SMART BUILDING MANAGER SOFTWARE IN CLOUD

Florin LACATUSU[1], Ioan DAMIAN[2], Anca Daniela IONITA[3], Marian LACATUSU[4]

Cloud applications have a significant presence these days and they are influencing our daily lives by providing accessible services that can be consumed from everywhere through multiple device types. This paper proposes a solution regarding management tasks of a smart building, using in-house Cloud-native software. Apart from providing a way of centralizing the data gathered from a sensor network, usually employed in a smart building, this research is focused on the configuration and monitoring of the smart building edge network in the context of generating alerts. The configuration files for edge nodes provide the necessary information in order to connect to the Cloud system.

Keywords: Smart buildings, Cloud-native applications, Containers, Monitoring systems

1. Introduction

Cloud implementations are more prevalent in our current social environment, hence many applications are developed or migrated to work in a public Cloud with the scope of providing a simple and accessible solution for its users. The importance of developing a Cloud-native application is validated in many fields of work and it was confirmed to be an improvement for multiple implementations with different objectives [1]. Another important topic that is more and more both present in the scientific literature and in many applications is related to smart buildings. The intelligence integrated at the construction level provides many benefits and aims to improve the life of the building occupants [2]. Internet of Things is the most important trigger for smart building development, concerning how all the smart devices integrate and communicate with each other to satisfy different needs, such as achieving energy efficiency, monitoring the environment, or providing different offerings for its tenants [3]. The

[1] PhD Student, Automation and Industrial Informatics Department, University POLITEHNICA of Bucharest, Romania, e-mail: florinlacatusu94@gmail.com
[2] PhD Student, Automation and Industrial Informatics Department, University POLITEHNICA of Bucharest, Romania, e-mail: ioandamian94@gmail.com
[3] Prof., Automation and Industrial Informatics Department, University POLITEHNICA of Bucharest, Romania, e-mail: anca.ionita@upb.ro
[4] PhD Student, Automation and Industrial Informatics Department, University POLITEHNICA of Bucharest, Romania, e-mail: marianlacatusu94@gmail.com

interconnection between these two topics - smart buildings leveraged by IoT devices, and Cloud computing for sustaining the management of software architecture components - has gained a lot of interest from the scientific community; it mostly benefits from the elastic computing and the ease of access for the building operators [4].

In this paper, we investigate the topic of developing a smart building management software that is located on a public Cloud, with the scope of gathering local data and also generating configuration files for the physical edge node network.

The novelty of our research is related to the proposal of a new system that leverages the advantages of Cloud technologies and the latest development methods to offer a platform that is used to both monitor and configure building edge nodes with the scope of detecting different alerts. Also, our solution is developed to be integrated into a broader system composed of multiple smart buildings with the scope of notifying each other of possible alerts.

Our objective was to propose a Cloud solution for a smart building monitoring and configuration system that provides the means of managing this kind of construction within a Cloud-native application, which can be accessed from any location using many device types, such as smartphones, tablets, laptops, etc. The aim is to improve the workflow and access of management systems within a smart building, by providing a flexible architecture to monitor and configure the local building edge network. The system was designed as a Cloud-native application, using container model deployment within a Kubernetes cluster. This brings the advantage of high availability and self-healing of containers, a necessary feature for a building monitoring system that can send alerts in case of emergency [5]. Another objective considered for this kind of system was to provide a way of communication with other smart buildings, with the scope of exchanging alerts in case of disaster. Such an integration contributes to the development of always connected buildings in the context of a smart city, as an application of the broader concept of communication between smart devices on a larger scale [6]. Regarding the configuration capabilities provided by our system, we propose a method of generating configuration files based on user preferences; this information will be further used within the setup of the edge network nodes, in order to automate their configuration process and the communication with the Cloud system. In consequence, Section 2 contains a study on the related work within the domains of Cloud computing, sensor networks, and smart buildings. Section 3 presents the architecture of the system, the application functionalities, and the results achieved with the current implementation, discussing then, in Section 4, what are the main aspects that such a solution may solve in the context of smart building configuration and monitoring.

