






Article

A Smart Building Fire and Gas Leakage Alert System with Edge Computing and NG112 Emergency Call Capabilities

Evangelos Maltezos , Konstantinos Petousakis , Aris Dadoukis, Lazaros Karagiannidis ,
Eleftherios Ouzounoglou , Maria Krommyda, George Hadjipavlis and Angelos Amditis 

Institute of Communication and Computer Systems (ICCS), 15773 Zografou, Greece;
konstantinos.petousakis@iccs.gr (K.P.); aristeidis.dadoukis@iccs.gr (A.D.); lkaragiannidis@iccs.gr (L.K.);
eleftherios.ouzounoglou@iccs.gr (E.O.); maria.krommyda@iccs.gr (M.K.); giorgos.hadjipavlis@iccs.gr (G.H.);
a.amditis@iccs.gr (A.A.)

* Correspondence: evangelos.maltezos@iccs.gr

Abstract: Nowadays, the transformations of cities into smart cities is a crucial factor in improving the living conditions of the inhabitants as well as addressing emergency situations under the concept of public safety and property loss. In this context, many sensing systems have been designed and developed that provide fire detection and gas leakage alerts. On the other hand, new technologies such edge computing have gained significant attention in recent years. Moreover, the development of recent intelligent applications in IoT aims to integrate several types of systems with automated next-generation emergency calls in case of a serious accident. Currently, there is a lack of studies that combine all the aforementioned technologies. The proposed smart building sensor system, SB112, combines a small-size multisensor-based (temperature, humidity, smoke, flame, CO, LPG, and CNG) scheme with an open-source edge computing framework and automated Next Generation (NG) 112 emergency call functionality. It involves crucial actors such as IoT devices, a Public Safety Answering Point (PSAP), the middleware of a smart city platform, and relevant operators in an end-to-end manner for real-world scenarios. To verify the utility and functionality of the proposed system, a representative end-to-end experiment was performed, publishing raw measurements from sensors as well as a fire alert in real time and with low latency (average latency of 32 ms) to the middleware of a smart city platform. Once the fire was detected, a fully automatic NG112 emergency call to a PSAP was performed. The proposed methodology highlights the potential of the SB112 system for exploitation by decision-makers or city authorities.

Keywords: fire detection; gas leakage detection; smart cities; smart building; critical infrastructure; edge computing; next-generation 112 call; emergency situation; public safety answering point; PSAP; common operational picture; COP



Citation: Maltezos, E.; Petousakis, K.; Dadoukis, A.; Karagiannidis, L.; Ouzounoglou, E.; Krommyda, M.; Hadjipavlis, G.; Amditis, A. A Smart Building Fire and Gas Leakage Alert System with Edge Computing and NG112 Emergency Call Capabilities. *Information* **2022**, *13*, 164. <https://doi.org/10.3390/info13040164>

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

Received: 18 February 2022

Accepted: 23 March 2022

Published: 24 March 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

1.1. Existing Challenges in Smart Cities

In recent years there has been a great deal of discussion and studies on transforming cities into smart cities. A smart city is a city that uses digital technology to protect, connect, and enhance citizen's lives, and mainly refers to six characteristics: (i) smart economy, (ii) smart mobility, (iii) smart environment, (iv) smart people, (v) smart living, and (vi) smart governance [1]. However, there are other sub-categories of the aforementioned, such as smart health, smart energy/water/waste, and smart safety. In this context, there are several objectives and research challenges that a smart city should consider: (i) data from different IoT sources/devices should be available for easy aggregation; (ii) data should be easily visualized and securely accessible while respecting privacy, (iii) data should be detailed and measurable, and real-time knowledge should be available at every level; (iv) analytics and decision-making systems should be used; (v) the city should incorporate state of the

art technologies for automation and further relevant extensibility; (vi) the city should have a network of collaborative spaces, and (vii) decision-making processes should be open and inclusive.

In recent years many sensing systems have been designed and developed to provide fire alerts under the smart city concept. Among others, public safety (in terms of injury and/or loss of life) and the reduction of property loss are the two main crucial issues that these systems seek to address. In the case of a fire threat, such sensing systems automatically notify homeowners or the relevant stakeholders about the emergency situation, and may perform certain actions to reduce the impact. Fire in buildings or infrastructure (either residential or professional/industrial) can occur due to several reasons, including (i) the burning of materials, gases, and electrical circuits, which can cause serious accidents, (ii) improper maintenance of equipment and overheating of machinery, or (iii) malicious action.

The development of intelligent applications in IoT has gained significant attention in recent years, with a special focus on safety and security applications in smart cities. New Internet of Things (IoT) applications, including Augmented Reality (AR), Virtual Reality (VR), Digital Twins (DT), virtual simulations, real-time search engines, and discovery services bring with them new challenges [2]. Such applications continuously generate enormous amounts of data and types of data which demand strict latency-aware computational processing capabilities, as well as a homogeneous approach for data processing and the generation of associated alerts, events, or raw information. Although cloud computing is able to provide benefits to IoT such as scalability, elasticity, multitenancy, storage capability, and resource pooling, it faces accessibility challenges. The unstable connection between the cloud and mobile devices is expected to prevent providers from achieving optimal performance. In order to solve these problems, edge computing attempts to decrease latency and support the massive machine type of communications.

Thus, the recent technology of edge computing is considered to be a flexible and viable solution that is able to overcome the limitations of cloud computing. Edge computing can be used for real-time smart city environments, enabling (i) context-awareness, (ii) geo-distributed capabilities, (iii) low latency, and (iv) migration of computing resources from the remote cloud to the network edge. In these terms, edge computing could scale from a single person to a smart individual building to even an entire city. Moreover, recent trends aim to integrate IoT-based systems with next-generation automated emergency calls, such as the single European emergency number (112) that, in the case of a serious accident, automatically performs an emergency call to the closest 112 public-safety answering point (PSAP).

1.2. Description of the Current State-of-the-Art

1.2.1. Fire and Gas Leakage Detection Systems

Several fire detection systems have been proposed in the bibliography that provide fire alerts for interior building cases based on various sensors (e.g., smoke, temperature, flame), and image processing techniques. In [3], an intelligent fire detection and mitigation system safe from fire was designed and performed using flame sensors. In [4], an IoT-based fire detection system was designed to protect people from fire by providing an alert message in emergency situations. Among others, an effective carbon monoxide (CO) sensor was used for measuring the amount of CO in the air. In [5], the authors designed a smoke detection and alarm system based on low range (LoRa) technology combining low power consumption and long distance transmission. The advantages of their system were its low power consumption, strong anti-interference ability, and long-distance transmission. On the other hand, in [6], an extensive study was carried out to improve the prediction accuracy of detector activation time by establishing a database of input parameters required for the fire simulation of various smoke detectors and combustibles. In [7], the authors proposed an architecture combining accurate and early sensing of a fire event, IP cameras for tracing people inside a building under fire, and long and inter-node communication capabilities, as well as an efficient user interface. In [8], an IoT-capable fire detection unit

with a camera system and enhanced Convolutional Neural Network (CNN) was effectively used for fire detection in indoor and outdoor environments. In [9], a fire detection system was proposed to improve fire focus in real time, allowing movement in confined spaces with no visibility or physical references. Their system adopted a computer vision algorithm designed for fire detection through thermal imaging able to operate with video sequences in low visibility conditions. The authors of [10] proposed a fire warning algorithm for indoor fire scenarios via a multisensor (temperature, smoke and CO) system. A solution known as a smart fire detection and deterrent system was recently introduced by [11] with the aim of detecting fire via multiple sensors (temperature, heat, humidity and smoke) by utilizing energy and relevant resources.

Gas leakage is another unwanted situation, as compressed natural gas (CNG) and liquefied petroleum gas (LPG) are used for several purposes in buildings and infrastructure. To avoid this dangerous situation, a gas leakage detection system is needed. The authors of [12] introduced a low cost gas leak detection and automatic alert system based on an LPG sensor. With a similar approach, [13] proposed a gas leak detection and rule-based automatic control system utilizing both LPG and CNG sensors. Another interesting study is the one proposed by [14]; in their study, an intelligent fire detector prototype was made with smoke, CO, temperature, and flame sensors. Upon confirmation of a fire within the building, relevant registered users are informed. Furthermore, messages from a central server database help to navigate a UAV with a visible camera and infrared (IR) temperature sensor, enabling the streaming of live videos to all the registered users.

In the works referenced in the bibliography, several efficient algorithms have been proposed for identifying and or classifying fire detection and gas leakage alerts, including (i) decision trees, (ii) rule-based and fuzzy logic, (iii) neural networks, (iv) probabilistic neural networks, (v) Hierarchical Linear Discriminant Analysis (LDA), (vi) Principal Component Analysis (PCA), and (vii) Nearest Neighbours (NN) classifiers [15]. The main advantage of the rule-based methods compared to the other, more complex techniques is that they are considered “white boxes”, as they are easy to interpret. Methods that ultimately rely on the definition of thresholds, sensitivity, and specificity can be adjusted by tuning the corresponding thresholds depending on the current operational conditions (e.g., room size and geometry, fire type, nuisances). In order to provide a more generalized model, dynamic features and complex schemes should be considered.

