IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 6, 2020, accepted November 25, 2020, date of publication November 30, 2020,
date of current version December 11, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3041394

Enhancing Rescue Operations With Virtualized Mobile Multimedia Services in Scarce Resource Devices

ASIER ATUTXA, JASONE ASTORGA, MAIDER HUARTE, EDUARDO JACOB, (Senior Member, IEEE), AND JUANJO UNZILLA
Department of Communications Engineering, Faculty of Engineering, University of the Basque Country, 48013 Bilbao, Spain

Corresponding author: Asier Atutxa (asier.atutxa@ehu.eus)

This work was supported in part by the Spanish Ministry of Economy, Industry and Competitiveness through the State Secretariat for Research, Development and Innovation under the Adaptive Management of 5G Services to Support Critical Events in Cities (5G-City) Project under Grant TEC2016-76795-C6-5-R, and in part by the Department of Economic Development and Competitiveness of the Basque Government through the 5G4BRIS Research Project under Grant KK-2020/00031.

ABSTRACT The aim of this article is to present an architecture to support reconfigurable multimedia services for a practical emergency environment use case of rescue operations. Specifically, radio communication and video surveillance services are provided by means of a small device carried by a mobile vehicle. This solution is validated by implementing a digital mobile radio (DMR) standard radio hotspot and a video streaming server in an unmanned aerial vehicle (UAV) that conveys a device with scarce resources in order to minimize power consumption. To achieve a fast and flexible deployment of the envisioned services, a virtualization-based approach is proposed. Namely, a Kubernetes orchestrator is used to manage the life-cycle of services deployed on a small resource device, endowing the architecture with scalability and management flexibility. This article describes the executed performance tests that measured key parameters such as deployment time and recovery time after disconnection. Highly promising results were obtained, showing that the proposed architecture can be deployed in less than 4 minutes and can recover from network disconnections in less than 10 seconds. Thus, the performance, reliability and flexibility of the overall solution are demonstrated.

INDEX TERMS Application virtualization, emergency services, multimedia communication, radio communication.

I. INTRODUCTION

Recent technological developments are changing society radically, and almost every aspect of life can be improved by taking advantage of those innovations. Revolution in the multimedia services field has been extremely important, and in the past decade this field has been completely transformed. The two main factors that have caused the growth in multimedia services are increased bandwidth in network communications and new compression algorithms that can be implemented due to more powerful CPUs. Compression algorithms such as H.264 [1] can achieve a compression factor of 1000:1 easing video transmission over all kinds of networks. Multimedia services such as video streaming have become popular with video streaming service subscriptions spreading from 52% to 74% of households in the U.S. between 2015 and 2019 [2]. Audio streaming is in a similar situation with on-demand audio streaming being the most consumed music source increasing from 62% of households in 2017 to almost 80% [3]. In fact, according to some studies [4] further growth will occur in the next few years, showing the importance of this market worldwide [5].

However, this technological revolution is not limited to entertainment, and other fields have started to implement streaming and other multimedia services to enhance their processes. For search and rescue operations, the latest technologies have improved performance and efficiency, making future applications more reliable in this matter.

One of the most crucial aspects of rescue activities is radio communication. In fact, these operations usually cover large

The associate editor coordinating the review of this manuscript and approving it for publication was Martin Reisslein.

areas where rescuers and volunteers must remain in contact with the base station through alternative networks due to main infrastructure being inspirational or nonexistent. As an example, in the case of mountain rescues, these areas may not have 2G-5G coverage due to landscape relief. Events such as hurricane Katrina or the tsunami in Sri Lanka in 2017 show a practical use of radio infrastructure in disasters, where volunteers and authorities have used radio communication technologies to remain in contact and coordinate rescue missions [6], [7].

Digital mobile radio (DMR) [8] has become a popular standard in these cases, while other alternatives such as TETRA [9] remain almost exclusively used by official institutions. Even though DMR has a global network through the Internet, there is only a limited number of master repeaters and the tendency is to build smaller repeaters or hotspots.

Mobility can be achieved through the use of Unmanned Aerial Vehicles (UAVs) that can carry small to medium objects. These have developed into powerful transports able to fly over complicated landscapes such as those in natural disasters or mountainous areas. UAVs can also be suitable mechanisms for providing additional services that can aid in rescue missions. In this sense, video streaming is a crucial service to obtain improved assessment of emergency situations because it provides visual information regarding the rescue area. In fact, having real-time images while searching for missing people can make a difference in performance and efficiency.

In this context, the main contribution of this work is a lightweight, flexible and low-cost system to support search and rescue operations in areas with scarce 2G-5G network coverage, such as mountainous areas. In such cases, rescuers rely on digital radio communication technologies for their coordination, and DMR has been widely used among radio hams since it does not require buying licenses and provides large coverage areas due to the use of their assigned VHF and UHF bands.

In such scenarios, the main benefit of the proposed system lies in the speed with which a new DMR hotspot can be deployed and fully operational to support search and rescue operations. This is achieved by running the DMR hotspot in a Raspberry Pi boarded in an airplane-type drone. On the one hand, placing the DMR hotspot in the air allows achieving the best possible coverage areas, as there are no obstacles between hams and the DMR base station. On the other hand, since the airplane type UAVs are remotely teleoperated, this approach results in a significant improvement of the deployment time compared to the classical solution where in the best cases, DMR hotspot is conveyed in an off-road vehicle and driven to a convenient place to provide a good coverage area, sometimes in mountainous regions by summiting their peaks. This system is illustrated in Fig. 1. Finally, it is also important to mention that airplane-type UAVs are the lowest-priced types of drones, and as a consequence, the proposed system results in a cost-effective solution. Furthermore, these

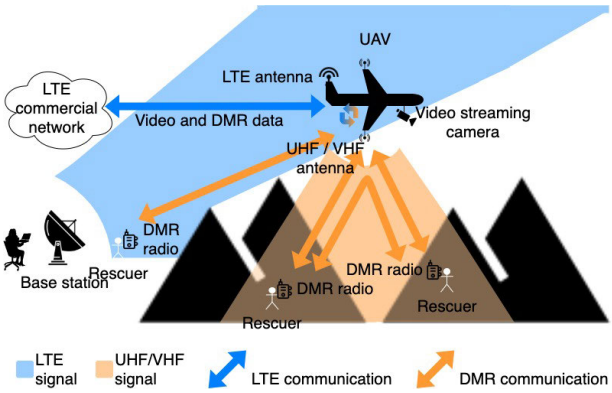

FIGURE 1. Proposed architecture.

UAVs are more efficient in terms of energy consumption, providing better characteristics for long-time missions.

