaves are running a REST-API as described in Section 3 and the master, according to each COP, will forward the packets accordingly. Fig. 6 shows an example of the packets exchanged using the REST-API protocol between the client (with IP address 192.168.0.112), the master (with external IP address 192.168.0.211 and also with an internal IP address in the cluster 10.255.0.2) and the slave in the cluster (with IP address 10.255.0.7). This protocol is the same for both COPs.

In particular for the application described in Section 5, we can see that first, there is a handshake and then it is sent the *post* (request) to the master, that it is forwarded by the master to the slave. Later, when the slave calculates the PA, it sends back the response to the external client, and finally it is finished. As mentioned before, it must be pointed out that the communication within the cluster (master and slaves) uses a VXLAN tunneling. This tunneling technique is based on MAC in IP encapsulation, adding 50 bytes to each packet. When it is sent the *post* (request) from the external client, this request is splitted in one initial packet of 300 bytes (66 bytes of header and 234 bytes of payload) followed by 122 packets of 1464 bytes (with 66 bytes as header and 1394 bytes of payload) that include the audio chunk. For these 122 packets, the slave usually sent 61 acknowledgments, as shown in Fig. 6. It must be pointed out that when the client does not receive these acknowledgments (because the slaves are busy when processing), the client will start sending retransmissions.

7. Performance evaluation

In this section, we perform the comparison between Docker Swarm (version 19.03.4) and Kubernetes (version 1.14) as COPs. We will use different type of clusters, both homogeneous and heterogeneous, with wired and wireless communication as described in Section 4. The performance evaluation is carried out as follows. The external client as shown in Fig. 5 will send requests to the master, and according to each COPs, it will distribute them among the slaves. During this process, we monitor the use of resources and processing time. We repeated several times the experiments till we achieved a confidence interval of 95%, to show our results in this section.

In the next sections we show the results, both in a homogeneous cluster based on wireless infrastructure as well as a heterogeneous cluster based on wired infrastructure. Notice that we do our tests trying to avoid congestion in the slaves as well as in the network, in order to get clear results at it was mentioned in Section 1. The former is given by the specific assigned load to each slave, as it is explained at each subsection. The latter is given by the number of slaves within the cluster in the slowest scenario (wireless infrastructure at 11 Mbps (IEEE 802.11b) within the communication range, even 200 m away), knowing the processing time for each audio chunk (shown in Table 3) and the bits exchanged for a complete transaction (1.552 Mbits with 196 packets as shown in Fig. 6). In particular, for the cluster based on wireless infrastructure with 3 RPi, they only use 3.14 Mbps, much lower than 11 Mbps and it would allow a maximum of 10 RPi connected.

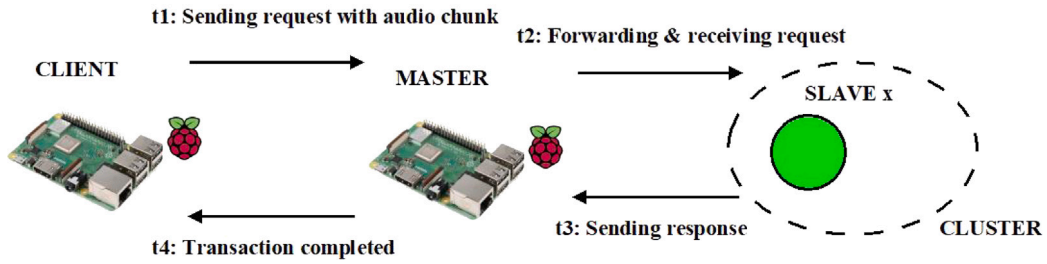


Fig. 5. Time stamps between external client, master and slave x in the cluster.

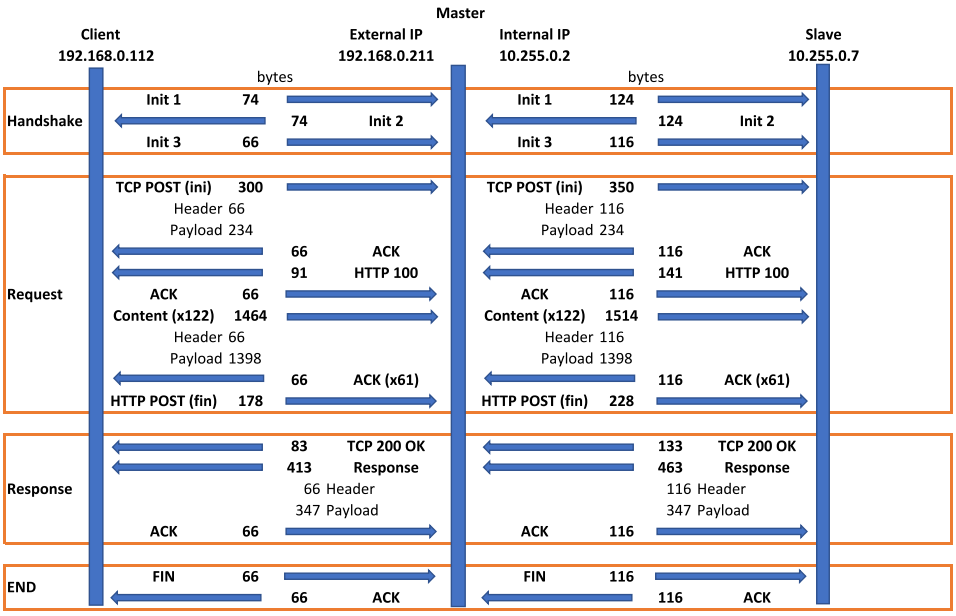


Fig. 6. Example of the packets exchanged using a REST-API protocol between the external client and the cluster, with the master and a slave.

It must be stressed that we do not include a heterogeneous cluster based on wireless infrastructure since the implementation of the different wireless cards and the characteristics of their antennas are different, among the SBC devices. This variability modifies and alters the results, being unfair. Nevertheless, the results from this mentioned scenario (heterogeneous cluster based on wireless infrastructure) are similar to the ones shown based on a wired infrastructure. In particular, the main reason is because in the IoT application (described in Section 5), the bottleneck as explained before, is not given by the communication infrastructure itself but the CPU resources of the low cost SBC devices as shown next in Section 7.2. Then, by using the network interface card based on Ethernet in these devices, we see that this variability (due to the implementation of the card) is null, allowing a fair comparison between the COPs.

7.1. Evaluation of a homogeneous cluster based on wireless infrastructure

We analyze the homogeneous cluster based on wireless infrastructure, shown in Figs. 1, with 4 RPi, 1 master and 3 slaves (described in Section 4.2) with both COPs. In this scenario, we will vary the number of slaves in the cluster from 1 to 3 and the number of *pods*/slave, from 1 to 4.

As said before, the slaves are running a REST-API and will receive the forwarded requests from the master that is running a COP. In this case, from the external client, we will send directly 100 audio chunks. The slaves will keep the audio chunks assigned temporarily if they are busy (while working with other audio chunks), according to the REST-API implementation, and they will process these audio chunks one by one in their different *pods*.

7.1.1. Use of resources

In this section, it must be pointed out that to extract the use of resources (% RAM memory, % CPU, and CPU-Temperature [°C]), we have used a python script using the `psutil` library (Giampaolo Rodola, 2020), that is a cross-platform library for retrieving information in Linux based operating systems.

In Figs. 7(a)–7(c), we compare the average use of resources in terms of % RAM, % CPU and CPU-Temperature (T) [°C] respectively, for all the combinations while processing the audio chunks.

In Fig. 7(a), we can see that Kubernetes always uses 14% approximately more of RAM compared with Swarm. This is partially due to the lower throughput in general as shown later in Section 7.1.2, needing to store audio chunks in memory. In general for both COPs, the memory use is proportionally reduced to the number of slaves (3% approx.) since the cluster always receives the same load, and increased (5% approx. per *pod*) with *pods*/slave.