In general, a fire is directly detected when the flame sensor is triggered. Furthermore, rules that include CO measurements result in faster detection of smoldering fires than smoke detectors. The rate of temperature rise results in similar or faster fire detection compared to smoke detectors [15]. However, a combination of rules can be considered for the detection of fire (e.g., in cases where the concentration values of two sensors such as smoke and temperature exceed their selected thresholds). False alarms can be reduced at the cost of reducing sensitivity to smoldering fires. Smoldering fires have been traditionally characterized by CO emissions, although many other volatiles appear, especially with the use of new building materials. CO is considered the most important and well-studied toxic emission from fires. Time to alarm can be longer for smoldering fires compared to open fires [15] in which other sensors such flame, smoke and temperature are more active.

1.2.2. Edge Computing

Several recent studies that integrate edge computing schemes with fire detection systems are described in this section. In [16], a cyber-physical social system for early fire detection was proposed, employing a scalable edge computing framework receiving captured information from several IoT nodes. A smart firefighting system for smart cities adopting fog/edge computing was introduced by [17]. Their system utilized heat, smoke, and flame detectors and applied a machine learning algorithm, thus identifying fire detection alerts. In [18], the authors developed a fire alarm system for smart cities using edge computing via different types of sensors, such as smoke, temperature, humidity,

flame, methane, and CO; whenever a node detects a fire, it signals the centralized node to alert the user.

1.2.3. Emergency Calls

According to the Strategic Research and Innovation Agenda (SRIA) document [19], IoT represents a research and innovation priority in crisis management and safety for buildings and infrastructure. In [20] it is suggested that upcoming services in emergency services network infrastructure adopt Next-Generation (NG) 112 architectures for efficient emergency call handling, along with other useful information such as spatial data. In this context, several studies have been proposed that make emergency calls after a serious accident. In [21], the integration of telematics and smart devices in emergency services and NG eCall, leveraging the three key enablers of total conversation, device data, and location-aware devices, was explored. In [22], the design of a NG112 emergency system based on the Pan-European Mobile Emergency Application (PEMEA) protocol was designed and extensively tested, including a demonstration event with all major system stakeholders (first responders, PSAP operators, and solution vendors) in attendance. Finally, an automatic information exchange in the early rescue chain using the International Standard Accident Number (ISAN) was presented in [23], connecting various information and communication technology (ICT) systems.

1.3. Our Contribution

Currently, there is a lack of studies that combine edge computing technology and emergency capability in a fire and gas leakage alert system. The proposed smart building sensor system, namely SB112, combines a small-size multisensor-based scheme with an open-source edge computing framework and an automated NG112 emergency call functionality, and involves crucial actors such as IoT devices, PSAP, the middleware of a smart city platform, and relevant operators in an end-to-end manner for real-world scenarios. Moreover, the technologies used in the implementation are state-of-the-art, such as Go programming language [24], microservices, and 3D printed cases.

The SB112 is able to publish both geolocated raw measurements from sensors and fire and gas leakage alerts in real time through a set of thresholds (or rules) in a smart city's custom data model to its middleware (e.g., the smart city platform). In this context, the SB112 system has great potential for use by decision-makers or city authorities under a common operational picture (COP) aspect through a smart city platform. More specifically, the main characteristics of the SB112 system are:

- It is an embedded system that adopts a low cost and small-size multisensor (temperature, humidity, smoke, flame, CO, LPG, and CNG) scheme in a Wireless Sensor Network (WSN) manner, ensuring reliable and expanded detection coverage of fire and gas leakage.
- It is integrated with the proposed edge computing platform, namely, the Distributed Edge Computing Internet of Things (DECIoT). The DECIoT is a scalable, secure, flexible, fully-controlled, and potential interoperable and modular open-source framework that ensures information sharing (raw measurements from the sensors and/or fire and gas leakage alerts) with other platforms or systems (utilizing proper middleware). Through this approach, optimal use of IoT nodes is ensured.
- Once a fire or gas leak is detected, an NG112 emergency call is carried out through the terms of a Common Alerting Protocol (CAP) message (fully automatically or upon confirmation from a COP operator or decision-maker) to a PSAP. In this study, the PSAP was simulated as an individual workstation considered to be located on a stakeholder's premises. Finally, the PSAP operator is able to request values from the SB112 system for monitoring reasons (e.g., the last fifteen values reported by the sensors during an event or operation).

This paper is organized as follows. Section 2 presents the proposed methodology as well as the hardware components of the SB112 system and describes the proposed

distributed edge computing platform and the NG112 emergency call scheme. Section 3 provides real-time experimental results associated with a fire alert triggered by physically activating a sensor node. Finally, Section 4 concludes the paper.

2. Materials and Methods

2.1. Proposed Methodology

In Figure 1, we present the architectural design of the SB112 system. The SB112 system consists of following main pillars: (i) the hardware components such as the sensor nodes (sensors and relevant microcontroller units) and the edge gateway (described in detail in Section 2.2); (ii) the edge computing platform, that is, the DECIoT platform (described in detail in Section 2.3); and (iii) the NG112 emergency call component (described in detail in Section 2.4). In this study, Apache Kafka [25] was used as middleware and was considered to represent the middleware of a smart city platform. The COP operator was considered to be the human ‘in the loop’ receiving and managing the smart city platform information. Apache Kafka is an effective and open-source distributed event streaming platform for high-performance data pipelines, streaming analytics, data integration, and mission-critical applications in real world conditions, such as smart cities [26].

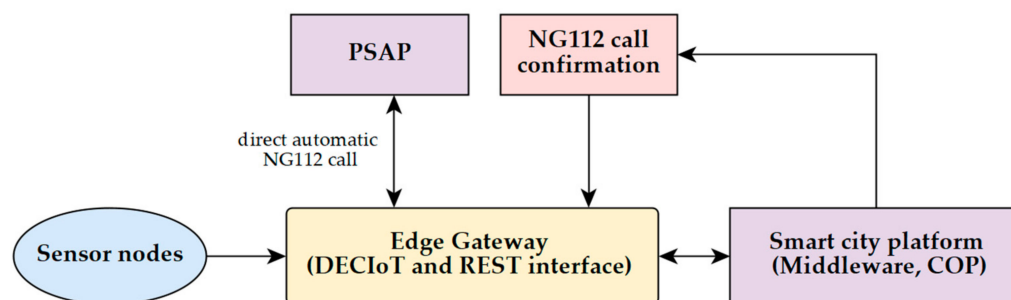


Figure 1. The proposed SB112 prototype architecture.

Sensor nodes are the source of information, i.e., raw measurements from the sensors. The sensor nodes periodically send raw measurements to the edge gateway; the frequency can be adapted depending on the use case. Such information is then pushed to the DECIoT, which is installed to the edge gateway. The DECIoT platform consists of multiple layers, and each layer contains multiple microservices by which the information from the sensor nodes is aggregated, persisted, filtered, and processed in order to generate/trigger relevant alerts (fire and/or gas leak detection). Then, the DECIoT has a dual role: (i) to publish the raw measurements and the fire detection and/or gas leakage alerts to Apache Kafka, and (ii) to be connected with a REST interface in order to make an NG112 call in the form of a CAP message (fully automatic or upon confirmation from the COP operator) to a PSAP. During the emergency situation, the PSAP operator is able to request values from the SB112 system for monitoring reasons. The SB112 system considers several crucial use-case requirements:

- Real time measurements, timely fire and gas leak detection, and generation of associated alerts. The ability of fires or gas leaks in buildings to spread extremely quickly makes their detection and suppression at an early stage a necessity. Such time-critical applications demand low-latency access to devices at the edge of the network, as well as the ability to perform rapid computations in order to take immediate decisions. Thus, the continuous and stable operation of the system is necessary in order to prevent fire from spreading, or unwanted situations associated with gas leakage.
- For optimal use of IoT nodes and interactive control, the SB112 system is able to perform and operate autonomously for prolonged periods of time, perform complex tasks depending on the use case, and to acquire measurements through commands either from specific nodes or from all of them. Thus, an optimal configuration is deployed providing automation, remote control, wireless data transferring, and the

ability to form scalable networks. The proposed DECIoT provides flexibility, modularity, and potential adaptiveness to incorporated nodes such the ones described in this paper (i.e., with embedded fire and gas leakage sensors) as well as to nodes with any other type of sensors (e.g., a camera) or legacy systems. Thus, the DECIoT is able to provide potential support in an interoperably and to integrate information from heterogeneous sources.

