2022 11th International Conference on Modern Circuits and Systems Technologies (MOCAST) | 978-1-6654-6717-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/MOCAST54814.2022.9837727

A Cloud Computing web-based application for Smart Farming based on microservices architecture

Vasileios Moysiadis
Dept. of Electrical and Computer Engineering
University of Western Macedonia
Kozani, Greece
bmoisiadis@uowm.gr

Konstantinos Tsakos
Intelligent Systems Laboratory
School of Electrical and Computer Engineering
Technical University of Crete
Chania, Greece
ktsakos@isc.tuc.gr

Panagiotis Sarigiannidis
Dept. of Electrical and Computer Engineering
University of Western Macedonia
Kozani, Greece
psarigiannidis@uowm.gr

Euripides G.M. Petrakis
Intelligent Systems Laboratory
School of Electrical and Computer Engineering
Technical University of Crete
Chania, Greece
petrakis@intelligence.tuc.gr

Achilles D. Boursianis
Department of Physics
Aristotle University of Thessaloniki
Thessaloniki, Greece
bachi@physics.auth.gr

Sotirios K. Goudos
Department of Physics
Aristotle University of Thessaloniki
Thessaloniki, Greece
sgoudo@physics.auth.gr

***Abstract*—The agriculture sector is envisioning a revolution of traditional farming supported by Information and Communications Technologies (ICT) and Cloud Computing is one of them. This tendency is called Smart Farming and promises to boost productivity while reducing production costs and chemical inputs. Cloud Computing aims to provide the necessary resources and the central orchestration of all devices involved in a Smart Farming scenario. To achieve high scalability, usability and performance in Cloud-based applications, we have to move from a monolithic development approach to microservices architecture using cutting edge technologies like containerisation. This paper presents a Smart Farming application based on Cloud Computing that promises to provide useful information to agronomists and farmers to support their decisions based on measurements from ground sensors and images captured from UAVs or ground cameras. Our implementation is based on microservices architecture using Docker Containers as the virtualisation technology. Each microservice runs on a different container and communicates through a RESTful API interface. The proposed architecture is highly scalable in future upgrades and promises high performance and security.**

***Index Terms*—Smart Farming, Cloud Computing, Microservices, Containerisation**

I. Introduction

Information and Communications Technologies (ICT) are involved in many different domains of modern life. One of them is Smart Farming, the new term in the agriculture sector. The integration of ICT with Smart Farming promises to bring many benefits in the following years. Increasing the productivity of the cultivation, improving the quality of the final product, decreasing the cost of production, reducing the chemical inputs, and reducing the labour effort are the most important aspects of this integration [1].

The primary ICTs used in Smart Farming are Unmanned Aerial Vehicles (UAVs), Unmanned Ground Vehicles (UGVs), Wireless Sensor Networks (WSNs), Big Data and Cloud Computing. Data captured from UAVs flying over the fields promise to provide relevant information regarding the cultivation status. This information is used to identify diseases or the stress of the cultivation or to predict yield production. Thus, they can be a valuable tool as they can provide input for Decision Support Systems (DSS) and assist agronomists and farmers with their actions on the field. UGVs can use computer vision to navigate autonomously in the cultivation and accomplish tasks like harvesting, monitoring, spraying, or weeding. Although this technology has to overcome many obstacles, shortly, UGVs will be able to operate continuously with accuracy and without human intervention, reducing labour effort.

Recent WSNs technologies promise to offer the communication channel between all involved devices, like UAVs, UGVs or sensors deployed on the field. Most Smart Farming applications require periodically taking measurements from ground sensors and transmitting them over long distances. Therefore, WSNs that promise transmission in long distances while reducing energy consumption for end nodes are of paramount importance. LoRa, SigFox and NB-IoT are three of the recent WSNs technologies that can meet such requirements [2]. Moreover, the massive amount of data produced by the enormous amount of sensors in large scale scenarios, along with data captured from UAVs and UGVs, make the role of Big Data analytics important. This analysis can extract valuable information and help agronomists and farmers to improve productivity [3].