Related work

The Cloud implementation benefits, provided by the features that this kind of deployment offers, can be used for multiple application types [1]. In this paper, we wish to underline the importance of deploying building monitoring and configuration software in the Cloud, so it can be easily and quickly accessed by authorized personnel, using multiple device types and sizes.

An objective of our solution is the communication between a sensor network located in a building and a Cloud monitoring application. The topic of the connection between a wireless sensor network and the Cloud is detailed in [7].

The implementation of a warning system represents another subject of interest in the context of the building monitoring system. After the data is collected from the sensor network, it is processed by the Cloud application, and then notifications are sent to the authorized personnel. An example of a warning system based on data fetched from an edge network is further detailed by Syafrudin et al. [8].

Cloud infrastructure is essential when designing an application and should be adapted for its needs and requirements. In our case, the application must be highly available and accessible, providing notification and information, through the "Smart Building Manager" portal for the authorized personnel. To satisfy these requirements we use a Kubernetes service in a public Cloud. This container implementation satisfies the above-mentioned requirements along with the microservices architecture based on the capability of services to be loosely coupled. Further details about Kubernetes features and various implementations are demonstrated by Dewi et al. in [9].

The development of a web dashboard to present the data collected from sensors, along with the information about the edge network, represents an important point for a building monitoring system. Thus, the aim is to provide a fast and responsive way of configuring and monitoring a smart building through a modern web application that can be queried from multiple device types and is intuitive for the users [10].

For a smart building software, it is also essential to provide integration between buildings that employ the same system. Hence, our solution targeted the interconnection between multiple smart buildings by providing the means necessary to exchange alert data between them. In the scientific literature, multiple smart buildings interconnected could be part of a smart city, as presented by Eremia et al. [11].

Regarding the security of the Smart Building Manager monitoring system, an encrypted connection is established between its components. Also, the backend application can be accessed only by authorized entities via a token. One interesting aspect that could improve the Smart Building Manager security is to implement an intrusion detection system such as the one in [18] which helps identify possible attacks or exploits on the IoT nodes that are employed at the building level.

2. Smart Building Manager

Smart Building Manager is our proposed solution for building monitoring and configuration. It is based on a container implementation, developed as a cloud-native application that can be deployed on any Cloud that provides a Kubernetes service. The main advantage of this kind of application is the separation between the monitored location and the actual software that analyzes and stores the data. In consequence, the system is easily accessible by the authorized personnel, hence is deployed in a public Cloud and it is not relying on the local building systems to operate. The data is collected from an edge network composed of edge nodes and sensors. The edge node is represented by a Linux-based low-power device (such as Raspberry Pi) that collects the data from sensors and sends it to the monitoring service located in the Cloud. An earlier version of this system was presented in [12]. Compared to this early representation our system evolved from a university-only approach to a fully cloud-native solution dedicated to smart buildings. Also, besides its alerting capabilities, underlined in the previous iteration, it also provides the necessary tools to configure the physical edge network by offering the possibility to generate configuration files used in edge nodes setup. Another capability that was introduced in the version presented here is the ability to send notification messages to other smart buildings that employ the same system. From the user experience point of view, the web portal and API backend were fully re-created from scratch, to provide a user-friendly interface that can be accessed from devices with multiple screen sizes and a well-documented API that can fuel further integrations with other services in the future.

2.1 Architecture

The architecture for the “Smart Building Manager” solution is presented in Fig 1.. It is composed of two main sections: The public Cloud Kubernetes deployed services that compose Smart Building Manager software and the building edge network.

Building Edge Network

In the architecture diagram (Fig 1.) we define the building edge network as being composed of multiple components with different roles. The edge nodes are represented by low-powered computing devices (such as a Raspberry Pi) that run a Linux-based operating system. These devices connect wirelessly to different sensors, to collect environmental data from them. On the edge nodes, a software analyzes the values that come from the sensors. If critical readings are detected, the alarm node will trigger a sound signal. Another component of the edge network is represented by the "virtual sensors" [13]. We refer to this term in our architecture to define building occupants' smartphones that are used to send reports in case of emergency.