- The SB112 system is highly scalable. The edge gateway is capable of supporting a large number of sensor nodes of any kind. Thus, by creating a network of edge gateways, each with its own sensors, the system can scale quickly and easily. It can cover large areas and handle a variety of different sensor types and measurements, which makes it capable of detecting various emergency conditions. Therefore, it can be applied to multiple use cases.
- For secure operation, recent advances in sensing technologies and data collection mechanisms make feasible the deployment of vast sensor networks that can potentially become intrusive and violate established regulations (e.g., GDPR). Edge computing frameworks are able to keep processing to the edge of the network, avoiding the indiscriminate transmission and storage of sensitive information such as image and video recordings. Although the SB112 system described in this paper does not collect and process image or video recordings, the measurements that it collects can be considered confidential, e.g., in the case of monitoring critical buildings or infrastructure. For this reason, the DECIoT incorporates a security mechanism (that is easy to enable or disable) in order to communicate with the relevant interfaces.
- The SB112 considers several crucial actors involved in emergency situations, such as sensors, IoT devices (sensor nodes and edge gateway), the COP operator of the city authority which manages the smart city platform, and the PSAP with its corresponding operator. Hence, the use case includes the following steps: (i) the SB112 is installed to infrastructure or a building and makes constant measurements of indoor air quality from its multiple sensors in a continuous operation manner; (ii) the SB112 detects an abnormal situation (fire or gas leak); (iii) the SB112 system initiates a fully automatic NG112 emergency call to a PSAP; (iv) the PSAP operator receives the NG112 emergency call; and (v) the PSAP operator requests values from the SB112 system. The above use case can be enriched in the case that a COP operator exists. The necessary condition is that the smart city platform and the SB112 system use the same middleware. Thus, once the SB112 detects an abnormal situation, the COP operator receives the relevant alert through the smart city platform. Then, the COP operator confirms the emergency event (or not) and provides an event response. In the case that the COP operator marks the alert as a false positive, this action terminates the use case. In case that the COP operator marks the alert as a true positive, the SB112 system receives the 'alert accepted' response and initiates an NG112 emergency call to the PSAP. It should be noted that the SB112 system ought to be pre-configured (i.e., to either make a fully automatic NG112 call or to make an NG112 call upon confirmation from the COP) depending on the final involved actors.

2.2. Hardware Components

The SB112 system utilizes the following low-cost and efficient sensors: (i) an SHTC3 temperature/humidity sensor, (ii) a TO-39 IR/flame sensor, (iii) an MQ-9B CO sensor, (iv) an MQ-2 smoke sensor, (v) an MQ-6 LPG sensor, and (vi) an MQ-4 CNG sensor. Such sensors have been successfully used in several studies in extended experiments proving their effectiveness for sensing applications [4,9,11–14]. The sensor node is based on an ESP32 microcontroller unit that has the capability to integrate multiple sensors according to the needs of various use cases.

In Figure 2, the physical outlook and pin configuration of the ESP32 microcontroller and its components is depicted, showing how the sensors are wired and which pins of the microcontroller are used. IO represents the Input/Output pins, while SCL and SDA are the

data and clock pins of the I2C interface, respectively. As an edge gateway, a Raspberry Pi 4 was utilized. Both ESP32 and the Raspberry Pi 4 have been used in several fire detection systems, proving their efficiency and flexibility [13,14,27]. Each sensor node requires a power supply for continuous operation, as does the edge gateway.

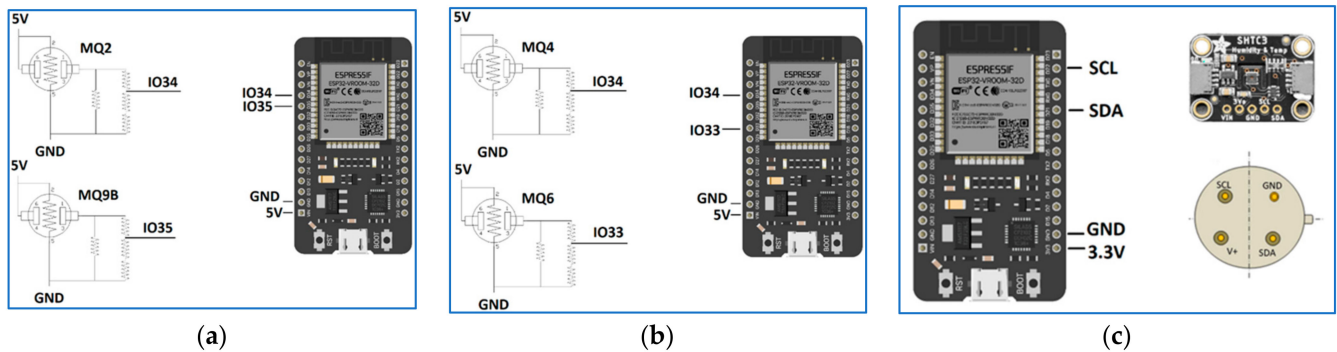


Figure 2. Physical outlook and pin configuration of the ESP32 microcontroller and its components for smoke/CO sensors (a), LPG /CNG sensors (b) and temperature/humidity/IR/flame sensors (c).

Within each sensor node two sensors were integrated via the ESP32, as follows: (i) sensor node 1 consists of an MQ-9B and MQ-2; (ii) sensor node 2 consists of an SHTC3 and TO-39; and (iii) sensor node 3 consists of an MQ-6 and MQ-4. These sensors nodes were created to be distributed and cover a typical room inside of a building (e.g., 25 m² and 3 m height) in a proof of concept of the SB112 prototype. Figure 3 shows the sensors, the microcontroller unit, and the edge gateway, while Figure 4 depicts the connectivity between them. The sensor nodes are connected with the edge gateway in a WSN manner; in this study, a Wi-Fi module was used.

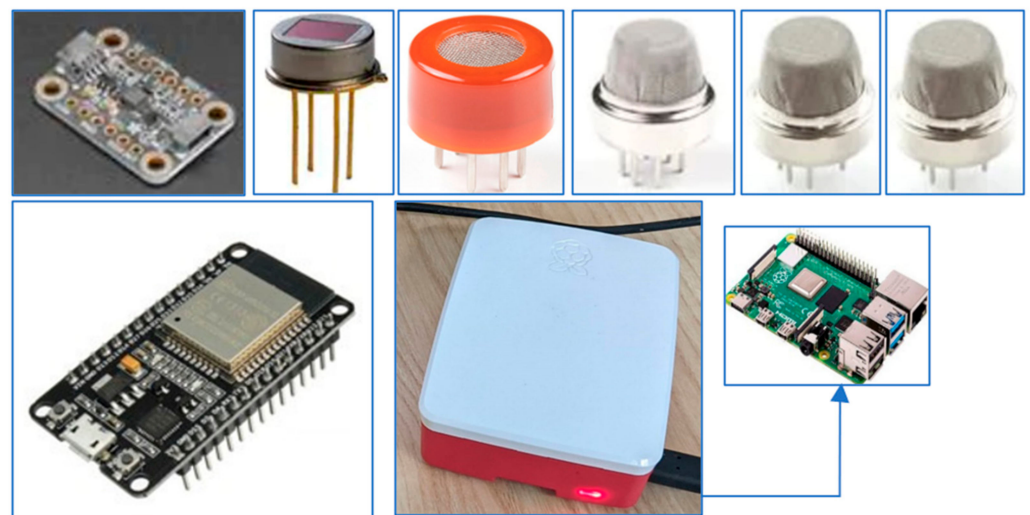


Figure 3. Top (from left to right): the sensors used (SHTC3, TO-39, MQ-9B, MQ-2, MQ-6, and MQ-4). Bottom left: the microcontroller unit (ESP32). Bottom right: the edge gateway (Raspberry Pi 4).

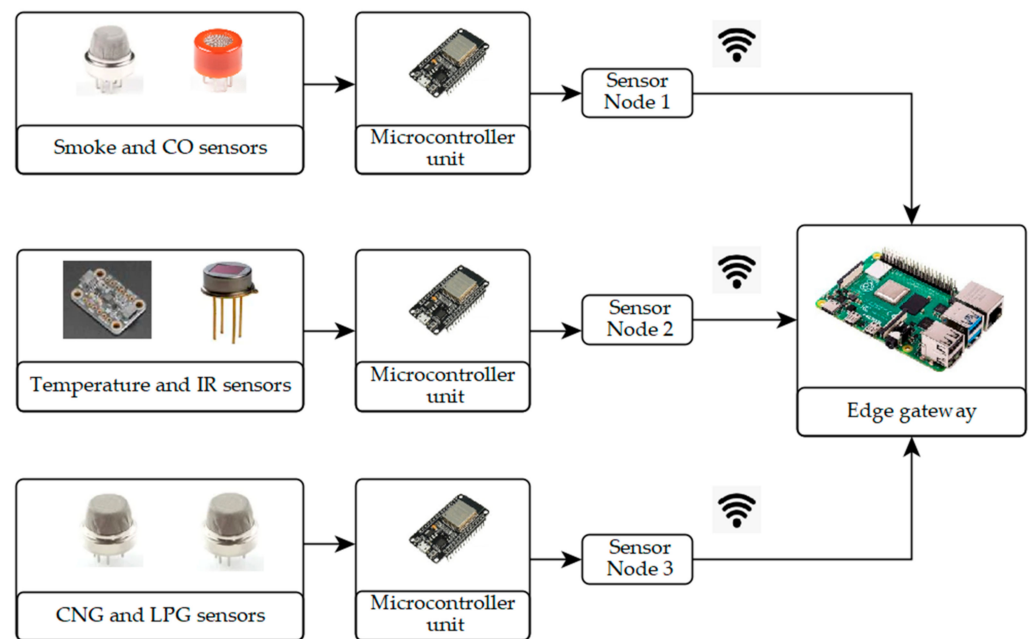


Figure 4. Connectivity diagram depicting the sensors, the microcontroller unit, and the edge gateway.