Finally, Cloud Computing plays a critical role in orchestrating all involved devices, especially when the supported Smart Farming deployment covers large areas. Theoretically, it offers an unlimited amount of computational and storage resources

suitable to accomplish every demanding task. Particularly in Big Data analytics scenarios, these features are of paramount importance, giving Cloud Computing an indispensable role in Smart Farming [4]. In addition, high availability and accessibility are two key aspects of Cloud Computing that every farmer and agronomist would appreciate.

Although many applications already exist to support farmers and agronomists, few are based on Cloud Computing. Recent trends for web-based applications lead to Cloud Computing solutions when high availability and high computational resources are essential. Furthermore, to achieve high scalability and upgradability, Microservice Architecture (MSA) seems more suitable than Monolithic Architecture (MA) or Service Oriented Architecture (SOA), as MSA is proper to quickly develop large and complex systems [5]. More specifically, MSA constitutes a newly introduced concept that promises many advantages in software development where each microservice implements a specific service, can be deployed and upgraded independently and communicates with a simple interface with other microservices. However, such an approach needs virtualisation technology to isolate each microservice. Docker Containers is a popular virtualisation solution based on containers, offering easy deployment, no conflicts in dependencies and little need for the configuration of each container. In [6] the authors present a research prototype of an MSA aiming to overcome the limits of existing IoT platforms in the Cloud.

In this paper, we present a Cloud Computing web-based application suitable to provide real-time information to end-users. In more detail, it provides measurements from ground sensors and images captured from UAVs or ground cameras. Furthermore, the appropriate vegetation indices and machine learning algorithms intend to extract useful information. The application is intended as a Decision Support System for agronomists and farmers to help them with their decisions. The application's development is based on microservices architecture using Docker Containers as the virtualisation method.

The remaining of this paper is organised as follows: In Section II, we give a brief analysis of Microservices Architecture compared to other approaches. In Section III, we are providing information about the Smart Farming scenario and the hardware setup used for all of the involved devices in our deployment. In Section IV, we are analysing the proposed software architecture for the implementation of the web-based application. In Section V, we present the main features of the web-based application, and finally, Section VI concludes this paper.

II. Microservices Architecture

For the implementation of our Smart Farming application, we are following the Microservices Architecture (MSA), which offers some benefits compared to the Monolithic Architecture (MA), and Service Oriented Architecture (SOA) [7], [8], [9]. Since now, many of the developed applications have been built with the monolithic approach. In this case, an application is built upon different components which are interdependent. This approach brings some obstacles when an application starts to grow. More specifically, the development is easier in the initial stage, but when it starts to grow by adding new features or upgrading existing ones, it is harder to implement [5]. In addition, new developers need extra time to get familiar with the existing code before starting to be efficient. SOA solves some of these issues by introducing the idea of services, which are independent components that communicate through a common interface called Enterprise Service Bus (ESB). However, MSA moves one step beyond by breaking services into smaller parts, and the communication occurs through RESTful APIs. Many researchers argue that MSA is the evolution of SOA [10], but it seems that they have some differences [8].

In essence, MSA offers many advantages in large scale applications. Although it is more demanding on the development in the first stages, it is safer when they scale as each part of the application runs independently. Thus, we can upgrade each microservice apart from the other. Moreover, it promotes code usability, as code segmentation in smaller parts makes it easier to find which ones can be reused in different tasks. In addition, we can select the most appropriate programming language for each microservice, which results in faster development and faster response times. Finally, when more resources are needed, it is easier to scale only those microservices that need them than scaling the whole infrastructure in the case of the monolithic application. To summarise, compared to the MA approach, some advantages of MSA are scalability, upgradability, and code reuse.

III. Experimental Setup and Hardware Architecture

The experimental areas are located in the region of Western Macedonia, Greece. Five different areas with cultivation are included at the prefectures of Kozani and Grevena. The covered area is around 6.200 acres in total, with crops such as cherry trees, peach trees, apple trees, wheat, and lentils. More specifically, two areas are located in the prefecture of Kozani, namely Bravas with 1.220 acres and Gratsanis with 1230 acres. The main crop in both of these areas is peach trees. In addition, three areas are located in the prefecture of Grevena. Amygdalies has 1.350 acres containing various crops like cereals (wheat, barley, oats, rye, corn), legumes (chickpeas, beans, lentils), and cherry trees. Klimataki has 850 acres and contains various cereals (wheat, barley, oats, rye, corn) and legumes (chickpeas, beans, and lentils). Finally, the area of Itea has 1.550 acres containing cereal crops (wheat, barley, oats, rye, corn) and legumes (chickpeas, beans, lentils).