Another important benefit of the proposed solution is the flexibility to perform on-field reconfigurations and adjustments, that is, to easily tune the configuration of running services or to load or unload new services. To achieve this goal, the proposed system is based on a virtualization solution where each service is loaded as an independent software container with all of its required dependencies and specific configurations and totally isolated from other containers running over the same hardware. The used virtualization architecture also allows automation, the advantages of which are remarkable mainly when a high amount of UAVs are deployed to provide coverage to extensive areas. In fact, automation provides an easy and cost-effective method for loading or unloading services from a centralized location in multiple remote devices at the same time.

Using the implemented testbed, critical performance parameters have been measured. The obtained results show that the proposed architecture can be deployed in less than 4 minutes and also displays a fast recovery from disconnections. This demonstrates the fast response time of this solution for meeting the needs of critical situations such as tsunamis or long-time devastated lands and also shows its high capacity to adapt to network events.

Having introduced the main aspects of the work and a brief background and motivation, we summarize the fundamental contributions of our work as follows:

- Design and implementation of a lightweight, flexible and low-cost system to allow dynamic deployment of services in scarce resource devices.
- Very fast deployment of rescue services infrastructure in emergency situations thanks to the use of an UAV.
- Easy reconfiguration and automation, thanks to the use of an orchestrator and virtual services.
- Showcase implementation of two services focused on emergency situations: DMR hotspot and video streaming service.
- Comprehensive performance evaluation of the proposed architecture in order to proof its effectiveness/suitability for the targeted scenarios.

The rest of this article is organized as follows: section II surveys the related work. Then, section III describes the proposed general architecture and section IV describe the evaluation of the system performance and the obtained performance results. Finally, section V discusses the obtained results and section VI summarizes the main conclusions and future research directions.

II. RELATED WORK

The aim of this section is to review alternative approaches to achieve the goal pursued by the architecture presented in this article. However, no other proposals have been found that specifically target the dynamic provision of radio and video streaming services for emergency situations. For this reason, this section has a broader scope and also reviews the implementation of DMR in devices with scarce resources as well as the use of UAVs and virtualization mechanisms for the provision of DMR and video streaming services.

In recent times, emergency service communications have been transitioning from analogue to digital infrastructures [10], to meet the demands of future applications and features such as spectral efficiency, group/broadcast calls, improved battery usage, and resilience to audio quality degradation. Professional Mobile Radio (PMR) digital technologies have been tested with standards such as TETRA, DMR and P25 [11], and have even been implemented by official institutions as a natural transition. However, this has been carried out through static implementations that only allow having an alternative communication architecture.

Some studies by Ganchev *et al.* [12] have developed DMR modules for use in M2M/IoT applications. In this case, a gateway module between a wireless sensor network and an IoT center of voice communication using an ARM [13] microcontroller was developed. This study demonstrates the use of a DMR module in low-cost devices with scarce resources.

Other work by Xie *et al.* [14] evaluated virtual machines (VMs) for container-based services, and containers were proved to be a an important candidate approach for obtaining scalable and dynamic networks. In fact, fast service deployments make it more suitable than VMs for some applications that are time-relevant, such as application for emergency situations. Further comparisons to ''bare-metal'' implementations have demonstrated that although Docker containers perform slightly worse in terms of CPU performance and time for the execution of the procedures, the difference is barely noticeable while still providing strong virtualization advantages. In fact, these investigations show that there is a difference in computation time of a half a second between native and Docker containers using *Y-cruncher* performance tester software, with both tests showing the execution time of approximately 70 sec. Comparison of VMs and containers also shows that execution times are lower for container virtualizations than for the hypervisor-based virtualization systems [15], as revealed by researchers Adufu *et al*. This is due to the start up times, because booting up is slower for a virtual machine than for a container.

The assessment of the performance of virtualization solutions is an important topic of current research. In particular, service deployment time has already been measured in other studies, and the investigations by Medel *et al.* [16] include multiple cases, taking into account CPU intensive applications, I/O intensive applications and network applications. Furthermore, their measurements also included creation and termination times for the deployments of preloaded Docker images that can be close to a real situation in which the target device for images is already prepared for deployment. Their results showed that the performance varies when deploying containers in different distributions among the resources. Yilma *et al.* have evaluated NFV MANO Systems [17] that are comparable to our work to some extent. They presented theoretical KPIs to benchmark the performance of two different systems, namely OSM Release 4 and ONAP-B. Their final KPIs used for evaluation obtained results for on-boarding process delay (OPD), deployment process delay (DPD) and CPU/RAM usage. The first two measure the times necessary to initialize a VM and deploy VNFs (virtual network function) and the time to start up the VNF itself, respectively. Our time measurements are similar, as we also split deployment times in distinct processes such as orchestrator initialization, the required time for a node to join a cluster and service deployment time. For ODP, their 5 services on average need 40 seconds each, and for DPD, the times are reduced to 26.4 seconds on average.

The use of video streaming in UAVs is a highly attractive concept because it has useful commercial applications. However, it may also have drawbacks that must be taken into account. In fact, some studies have analyzed the reliability of communications to and from the drone that necessitates a simple and reliable data link to be always available for connectivity. Using IEEE 802.11, as proposed in [18], rather than the point-to-point proprietary wireless links that are often used currently allows the UAV to broadcast to multiple receivers in the ground in a secure manner. However, the limited radio coverage of IEEE 802.11 signals is challenging for cases where the UAV must cover large areas or must operate at a significant distance from the base station.

Tipantuña *et al.* have developed an NFV-based energy scheduling algorithm for UAVs and 5G network infrastructures [19]. In the context of deploying services in UAVs such as routing tasks, video surveillance or Internet connectivity, they have developed a scheduling algorithm to maximize service time. Their solution consists of defining two main states for UAVs, called service execution state and replacement state. Their algorithm considers the energy consumption of executing the service and the necessary energy to replace the UAV that takes into account the required time for the UAV to return to the base station, migrate the VNF to the new device and the time to reach the location. This optimization and energy management enable the calculation of the number of UAVs necessary to provide a service in determined conditions, dimensioning and planning the deployment. Their results show that it is possible to achieve 100% service time in

TABLE 1. Summary of research works.

Authors	Virtualization technology	Multimedia services	Scarce resource hardware	Mobility (UAVs)	Emergency situations	Arch. performance assessment
Ganchev *et al.* [12]		X	X			
Xie *et al.* [14]	X					X
Adufu *et al.* [15]	X					X
Medel *et al.* [16]	X					X
Yilma *et al.* [17]	X					X
Van den Bergh *et al.* [18]		X		X		
Tipantuña *et al.* [19]	X	X		X		X
Gao *et al.* [20]	X			X		X
Zarakovitis *et al.* [21]		*		X		X
Nogales *et al.* [22]	X	X (low bandwidth)	X	X	X	
Salamí *et al.* [23]			X	X		X
Fernandez-Pacheco *et al.* [24]				X	X	
Pardo *et al.* [25]		X		X	X	X

* LTE signalling

certain conditions, demonstrating that continued service can be provided using UAV networks.