Regarding % CPU, as shown in Fig. 7(b), both COPs have a similar behavior, increasing it with the number of *pods*. They show a minimum of 35% approximately, adding 20% approximately when each *pod* is included. The use of CPU is reduced as the number of slaves increases. Nevertheless, Swarm shows a slightly greater use of CPU. The maximum is reached by Swarm with a 95% (with 4 *pods*/slave) and Kubernetes has 10% less. Also, CPU Temperature (Fig. 7(c)), according to the CPU use, Swarm gets between 2 and 3 °C higher than Kubernetes for the different combinations. This issue agrees with a higher throughput for Swarm, being more efficient, as shown next.

7.1.2. Processing time

Once the external client sends the audio chunks and they are forwarded by the master, the slaves will start processing by the first

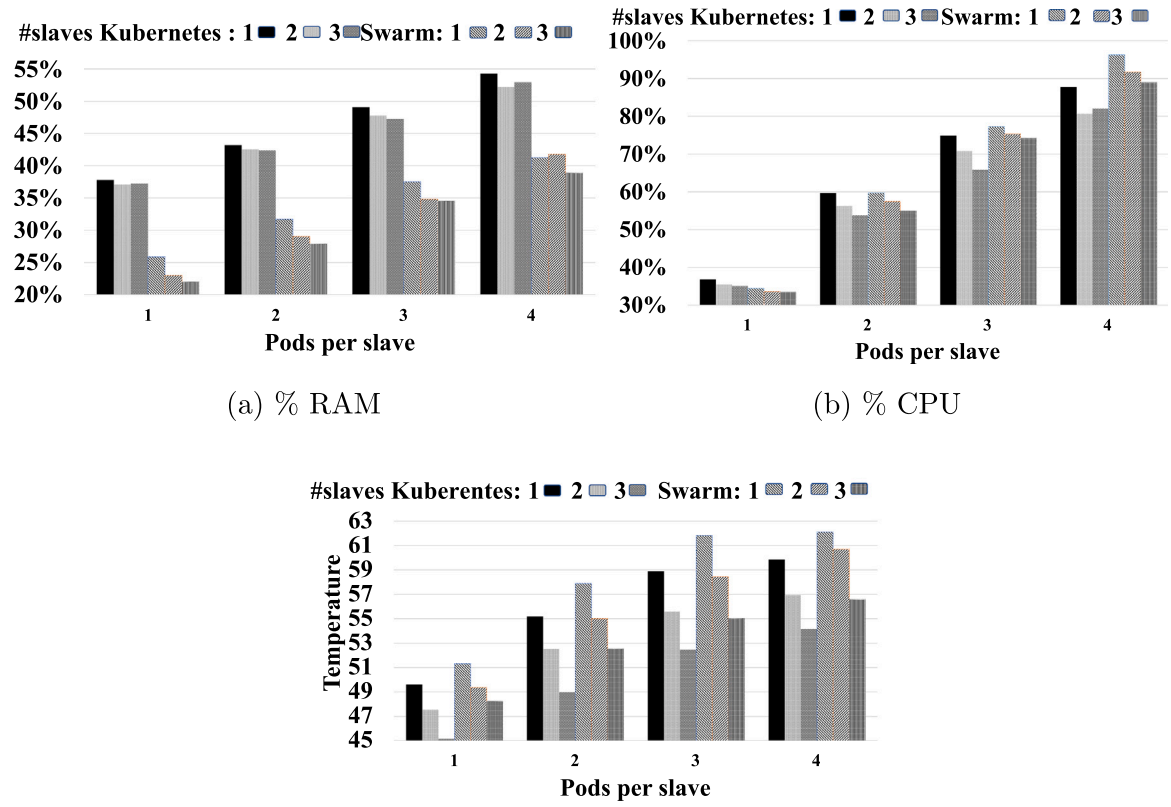


Fig. 7. Use of resources comparing Kubernetes vs Swarm in a homogeneous cluster based on wireless infrastructure of RPi with different slaves and different *pods*/slave: 7(a) % RAM memory, 7(b) % CPU, and 7(c) CPU-Temperature [°C].

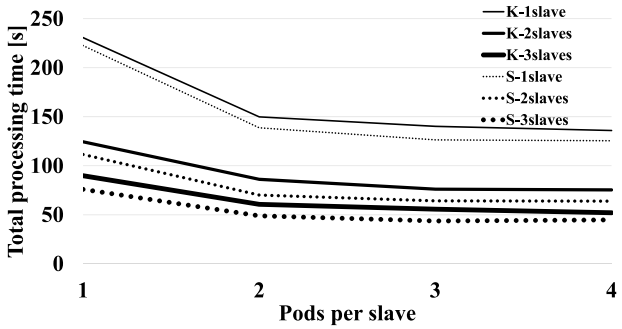


Fig. 8. Total processing time in seconds for 100 audio chunks with different slaves and *pods*/slave, comparing Kubernetes vs Swarm.

one arrived and meanwhile they will keep the rest. If an audio chunk has to wait for processing, the queuing time will depend on the number of pending audio chunks at each slave.

Fig. 8 shows the total processing time ($t_4 - t_1$) to process the 100 audio chunks (the total load), comparing Kubernetes vs Swarm with different slaves and *pods* per slave. We can see clearly that for all the combinations (1 slaves with 1 *pod* till 3 slaves with 4 *pods* per slave) Swarm gets shorter time (dotted line). This improvement is nearly constant while increasing the *pods* per slave and it increases as the number of slave increase, because the improvement is aggregated per slave. From this figure, we can estimate approximately the computation time per audio chunk, dividing by 100.

To compare the internal behavior between Kubernetes vs Swarm to process the containers, Figs. 9(a) and 9(b) show the processing time per audio chunk with both COPs, with 1 slave, but with 2 and 3 *pods* respectively. In the *x-axis* we indicate each audio chunk. In these figures, we can see the queuing phenomenon since each chunk at each *pod* has to wait for the previous ones, in a FIFO discipline. Since each audio chunk takes an average of 1.479 s (as seen in Table 3 for the RPi), this time grows linearly. In Swarm (gray dots), this time increases linearly with the same slope, showing that each *pod* is working in a uniform way. However with Kubernetes (black dots), Figs. 9(a) and 9(b) show 2 and 3 slopes respectively. That is, *pods* inside the same slave, are working at different pace. It must be noticed that the total time to process a request ($t_4 - t_1$) includes the calculation time, the communication time (that is the same for all the chunks) and an internal management time at each slave. Thus, the management time introduced with Kubernetes is not uniform between the *pods*, showing different slopes.

In Figs. 10(a)–10(c) we show the delay factor comparing Kubernetes vs Swarm with different *pods*/slave and with 1 till 3 slaves respectively but in a dynamic way, modifying the audio chunks sent per second (λ) from the external client, between 1 to 10. The delay factor is defined as the average slope for the *pods* shown in Figs. 9(a) and 9(b). This could be seen as the incremental delay introduced at each audio chunk compared with the first one. Swarm always gets a shorter delay factor but this advantage is smoothed when the number of total slaves and *pods* increase. Also as it could be expected, this delay factor is reduced proportionally by the number of slaves and *pods*, because they are working in parallel.

7.2. Evaluation of a heterogeneous cluster based on wired infrastructure

Next, we show the results for the performance evaluation in the heterogeneous cluster (with RPi master, RPi slave, Tinker, Udoo, Jetson

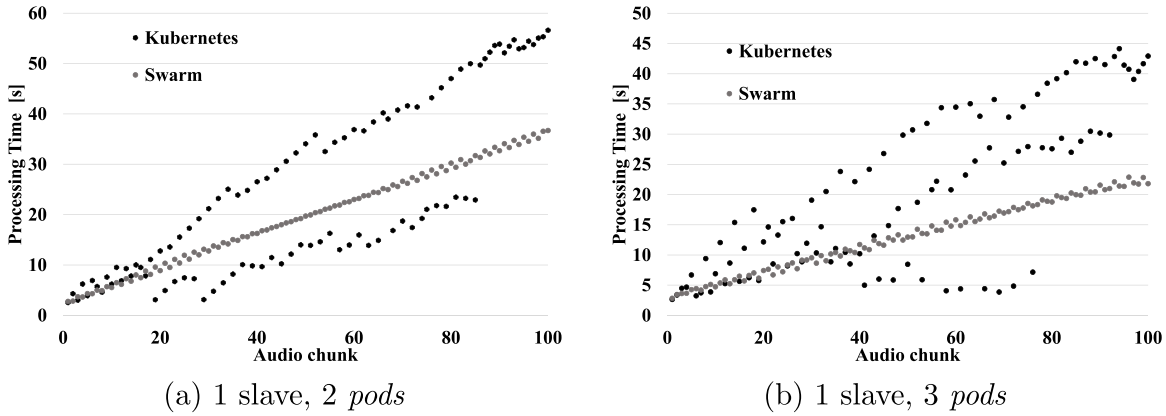


Fig. 9. Processing time in seconds per audio chunk, comparing Kubernetes vs Swarm in 1 slave with: 9(a) 2 *pods* and 9(b) 3 *pods*.