For the analog sensors, we used a voltage divider in parallel in order to convert the output voltage range of the sensor from 0–5 V to 0–3.3 V, which is the operating range of the analog to digital converter (ADC) of the microcontroller. In order to be independent of the existing network infrastructure for the microcontroller–edge gateway communication, the edge gateway serves as a Wi-Fi access point which the microcontroller connects to. The microcontrollers are programmed to read the sensor values every 30 s and send them through MQTT to the edge gateway and the DECIoT. Additionally, various parameters of the program, such as the sampling period and other sensor related parameters, can be changed with a command through MQTT [28].

2.3. Distributed Edge Computing IoT Platform (DECIoT)

Edge computing refers to the enabling technologies that allow computation to be performed at the network edge, such that the computing happens near the data sources. An edge device is any computing or networking resource residing between data sources and cloud-based datacenters. In edge computing, the end device both consumes and produces data. Such devices request services and information from the cloud and perform several real time computing tasks (e.g., storage, caching, filtering, processing) using data sent to and from the cloud. The edge computing framework designed here should be efficient, reliable, and secure while having extensibility.

In this study, a proposed edge computing platform called DECIoT was designed and developed with the intent of integrating it into the SB112 system. The DECIoT platform addresses, among others, the problems of gathering, filtering, and aggregating data while interacting with the IoT devices, providing security and system management, providing alerts and notifications, executing commands, temporarily storing data for local persistence, transforming and processing data, and ultimately exporting the data in formats and structures that meet the needs of other platforms. This process carried out using open-source microservices that are state of the art in the area of distributed edge IoT solutions.

The DECIoT (Figure 5) is built upon a collection of open-source microservices that function on the edge of the physical realm (the Device Services Layer), with the Core Services Layer and the Support Layer at the center and the Application Layer on the top. The DECIoT is based on the EdgeX foundry open source framework [29]. The EdgeX foundry is considered in the literature to be a highly flexible and scalable edge computing

framework that facilitates interoperability between devices and applications at the IoT edge, such as industries, laboratories, and data centres [27,30–32]. According to several organizations (e.g., the Alliance for Internet of Things Innovation, AIOTI [19]) EdgeX is recognized among others as one of the Open Source Software (OSS) initiatives currently focusing on edge computing.

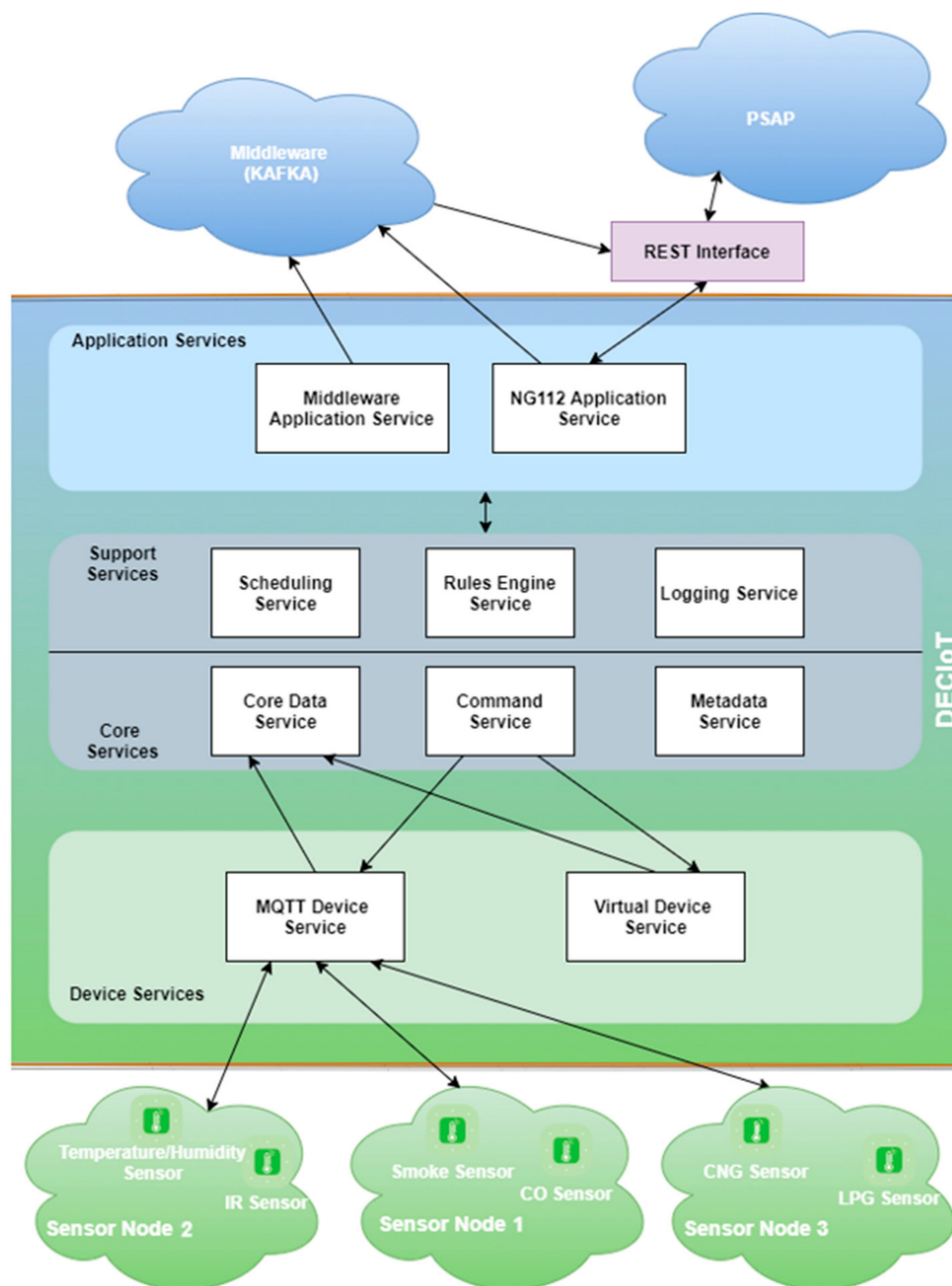


Figure 5. DECIoT architecture.

The DECIoT platform follows the microservice architecture pattern, not the traditional monolithic architecture pattern. The main principle of microservice architecture is that an application can be designed as a collection of loosely-coupled services, with each service being self-contained software responsible for implementing a specific functionality of the application. The DECIoT platform consists of multiple layers, and each layer contains multiple microservices. All microservices can communicate with each other in one of two ways, either through standardized REST APIs, for quick and direct communication, and/or

through the message bus, which is implemented as a pub/sub mechanism. Message bus via Redis Pub/Sub is used by default; alternatively, ZeroMQ or MQTT can be used.

Security can be enabled in DECIoT with the use of two major security microservices. The Security Store is a safe place to store all classified DECIoT information, and all other microservices can interact with this store to save and retrieve such classified information; an edge gateway acts as a reverse proxy for all other APIs, restricting access to them. In all layers in the DECIoT ecosystem, an SDK is provided, allowing a user to implement a different version of a microservice, or an entirely new one. Below, we present the different layers of DECIoT, from southbound physical devices to northbound external middleware/infrastructures and applications (in this study, Apache Kafka was used as middleware and considered to represent the middleware of a smart city platform).