Following the approach of virtual services in UAVs, Ge Gao *et al.* [20] have studied the use of a hypervisor for real-time application deployments in UAVs. They developed a hierarchic architecture to plan and dimension virtual machines and applications depending on the level of criticality.

The use of UAVs for telecommunication networks has also been widely studied. Xilouris *et al.* [21] proposed the use of UAVs to deploy telecommunication nodes, presenting a possible architecture. They propose that future technologies such as 5G will implement network slicing to separate the network into logical segments, preparing each segment for determined purposes, and suggest that UAV infrastructures must adapt to this concept. Based on their concern about the current telecommunication network, their designs favor terrestrial devices due to antenna orientations in transmitters. However, their measurements show that it is possible to maintain stable communication with an aerial device through the current infrastructure.

Nogales *et al.* have also studied the use of UAV systems as programmable networks for 5G application deployments with NFVs [22]. The main advantages of using NFVs in this context is the flexibility to adapt to different requirements, the ability to configure services in critical times, the possibility to deploy additional services and functions in working environments and cost reduction in maintenance and investments. They deployed virtualized services with OpenStack, using FANET (flying ad hoc network) routing protocol in each UAV and coordinating the NFV environment with a management and orchestration (MANO) framework. Accomplished tests measure the throughput of data and management communications that take into account NFVI control and transmissions among UAVs over UDP protocol. Battery usage is also measured in multiple contexts and varies from 1 hour in normal conditions to 20 minutes in extreme situations.

The communication between base stations and UAV systems may not be optimal in some situations, particularly for data-heavy applications. Therefore, researchers Salamí *et al.* studied on-board processing of information in UAVs prior to its transmission to terrestrial stations [23]. In fact, data links used in UAV communications may not be sufficient in some cases although preprocessing also has remarkable drawbacks. UAVs must carry powerful hardware and software for this purpose, and therefore face the problem of high energy consumption. These researchers have studied multiple alternatives in both hardware and image processing software, concluding that there are differences amongst devices and that parallelization of processes does not improve time measurements drastically because a deep knowledge of the specific hardware used in the application is needed.

It has been proved that UAVs can have a strong beneficial effect in emergency situations, as shown in the study by Fernandez-Pacheco *et al.* [24]. Their results show that video information recorded using UAVs is beneficial in training rescuers and medical staff. Moreover, Pardo *et al.* measured the improvement in rescues of missing people by using UAV-assisted video [25]. It was found that for all of the performed tests, better performance was obtained for the group using the UAV compared to the control group.

Table 1 shows a summary of all studies analyzed in this section, identifying the characteristics they develop. We observe that there is still room for improvement in lightweight service deployment in critical situations. In fact, although there is some literature about the use of virtualization with scarce resource devices, we consider that there should be an architectural design to deploy multimedia services over low capacity hardware to fulfill the required performance features. In this aspect, the closest study is the work by Nogales *et al.* [22], although their solution is based on OSM which makes the overall framework heavier. Furthermore, their work focuses on assessing the performance of the implemented IP telephony service, but the performance of the operation of the architecture itself is not analyzed. Taking all that into account, there is a need to evaluate the stability, availability, and operational times of the proposed virtualization networking infrastructure.

Therefore, the work presented in this article represents an advance with respect to the current state of the art by allowing a fast and flexible deployment of a DMR architecture based on Docker containers and implementing useful services such as video streaming. While other studies focus on high-performance environments and enhancing effectiveness through virtualization, this work presents the framework in which to deploy multimedia services using scarce resource devices, which in this background can be applied to a practical use case in emergencies. This is an innovative proposal for addressing a current problem based on multiple technologies, along with a thorough performance assessment.

The next section describes the general architecture and each component of the infrastructure and analyses the proposed design.

III. GENERAL ARCHITECTURE

This section presents the overall architecture proposed to allow a fast deployment of a DMR hotspot and a video streaming service to aid rescue operatives in emergency situation where no communication infrastructures are available.

A. DESIGN CONSIDERATIONS

Taking into account the characteristics of the specific targeted scenarios, the designed architecture must comply with several requirements.

Most importantly, most emergency situations require a fast response. Therefore, the first design objective of the proposed architecture must be to allow providing rescue teams with radio communications and live video streaming of the affected area as soon as possible. To achieve this goal, we have chosen to use an unmanned aerial vehicle (UAV) to carry the necessary infrastructure. The UAV is able to reach the targeted area sooner than any other vehicle, particularly in the cases where the emergency situation has made roads impassable or if the landscape makes it difficult to access the require zone, such as in mountainous areas. However, the use of UAVs implies some restrictions on the equipment that can be carried, because the equipment must be light and must consume little power. Therefore, we chose to use a Raspberry Pi as part of our architecture for the deployment of the DMR and video streaming services.

Another important requirement of the targeted scenarios is that the solution must be reconfigurable in a fast and flexible manner. To achieve these goals, we have opted for a virtualization-based approach, and more specifically, for a solution based on Docker containers. These containers provide environments tailored to a specific purpose, usually implying the installation of services and libraries with suitable software versions and packages. Quite commonly, an unexpected event during a rescue operation requires a modification of the operation of the deployed services or the deployment of new service that were not considered in advance. In such a case, expert programmers and developers can work from their locations anywhere in the world on the modification of the deployed virtual containers or in the creation of new containers. These virtual containers can then be easily deployed in the Raspberry by non-expert people working on-site in the emergency field, even without having to land the UAV. Additionally, as previously mentioned, virtualization is a powerful tool for service deployments, because it enables development and testing prior to implementation and is more fault-tolerant than the 'bare-metal' deployments. Moreover, virtualization provides the possibility to deploy services and software over distinct hardware without having to configure it specifically.

Another key requirement of the designed architecture is that it must allow for automation. The considered rescue operations will frequently cover vast geographical areas and therefore, in many cases, more than one UAV will be necessary to provide the service to the whole deployed operational effort. In such a case, it is essential to have a mechanism that allows to automatically deploy and manage the lifecycle of the deployed services in all of the UAVs. Thus, it is possible to achieve high savings of time and effort. In this regard, the use of virtual containers makes it easier to ensure the scalability and carry out the set up of the architecture, because these virtual containers are ready-to-go images that can be managed by a centralized orchestrator. This orchestrator provides the ability to deploy, control and stop virtual containers, and is based on Kubernetes. In fact, Kubernetes is a container orchestrator that provides necessary tools for the management and control of all Docker containers. In particular, Kubernetes is preferred over the alternative orchestrators due to strong support from the developers and extensive use in all kinds of deployments. Kubernetes operates in a master-slave hierarchy using a device that controls other workers that will be those with scarce resources in this design. The network manager will use the Kubernetes API to deploy and control services running on the worker nodes solely by using virtualized containers.