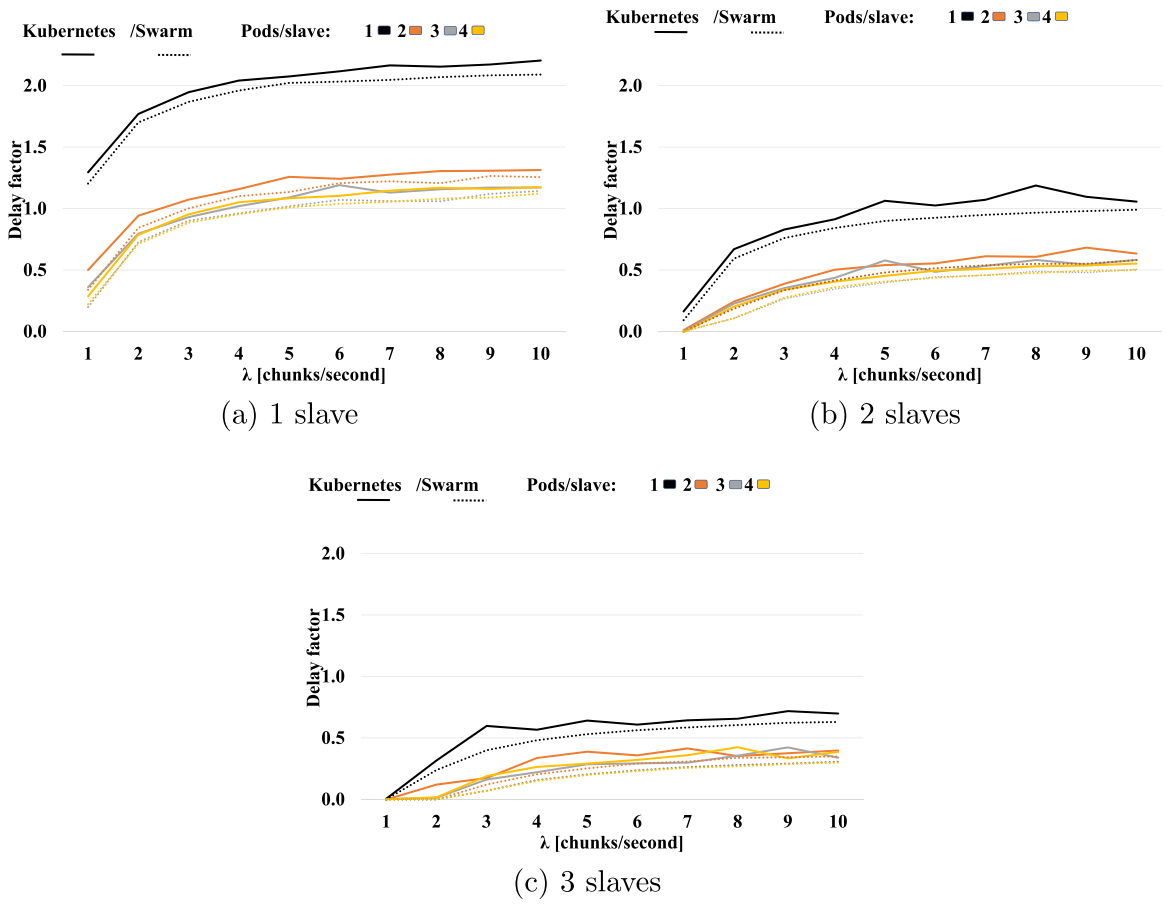


Fig. 10. Delay factor comparing Kubernetes vs Swarm with different slaves (10(a) 1 slave, 10(b) 2 slaves and 10(c) 3 slaves) and modifying *pods*/slave.

boards and the computer), shown in Figs. 2 and 3 and described in Section 4.3.

At this point, we realized that when working with multibrand SBC devices, by default Kubernetes initially checks the different slaves and based on their performance will weigh the amount of load assigned. By doing this, RPi got always less audio chunks than the others. This have pros and cons. Swarm also has other mechanisms to deploy containers. Then, since we want to carry out a performance evaluation of the different SBC devices with the same load, for a fair comparison, we canceled this behavior of Kubernetes.

Once we fixed this, in this case, the load generated by the external client was 200 audio chunks sent to each slave in the cluster consecutively. Besides, for the results we will vary the *pods* per slave from 1 to 10. With these tests we cover all the scenarios with different congestion degree. This is clearly seen later with the use of CPU, reaching in general more than 97%.

7.2.1. Use of resources

To show the use of resources in the heterogeneous cluster, we have defined two different scenarios: with full load (with 10 *pods* assigned to each slave while processing 200 audio chunks) and without load, while running only the COPs. We use the same tools described in Section 7.1.1. In addition, to measure the power consumption we use an oscilloscope with milliamp clamp meters.