- The Device Service Layer acts as an interface between the system and physical devices, and is tasked with the functionality of collecting data and actuating the devices with commands. It supports multiple protocols for communication through a set of device services (MQTT Device Service, REST Device Service, Virtual Device Service) and an SDK for creating new device services. In the SB112 system, the MQTT Device Service was used to receive information (i.e., raw measurements) from the sensor nodes. Between the sensor nodes and the MQTT Device Service, there is an MQTT broker.
- The Core Services Layer is at the center of the DECIoT platform and is used for storing data and for commanding and registering devices. The Core Data Service is used for storing data, the Command Service initiates all the actuating commands to devices, and the Metadata Service stores all the details for the registered devices. Specifically, the Metadata Service manages information about devices connected to and operated by the DECIoT, knows the type and organization of data reported by the devices, and knows how to command the devices. These microservices are implemented with the use of Consul, Redis, and adapters developed with the Go programming language for integration with all other microservices. In the SB112 system, all of the above microservices are exploited.
- The Support Services Layer includes microservices for local/edge analytics and typical application duties such as logging, scheduling, and data filtering. The Scheduling Service is a microservice capable of running periodic tasks within DECIoT (for example, cleaning the database of the Core Data Service each day) and is capable of initiating periodic actuation commands to devices using the Command Core Service; this is an implementation in Go that exploits features of Consul and Redis. The Rules Engine Service performs data filtering and basic edge data analytics; Kuiper is used in this microservice. The Logging Service, a Go language implementation, is used for logging the messages of other microservices. In the SB112 system, the Rules Engine microservice was used in order to generate and trigger alerts using raw measurements from the sensor nodes. Here, selected thresholds are considered to generate and trigger the relevant alerts; see Section 3.1.
- The Application Services Layer consists of one or more microservices with the functionality of communicating with external infrastructure and applications. Application Services are the means by which data are extracted, transformed, and sent from the DECIoT to other endpoints or applications. Using this layer, the DECIoT can communicate with a variety of middleware brokers (e.g., MQTT, Apache Kafka) or REST APIs with the goal of reaching external applications and infrastructure. At the same time, the support of an SDK allows the implementation of new application services that fit the use case. For the SB112 system, two application services were implemented, the NG112 application service and the Middleware application service. The NG112 application service is used for two purposes: (i) to send alerts to the middleware, and (ii) to initiate an NG112 emergency call to a PSAP. The Middleware application service is used to send the information received from the Core Data Service to the middleware (i.e., raw measurements from the sensor nodes). Between the NG112 application service and the PSAP, a REST Interface exists.

2.4. NG112 Emergency Call Component and PSAP

The module that starts the NG112 call procedure is the Kuiper rules engine [33]. The engine can be configured with various and complex rules to filter incoming data from the Core Data Service and take an action. In this case, it sends an MQTT message to the alert topic (the topic changes depending on the type of the alert). The NG112 application service in the Application Services layer of the DECIoT runs an MQTT consumer and listens to the alert topics. When an alert arrives, the application checks whether the same alert has been fired again recently in a configurable timeframe and stops it, in order to avoid bloating the system. Afterwards, it forwards it to the smart city platform (Apache Kafka) and to the REST interface with a REST call.

The REST interface is responsible for formatting the alert into a CAP message and initiating the NG112 call with the Session Initiation Protocol (SIP). There are two modes for the NG112 call, fully automatic or with confirmation from the COP operator.

In case that the second mode is selected, the REST interface awaits a message from the COP operator through Apache Kafka in order to proceed with the NG112 call. The COP operator receives the alert from the smart city platform, processes it, decides whether an NG112 call should be made, and carries out the notification accordingly. Finally, past data from the DECIoT can be requested. After receiving the NG112 message, the PSAP can make a request to obtain the recent measurements, e.g., the temperature of the last hour. Then, the REST interface requests the data from the DECIoT and forwards them to the PSAP. It should be noted that in this study, the PSAP was simulated as an individual workstation considered to be located on a stakeholder's premises.

As for the software, the two application services were written in Go programming language using the application service SDK provided. The REST interface was written in Python programming language, and uses the Flask library to receive and send REST API calls and the Linphone SIP library to send SIP messages.

3. Results

3.1. Experimental Parameters and Setting

In order to ensure that the DECIoT platform is able to support real time streaming of data the system latency was measured, considering the time a measurement is been received on MQTT as the "start time" and the time a consumer consumes the measurement from Apache Kafka as the "end time". One hundred measurements were sent to DECIoT each time we executed the experiment; the experiment was executed ten times, with one hour between successive runs of the experiment. The experiment was conducted within the same physical machine in order to avoid measuring latencies caused by networking issues. The results of this experiment were very encouraging; on average, 32 ms latency was measured. In the best case, 6 ms latency was measured, and in the worst case, 113 ms latency was measured.

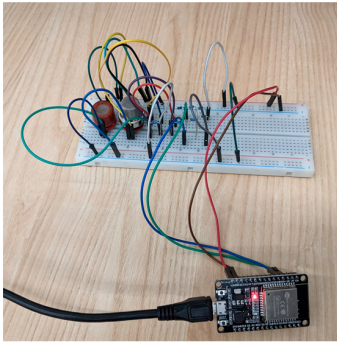
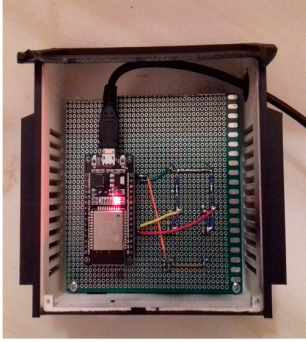

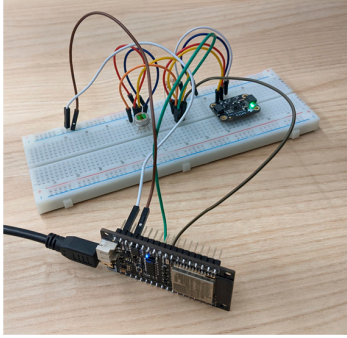
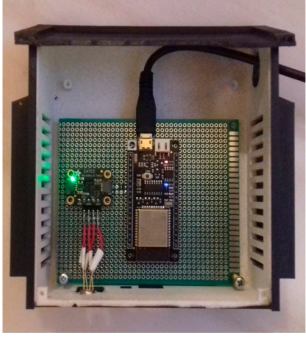

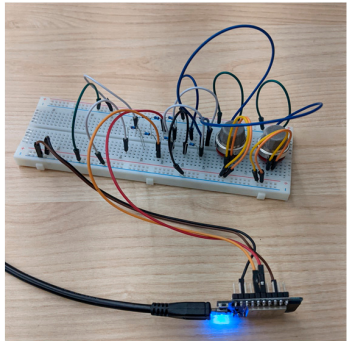
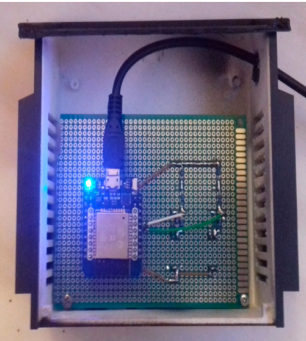

The conversion of the raw readings (voltage) of the used sensors to concentration parts per million (ppm) was achieved in this study using the calibration graphs supplied by the manufacturer's datasheet along with the approaches used in [34,35]. In this study, we adopted a traditional rule-based approach, taking into account readings from multiple sensors and building a set of thresholds. Hence, we adopted the thresholds (shown in Table 1) upon of which a fire or gas leak is considered detected. Such thresholds are associated with a typical case of room inside a building, and were selected from [9,10,12,13].

Table 1. Adopted thresholds for fire and gas leak detection selected from [9,10,12,13].

Sensor	Threshold
Temperature	60 Celsius
CO	50 ppm
Smoke	300 ppm
LPG	2100 ppm
CNG	1000 ppm

For installation and protection purposes, relevant packaging was prepared for each sensor node. The case cover of each sensor node was printed via 3D printer. The physical integration of the microcontroller units with the sensors as well as the final packaging of the sensor nodes are presented in Table 2. Each packaged node has the physical dimensions 13 cm height, 13 cm width, and 5 cm length.

Table 2. Physical integration of the microcontroller units with the sensors and packaging of the sensor nodes.

Sensor Node Id	Physical Integration of the Microcontroller Units with the Sensors	Packaged Sensor Node (Left: Inside; Right: Outside)	
1			
2			
3			

3.2. Experiment

In this paper, a representative end-to-end experiment was carried out from an applied perspective in order to identify a fire alert by physically triggering sensor node 1 under the terms of a fire event. A sequence diagram that depicts the data flow is presented in Figure 6.