Finally, it is also important to mention that the proposed solution based on UAVs and Raspberry Pis results in a cost-effective system that can be easily affordable by rescue services and volunteer groups who are the most common potential users of our solution.

B. FUNCTIONAL BLOCKS OF THE ARCHITECTURE

The proposed architecture consists of different modules deployed on the ground and on-board the UAV. Therefore, the use of a UAV requires the deployment of different communication systems for the control and teleoperation of the UAV itself, and also to enable the communication between the architecture modules deployed in the ground and the ones deployed in the Raspberry Pi on-board the UAV. Fig. 2 provides a graphical representation of the different communication systems available in the UAV. First, the UAV is used to interconnect (1) DMR mobile equipment. Additionally, it is connected to the ground station for control purposes by either (2) 1200 FSK modems or (3) WiFi interfaces. The radio control is carried out in either the 868 MHz or 2.4 GHz frequency band. On the other hand, the UAV sends

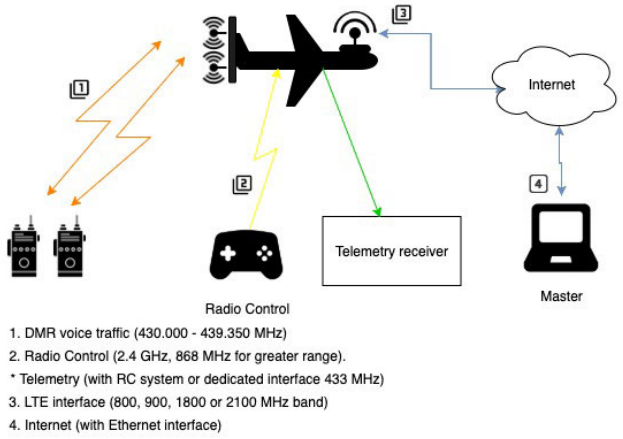

FIGURE 2. General architecture.

(4) telemetry data regarding the flight and (5) analogue video in order to allow the control of the drone in First Person View (FPV) mode or direct view. Finally, the UAV also has (6) an LTE connection with an LTE base station deployed in the ground. This connection is essential for the communication between the architecture elements deployed on the ground and on-board the UAV and to allow the integration of the DMR hotspot in the Brandmeister network.

Regarding the different functional blocks that constitute the overall architecture and make its operation possible, Fig. 3 provides a global view of the proposed architecture design. In a first glimpse, we can see that there are two main parts in the figure, a black square that gathers the modules inside the UAV and the rest of blocks that are in the ground. Apart from that, left side consists of blocks related to the services deployed and the right side takes into account all the other functions that are necessary for the operation.

Therefore, on the right-hand side, mission control and avionics systems between the UAV and the ground base station interact for a successful teleoperation of the UAV. This part of the architecture gathers control and telemetry systems that give the teleoperator real-time information about the flight and allow to remotely control it. These systems normally use 2.4 GHz interfaces, and also have functions such as autopilot and RTH (return to home) to protect the UAV in case of disconnection. The autopilot function is based on multiple flight patterns that can be circular or oval, enabling an automatic and iterative flight over an area. GPS units that include magnetometers are used to improve flight paths and to manage area coverage in each operation. There is also an option to incorporate first person view (FPV) mode to the UAV, using the required 5.8 GHz video interface connected to specific gogles. It must be emphasized that this video is completely different from the video streaming service that we deploying through Kubernetes, because this FPV video is based on specific hardware and is only used for piloting the UAV.

On the left-hand side, we show the functional blocks related to the Raspberry Pi device installed onboard the UAV. An LTE module is used to enable connectivity between the architecture modules on the ground and the ones deployed in the UAV, as well as to enable Internet access, mainly for the video streaming service and connection to the DMR worldwide network. Next to the LTE module, there is a UHF (ultra-high frequency) interface for providing DMR signal coverage to the rescuers below the flight zone of the UAV.

Finally, there are some blocks that belong to the virtualization system, as show in the green square called 'Configurable System' and the top left side, including the desktop computer (master) located at the ground base station. The solution works in a master-slave hierarchy, with a centralized master on the ground that controls multiple workers running in the Raspberry Pis (onboard the UAVs). More specifically, there is a base software layer that supports all *pods* and containers, and therefore the services that will be deployed. Services will be implemented on Docker containers that are deployed inside pods. The pods are Kubernetes units for container organization, and containers are standard units of software that create an environment for applications. The containers inside the same pod share some resources such as storage or network. *Kubelet* is an active daemon that communicates with the API server in the master and controls pods in the node, ensuring their appropriate operation. The most important service in the master is *etcd*, which is a distributed key-value store that keeps the cluster's configuration in a secure manner. Then, the API server presents these key-value pairs and provides an external interface for the nodes to access the master. The controller manages and supervises the cluster, ensuring its availability and condition. Finally, the scheduler defines the mapping for the pods that are newly created, assigning a node to each pod and defining how they will be distributed through the cluster.

In the following subsections, each service is discussed in-depth, explaining their operation and convenience in this use case.

C. DMR SERVICE

One of the most innovative aspects of this use case is to provide DMR radio signal coverage using an UAV, so that the operation of this service is of major importance. DMR is a digital radio standard defined by ETSI (European Telecommunications Standards Institute) that operates in VHF (Very High Frequency) and UHF (Ultra High Frequency) bands. It uses 12.5 kHz bandwidth with two time slots, and the network is divided in various hierarchy levels that guarantee communication among the users. Brandmeister demonstrated [26] one of the most successful DMR networks for which the core is based on 58 master stations that are connected through the Internet. Then, multiple repeaters are connected to these masters, and provide access points for the users. Similar to repeaters, another option is to build hotspots, and the main difference is that repeaters can perform local radio to radio contact as well as global communications, while hotspots are used to increase the coverage of a repeater.