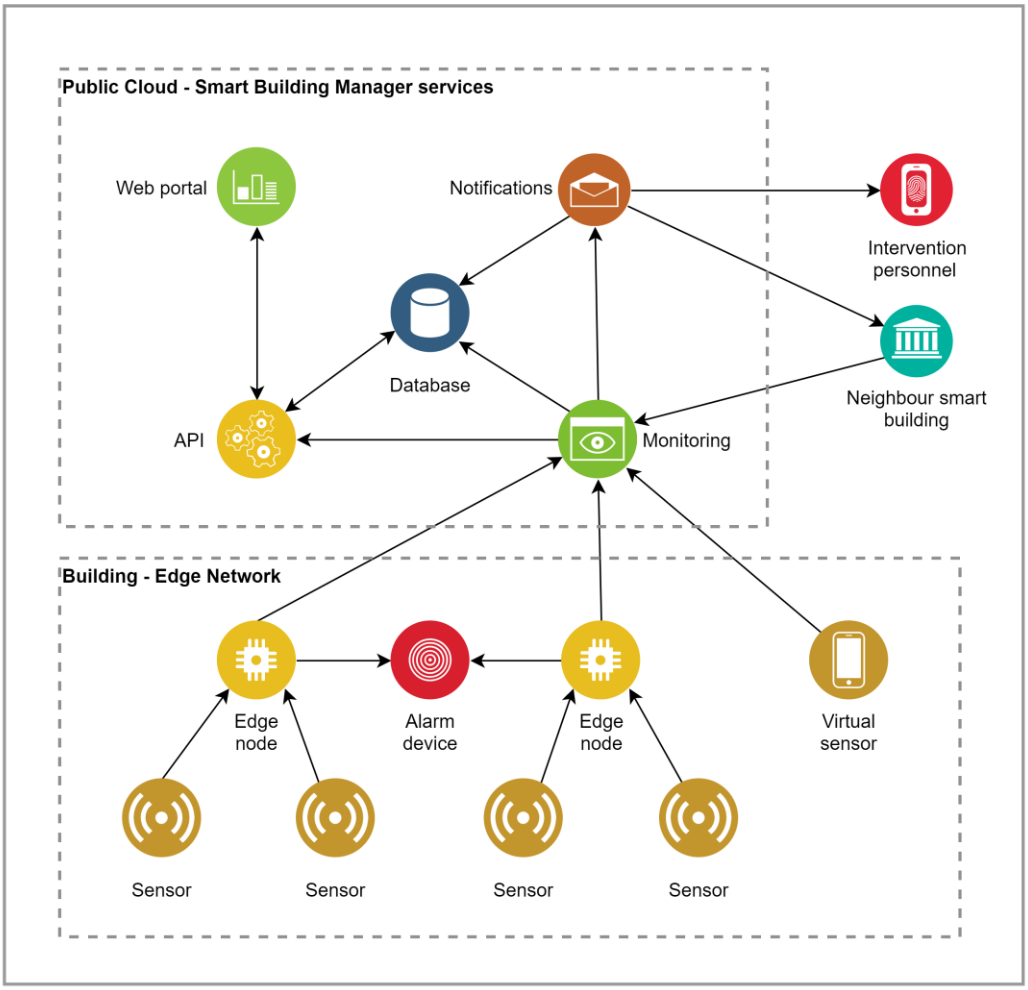

Fig 1. Smart Building Manager architecture

Cloud components

The Cloud layer is represented by the Smart Building Manager software components deployed on a Kubernetes cluster located in a Public Cloud, such as Amazon Web Services, Microsoft Azure, or IBM Cloud. The software architecture is represented by loosely coupled services implemented on Kubernetes pods [14]. There is a pod for each of the services presented in Fig. 1. Regarding the data flow between the components, two scenarios can be described, which are representative of the two main functionalities that this application offers: (i) building data gathering and alerting, (ii) edge nodes configuration.

For *Scenario 1*, the data is collected from the edge nodes and virtual sensors by the monitoring service. This component verifies if the data received from both sources is critical or not. If critical data is detected, the notification service will generate an alert that will be sent to the intervention personnel and to registered neighbor buildings that employ the same system. These alerts and critical values are logged to a database that can be queried by other services, such as Smart Building Manager API. The service provides a dedicated Swagger page [15] that describes all the methods implemented within this component and supports REST calls that can be executed by other applications. The Smart Building Manager web portal is used to display the critical data, collected by the monitoring component, through a call on the dedicated API service. Another path, also related to *Scenario 1*, is represented by the notifications sent by neighbor smart buildings that are also collected by the Monitoring service (Fig 1.). In this case, the message will be directly sent to authorized personnel smartphones, to provide the help needed.