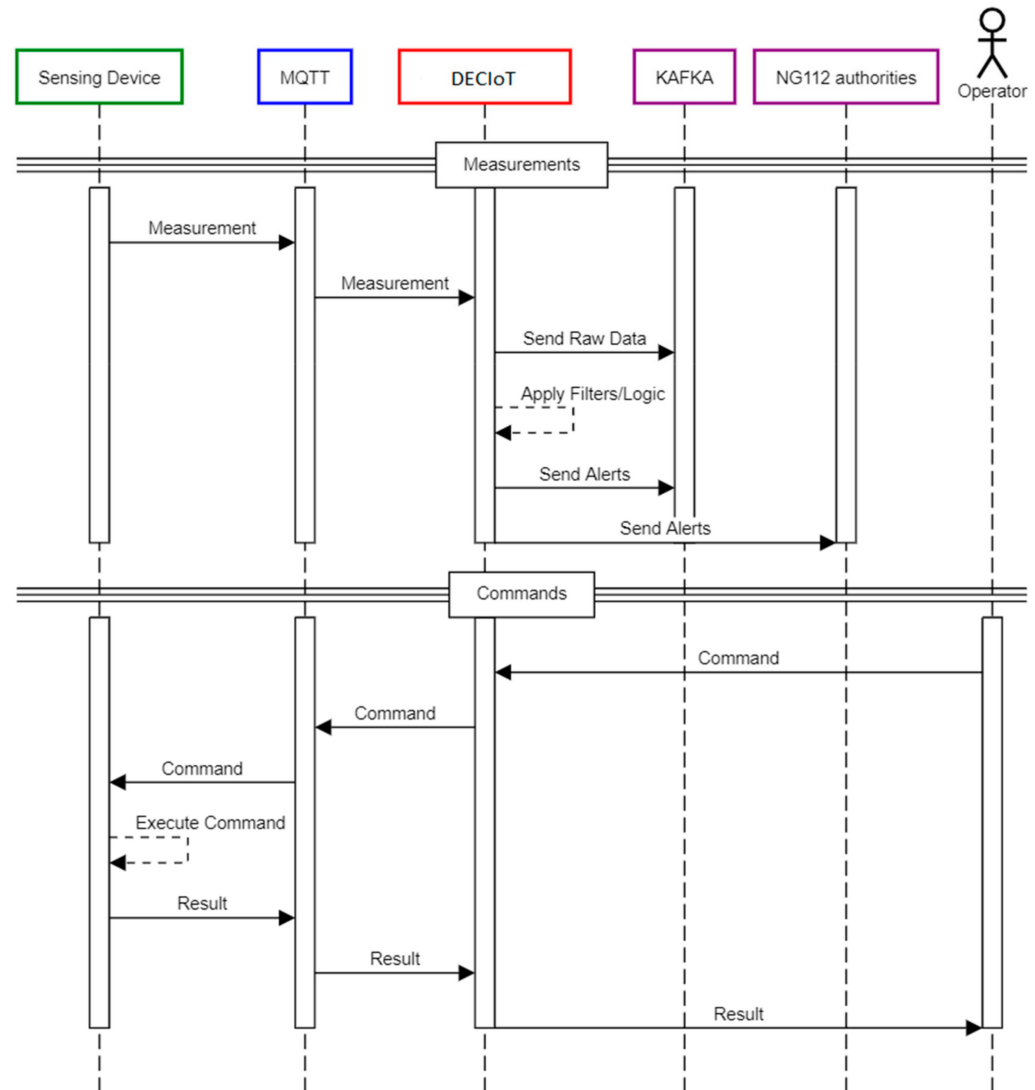


Figure 6. Sequence diagram depicting the data flow.

Initially, verification was carried out to confirm that Apache Kafka (Figure 7) received the raw measurements of sensor node 1 from the DECIoT in the JSON format on the specified topics ("smoke_concentration" and "CO_concentration"). It should be mentioned that, in this study, the Apache Kafka 'consumer' was executed in another external device. The geolocation (latitude, longitude, elevation) of sensor node 1 as well as the date and timestamp are additional information that is shared with the Apache Kafka user.

<pre>{ "source": "SB112_SENSOR01", "data_id": "0cb6e32a-51b9-4d74-b92a-c0b300awade9", "time": "2021-12-10T13:35:05.46+02:00", "unit": "ppm", "data_value": "0.778", "localization": { "lat": 37.97811429417777, "lon": 23.783215824380974, "elevation": 10 }, "data_type": "Smoke", "metadata": "SB112, smoke_concentration" }</pre>	<pre>{ "source": "SB112_SENSOR01", "data_id": "b67ff7fe-bffa-4b0c-8442-54-33d5a298d6", "time": "2021-12-10T13:35:05.46+02:00", "unit": "ppm", "data_value": "1.411", "localization": { "lat": 37.97811429417777, "lon": 23.783215824380974, "elevation": 10 }, "data_type": "CO", "metadata": "SB112, co_concentration" }</pre>
--	---

Figure 7. Verification that the smart city middleware (Apache Kafka) received the raw measurements of sensor node 1 from the DECIoT ((left): smoke; (right): CO).

After sensor node 1 was physically triggered by burning a piece of paper (Figure 8), the concentration values (in ppm) of the smoke and CO sensors were increased. Figures 9 and 10 show the concentration values of the smoke and CO sensor, respectively, before the event (time period from 18:24:20 to 18:34:00 24 h/pm), during the event (time period from 18:34:00 to 18:35:50 24 h/pm), and after the event (time period from 18:35:50 to 18:42:00 24 h/pm).



Figure 8. Physical triggering of sensor node 1 by burning a piece of paper.

Through the Support Services Layer of the DECIoT, a rule was associated with the fire alert to the effect that if the concentration values from both the smoke and the CO sensor exceed the adopted thresholds (shown in Table 1) at the same time, then a fire is detected. Figure 11 verifies that the Middleware Application Service printed the concentration values from the smoke and CO sensors during the event. The concentration values of 1560 ppm and 56 ppm from the smoke and CO sensors, respectively, (see Figures 9 and 10) exceeded the adopted thresholds; thus, a fire was detected. Detection of the fire triggered a fully automatic NG112 emergency call to the PSAP in the form of a CAP message (Figure 12).

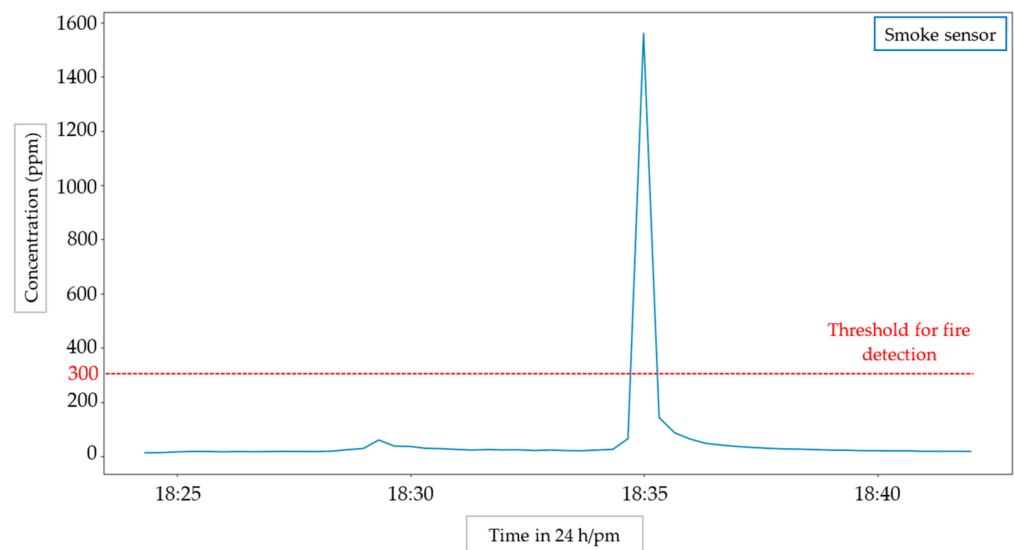


Figure 9. Concentration values for the smoke sensor before the event, during the event (time period from 18:34:00 to 18:35:50 24 h/pm), and after the event. The red line represents the threshold associated with fire detection.

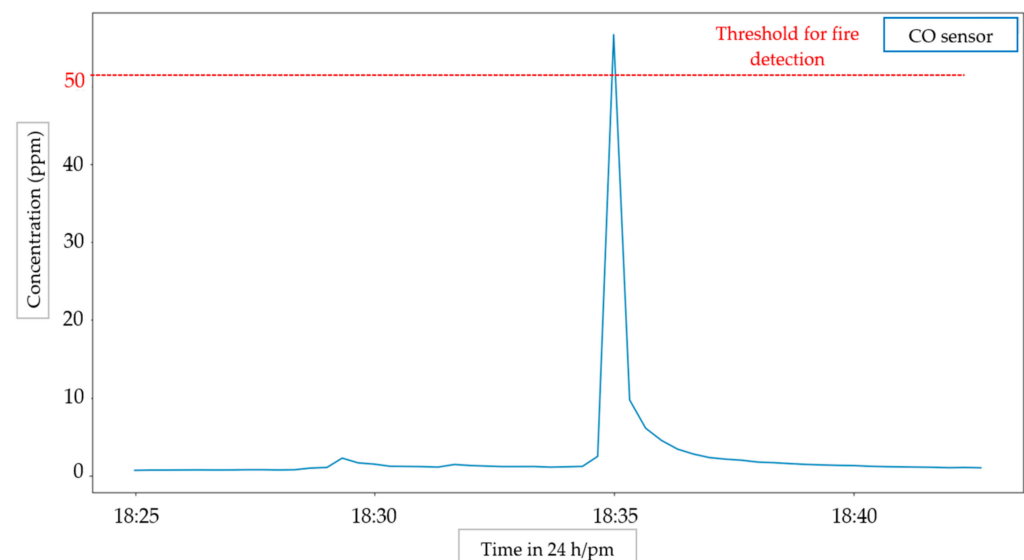


Figure 10. Concentration values for the CO sensor before the event, during the event (time period from 18:34:00 to 18:35:50 24 h/pm), and after the event. The red represents the threshold associated with fire detection.

```
Jan 20 16:34:59+02:00 sb112gateway app-service[6306]:print:
Jan 20 16:34:59+02:00 sb112gateway app-service[6306]:{"id":"66a09f52-dc83-4e1c-a6b2-79637ad29b3f","device":"NG112-MQTT-device",
"name":"smoke", "value":"1560.422"}
Jan 20 16:34:59+02:00 sb112gateway app-service[6306]:msg
Jan 20 16:34:59+02:00 sb112gateway app-service[6306]:{SB112_SENSOR01 2022-01-20T16:34:59.16+02:00 Smoke ppm {37.97811429417777, 23.783215824380974, 10}, smoke concentration
Jan 20 16:34:59+02:00 sb112gateway app-service[6306]:print:
Jan 20 16:34:59+02:00 sb112gateway app-service[6306]:{"id":"1a0511f0-1db5-4016-ba35-129a93bd8be1","device":"NG112-MQTT-device",
"name":"co", "value":"56.630"}
Jan 20 16:34:59+02:00 sb112gateway app-service[6306]:msg
Jan 20 16:34:59+02:00 sb112gateway app-service[6306]:{SB112_SENSOR01 2022-01-20T16:34:59.16+02:00 CO ppm {37.97811429417777, 23.783215824380974, 10}, co concentration
```