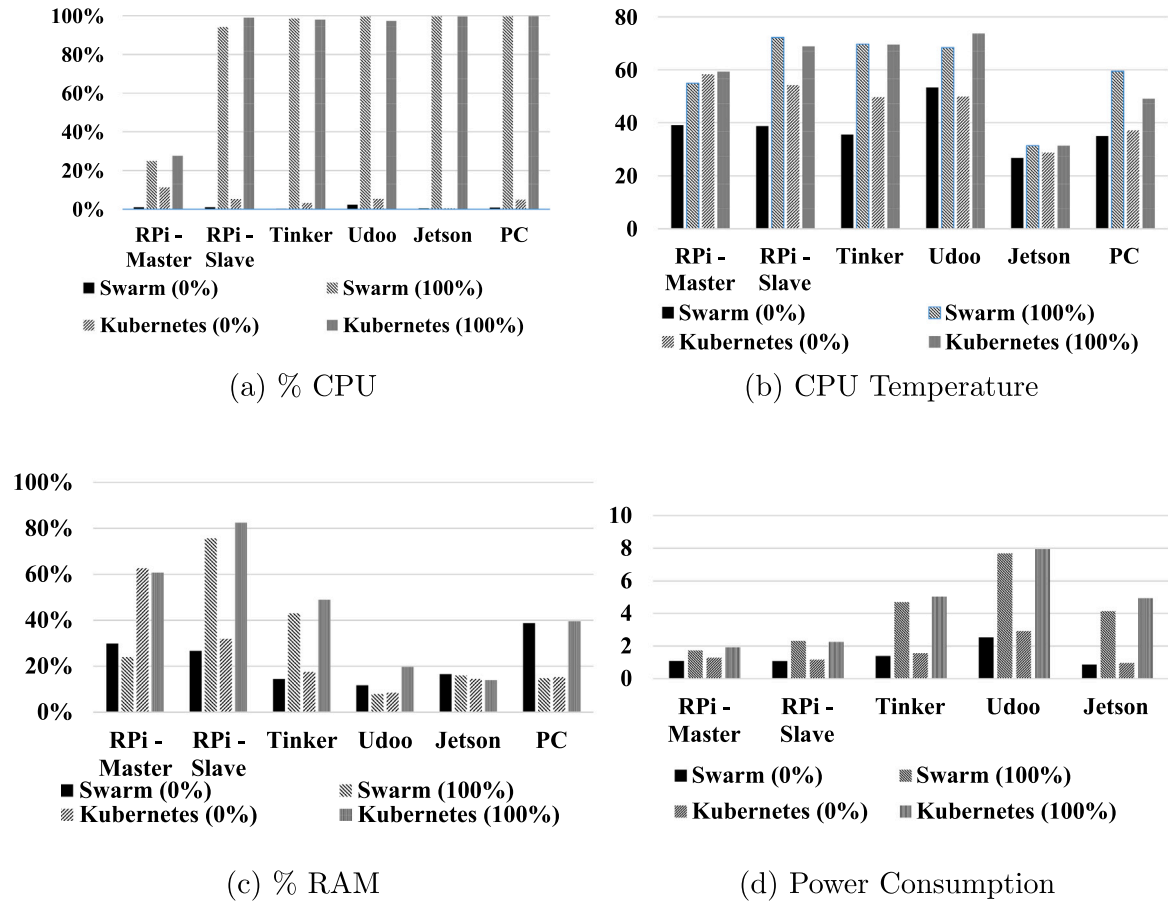


Fig. 11. Use of resources comparing Kubernetes vs Swarm in a heterogeneous cluster based on wired infrastructure without load (0%) and full load (100%): 11(a) % CPU, 11(b) CPU Temperature [°C], 11(c) % RAM and 11(d) Power Consumption [W].

Figs. 11(a)–11(d) show the use of resources % CPU (and its T [°C]), RAM used and Power Consumption for each COP, denoting full load (100%) and without load (0%).

In Fig. 11(a) is shown the use of CPU. Without load (0%), Kubernetes shows a higher use compared with Swarm. In particular, the highest use (4% approximately) is shown by RPi and Udoo, while Swarms has a use around 1% for all of them. Also, notice that the master (RPi) shows a higher difference in terms of CPU, with 1% and 10% for Swarm and Kubernetes respectively. With full load (100%), all of them are above 97%, and in general Kubernetes has higher CPU use compared with Swarm, except with Udoo. It must be stressed, that in this scenario, the RPi slave shows 94% and 98% of CPU use for Swarm and Kubernetes respectively, but with a big difference, that is with Kubernetes, RPi starts losing packets, since it cannot respond the external client by the REST-API. In this case, the external client will start sending retransmissions, as mentioned before in Fig. 6, till the RPi is able to answer or till the RPi closes the TCP session with a reset, both scenarios are feasible. Actually, we see that this behavior with RPi and running Kubernetes starts with 8 *pods* or more running in the RPi. Notice that in our preliminary study in Fayos-Jordan et al. (2019) as well as we saw in Section 7.1 with only RPi, we set the number of *pods* as the number of cores, and then in this scenario we did not lose any packet since we did not reach this limit.

In Fig. 11(b) is shown T of CPU. There is a clear difference between with and without load. In the RPi slave, Docker Swarm without load, T is lower than Kubernetes. However, with full load it is higher with Docker Swarm, and with the Udoo is just the opposite. Also, we see that Jetson board has always the lowest T. Also, it is seen that Docker Swarm heats more the CPU with full load, compared with Kubernetes, except with Udoo. This is clearly seen both in the PC and in RPi.

In Fig. 11(c), we can see that RPi uses more memory compared with the other SBC devices in %. This is because in comparison, it has less memory in total. We can appreciate, that in RPi and Tinker board, Kubernetes has higher use since they are slower. Also, in this scenario, we can see that the master (a RPi too), Kubernetes has higher memory use compared with Docker Swarm.

Finally, in Fig. 11(d) is shown the consumption where clearly RPis in all the scenarios are consuming less. With Docker Swarm, RPi consumes 1 W without load and 2.35 W with full load, while with Kubernetes, it is consuming 1.2 W without load and 2.35 W with full load. In the other platforms, Jetson is consuming less than RPi without load but with full load is consuming more than 4 and 4.85 W for Docker Swarm and Kubernetes respectively. The highest consume is given by Udoo with full load, with 7.5 W and 7.95 W for Docker Swarm and Kubernetes respectively.

7.2.2. Processing time

In Figs. 12(a)–12(b), we can see the processing time for the different SBC devices with Docker Swarm and Kubernetes respectively, with different *pods*/slave (from 1 to 10). It must be highlighted that the processing time ($t_3 - t_2$) increases in general with the number of *pods*, due to the internal *pod* management and overhead, inside the slaves. These *pods* are internally distributed in the slaves among the different cores. Besides, the processing time for the RPi is greater than the other SBC devices and followed by the Tinker board, because they have simpler architectures and lower performance. In addition with RPis, it is seen that when the number of *pods* in Kubernetes increases above 7 *pods*, the processing time starts to decrease. This is because the use of CPU is more than 98%, as mentioned before, and then the RPi cannot complete the protocol of the REST-API, losing approximately 25% of

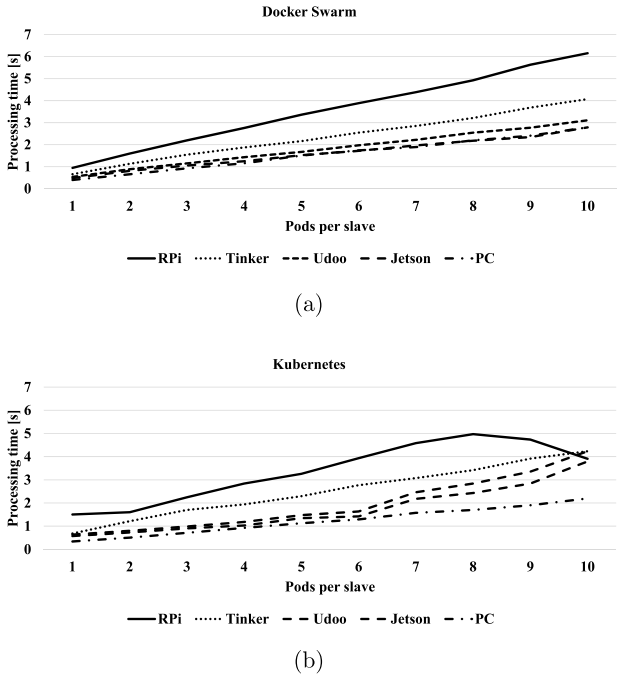


Fig. 12. Average processing time per audio chunk at the slaves for the different SBCs from 1 to 10 *pods* per slave, with 12(a) Docker Swarm and 12(b) Kubernetes.

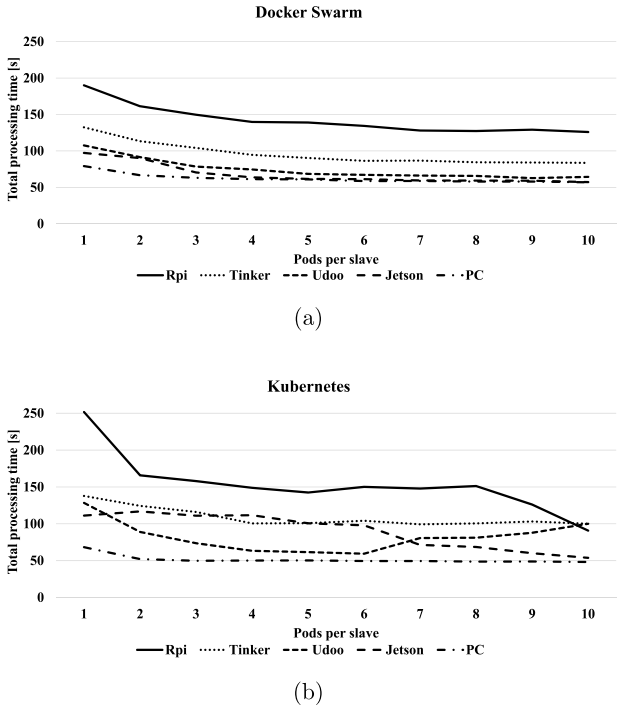


Fig. 13. Total processing time for the different SBCs devices for 200 audio chunks per slave, with 13(a) Docker Swarm and 13(b) Kubernetes, from 1 to 10 *pods* per slave.

audio chunks with 8 *pods* and more than 50% with 9 and 10 *pods*. Then, because the RPi losses these audio chunks, the calculated processing time is lower.