Regular flights from UAVs take place throughout the year, capturing images from all areas. We use the UAV eBee X from SenseFly for the flights, which is a fixed-wing model able for a continuous flight of up to 90 minutes, covering approximately 1.200 acres at the height of 400 feet. In addition, it can carry a range of groundbreaking cameras that makes it proper for a wide range of applications like surveying, mapping, engineering, environmental monitoring and agriculture. In our

deployment, we are using the multispectral camera Parrot Sequoia+, capable of capturing images in RGB, and in other four bands, GREEN (550nm ± 40nm), RED (660nm ± 40nm), Red Edge (735nm ± 10nm), and Near Infrared (NIR) (790nm ± 40nm). Furthermore, when the agronomists identify possible diseases of an area after examining the images from UAVs, photos from the ground are captured with the same camera for more thorough research.

We use Pix4D Studio to process captured photos before uploading them to the web-based application. More specifically, we use Pix4D Studio to concatenate all captured photos and create an orthomosaic image representing the entire flight area. Finally, this image is used in our implementation to generate the vegetation indices and apply machine learning algorithms to extract useful information for agronomists and farmers.

Furthermore, sensors have been deployed in the monitoring areas to measure parameters such as soil moisture in two different depths (50cm, 20 cm) and soil temperature (40 cm). We use the model Agriculture Pro from Libelium as the sensor nodes, equipped with a LoRa module. LoRa is a Low Power Wireless Area Network (LPWAN) capable of transmitting messages up to 20 Km in rural areas and up to 5km in urban areas, with low data rates and low energy consumption. It is also highly scalable as it can support multiple nodes per gateway, while adding new gateways to the network can significantly increase the reliability and the number of supported devices. Moreover, it is working at 868MHz in the EU, a free band to use without a license. These features make it suitable for agriculture applications [11] in which most of the deployed sensors need to transmit data in long distances and work without human intervention for a lot of months or even years.

We use Lorix One as the LoRa gateway, with a 4.15 dBi antenna, with the chipset SX1301 from Semtech integrated. It can receive and demodulate messages from eight channels and six different Spreading Factors (SF) simultaneously. We have placed gateways to the areas of interest to receive incoming packets from the deployed nodes in the cultivation. They have access to the Internet through the local network and are configured to transmit messages to The Things Network (TTN). TTN is an open-source infrastructure that provides all necessary services to LoRa end-devices. All messages received in TTN are transmitted to our web-based application on the Cloud through a RESTfull web service.

Our web-based application is hosted by a virtual machine in the Cloud, with a Linux Debian 10.7 as the operating system with 16 GB of RAM and 16 CPU cores. The virtual machine is located upon the Okeanos-Knossos infrastructure, the Cloud Service of GRNET, dedicated to the Greek Research and Academic Community. The infrastructure is scalable to add extra resources such as storage, memory or CPU cores. That makes it flexible and able to support more load when it is needed.

Figure 1 depicts the described hardware architecture.

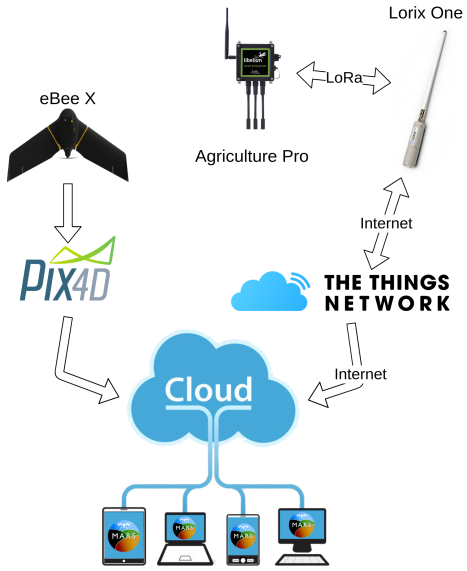

Fig. 1. Hardware Architecture

IV. Software Architecture

To implement our web-based application based on MSA, we use Docker to isolate each microservice in a different container. Docker is a promising virtualisation technology based on containers. Compared to virtual machines, it needs fewer resources as it loads only the necessary binaries and libraries in each container and not the whole operating system.