For *Scenario 2* the concern is on the edge nodes configuration, which is an important feature of the Smart Building Manager. The objective that we want to achieve with this feature is the preparation of the Cloud system to accept connections from the edge nodes, by setting up the necessary database configurations. The flow for this scenario is as follows. An administrator connects to the Smart Building Manager web portal. Here, there is an option to create one or multiple buildings to manage. After a building is created, the next step is to add floors to the building. For each of the building floors, edge nodes must be created. The last step is to configure sensors for each node. All these virtual components are persisted in a database that is now prepared to be filled by the data that come from the physical edge nodes. This kind of computing device located in the building will be properly set up by using JSON configuration files generated by the Smart Building Manager, based on the user-selected settings. This config file will contain all the information needed, such as Cloud monitoring service API endpoints, database ids, sensor critical values thresholds, sensor information. With this configuration, we provide an automated way to set up edge nodes to integrate with the Cloud monitoring system, by preparing all the necessary aspects related to the database setup and physical computing devices.

Regarding the technologies used to develop the Cloud services, we chose Node.js for the Monitoring, Notifications, and API services. For the Web frontend, Angular was used for the implementation, along with the Angular

Material library for frontend components and design. The database that is used to persist all the configurations and collected data is MySQL. Each of the above-mentioned services are running on their dedicated pods on a Kubernetes cluster.

2.2 Web Frontend

The Smart Building Manager portal is an Angular-based frontend that executes calls on the Node.js API backend service, to obtain information about the collected data and to edit the configurations related to the edge network. The application is developed to be accessible from multiple device types and sizes. Therefore, it is a responsive web application that can be accessed from everywhere, since it is deployed in a public Cloud, and from any device that supports a Web browser. The advantages that come from this type of implementation are very useful for a building manager because the administration can query the data and execute administrative operations from any location where the Internet is available.

For the authentication to this application, the user can log in through a username and password. After the user is authenticated, a token is provided, which is used to access every page of the application. From the authorization point of view, there are two roles available for the application: regular user and application administrator. The difference between the two roles is that the latter can perform user management for the application.

After a user is provided access to the application, the first thing that one is advised to do is to select or create the building that will be managed. The concept of building in Smart Building Manager is very similar to the concept of “workspace” or “project” in popular Cloud applications. Therefore, every change that is performed on a building is done only for that building and not for the other buildings that are owned by the user.

In the configuration process, the step of adding edge nodes to the system is done after the building is selected and at least one floor is defined. The building context is always shown in the toolbar to provide a clearer way for the user to view the building for which changes are done. New nodes can be added for each of the floors defined. In the dedicated menu, each node is represented by a box that shows the node name and the room where it is located. This box design, used for components display, is adopted for every other part of the system and it is very convenient for responsive applications, especially for the case when it is accessed from smaller mobile screens. All the components dedicated menus provide the facility to create, read, update, or delete system components (Fig.2).