With regard to the total time shown in Figs. 13(a)–13(b), once again the RPi is the slowest and in general, Swarm is more efficient than Kubernetes. We can estimate the total time per audio chunk dividing by 200, the total number of chunks sent. In particular, RPi can process the audio chunks with less than one second (the duration of the audio

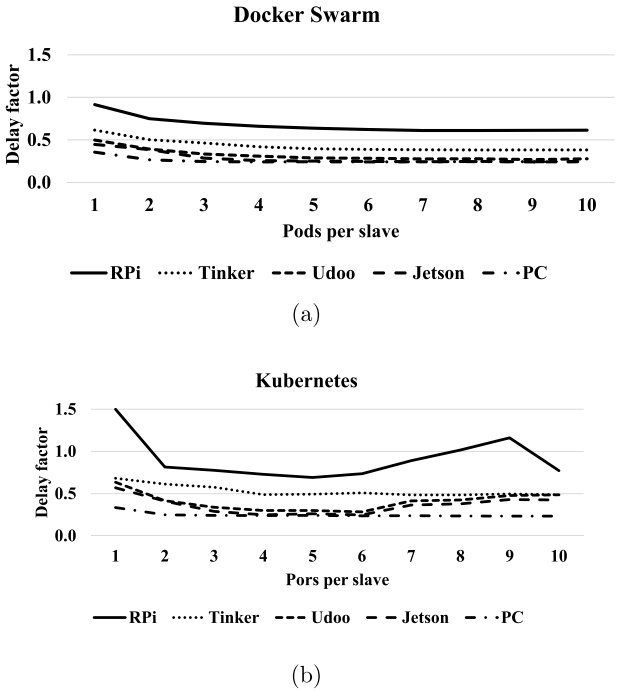


Fig. 14. Delay factor for the different SBCs, with 14(a) Docker Swarm and 14(b) Kubernetes, from 1 to 10 *pods* per slave.

chunk) with more than 2 *pods* and the optimum is achieved with 5 *pods*. However for the Jetson device and computer this optimum is higher, although with more than 10 *pods* we already have a use of CPU close to 100%.

In Figs. 14(a)–14(b), we show the delay factor comparing Docker Swarm and Kubernetes with different *pods*/slave. Swarm always gets a shorter delay factor. Also as it could be expected, this delay factor is reduced proportionally by the number *pods* per slave. When we have more *pods* per slave, the total throughput increases and the delay factor decreases. With regard to RPi, the delay factor increases from 8 *pods* due to the mentioned lost packets.

8. Conclusions

To ease the ubiquitous computing on IoT, we need a flexible and scalable network design. For this purpose, we have introduced fog computing using a cluster of low cost SBC devices. Our solution is based on Linux containers and an orchestration platform, to run on the top of the cluster.

We have presented as proof of concept a containerized application for Soundscape monitoring. This application requires high level of computing that is solved by offloading onto this cluster. Based on this application and using the proposed testbed, we have compared two different Container Orchestration Platform (COP), Docker Swarm and Kubernetes in terms of use of resources and processing time.

The experimental results showed in general an improved performance with the proposed offloading method in terms of execution time, latency and throughput when using these COPs. Also, we realized that both COPs can run several *pods* (or Docker stacks in Docker Swarm), even more than cores the SBC devices have. But increasing the number of *pods* (or stacks), we increase the processing time of the individual containers (due to internal overhead), as well as RAM and CPU use, at the same time we increase the throughput of the devices since they can compute more containers in parallel, with a certain limit. It must be stressed that in the RPi (the simplest device) with 8 *pods* or

more, running Kubernetes, we achieve 98% use of CPU and it cannot carry out correctly the REST-API, losing packets from the application. Meanwhile with Docker Swarm, since it has a smoothed behavior to process the workload, RPi does not lose any. Internally Kubernetes adds extra overhead that is a drawback for this kind of simple devices.

On one hand, we have seen that Docker Swarm always outperforms Kubernetes, both in use of resources and processing time on these SBC devices and with different topologies. On the other hand, we have seen that Kubernetes includes a feature set that could be interesting for higher performance devices (servers) that are not included natively in Docker Swarm, such as auto scaling and the initial test to weight the different slaves in order to perform a default load balancing but at a cost of extra CPU use. This default behavior of Kubernetes is useful when using a heterogeneous cluster of mid/high range performance devices. Notice that we did not use this feature because we wanted to distribute uniformly the load for a fair comparison among the slaves.

Finally, from the different SBC devices analyzed, we have seen that RPi (model 3B) has the lowest power consumption and the lowest throughput, although it has a good quality/price ratio.

CRediT authorship contribution statement

Rafael Fayos-Jordan: Software, Validation, Investigation. **Santiago Felici-Castell:** Funding acquisition, Investigation, Methodology, Project administration, Writing - review & editing, Supervision, Writing - original draft. **Jaume Segura-Garcia:** Funding acquisition, Investigation, Methodology, Project administration, Supervision, Resources. **Jesus Lopez-Ballester:** Data curation, Formal analysis. **Maximo Cobos:** Investigation, Methodology, Supervision.

Acknowledgments

This work is supported by the Ministry of Economy under the project Urbauramon BIA2016-76957-C3-1-R and the grant by University of Valencia UV-INV-EPDI19-995284.

References

Alam, M.G.R., Hassan, M.M., Uddin, M.Z., Almogren, A., Fortino, G., 2019. Autonomic computation offloading in mobile edge for IoT applications. Future Gener. Comput. Syst. 90, 149–157. http://dx.doi.org/10.1016/j.future.2018.07.050, URL http://www.sciencedirect.com/science/article/pii/S0167739X18303996.

Apache Mesos, 2018. URL http://mesos.apache.org/. (Accessed 11 January 2019).

Bellavista, P., Berrocal, J., Corradi, A., Das, S.K., Foschini, L., Zanni, A., 2019. A survey on fog computing for the internet of things. Pervasive Mob. Comput. 52, 71–99. http://dx.doi.org/10.1016/j.pmcj.2018.12.007, URL http://www.sciencedirect.com/science/article/pii/S1574119218301111.

Beserra, D., Pinheiro, M.K., Souveyet, C., Steffenel, L.A., Moreno, E.D., 2017. Performance evaluation of OS-level virtualization solutions for HPC purposes on SoC-based systems. In: 2017 IEEE 31st International Conference on Advanced Information Networking and Applications, AINA, pp. 363–370.

Cobos, M., Perez-Solano, J.J., Felici-Castell, S., Segura, J., Navarro, J.M., 2014. Cumulative-sum-based localization of sound events in low-cost wireless acoustic sensor networks. IEEE/ACM Trans. Audio Speech Lang. Process. 22 (12), 1792–1802. http://dx.doi.org/10.1109/TASLP.2014.2351132.

de Alfonso, C., Calatrava, A., Moltó, G., 2017. Container-based virtual elastic clusters. J. Syst. Softw. 127, 1–11. http://dx.doi.org/10.1016/j.jss.2017.01.007, URL http://www.sciencedirect.com/science/article/pii/S0164121217300146.

Docker Containers. 2018. URL https://docs.docker.com/engine/examples/. (Accessed 2 March 2019).

Docker Swarm. 2017. URL https://docs.docker.com/swarm/. (Accessed 11 November 2018).

EDUCBA (Corporate Bridge Consultancy Pvt Ltd) , 2020. Key differences between LXC and docker. In: Head To Head Comparison Between LXC and Docker. URL https://www.educba.com/lxc-vs-docker/. (Accessed 24 July 2020).

Fastl, H., Zwicker, E., 2007. Psychoacoustics: Facts and models. In: Springer Series in Information Sciences, Springer.