In Figure 2, we present the proposed architecture with each microservice running in a different Docker Container. The implemented microservices are the Back-End (BE), the Data Storage (DS), the Identity Manager (IDM), the User Data Storage (UDS), and the Geospatial Data Storage (GDS). In the following paragraphs, we give a brief description of each of them as well as their role in the project.

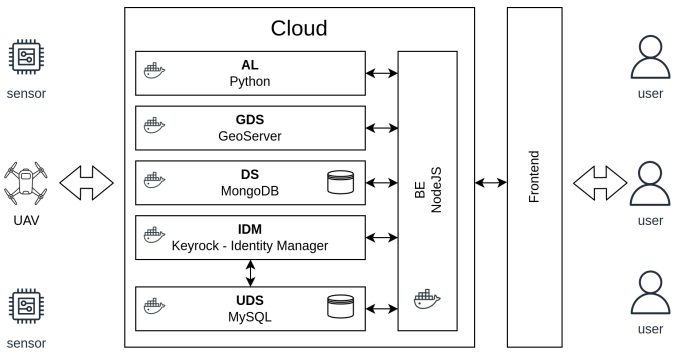

Fig. 2. Microservices architecture.

Back-End (BE): The Back-End microservice is responsible for providing a RESTful API to serve the appropriate information to end-devices or other infrastructure components, as it is responsible for interacting with all other microservices.

In more detail, it implements a RESTful API with endpoints for adding, removing or querying measurements from ground sensors. All incoming data are sent to the Data Storage microservice to store them for future usage. Furthermore, another part of BE is responsible for uploading images from UAVs flights or images from ground cameras. The incoming images are sent to Geospatial Data Storage microservice to store them for later usage. Finally, when a user tries to log in, BE is responsible for sending the request to the Identity Manager microservice to authorise him to log in.

For the implementation of the BE microservice, we are using NodeJS wrapped in a Docker Container. NodeJS is a Javascript runtime environment suitable to run scalable web applications. It is one of the most used back-end technologies supporting dynamic web-based applications and offering fast response times, with an asynchronous response if needed. It is open-source and working in multiple platforms providing numerous modules for every purpose.

Data Storage (DS): In Smart Farming applications, an enormous amount of data originates from heterogeneous sources. In our experimental setup, we have images captured from UAVs and images taken from the ground. All of these images are uploaded in the Cloud, while additional information about the area, date, and time of the flight is stored in the database. Additionally, ground sensors deployed in the field are continuously sending their measurements through the LoRa wireless network. All of these types of data are proper for document-based data stores.

We have chosen MongoDB for this service as it is one of the most suitable solutions for document-based data stores. MongoDB uses BSON (Binary JSON) to store data, an extension of JSON (JavaScript Object Notation), that uses a binary structure encoding and offers faster parse times. Also, BJSON has extended to support additional data types such as dates and binary data, which are not supported by the JSON format. JSON is an open standard for data interchange that uses a human-readable format. In our implementation, JSON is used to describe the sensors' measurements, and the data describe each of the captured images from UAVs or ground cameras.

Identity Manager (IDM): A reliable and secure service for user authentication is always crucial in Cloud Computing deployments. To avoid building a new service from scratch, we have selected to use a reliable open-source solution working out-of-the-box, including rich features. Thus, we are using FIWARE Keyrock, an identity manager offering management for users, organisations, and applications. All functions are available through a RESTful API interface that makes it simple to integrate with our BE microservice. FIWARE Keyrock needs a MySQL database to store all necessary information for user authentication. For that reason, we use MySQL as a database in the User Data Storage microservice.