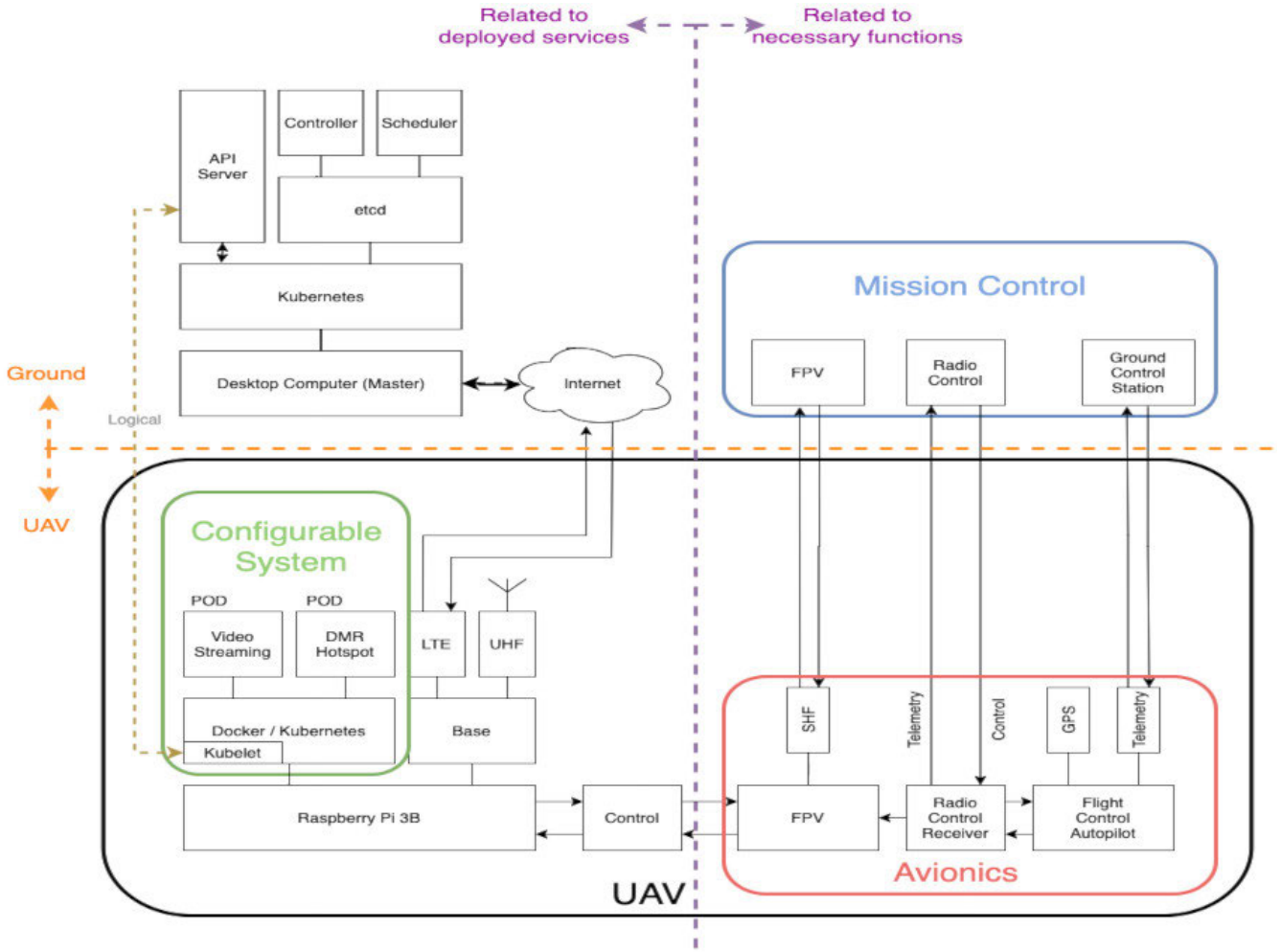

FIGURE 3. Proposed architecture.

Therefore, hotspots must always forward radio communications to a repeater. However, hotspots are much cheaper than repeaters and have full-time access to all talk groups that are the groups used to organize radio traffic into digital contacts [27]. Hotspots may vary in power and coverage but can be mobile as long as an Internet connection is available, making them attractive for rescue operations in mountainous regions and disaster zones.

The image used for this service is based on Alpine in order to reduce the libraries and overall software resources. Then, essential software such as *make*, *git* and *MMDVMHost* are installed, followed by the installation and adding the necessary configuration file. This file has to be preconfigured with the parameters of the hotspot in order to save time.

D. VIDEO STREAMING

Video streaming service is another important aspect of the proposed solution, providing real-time video to the rescuers. Bandwidth, resolution, latency and other parameters are crucial in this case, due to the fact that the devices used have scarce resources. Taking this into account, this work has implemented a real-time video transmission system with the appropriate use of coding tools. H.264 is the code used to compress the video after its recording because it provides native integration with Raspberries and suitable transmission tools for it such as UV4L [28] are already available. For transmission, transmission over UDP is used.

The image for this service is more complex than the previous one. In fact, it is based on a Raspbian Jessie image that is originally heavier. Furthermore, the video streaming service require the installation of additional libraries including *python3*, *libraspberrypi* and *picamera*.

IV. TESTBED

This section describes the practical implementation used to test the proposed system. The DMR hotspot is implemented on a Raspberry Pi 3 Model B with 1 GB of RAM and ARM Cortex A53 processor at 1.20 GHz, and the hotspot hardware attached is a STM32F103C8T6 module with two UHF antennas. The software used in the Raspberry is MMDVM Host that creates an interface between the MMDVM module and the network, and it is configured to provide an access point at 430 MHz while connecting to the Spanish Brandmeister master node through the Internet network interface.

The master computer that controls the node runs an Ubuntu 18.04 OS, and has 16 GB of RAM and an Intel i7 processor with eight cores at 3.60 GHz. These devices run Kubernetes version 1.15.0 and Docker 19.03.

For Internet connectivity, the master computer has an Ethernet interface at 1 Gb/s and the node device uses an LTE interface at 15 Mb/s. This interface consists of an LTE modem plugged in the Raspberry that is connected to a Software Defined Radio (SDR) eNB. As depicted in Fig. 4, the service is provided by the Amari Callbox Mini that includes an EPC and HSS system with connectivity to the Internet through a local network with public IP addresses and a firewall. In many real-world emergency situations, this architecture could be simplified by using commercial LTE network coverage if available at the altitude where the UAV is flying, but we have chosen to use this setup to obtain better access to the whole system. In this manner, we can measure multiple parameters in different points of the network.

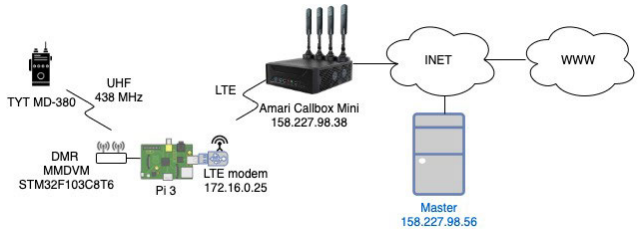

FIGURE 4. LTE network infrastructure.

For the UAV selected to carry the mobile part of the architecture, we have opted for a fixed-wing type of drone, mostly because its increased flight time and weight charge capacity are far superior to those for quadcopters even if take-off and landing operations are somewhat more complicated [29]. Battery efficiency is an important aspect to bear in mind, given that the drone carries other essential elements that add weight and consumption, such as the LTE modem and the DMR module. Those advantages are the reason why many object transporting and terrain mapping projects choose fixed-wing drones over quadcopters [30], [31]. The drone chosen for this use case is a SkyWalker X7 that is shown in Fig. 5. The radio used to control the UAV is a FrSky Horus X10 [32] that communicates with the drone in the 2.4 GHz band. Moreover, the power supply carried in the drone is an 8000 mAh 30C lithium polymer battery that is sufficient for powering all of the components included in the system. In fact, the Raspberry itself consumes roughly 250 mAh, including the peripherals attached to it such as the LTE modem, STM module and Pi camera. This setup will allow for an estimated flight time of the UAV with all the architecture operative of at least one hour.