Fayos-Jordan, R., Felici-Castell, S., Segura-Garcia, J., Pastor-Aparicio, A., Lopez-Ballester, J., 2019. Elastic computing in the fog on internet of things to improve the performance of low cost nodes. Electronics 8 (12), http://dx.doi.org/10.3390/electronics8121489, URL https://www.mdpi.com/2079-9292/8/12/1489.

Foundation, R.P., 2019. Download raspbian for raspberry pi. URL https://www.raspberrypi.org/downloads/raspbian/.

Giampaolo Rodola, 2020. Cross-platform lib for process and system monitoring in Python, Psutil 5.7.0. URL https://pypi.org/project/psutil, (Accessed 12 February 2020).

Gomes, M., Pardal, M.L., 2018. Cloud vs Fog: assessment of alternative deployments for a latency-sensitive IoT application. In: The 9th International Conference on Ambient Systems, Networks and Technologies (ANT 2018) / The 8th International Conference on Sustainable Energy Information Technology (SEIT-2018) / Affiliated Workshops. Procedia Comput. Sci. 130, 488–495. http://dx.doi.org/10.1016/j.procs.2018.04.059, URL http://www.sciencedirect.com/science/article/pii/S1877050918304125.

Hindman, B., Konwinski, A., Zaharia, M., Ghodsi, A., Joseph, A.D., Katz, R., Shenker, S., Stoica, I., 2011. Mesos: A platform for fine-grained resource sharing in the data center. In: Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation. NSDI'11, USENIX Association, Berkeley, CA, USA, pp. 295–308, URL http://dl.acm.org/citation.cfm?id=1972457.1972488.

Jawarneh, I.M.A., Bellavista, P., Bosi, F., Foschini, L., Martuscelli, G., Montanari, R., Palopoli, A., 2019. Container orchestration engines: A thorough functional and performance comparison. In: ICC 2019 - 2019 IEEE International Conference on Communications, ICC. pp. 1–6. http://dx.doi.org/10.1109/ICC.2019.8762053.

Jetson, Jetson Nano. 2018. URL https://www.nvidia.com/. (Accessed 11 August 2019).

Kovács, A., 2017. Comparison of different Linux containers. In: 2017 40th International Conference on Telecommunications and Signal Processing, TSP, pp. 47–51.

Kubernetes (K8s). 2018. URL https://kubernetes.io/. (Accessed 11 October 2018).

Liu, L., Liu, J., Qian, H., Zhu, J., 2018. Performance evaluation of BATMAN-adv wireless mesh network routing algorithms. In: 2018 5th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/2018 4th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom), pp. 122–127.

Luong, D., Thieu, H., Outtagarts, A., Ghamri-Doudane, Y., 2018. Cloudification and autoscaling orchestration for container-based mobile networks toward 5g: Experimentation, challenges and perspectives. In: 2018 IEEE 87th Vehicular Technology Conference, VTC Spring, pp. 1–7. http://dx.doi.org/10.1109/VTCSpring.2018.8417602.

Medel, V., Tolosana-Calasanz, R., Ángel Baares, J., Arronategui, U., Rana, O.F., 2018. Characterising resource management performance in kubernetes. Comput. Electr. Eng. 68, 286–297. http://dx.doi.org/10.1016/j.compeleceng.2018.03.041, URL http://www.sciencedirect.com/science/article/pii/S0045790617315240.

Morabito, R., 2015. Power consumption of virtualization technologies: An empirical investigation. In: 2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing, UCC, pp. 522–527.

Neumann, A., Aichele, C., Lindner, M., Wunderlich, S., 2008. Better Approach To Mobile Ad-hoc Networking (B.A.T.M.A.N.), IETF Draft.

Noriega-Linares, J.E., Rodriguez-Mayol, A., Cobos, M., Segura-García, J., Felici-Castell, S., Navarro, J.M., 2017. A wireless acoustic array system for binaural loudness evaluation in cities. IEEE Sens. J. 17 (21), 7043–7052. http://dx.doi.org/10.1109/JSEN.2017.2751665.

Open Mesh, 2019. Doc-overview - batman-adv - open mesh. URL https://www.open-mesh.org/projects/batman-adv/wiki.

Pastor-Aparicio, A., Lopez-Ballester, J., Segura-Garcia, J., Felici-Castell, S., Cobos, M., Fayos-Jordan, R., Perez-Solano, J., 2019. Zwicker's annoyance model implementation in a WASN node. In: Inter Noise 2019, vol. 1. p. 1.

Perles, A., Pérez-Marín, E., Mercado, R., Segrelles, J.D., Blanquer, I., Zarzo, M., Garcia-Diego, F.J., 2018. An energy-efficient internet of things (IoT) architecture for preventive conservation of cultural heritage. Future Gener. Comput. Syst. 81, 566–581. http://dx.doi.org/10.1016/j.future.2017.06.030, URL http://www.sciencedirect.com/science/article/pii/S0167739X17313663.

Raspberry Pi 3B+. 2018. URL https://www.raspberrypi.org/. (Accessed 2 January 2019).

Rodriguez, M., Buyya, R., 2019. Container based cluster orchestration systems: A taxonomy and future directions. J. Softw.: Pract. Exp. 49 (5), 698–719.

SBC: Tinker-Board. 2019. URL https://www.asus.com/. (Accessed 12 October 2019).

UDOO X86 II. 2019. URL https://www.udoo.org/. (Accessed 15 September 2019).

Verba, N., Chao, K.-M., James, A., Goldsmith, D., Fei, X., Stan, S.-D., 2017. Platform as a service gateway for the fog of things. Adv. Eng. Inform. 33, 243–257. http://dx.doi.org/10.1016/j.aei.2016.11.003, URL http://www.sciencedirect.com/science/article/pii/S1474034616301872.

Zao, J., et al., 2018. IEEE Standard for adoption of openfog reference architecture for fog computing. IEEE Std 1934-2018 1–176. http://dx.doi.org/10.1109/IEEESTD.2018.8423800.

Rafael Fayos Jordán received the B.S. Degree in Telematics Engineering from the University of Valencia in 2018. He is currently in the last year of master degree in Web Technology, Cloud Computing and Mobile Applications and working as researcher in Department of Computer Science, University of Valencia. His interest is in the IoT platform and node management, cloud computing and network management.

Santiago Felici-Castell received the M.Sc. and Ph.D. degrees in telecommunication engineering from the Polytechnical University of Valencia, Valencia, Spain, in 1993 and 1998, respectively. He is currently an Associate Professor with the University of Valencia, Valencia. His research has focused on networking, communication systems, and multiresolution techniques for data transmission with quality of service. He is also a Cisco Systems Certificated Instructor, and has authored over 25 technical papers in international journals and conferences.

Jaume Segura-Garcia received the M.Sc. and Ph.D. degrees in physics from the University of Valencia, Valencia, Spain, in 1998 and 2003, respectively. After completing his Ph.D. study, he was with the Robotics Institute, University of Valencia, where he was involved in several projects related to intelligent transportation systems. Since 2008, he has been with the Department of Computer Science, University of Valencia, where he is currently an Associate Professor. He has been a Visiting Researcher with multiple European research centers. He has co-authored over 90 publications at national and international journals, book chapters, and conferences. He was on the Organizing Committee of several national and international conferences, including the International Workshop on Virtual Acoustics (2011). He is a member of the Spanish Acoustics Society and the European Acoustics Association.

Jesus Lopez-Ballester received the B.S. degree in Telecommunications Engineering specialized in Electronic Systems and the M.Sc. of Telecommunications Engineering at University of Valencia in 2009 and 2014, respectively, obtaining the Extraordinary Prize Final Project of the College of Telecommunications Engineers. Before and after finishing his M.Sc. study, work for several years with hospitals in the city of Valencia, where he specialized in the acquisition and processing of biomedical signals and the design of systems of care and remote monitoring of patients. His interest in signal processing hardware, software and human machine interfaces, led him to work in University Research Institute on Robotics and Information and Communications Technologies (IRTIC), developing advanced training systems. He is currently pursuing her Ph.D in Information Technology, Communications and Computation at the University of Valencia and his topics of interest are analysis of acoustic events, deep learning and signal processing.