User Data Storage (UDS): The purpose of the User Data Storage microservice is to store users' credentials in order to authorise them to log in and use the web-based application.This microservice communicates directly with the Identity Manager microservice in order to store all the appropriate information regarding users, organisations and applications as described before.

MySQL wrapped in a Docker container is used as a database to store users' credentials. It is an open-source Relational Database Management System (RDBMS) widely used in web-based applications.

Geospatial Data Storage (GDS): Captured images from UAVs are not suitable to be displayed directly in a web-based application. We first need to georeference them in order to display them in the correct position over a map of the GIS. Besides, captured images have potentially large sizes, making it difficult to handle through an internet connection and display them directly in a web browser. Thus, we need a method to break them into small parts, each of them corresponding to specific geographical coordination and at a particular zoom level. Web Map Tile Service (WMTS) is a convenient service that can efficiently render and serve large georeferenced images over the web.

For this microservice, we have selected to use the GeoServer, an open-source geospatial data sharing service able to handle large geospatial images or other information from heterogeneous sources. It comes with a web interface for configuration, but its functionalities are also available through a RESTful API. In addition, it supports the most popular services for geospatial data, such as the Web Map Service (WMS), Web Coverage Service (WCS), Web Feature Service (WFS) and Web Map Tile Service (WMTS).

Application Logic (AL) : This microservice includes the implementation of the machine learning algorithms and other functionalities of the application. For example, it is used to calculate several vegetation indices, which show various crop health parameters. In addition, AL is responsible for raising an alert and notifying agronomists and farmers when the parameters from the ground sensors are out of the specified bounds. Finally, machine learning algorithms are implemented in AL, giving more accurate advice about the cultivation's status. We use Python as the programming language in this microservice, and all required dependencies are installed at the build time.

Docker containers are interconnected via a virtual network set up at the initialisation stage. Besides, the communication between microservices occurs through a RESTful API. For example, when the BE microservice needs to authorise a user, it sends a request to IDM microservice through the RESTful API interface. In addition, when the BE microservice needs to retrieve the orthomosaic images from the flights of the specific area, it sends a specific request to the GDS microservice.

V. Presentation

Our implementation focuses on data aggregation from ground sensors, images captured from UAVs, or images captured from the ground. After the stage of data processing, the web-based GIS application visualize the available information to help agronomists and farmers identify possible diseases and estimate crop production. Figure 3 displays a layout of our implementation. The left panel shows the corresponding

measurements of the selected ground sensor on the map. More specifically, it shows measurements for soil moisture in two different depths (50cm, 20cm) and soil temperature in 40 cm depth. On top of the left panel, the first graph provides the soil moisture from both depths, while the second graph provides the soil temperature.

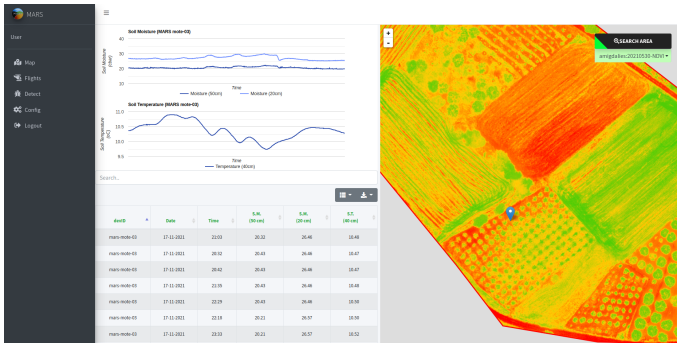

Fig. 3. Cloud application.

We can see a specific orthomosaic image taken from a UAV by selecting it from the dropdown list on the right. Images can be in RGB format or display a particular vegetation index. For the example, in Figure 3 we see an NDVI image from the specified area produced by the NIR and Red channel. More specifically, an NDVI image is a pseudocolour image, where green areas indicate healthy crops and red or yellow areas indicate the unhealthy, which could be due to stress or disease of the cultivation. Apart from NDVI, the users can observe other vegetation indices like GNDVI (Green Normalized Difference Vegetation Index), NDRE (Normalized Difference Red Edge), SAVI (Soil Adjusted Vegetation Index).