Figure 11. The Middleware Application Service prints concentration values from the smoke and CO sensors that exceed the thresholds.

```
message received: {"info":{"severity":"Severe","area":{"areaDescription":"Sensor_Installation","circle":37.97811429417777 23.783215824380974 10}},
"eventType":"Fire_detected","certainty":"observed","eventCategory":"Fire","urgency":"Expected"},"sentTime":1642696500275,"senderID":"SB112",
"messageType":"Alert","messageStatus":"Test","messageID":"b7cb3a38d5241e9a","scope":"Restricted"}
```

Figure 12. CAP message associated with the NG112 emergency call that the PSAP operator receives.

3.3. Installation of SB112 System in Indoor Buildings and Infrastructure

The SB112 system is a prototype that has been designed and developed in the context of the European Union's Horizon 2020 research and innovation programme, namely, the Spaces Safety And Security (S4AllCities) project [36]. As mentioned above, in this paper, a representative experiment was carried out from an applied perspective to identify a fire alert. Similar extended experiments described in Section 3.2 have been carried out by physically triggering of the sensor nodes (either individually or a combination) in the circumstances of a fire and/or gas leak case event using multiple rules according to the thresholds shown in Table 1. To serve the S4AllCities pilot project, the SB112 system was installed into a building of interest for continuous operation and monitoring (Figure 13).

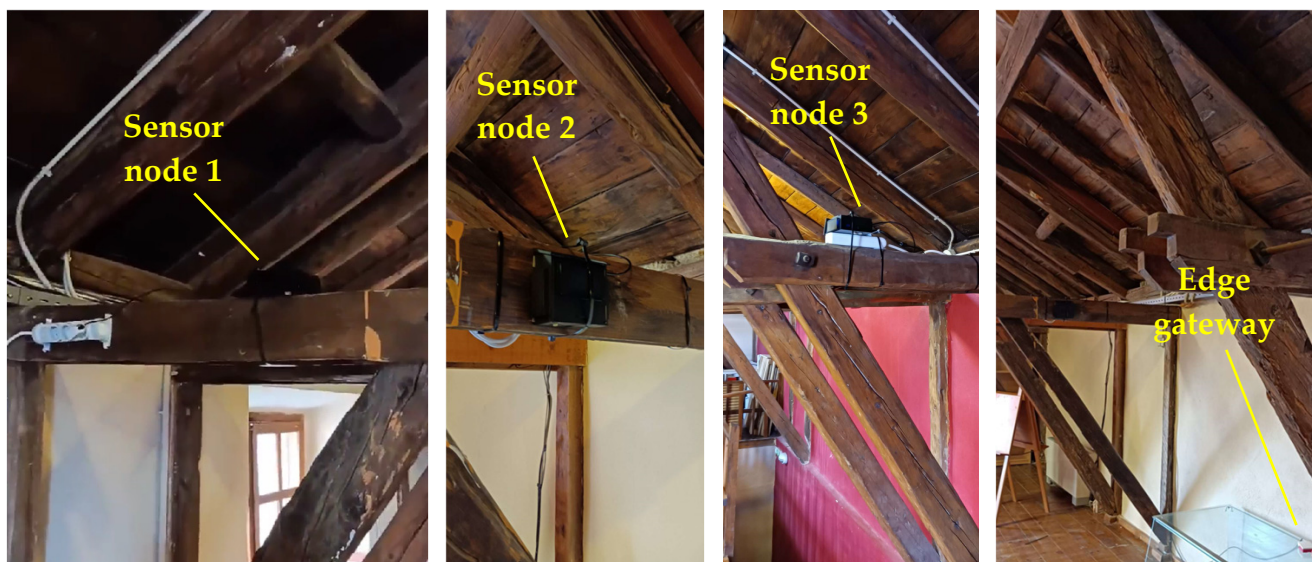


Figure 13. Installation of SB112 system into a building of interest for continuous operation and monitoring.

4. Conclusions

Public safety and the reduction of property loss are the two main crucial issues that a smart city seeks to address. The ability of fires or gas leaks to spread out extremely quickly in critical buildings or infrastructure makes their detection and suppression at an early stage a necessity. Recent technologies such edge computing and initiation of emergency calls through Next Generation calls to the European emergency number (112) have been considered among the recent trends contributing to the smart cities concept, especially in the emergency situation domain. The proposed system of this study, namely Smart Building (SB112), is an embedded fire and gas leakage detection system that adopts both technologies and utilizes multiple efficient sensors (temperature, humidity, smoke, flame, CO, LPG, and CNG). The proposed Distributed Edge Computing Internet of Things (DECIoT) platform that is integrated with the SB112 is based on the EdgeX foundry open-source framework, which is considered a highly flexible and scalable edge computing framework. Moreover, the technologies used in the implementation are state-of-the-art, such as the Go programming language, microservices, and 3D printed cases.

The SB112 system is highly scalable; hence, it is able to cover large areas and handle a variety of different sensor types and measurements, which makes it capable of detecting various emergency conditions. Moreover, it considers several actors throughout an ongoing emergency situation, such as sensors, IoT devices (sensor nodes and edge gateway), city au-

thority operators that receive and manage crucial information from the smart city platform, and a public safety answering point (PSAP) along with its corresponding operator.

To ensure that the DECIoT platform can support real-time streaming of data, the system latency was measured through an extended experiment. The results indicated an average latency of 32 ms, which is considered to be satisfactory for emergency response applications. To verify the utility and functionality of the SB112 system, an end-to-end representative physical experiment was successfully carried out from an applied perspective in order to identify a fire alert from sensor node 1 (smoke and CO). In this study, Apache Kafka was considered as the middleware of a smart city platform. The effective integration of the DECIoT to the SB112 system was verified by sharing information with Apache Kafka. Furthermore, the SB112 system effectively published raw measurements identifying an emergency situation associated with detection of a fire in real time. Once the fire was detected, an NG112 emergency call was automatically placed to a PSAP in the form of a Common Alerting Protocol (CAP) message. The effective end-to-end experiment highlighted the potential of the SB112 system for use by decision-makers or city authorities in real-world scenarios.

Similar extended experiments were performed by physically triggering sensor node 2 (temperature and flame) and sensor node 3 (CNG and LPG) using multiple rules, either individually or in combination, in the context of a fire and/or gas leakage case event. The results were equally satisfactory. Hence, taking into account the stability and efficiency of the SB112 system, its installation into a building of interest for continuous operation and monitoring was carried out. For installation and protection purposes, relevant packaging was installed to cover each sensor node.

Future work is intended to extend the SB112 system with several additional sensor nodes as well as edge gateways employing additional sensors (e.g., camera sensors with video streaming, CO₂), and additional dynamic features and complex schemes to provide a more general model associated with fire and gas leak detection may be considered as well.

Author Contributions: Conceptualization, E.M., K.P., A.D., L.K., E.O. and M.K.; methodology, E.M., K.P., A.D. and L.K.; software, K.P. and A.D.; validation, E.M., K.P., A.D., L.K., M.K. and G.H.; writing—review and editing, E.M., K.P., A.D., L.K., E.O. and M.K.; supervision, E.M., L.K., E.O., M.K. and A.A. All authors have read and agreed to the published version of the manuscript.

Funding: This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 883522.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data sharing is not applicable to this article.