Maximo Cobos received the telecommunications engineer degree, the M.S. degree in telecommunication technologies, and the Ph.D. degree in telecommunications, all of them from the Universitat Politècnica de València, Valencia, Spain, in 2006, 2007, and 2009, respectively. His Ph.D. dissertation was awarded with the Ericsson Best Thesis Award on Multimedia Environments from the Spanish National Telecommunications Engineering Association (COIT). He completed with honors his Ph.D studies under the University Faculty Training program (FPU). Since 2011, he has been with the Computer Science Department of the Universitat de València, where he is now an Associate Professor. His work is focused on the area of digital signal processing for audio and multimedia applications, where he has published more than 90 technical papers in international journals and conferences. Dr. Cobos is a Senior Member of the Institute of Electrical and Electronics Engineers (IEEE), and a Full Member of the Acoustical Society of America (ASA).

Anexxes II: Research Projects

F Herramientas inteligentes para la gestión y control del paisaje sonoro urbano. Definición de protocolos de monitorización y auralización. Intervención en el Patrimonio Sonoro (Proyecto coordinado) (URBAURAMON)

Programme	PIPN - Research Projects of the National R+D+I Programme
Funder	MINECO. Ministry of Economy and Competitiveness
Project N.	BIA2016-76957-C3-1-R
Amount	77,440.00
Duration	2016 - 2020
P. Researcher.	Jaume Segura-Garcia, Santiago Felici-Castell
Researchers	4
Keywords	Paisatge Sonor, Paisaje Sonoro, Soundscape, Redes de sensores distribuidos, Auralización

G ECO4RUPA: 'Creating inclusive and intelligent Ecosystems for improving citizens health with Real-time Urban Pollution hazard Avoidance'

Programme	PIPN - Research Projects of the National R+D+I Programme
Funder	State Secretariat for Universities and Research
Entities	Ministry of Science and Technology
Project N.	PID2021-126823OB-I00
Amount	36,300.00
Duration	2022 - 2025
P. Researcher	Jaume Segura-Garcia, Santiago Felici-Castell
Researchers	8
Keywords	Internet de las Cosas, WSN, Enrutador inteligente, Medio ambiente urbano, Smart City

H Building new ecosystems with global-context aware mobility for smart cities relying on green 5G/6G Communications and AI-IoT monitoring tecnologies (GREENISH)

Programme	PIGV - Research Projects (Valencian Regional Government)
Funder	Government of Valencia - Regional Ministry of Education, Universities and Employment
Entities	Ministry of Science and Technology
Project N.	CIAICO/2022/179
Amount	90,000.00
Duration	2023 - 2025
P. Researcher	Santiago Felici-Castell, Sandra Roger-Varea
Researchers	7
Keywords	Internet of the Things (IoT), Artificial Intelligence, Vehicular to X Communications, Air Quality

References

[1] I. Statista, *Iot connected devices worldwide 2019-2030 — statista*, 2023. [Online]. Available: https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/.

[2] I. Foster and C. Kesselman, *The Grid: Blueprint for a New Computing Infrastructure (The Elsevier Series in Grid Computing)*, I. Foster and C. Kesselman, Eds. Morgan Kaufmann, Jan. 1998, p. 675, cloud definitions, ISBN: 978-1558604759.

[3] A. Eldow, M. Shakir, M. A. Talab, A. K. Muttar and R. M. Tawafak, 'Literature review of authentication layer for public cloud computing: A meta-analysis,' vol. 97, p. 12, 2006, cloud definitions.

[4] L. Wang, G. von Laszewski, A. Younge *et al.*, 'Cloud computing: A perspective study,' vol. 28, pp. 137–146, 2 Apr. 2010, ISSN: 0288-3635. DOI: 10.1007/s00354-008-0081-5. [Online]. Available: http://link.springer.com/10.1007/s00354-008-0081-5.

[5] A. K. Sandhu, 'Big data with cloud computing: Discussions and challenges,' vol. 5, pp. 32–40, 1 Mar. 2022, cloud definitions, ISSN: 2096-0654. DOI: 10.26599/BDMA.2021.9020016. [Online]. Available: https://ieeexplore.ieee.org/document/9663258/.

[6] F. Bonomi, R. Milito, J. Zhu and S. Addepalli, 'Fog computing and its role in the internet of things,' fog computing definitions, Association for Computing Machinery, 2012, pp. 13–16, ISBN: 9781450315197. DOI: 10.1145/2342509.2342513. [Online]. Available: https://doi.org/10.1145/2342509.2342513.

[7] M. Hajibaba and S. Gorgin, 'A review on modern distributed computing paradigms: Cloud computing, jungle computing and fog computing,' vol. 22, 2 2014, Fog computing definitions, ISSN: 13301136. DOI: 10.2498/cit.1002381.

[8] H. Atlam, R. Walters and G. Wills, 'Fog computing and the internet of things: A review,' vol. 2, p. 10, 2 Apr. 2018, fog computing definitions, ISSN: 2504-2289. DOI: 10.3390/bdcc2020010. [Online]. Available: https://www.mdpi.com/2504-2289/2/2/10.

[9] H. K. Apat, R. Nayak and B. Sahoo, 'A comprehensive review on internet of things application placement in fog computing environment,' vol. 23, p. 100 866, Oct. 2023, fog computing definitions, ISSN: 25426605. DOI: 10.1016/j.iot.2023.100866. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2542660523001890.

[10] H. Pang and K.-L. Tan, 'Authenticating query results in edge computing,' Edge computing definitions, IEEE Comput. Soc, 2004, pp. 560–571, ISBN: 0-7695-2065-0. DOI: 10.1109/ICDE.2004.1320027. [Online]. Available: http://ieeexplore.ieee.org/document/1320027/.

[11] M. Desertot, C. Escoffier and D. Donsez, 'Towards an autonomic approach for edge computing,' vol. 19, pp. 1901–1916, 14 Sep. 2007, edge computing definitions, ISSN: 1532-0626. DOI: 10.1002/cpe.1135. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1002/cpe.1135.

[12] B. Varghese, N. Wang, S. Barbhuiya, P. Kilpatrick and D. S. Nikolopoulos, 'Challenges and opportunities in edge computing,' edge computing definitions, IEEE, Nov. 2016, pp. 20–26, ISBN: 978-1-5090-5263-9. DOI: 10.1109/SmartCloud.2016.18. [Online]. Available: https://ieeexplore.ieee.org/document/7796149/.

[13] H. Hua, Y. Li, T. Wang, N. Dong, W. Li and J. Cao, 'Edge computing with artificial intelligence: A machine learning perspective,' vol. 55, pp. 1–35, 9 Sep. 2023, edge computing definitions, ISSN: 0360-0300. DOI: 10.1145/3555802. [Online]. Available: https://dl.acm.org/doi/10.1145/3555802.

[14] Z. M. Fadlullah, F. Tang, B. Mao *et al.*, 'State-of-the-art deep learning: Evolving machine intelligence toward tomorrow's intelligent network traffic control systems,' vol. 19, pp. 2432–2455, 4 2017, ISSN: 1553-877X. DOI: 10.1109/COMST.2017.2707140. [Online]. Available: http://ieeexplore.ieee.org/document/7932863/.

[15] S. I. E. Ltd, *What does it cost to build a data centre?* [Online]. Available: https://siteltd.co.uk/blog/what-does-it-cost-to-build-a-data-centre/.

[16] D. I. LLC, *How much does it cost to build a data center? - dgtl infra*. [Online]. Available: https://dgtlinfra.com/how-much-does-it-cost-to-build-a-data-center/.

[17] KIO, *Costs of a data center*. [Online]. Available: https://www.kio.tech/en-us/blog/data-center/costs-of-a-data-center.

[18] J. H. Kim, S. Ullah and D. H. Kim, 'Gpu-based embedded edge server configuration and offloading for a neural network service,' vol. 77, pp. 8593–8621, 8 Aug. 2021, ISSN: 15730484. DOI: 10.1007/S11227-021-03623-9/TABLES/8. [Online]. Available: https://link.springer.com/article/10.1007/s11227-021-03623-9.