We have three different groups of users, each one with different roles. The first group includes user administrators responsible for system administration, such as system maintenance, system resources monitoring, and images uploading from UAVs or ground cameras. In addition, they have the responsibility to create or delete authorised users. The second group corresponds to agronomists, who are liable for evaluating the captured images from UAVs or ground cameras and the data from the sensors. Finally, the third group is related to the farmers who have access to the platform to observe the available information for their cultivation.

VI. Conclusion

Supporting Smart Farming with ICT technologies can leverage productivity and reduce cost production. One of these technologies is Cloud Computing and it can play a vital role to support the whole infrastructure. It is able to support the huge amount of end devices, orchestrate them and offer a plethora of computational and storage resources. In addition, the enormous amount of data needs high processing capabilities in order to extract useful information based on machine learning algorithms.

Until now, a monolithic web-based application was able to fulfil these requirements of the implementation. However, when more than a few services are required, we have to adopt an alternative approach called Microservices Architecture that can overcome the obstacles that large deployments bring.

This paper presents a web-based application for Cloud Computing built with the concept of MSA and promising high scalability, upgradability and security. Our implementation uses Docker Containers as the virtualisation technology to isolate each microservice. Moreover, communication between microservices occurs through a RESTfull interface.

The developed web-based application provides useful information to agriculturists and farmers from heterogeneous sources like images captured from UAVs or ground cameras and measurements from ground sensors. In the future, we are intending to enrich it with more features like measurements from weather stations and additional machine learning algorithms for stress or disease detection.

Acknowledgment

This research was co-funded by the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship, and Innovation, grant number T1EDK-04759.

References

[1] Vasileios Moysiadis, Panagiotis Sarigiannidis, Vasileios Vitsas, and Adel Khelifi. Smart farming in europe. *Computer Science Review*, 39:100345, 2021.

[2] Vasileios Moysiadis, Thomas Lagkas, Vasileios Argyriou, Antonios Sarigiannidis, Ioannis D Moscholios, and Panagiotis Sarigiannidis. Extending adr mechanism for lora enabled mobile end-devices. *Simulation Modelling Practice and Theory*, 113:102388, 2021.

[3] Andreas Kamilaris, Andreas Kartakoullis, and Francesc X Prenafeta-Boldú. A review on the practice of big data analysis in agriculture. *Computers and Electronics in Agriculture*, 143:23–37, 2017.

[4] Sjaak Wolfert, Lan Ge, Cor Verdouw, and Marc-Jeroen Bogaardt. Big data in smart farming–a review. *Agricultural systems*, 153:69–80, 2017.

[5] Miika Kalske, Niko Mäkitalo, and Tommi Mikkonen. Challenges when moving from monolith to microservice architecture. In *International Conference on Web Engineering*, pages 32–47. Springer, 2017.

[6] Xenofon Koundourakis and Euripides GM Petrakis. ixen: context-driven service oriented architecture for the internet of things in the cloud. *Procedia Computer Science*, 170:145–152, 2020.

[7] Abdul Razzaq. A systematic review on software architectures for iot systems and future direction to the adoption of microservices architecture. *SN Computer Science*, 1(6):1–30, 2020.

[8] Muhammad Waseem, Peng Liang, and Mojtaba Shahin. A systematic mapping study on microservices architecture in devops. *Journal of Systems and Software*, 170:110798, 2020.

[9] Shanshan Li, He Zhang, Zijia Jia, Chenxing Zhong, Cheng Zhang, Zhihao Shan, Jinfeng Shen, and Muhammad Ali Babar. Understanding and addressing quality attributes of microservices architecture: A systematic literature review. *Information and Software Technology*, page 106449, 2020.

[10] Olaf Zimmermann. Microservices tenets. *Computer Science-Research and Development*, 32(3):301–310, 2017.

[11] Badreddine Miles, El-Bay Bourennane, Samia Boucherkha, and Salim Chikhi. A study of lorawan protocol performance for iot applications in smart agriculture. *Computer Communications*, 164:148–157, 2020.