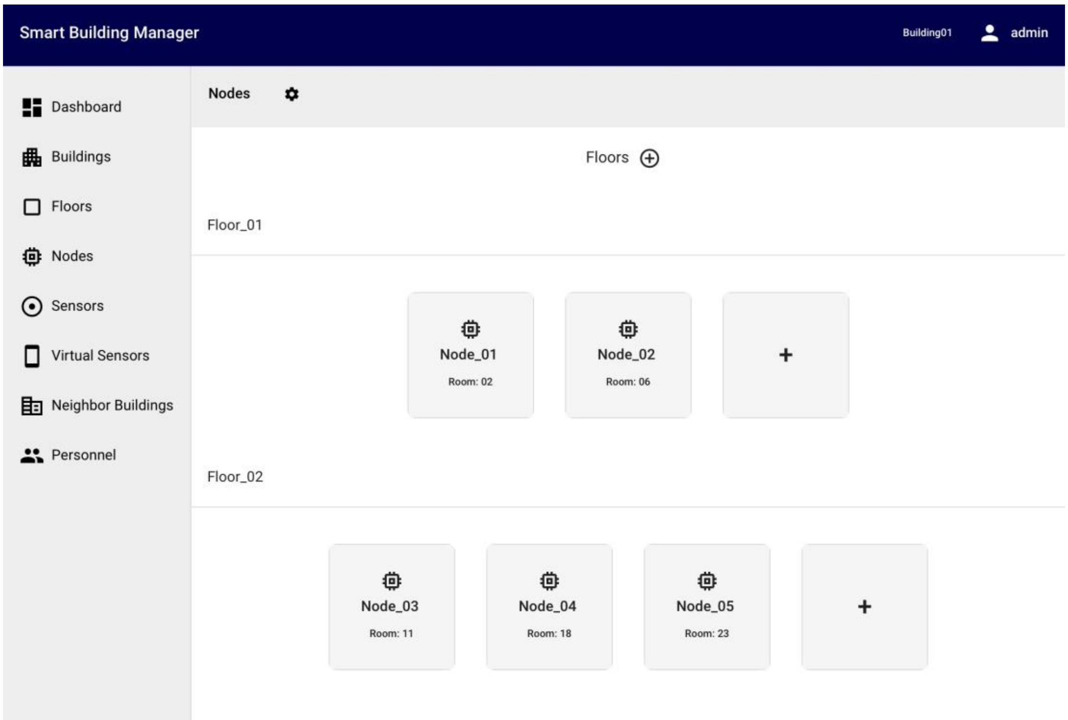

Fig 2. Smart Building Manager web view

The step that follows the node configuration is represented by the sensor setup. A new sensor is attached to one node. With this sensor configuration, we provide a way of defining the sensors, at the database level, that will be wirelessly connected to the node and will send environmental data to it. Practically, with the Smart Building Manager, one prepares the monitoring system to receive the collected data, by defining the edge network configuration, at the database level, with the facility of exporting the configuration files to be used within the setup of each physical node installed at the building level. There is also a dedicated menu for configuration and monitoring purposes, for components such as Buildings, Floors, Nodes, Sensors, Virtual Sensors, Neighbor Buildings, and Personnel. Regarding neighbor buildings, in the menu related to this topic, one can add one or multiple neighbor buildings to send alerts to it and also provides the offering to see other notifications that came from neighbor buildings that employ the same Smar Building Manager system. The *Virtual sensors* section shows the reports that came from the building occupants. For the *Personnel* section, there is a dedicated menu to add new building staff that will be notified in case the system will detect a possible alert. Regarding the *Dashboard* section, this part provides dedicated cards that show information about all the building components, such as the number of configured floors, nodes, sensors, as well as sections related to alerts from sensors, building occupants, and neighbor buildings.

With this Web application, the Smart Building Manager provides a way of executing management tasks, such as edge component configuration and monitoring, with the facility of showing information that can be accessed from any device and any location that facilitates a connection to the Internet.

2.3 Evaluation

Regarding the evaluation topic, we are defining three criteria for comparing our solution with other implementations from the scientific literature: monitoring capabilities, monitoring application location, and ease of configuration. For the first aspect, compared to solutions described in [16] and [17] our system is focused more on alert detection, also providing the possibility of informing neighbor smart buildings of a possible critical event. This offering is essential during an emergency where neighbors could provide vital help during such an event given their very near location. One example could be a large building complex such as a university campus where all the smart buildings that are in its perimeter are alerted when a critical event happens. In comparison to the solution described in [17], our monitoring application is located in the Cloud. This condition is very important considering our smart building monitoring system is focused on detecting alerts, therefore having the monitoring application implemented in Cloud is a huge benefit since this system will not be affected in case of emergency. This aspect could also benefit other systems concerning costs that are cheaper in the Cloud compared to the acquisition and maintenance of a local data center. Being located in the Cloud, there are some performance questions regarding the possible delays that may be caused by the distance between the local nodes and the Cloud system. Therefore, we tested the response time of our Monitoring API by replicating requests from twenty edge nodes that mimic real-world alerts. In Fig. 3, we can see that our system provides very good performance, obtaining an under 440 ms maximum response time from Smart Building Manager API using the default Kubernetes cluster provided by IBM Cloud. This result confirms that our solution is capable of leveraging Cloud technologies and at the same time providing strong performance, being capable to process the alerts that come from the monitored building.