The tests developed in this use case measure various Key Performance Indicators (KPIs) that are critical to assess the performance of the proposed system to provide the envisioned services. The main effort of testing analyzes time-relevant factors that are crucial for a fast preparation of the system and restoration after occasional connectivity failures. Other tests focus on performance parameters such as bit error rate (BER) or throughput that also offer sufficient information regarding the quality of the architecture used.

All of the performed tests were set up to measure these parameters, and are explained in-depth in the subsections below together with their results. These tests have been repeated over 50 times to consider a normal distribution of the results.

A. TEST 1: SETTING UP THE CLUSTER

The first test measures the master/cluster setup time. This is the first activity that must be analyzed because it is the main task that must performed prior to the deployment of any service. This process generates various services that are necessary for correct operation such as DNS, API server and node/pod controller. Due to the number of services deployed, time measuring tests must be designed correctly to ensure accurate and valuable results. The test is controlled by a script that uses Kubernetes API and its multiple functions, and in addition to the setup times, this test measures the time required to reset the cluster completely.

Fig. 6 shows that cluster start up time is close to 2 minutes (1 minute and 52.6 seconds) while reset time is 12.5 seconds on average. This is an acceptable time considering that this process is performed at the start of the deployment and will not be repeated.

B. TEST 2: CONNECTING NODES

This test measures the time needed to connect and disconnect the nodes from the cluster. This is an important feature that must be timed precisely because it affects other events such as the joining of the nodes after the connectivity is dropped. The main difference is that the first connection of a node to a cluster requires an association using the Kubelet service and exchange of the information about the cluster and the overall connection parameters. Meanwhile, reconnection after temporarily losing connection only consists of commusnication-restoring messages exchanged between the master and the node; this is measured in test 4, as described below.

Fig. 7 depicts a more fluctuating response than test 1. In this case, the average times are 1 minute and 40.1 seconds for connection and 2.1 seconds for disconnection, showing that associating a node to a cluster may be almost as slow as bringing up a cluster, while disconnecting a node is a very fast procedure. This is another process that must be done only once every deployment, and therefore, the obtained results are satisfactory overall.

C. TEST 3: DEPLOYING THE SERVICES

The aim of this test is to assess the time required to deploy the services on a Kubernetes cluster. In this case, two different Docker images have been implemented, namely a DMR hotspot and a video streaming service. The DMR image is much simpler than the video streaming image, and requires

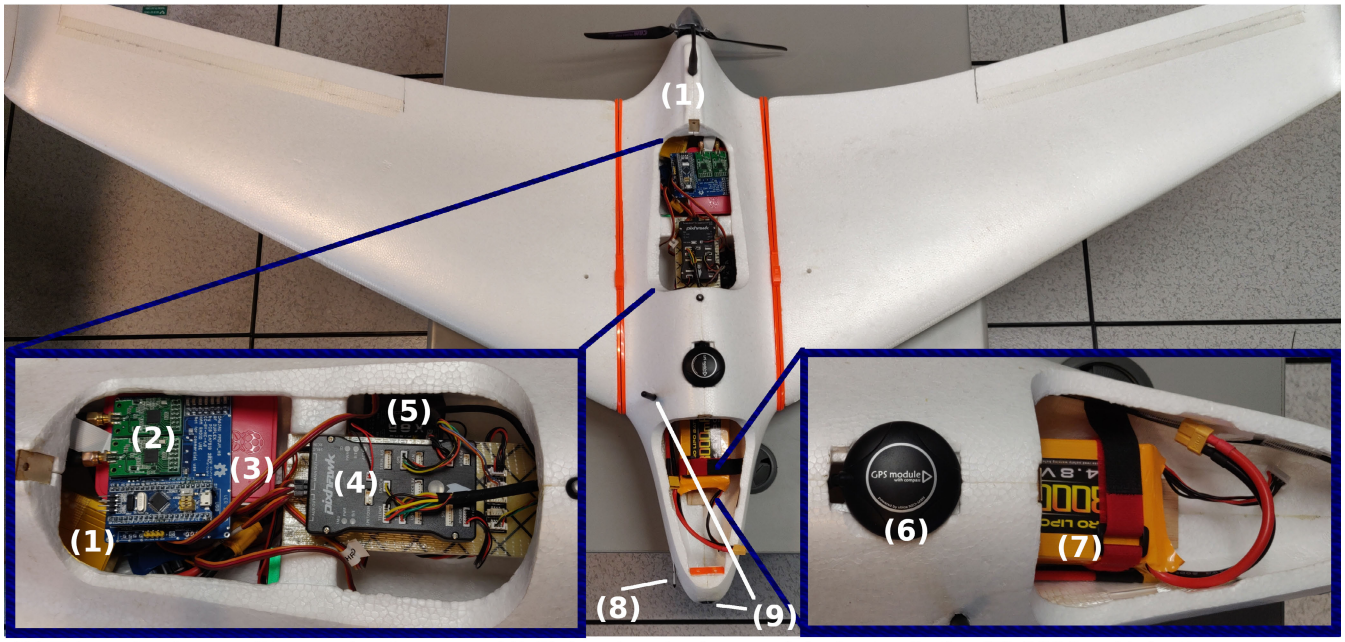

FIGURE 5. SkyWalker X7 fixed wing drone.

FIGURE 6. Test 1: Cluster start up and reset times.

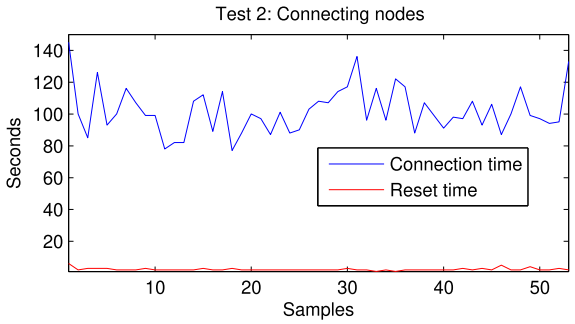

FIGURE 7. Test 2: Node connection start up and reset times.

less software, fewer libraries and less processing. The DMR service image results in a load of approximately 250 MB of, while the video streaming image load can rise up to 800 MB. Therefore, the first result provides information regarding the behavior of light-load services while the second one provides the same information for heavy-load services.

As demonstrated by the following results depicted in Figs. 8 and 9, service deployments in test 3 are faster than in the previous tests, which is a positive outcome considering emergency situations. Video streaming service statistics in this case show an average deployment time of 18.7 seconds and service stop average time of 42.8 seconds. Moreover, the DMR service produced similar but faster results in all aspects due to its lighter load in terms of resource consumption. Deployment statistics show an average time of 16.7 seconds for start up and 41.8 seconds for finishing.