[19] V. K. Rathi, N. K. Rajput, S. Mishra *et al.*, 'An edge ai-enabled iot healthcare monitoring system for smart cities,' vol. 96, p. 107 524, Dec. 2021, ISSN: 0045-7906. DOI: 10.1016/J.COMPELECENG.2021.107524.

[20] L. Gao, T. H. Luan, B. Liu, W. Zhou and S. Yu, 'Fog computing and its applications in 5g,' pp. 571–593, Jan. 2016. DOI: 10.1007/978-3-319-34208-5_21/FIGURES/9. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-34208-5_21.

[21] S. González-Díaz, A. Arias-Cruz, C. Macouzet-Sánchez and A. Partida-Ortega, 'Impact of air pollution in respiratory allergic diseases,' vol. 18, 73 2016, ISSN: 16655796. DOI: 10.1016/j.rmu.2016.10.006.

[22] G. Molinari, G. Colombo and C. Celenza, 'Respiratory allergies: A general overview of remedies, delivery systems, and the need to progress,' vol. 2014, 2014. DOI: 10.1155/2014/326980.

[23] W. H. Organitation, *Air quality, energy and health*. [Online]. Available: https://www.who.int/teams/environment-climate-change-and-health/air-quality-and-health/health-impacts/types-of-pollutants.

[24] Eurostats, *Deaths from diseases of the respiratory system*, 2020.

[25] K. Levak, M. Horvat and H. Domitrovic, 'Effects of noise on humans,' vol. 1, pp. 333–336, 2008. [Online]. Available: https://api.semanticscholar.org/CorpusID:36958789.

[26] H. Fastl and E. Zwicker, *Psychoacoustics: Facts and models*. 2007. DOI: 10.1007/978-3-540-68888-4.

[27] B. C. Moore, B. R. Glasberg and T. Baer, 'A model for the prediction of thresholds, loudness, and partial loudness,' vol. 45, 4 1997, ISSN: 00047554.

[28] T. R. P. Foundation, *Teach, learn, and make with the raspberry pi foundation*. [Online]. Available: https://www.raspberrypi.org/.

[29] J. Segura-Garcia, J. Lopez-Ballester, S. Felici-Castell *et al.*, 'Soundscape monitoring of modified psychoacoustic annoyance with next-generation edge computing and iot,' 2022. DOI: 10.1145/3544538.3544666.

[30] E. Systems, *Wireless socs, software, cloud and aiot solutions — espressif systems*. [Online]. Available: https://www.espressif.com/.

[31] P. Ltd, *Fipy*. [Online]. Available: https://docs.pycom.io/datasheets/development/fipy/.

[32] R. K. Bhagat, M. S. D. Wykes, S. B. Dalziel and P. F. Linden, 'Effects of ventilation on the indoor spread of covid-19,' vol. 903, 2020, ISSN: 14697645. DOI: 10.1017/jfm.2020.720.

[33] ECDC, 'Heating, ventilation and air-conditioning systems in the context of covid-19,' June 2020.

[34] K. Azuma, U. Yanagi, N. Kagi, H. Kim, M. Ogata and M. Hayashi, *Environmental factors involved in sars-cov-2 transmission: Effect and role of indoor environmental quality in the strategy for covid-19 infection control*, 2020. DOI: 10.1186/s12199-020-00904-2.

[35] Z. Liu, P. Ciais, Z. Deng *et al.*, 'Near-real-time monitoring of global co2 emissions reveals the effects of the covid-19 pandemic,' vol. 11, 1 2020, ISSN: 20411723. DOI: 10.1038/s41467-020-18922-7.

[36] R. Fayos-Jordan, J. Segura-Garcia, A. Soriano-Asensi, S. Felici-Castell, J. M. Felisi and J. M. Alcaraz-Calero, 'Ventqsys: Low-cost open iot system for co2 monitoring in classrooms,' vol. 27, pp. 5313–5327, 8 2021, ISSN: 15728196. DOI: 10.1007/s11276-021-02799-5. [Online]. Available: https://doi.org/10.1007/s11276-021-02799-5.

[37] P. Orellano, N. Quaranta, J. Reynoso, B. Balbi and J. Vasquez, *Effect of outdoor air pollution on asthma exacerbations in children and adults: Systematic review and multilevel meta-analysis*, 2017. DOI: 10.1371/journal.pone.0174050.

[38] S. Vargas, W. Onatra, L. Osorno, E. Páez and O. Sáenz, 'Contaminación atmosférica y efectos respiratorios en niños, en mujeres embarazadas y en

adultos mayores,' vol. 11, 1 Jun. 2008, ISSN: 01234226. DOI: 10.31910/rudca.v11.n1.2008.600.

[39] R. Fayos-Jordan, R. Araiz-Chapa, S. Felici-Castell, J. Segura-Garcia, J. J. Perez-Solano and J. M. Alcaraz-Calero, 'Eco4rupa: 5g-iot inclusive and intelligent routing ecosystem with low-cost air quality monitoring,' vol. 14, 8 2023, ISSN: 20782489. DOI: 10.3390/info14080445.

[40] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hoßfeld and P. Tran-Gia, 'A survey on quality of experience of http adaptive streaming,' vol. 17, 1 2015, ISSN: 1553877X. DOI: 10.1109/COMST.2014.2360940.

[41] A. Neumann, C. Aichele, M. Lindner and S. Wunderlich, 'Better approach to mobile ad-hoc networking (batman),' pp. 1–24, 2008.

[42] S. Felici-Castell, M. Garcia-Pineda, J. Segura-Garcia, R. Fayos-Jordan and J. Lopez-Ballester, 'Adaptive live video streaming on low-cost wireless multihop networks for road traffic surveillance in smart cities,' vol. 115, pp. 741–755, 2021, ISSN: 0167739X. DOI: 10.1016/j.future.2020.10.010.

[43] R. Fayos-Jordan, S. Felici-Castell, J. Segura-Garcia, J. M. Alcaraz-Calero and A. Cervello-Duato, 'Exploiting multihoming capabilities in 5g-enabled iot nodes,' vol. 11, pp. 134 940–134 950, 2023, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2023.3338180. [Online]. Available: https://ieeexplore.ieee.org/document/10336763/.

[44] N. Verba, K. M. Chao, A. James, D. Goldsmith, X. Fei and S. D. Stan, 'Platform as a service gateway for the fog of things,' vol. 33, 2017, ISSN: 14740346. DOI: 10.1016/j.aei.2016.11.003.

[45] R. Fayos-Jordan, S. Felici-Castell, J. Segura-Garcia *et al.*, 'Thorough analysis of raspberry pi devices in outdoor/indoor communications in terms of qos,' vol. Part F1667, Association for Computing Machinery, 2020, pp. 1–7, ISBN: 9781450377119. DOI: 10.1145/3401895.3401915.

[46] R. Fayos-Jordan, S. Felici-Castell, J. Segura-Garcia, A. Pastor-Aparicio and J. Lopez-Ballester, 'Elastic computing in the fog on internet of things to improve the performance of low cost nodes,' vol. 8, 12 2019, ISSN: 20799292. DOI: 10.3390/electronics8121489.

[47] A. C. Inc, *Tinker board — aiot & industrial solution — asus united kingdom.* [Online]. Available: https://www.asus.com/uk/networking-iot-servers/aiot-industrial-solutions/all-series/tinker-board/.

[48] S. S.P.A., *Udoo single board computer — x86 & arm powerful arduino mini pc.* [Online]. Available: https://www.udoo.org/.

[49] N. Corporation, *Embedded systems developer kits & modules — nvidia jetson.* [Online]. Available: https://www.nvidia.com/en-gb/autonomous-machines/embedded-systems/.

[50] R. Fayos-Jordan, S. Felici-Castell, J. Segura-Garcia, J. Lopez-Ballester and M. Cobos, 'Performance comparison of container orchestration platforms with low cost devices in the fog, assisting internet of things applications,' vol. 169, 2020, ISSN: 10958592. DOI: 10.1016/j.jnca.2020.102788.