Acknowledgments: This work is a part of the S4AllCities project. This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 883522. Content reflects only the authors’ views; the Research Executive Agency (REA)/European Commission is not responsible for any use that may be made of the information it contains.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zanella, A.; Bui, N.; Castellani, A.; Vangelista, L.; Zorzi, M. Internet of Things for Smart Cities. *IEEE Internet Things J.* **2014**, *1*, 22–32. [CrossRef]
2. New Waves of IoT Technologies Research—Transcending Intelligence and Senses at the Edge to Create Multi Experience Environments. SINTEF. Available online: <https://www.sintef.no/publikasjoner/publikasjon/1896632/> (accessed on 11 January 2022).
3. Mobin, M.I.; Abid-Ar-Rafi, M.; Islam, M.N.; Hasan, M.R. An Intelligent Fire Detection and Mitigation System Safe from Fire (SFF). *Int. J. Comput. Appl.* **2016**, *133*, 1–7.
4. Bhoi, S.K.; Panda, S.K.; Padhi, B.N.; Swain, M.K.; Hembram, B.; Mishra, D.; Mallick, C.; Singh, M.; Khilar, P.M. FireDS-IoT: A Fire Detection System for Smart Home Based on IoT Data Analytics. In Proceedings of the 2018 International Conference on Information Technology (ICIT), Bhubaneswar, India, 19–21 December 2018; pp. 161–165. [CrossRef]

5. Li, Z.; Liu, J.; Guo, C.; Li, S.; Li, L.; Xu, L. Design of Smoke Detection and Alarm System Based on LoRa. In Proceedings of the 2020 Chinese Automation Congress (CAC), Shanghai, China, 6–8 November 2020; pp. 5086–5090. [\[CrossRef\]](#)
6. Jang, H.-Y.; Hwang, C.-H. Obscuration Threshold Database Construction of Smoke Detectors for Various Combustibles. *Sensors* **2020**, *20*, 6272. [\[CrossRef\]](#) [\[PubMed\]](#)
7. Birajdar, G.S.; Singh, R.; Gehlot, A.; Thakur, A. Development in building fire detection and evacuation system—a comprehensive review. *Int. J. Electr. Comput. Eng. (IJECE)* **2020**, *10*, 6644–6654. [\[CrossRef\]](#)
8. Jadon, A.; Omama, M.; Varshney, A.; Ansari, M.S.; Sharma, R. FireNet: A Specialized Lightweight Fire & Smoke Detection Model for Real-Time IoT Applications. *arXiv* **2019**, arXiv:1905.11922.
9. Rodriguez-Sanchez, M.; Fernández-Jiménez, L.; Jiménez, A.; Vaquero, J.; Borromeo, S.; Lázaro-Galilea, J. HelpResponder—System for the Security of First Responder Interventions. *Sensors* **2021**, *21*, 2614. [\[CrossRef\]](#)
10. Wu, L.; Chen, L.; Hao, X. Multi-Sensor Data Fusion Algorithm for Indoor Fire Early Warning Based on BP Neural Network. *Information* **2021**, *12*, 59. [\[CrossRef\]](#)
11. Rehman, A.; Qureshi, M.A.; Ali, T.; Irfan, M.; Abdullah, S.; Yasin, S.; Draz, U.; Glowacz, A.; Nowakowski, G.; Alghamdi, A.; et al. Smart Fire Detection and Deterrent System for Human Savior by Using Internet of Things (IoT). *Energies* **2021**, *14*, 5500. [\[CrossRef\]](#)
12. Khan, M.M. Sensor-Based Gas Leakage Detector System. *Eng. Proc.* **2020**, *2*, 28. [\[CrossRef\]](#)
13. Swetha, S.V.; Swetha, G.N. LPG/CNG Gas Leakage Detection and Prevention using IoT. *Int. Res. J. Eng. Technol.* **2021**, *8*, 4.
14. Nembhnan, J.; Yadav, A.; Biswas, S. Developing an intelligent fire alarming, monitoring and rescuing system using UAV. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2020**, *43*, 409–414. [\[CrossRef\]](#)
15. Fonollosa, J.; Solórzano, A.; Marco, S. Chemical Sensor Systems and Associated Algorithms for Fire Detection: A Review. *Sensors* **2018**, *18*, 553. [\[CrossRef\]](#)
16. Avgeris, M.; Spatharakis, D.; Dechouniotis, D.; Kalatzis, N.; Roussaki, I.; Papavassiliou, S. Where There Is Fire There Is SMOKE: A Scalable Edge Computing Framework for Early Fire Detection. *Sensors* **2019**, *19*, 639. [\[CrossRef\]](#)
17. Ilayarani, P.P.; Dominic, M.M. Smart Firefighting System for Smart Cities Adopting Fog/Edge Computing. *Int. J. Innov. Technol. Explor. Eng.* **2019**, *8*, 1–7.
18. Mahgoub, A.; Tarrad, N.; Elsherif, R.; Ismail, L.; Al-Ali, A. Fire Alarm System for Smart Cities Using Edge Computing. In Proceedings of the 2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT), Doha, Qatar, 2–5 February 2020; pp. 597–602. [\[CrossRef\]](#)
19. AIOTI. The Alliance for the Internet of Things Innovation. Available online: <https://aioti.eu/> (accessed on 13 January 2022).
20. imthefastlane. The Value of Spatial Information for Emergency Services. EENA. Available online: <https://eena.org/wp-content/uploads/GIS-The-value-of-spatial-information-for-emergency-services.pdf> (accessed on 28 January 2022).
21. Sdongos, E.; Bolovinou, A.; Tsogas, M.; Amditis, A.; Guerra, B.; Manso, M. Next Generation Automated Emergency Calls—Specifying Next Generation Ecall Sensor-Enabled Emergency Services. In Proceedings of the 2017 14th IEEE Annual Consumer Communications & Networking Conference (CCNC), Las Vegas, NV, USA, 8–11 January 2017; pp. 1–6. [\[CrossRef\]](#)
22. Sedlar, U.; Winterbottom, J.; Tavcar, B.; Sterle, J.; Cijan, J.; Volk, M. Next Generation Emergency Services Based on the Pan-European Mobile Emergency Application (PEMEA) Protocol: Leveraging Mobile Positioning and Context Information. *Wirel. Commun. Mob. Comput.* **2019**, *2019*, e1408784. [\[CrossRef\]](#)
23. Haghi, M.; Barakat, R.; Spicher, N.; Heinrich, C.; Jageniak, J.; Öktem, G.; Krips, M.; Wang, J.; Hackel, S.; Deserno, T. Automatic Information Exchange in the Early Rescue Chain Using the International Standard Accident Number (ISAN). *Healthcare* **2021**, *9*, 996. [\[CrossRef\]](#) [\[PubMed\]](#)
24. The Go Programming Language. Available online: <https://go.dev/> (accessed on 10 February 2022).
25. Apache Kafka. Available online: <https://kafka.apache.org/> (accessed on 26 January 2022).
26. Nasiri, H.; Nasehi, S.; Goudarzi, M. A Survey of Distributed Stream Processing Systems for Smart City Data Analytics. In *SCIOT 18: Proceedings of the International Conference on Smart Cities and Internet of Things*; Association for Computing Machinery: New York, NY, USA, 2018; pp. 1–7. [\[CrossRef\]](#)
27. Xu, R.; Jin, W.; Kim, D. Enhanced Service Framework Based on Microservice Management and Client Support Provider for Efficient User Experiment in Edge Computing Environment. *IEEE Access* **2021**, *9*, 110683–110694. [\[CrossRef\]](#)
28. Eclipse Mosquitto. 2018. Available online: <https://mosquitto.org/> (accessed on 26 January 2022).
29. Foundation, T.L. Welcome. Available online: <https://www.edgexfoundry.org> (accessed on 13 January 2022).
30. Cao, K.; Liu, Y.; Meng, G.; Sun, Q. An Overview on Edge Computing Research. *IEEE Access* **2020**, *8*, 85714–85728. [\[CrossRef\]](#)
31. Jin, S.; Sun, B.; Zhou, Y.; Han, H.; Li, Q.; Xu, C.; Jin, X. Video Sensor Security System in IoT Based on Edge Computing. In Proceedings of the 2020 International Conference on Wireless Communications and Signal Processing (WCSP), Nanjing, China, 21–23 October 2020; pp. 176–181. [\[CrossRef\]](#)
32. Villali, V.; Bijivemula, S.; Narayanan, S.L.; Prathusha, T.M.V.; Sri, M.S.K.; Khan, A. Open-source Solutions for Edge Computing. In Proceedings of the 2021 2nd International Conference on Smart Electronics and Communication (ICOSEC), Trichy, India, 7–9 October 2021; pp. 1185–1193. [\[CrossRef\]](#)
33. eKuiper-LF Edge. Available online: <https://www.lfedge.org/projects/ekuiper/> (accessed on 16 March 2022).
34. Mane, S.A.; Nadargi, D.Y.; Nadargi, J.D.; Aldossary, O.M.; Tamboli, M.S.; Dhulap, V.P. Design, Development and Validation of a Portable Gas Sensor Module: A Facile Approach for Monitoring Greenhouse Gases. *Coatings* **2020**, *10*, 1148. [\[CrossRef\]](#)

-
35. Fakra, D.A.H.; Andriatoavina, D.A.S.; Razafindralambo, N.A.M.N.; Amarillis, K.A.; Andriamampianina, J.M.M. A simple and low-cost integrative sensor system for methane and hydrogen measurement. *Sens. Int.* **2020**, *1*, 100032. [[CrossRef](#)]
 36. Smart Spaces Safety And Security. Greece. S4AllCities. Available online: <https://www.s4allcities.eu> (accessed on 9 February 2022).