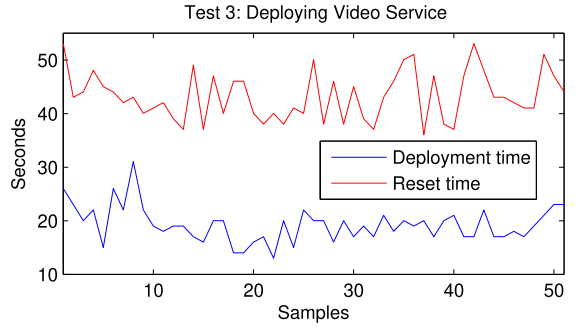

FIGURE 8. Test 3: Video streaming service deployment and stop times.

D. TEST 4: NODE RECOVERY TIME AFTER DISCONNECTION

After measuring the node connection/disconnection times, it is important to analyze node behavior in critical situations such as losing network connectivity or sudden shutdowns due

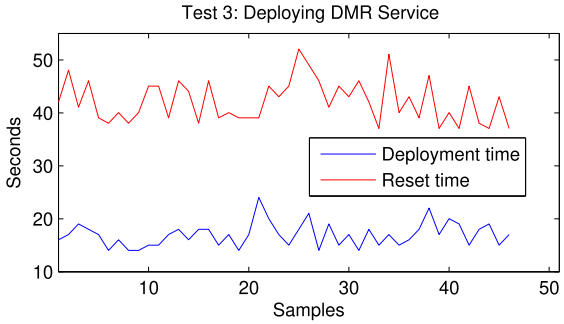

FIGURE 9. Test 3: DMR service deployment and stop times.

to power issues. In fact, a fast recovery time in these cases may be necessary to maximize availability.

Test 4 results evaluate the node recovery time after disconnection from the cluster with the results gathered in Fig. 10. In spite of a few rare outliers, the tendency of recovery is steady and does not vary for more than a couple of seconds. In fact, the average time is 9.3 seconds, while the outlier samples result in a maximum deviation of 22 seconds. This is most likely due to the Kubelet not being able to restore the communication with the master in the first attempt, and then having to wait between multiple retries.

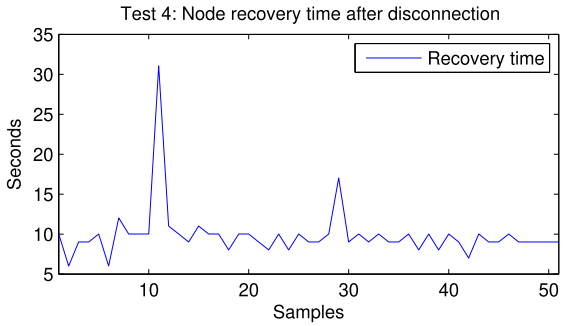

FIGURE 10. Test 4: Node recovery time after disconnection.

E. TEST 5: MASTER RECOVERY TIME AFTER DISCONNECTION

This test is quite similar to the previous test and measures the recovery time for a master after disconnection. However, Kubernetes clusters are controlled by masters, and rebooting of the master will break the cluster down entirely. Although it is possible to configure the master to start the Kubernetes service when turning on, previous configuration and attached services are not maintained. Taking this into account, a more probable case is analyzed here, which is the loss of network connectivity in the master.

It is important to note that when the master loses connectivity, the nodes continue to operate as expected. Therefore, the continuation of the functioning of the services deployed on worker nodes and their availability while the master recovers are advantageous.

The test results are depicted in Fig. 11, and show an average time of 7.8 seconds. A similar pattern to that of the previous test can be seen, as there are some peaks in this case too.

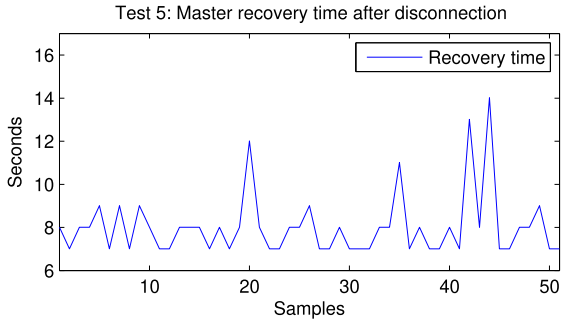

FIGURE 11. Test 5: Master recovery time after disconnection.

F. TEST 6: PACKET ERROR RATIO IN DMR

This test measures the packet loss percentage in a voice transmission and the BER percentage in multiple communications and extracts results from the measured data. *MMDVMHost* software used in this implementation provides information about these parameters, making it a useful tool for the monitoring of the performance during the activity.

First, a tool called *parrot* is implemented in DMR that is used to test whether there is connectivity with the repeater. It consists of sending an audio message and returning it to the sender after reaching the repeater. Using this tool, the average BER percentage obtained is 1.67% of the total sent, meaning that only a low percentage of bits are lost or returned with errors. Moreover, receiving the message from the repeater results in a BER percentage of 0.1%, which is a much lower value. It must be noted that this may vary depending on the distances from each element of the infrastructure, namely the user device, hotspot and repeater. However, depending on the environment and conditions, these variations can be countered through the use of more powerful antennas. For the data received from other users, the average BER result was 0.1764% and the obtained packet loss percentage was 4.1778%. These values show that the loss of information in this architecture is quite low and enables correct transmission for all messages.

G. TEST 7: VIDEO STREAMING PERFORMANCE INFORMATION

This test measures multiple KPIs in the LTE connection of the Raspberry to evaluate video streaming performance. Preliminary measurements were performed using a network protocol analyser while the video transmission is on. These measurements show that during 10 minutes, the video transmission has transferred 1167M bytes, in 805673 packets of 1410 bytes on average. In other words, the measured bandwidth measured is 15 Mbps (1.875 MByte/s), which is an acceptable value.

To obtain truthful results about the capacity of the LTE link, the tool used in the measurements is *iperf3* that gathers bandwidth, jitter and packet loss percentage among the other parameters. These values are used to describe the performance of the LTE link in a wide range of possibilities in order to obtain a reference for the potential response to the variation in the quality of the transmitted video. Fig. 12 presents the obtained results for the bitrate as a function of the lost packet percentage, and Fig. 13 depicts the related throughput.

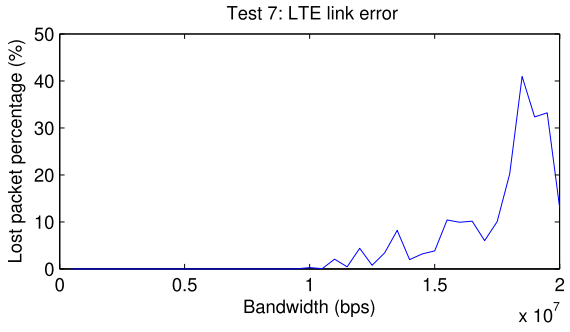

FIGURE 12. Test 7: LTE link error.