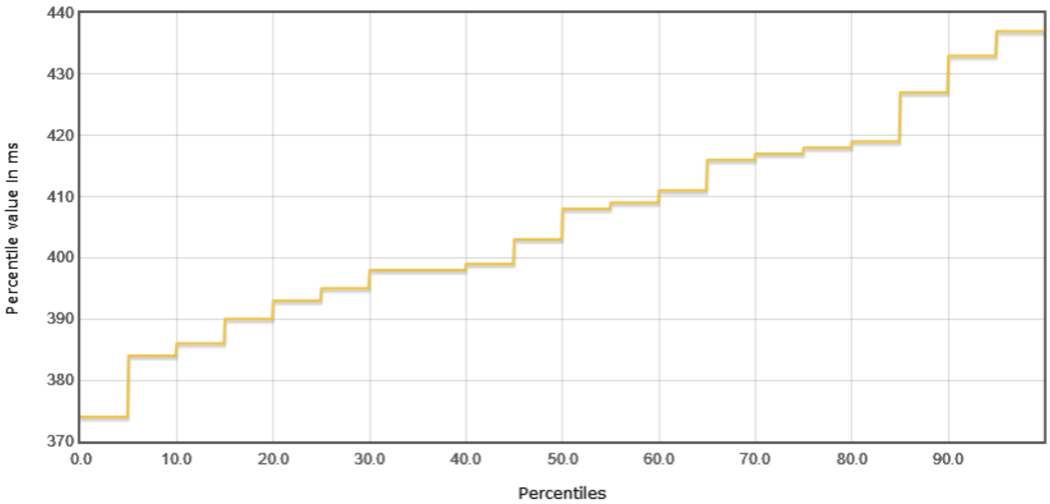

Fig. 3 Smart Building Manager simulated requests

The last comparison point is related to the ease of configuration of the building monitoring system. Smart Building Manager software is leveraging container technologies that provide great portability and deployment ease. After the monitoring software is deployed, the building edge nodes can be configured from this portal by generating JSON configuration files that will serve in the edge nodes setup. As a conclusion to this evaluation, our smart building monitoring solution provides huge benefits in regards to the detection of critical events and system configuration ease. Yet, other solutions such as the one in [16] and [17] also tackle the energy efficiency topic that is very important for a smart building management system. As a future development plan, we also want to integrate this efficiency topic in our offering to provide a complete building management system application offered as a Cloud solution.

Discussion and conclusion

Regarding the verification of our application capabilities and functions, we deployed Smart Building Manager on a Kubernetes service, and the IBM Cloud Kubernetes cluster was used for testing purposes. Yet, this application can be installed on any Cloud Computing provider that hosts a Kubernetes service. Being available as a Cloud service, such a building manager helps users to access the offerings provided by it from any location they see fit. From the application monitoring and logging capabilities points of view, any container monitoring platform can be used to monitor this application. One example of a platform that is usually employed for monitoring Kubernetes pods is ELK (Elasticsearch, Logstash, Kibana). Therefore, all the logs and metrics will be collected by this platform and the system administrators will easily inspect all these data regarding the application function.

The paper proposed an architecture of Smart Building Manager that can be deployed to configure and monitor different buildings, with the facility to provide an automated way for the physical edge network set up, along with the integrations with different neighbor facilities. The most evident benefit compared with other monitoring solutions consists in the deployment flexibility, as it can be deployed on any Cloud that supports a Kubernetes service; it thus ensures the separation of the building management software from the actual monitored building environment. Therefore, such a service will remain highly available to monitor data and send alerts based on the configured thresholds, being accessible at any time by the authorized personnel to query the collected data. Whereas the current development and tests are realized for a single building, future work will consider the communication between buildings and will verify what are the challenges when scaling up.

R E F E R E N C E S

[1]. *Marian Lacatusu, Anca Daniela Ionita*, Metamodeling Environment in Cloud, U.P.B. Sci. Bull., Series C, **Vol. 82**, Iss. 3, 2020, pp 27-34

[2]. *Froufe, M.M.; Chinelli, C.K.; Guedes, A.L.A.; Haddad, A.N.; Hammad, A.W.A.; Soares, C.A.P*. Smart Buildings: Systems and Drivers. Buildings 2020, *10*, 153.

[3]. *Sittón-Candanedo, I.; Alonso, R.S.; García, Ó.; Muñoz, L.; Rodríguez-González, S.* Edge Computing, IoT and Social Computing in Smart Energy Scenarios. Sensors 2019, *19*, 3353.

[4]. *Hang, L.; Jin, W.; Yoon, H.; Hong, Y.G.; Kim, D.H.* Design and Implementation of a Sensor-Cloud Platform for Physical Sensor Management on CoT Environments. Electronics 2018, *7*, 140

[5]. *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.

[6]. *Tcholtchev, N.; Schieferdecker, I.* Sustainable and Reliable Information and Communication Technology for Resilient Smart Cities. Smart Cities 2021, **4,** pp 156-176.

[7]. *George Manea, Sorin Popa*, Integration of Sensor Networks in Cloud Computing, U.P.B. Sci. Bull., Series C, **Vol. 78**, Iss. 2, 2016

[8]. *Syafrudin, M.; Fitriyani, N.L.; Alfian, G.; Rhee, J.* An Affordable Fast Early Warning System for Edge Computing in Assembly Line. Appl. Sci. 2019, *9*, 84.

[9]. *L. P. Dewi, A. Noertjahyana, H. N. Palit and K. Yedutun,* "Server Scalability Using Kubernetes," 2019 4th Technology Innovation Management and Engineering Science International Conference (TIMES-iCON), Bangkok, Thailand, 2019, pp. 1-4

[10]. *Pauwels, Koen & Ambler, Tim & Clark, Bruce & LaPointe, Pat & Reibstein, David & Skiera, Bernd & Wierenga, Berend & Wiesel, Thorsten*. (2009). Dashboards as a Service: Why, What, How, and What Research Is Needed?. Journal of Service Research. 12. 175-189. 10.1177/1094670509344213.

[11]. *Mircea Eremia, Lucian Toma, Mihai Sanduleac,* The Smart City Concept in the 21st Century, Procedia Engineering, **Volume 181**, 2017, Pages 12-19, ISSN 1877-7058

[12]. *Florin Lăcătuşu, Anca-Daniela Ioniță*, Architecture for Monitoring Risk Situations in a University Environment Rev. Roum. Sci. Techn.– Électrotechnique et ÉnergétiqueVol. 65, 3–4, pp. 259–263, Bucarest, 2020

[13]. *Armando, Ngombo; Rodrigues, André; Pereira, Vasco; Sá Silva, Jorge; Boavida, Fernando*. "An Outlook on Physical and Virtual Sensors for a Socially Interactive Internet" Sensors 18, no. 8: 2578, 2018.

[14]. Kubernetes Concepts - Pods: https://kubernetes.io/docs/concepts/workloads/pods/, Accessed October 5 2020.

[15]. API Documentation - Swagger: https://swagger.io/, Accessed October 10 2020.

[16]. *Shoumi Sen and Simon Leblond* - Leveraging Cloud-based Building Management Systems for Multi-site Facilities, Schneider Electric –White Paper 998-2095-11-05-15AR0Rev 0- https://download.schneider-electric.com/files?p_Doc_Ref=998-2095-11-05-15AR0_EN, Accessed 2021

[17]. Stavropoulos, Thanos & Tsioliaridou, Ageliki & Koutitas, George & Vrakas, Dimitris & Vlahavas, I.. (2010). System Architecture for a Smart University Building. 477-482. 10.1007/978-3-642-15825-4_64.

[18]. Sudqi Khater, B.; Abdul Wahab, A.W.B.; Idris, M.Y.I.B.; Abdulla Hussain, M.; Ahmed Ibrahim, A. A Lightweight Perceptron-Based Intrusion Detection System for Fog Computing. *Appl. Sci.* **2019**, *9*, 178. https://doi.org/10.3390/app9010178