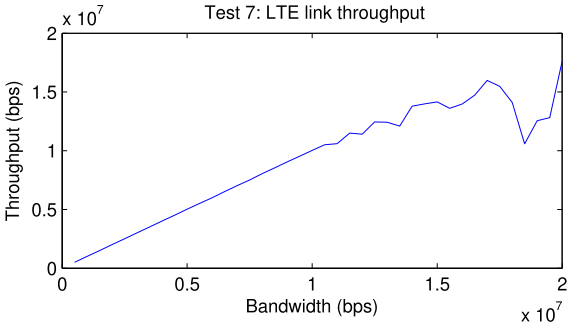

FIGURE 13. Test 7: LTE link throughput.

An examination of the results shows that this link is sufficiently robust with respect to lost packets to transmit the required video information. In fact, the video transmission requires a rate of 15 Mbps and at this level, the error rate is lower than 5%, which is sufficiently low for a good transmission. For the throughput, although at this rate, the throughput starts to decrease, it is still above the limit. This also shows that there is still room for acceptable error that may be required for future applications that may need higher video quality, requiring higher bitrate. Therefore, a greater bitrate can increase the error percentage, but taking into account that the transmission is carried out over UDP, the loss of some packets in a long transmission is acceptable.

V. DISCUSSION

Following the description of the performed tests in the previous sections, this section analyses all of the obtained results. Table 2 presents a summary of the performed time-measurement tests. First, the Kubernetes master and cluster preparation measurements show that this is the slowest part of the deployment. The cluster requires approximately 2 minutes on average to be ready for operation, but the standard deviation is steady and makes the process predictable. However, although the addition of nodes to the cluster requires almost the same amount of time on average, this process entails a greater uncertainty owing to its greater variation, with a standard deviation of almost 15 seconds. Combined, these two processes complete the initial preparation of the environment in which the services are deployed. Combining the results, it is safe to say that the average time required for the deployment of DMR and video streaming services is 3 minutes and 32.7 seconds. These values are particularly favorable in emergency situations that have been examine in this use case, because they show that the entire architecture can be deployed in less than 4 minutes. For ''bare-metal'' deployment, the time required to prepare and start up the service is similar to that of the virtualized procedure, but does not provide automatization for the process. Therefore, in a practical environment with multiple devices, the time difference will be much higher due to the need to configure each device manually. Furthermore, Kubernetes has scalability tools in addition to the monitoring of the service in execution, bringing it up automatically if there is any problem.

Moreover, services require less than 20 seconds to deploy, which is a highly satisfactory result compared to traditional services. These results are even more positive due to the fact that a virtualization layer is added. Moreover, the deployment times display a remarkably low standard deviation of only 3 and 2 seconds for video streaming and DMR, respectively. Differences between both services are lower than expected, even though their complexity and the sizes of the resulting images are quite different. Taking this into account, we can predict an estimated default time for deployments of any kind of service. To put this result into context, other studies [33] using similar virtualization environments (Kubernetes and microservices) have also measured deployment times for services. In this case, they evaluated service starting times without taking into account the host machine setup times or Kubernetes implementation, and therefore their results are directly comparable to ours. In fact, their results are on the same order of magnitude, as they need approximately 14 seconds to deploy the services in a single pod, a few seconds below our mean times (19.3 and 17.0). This difference may originate from the images used and the hardware in which they deploy the services, which is more powerful than our scarce resource devices, leading to a faster deployment.

For fault-tolerance, the tests performed show a great resiliency to network signal disconnections, with less than 10 seconds required for both master's and node's recovery. This means that in a critical situation where network infrastructure is not stable and may become unavailable, this architecture has an excellent recovery time.

Therefore, the measurements demonstrate excellent overall performance of the setup, achieving acceptable results on

TABLE 2. Summary of the results of the performed tests.

Measured Time	Mean Time (min:s)	Std. Deviation (s)	Max. Deviation (s)	95% Conf Interval (min:s)
Test 1: Start up	01:51.6	5.5	18.6	01:50.4 - 01:52.8
Test 1: Reset	00:12.7	1.0	2.3	00:12.4 - 00:12.9
Test 2: Start up	01:42.0	14.4	43.0	01:38.8 - 01:45.1
Test 2: Reset	00:02.3	0.8	3.7	00:02.1 - 00:02.5
Test 3: Video Start up	00:19.3	3.3	11.7	00:18.5 - 00:20.0
Test 3: Video Reset	00:43.2	4.5	9.8	00:42.2 - 00:44.2
Test 3: DMR Start up	00:17.0	2.2	7.0	00:16.5 - 00:17.4
Test 3: DMR Reset	00:42.2	4.0	9.8	00:41.3 - 00:43.0
Test 4: Reconnection	00:09.9	3.4	21.1	00:09.2 - 00:10.6
Test 5: Reconnection	00:08.0	1.6	8.0	00:07.6 - 00:08.3

timing and other parameters. It can be safely said that we achieved positive results and we improved current work in this research area. In comparison to the studies analyzed in section II and particularly [22], which is the closest to our work, we must underline that our solution is based on a lighter virtualization architecture, as demonstrated by the performance evaluation carried out. It must be noted that reducing processing time implies a proportional reduction of the energy consumption, since usage of the processor directly translates into energy utilization. And efficient use of batteries is a critical point when deploying solutions based on UAVs, as it allows for longer missions without having to recharge the battery.

VI. CONCLUSION AND FUTURE WORK

This article presents a general architecture that has been tailored to a specific practical use case, providing a solution to rescue missions in areas where mobile network coverage is not available by providing DMR radio service and video streaming. To achieve this aim, a UAV is equipped with a Raspberry and all of the necessary antennas and peripherals to implement the aforementioned services. To achieve deployment flexibility and agility, a virtualization-based approach is proposed, based on a Kubernetes orchestrator and Docker containers. It is important to mention that this solution is a general architecture and each implementation must consider the specific proposal.

To test the feasibility of the proposed overall solution, Key Performance Indicators were identified and measured, demonstrating a fast set up of the system and a highly satisfactory response time in case of temporary connection losses between the Kubernetes master and the raspberries that behave as slaves. Additionally, the performance results related to the quality of the provided DMR and video streaming services are also highly satisfactory.

Future research may consider the potential effects of implementing a video processing system in the device inside the drone that will replace the streaming service instead. For example, real-time image recognition on-board can be used to detect color patches that can represent people or fires, and transmit a GPS position and picture only in case of a positive match. Of course, a study of the power balance between the decrease in the power consumption related to the selective transmission and the increase involved in the processing should be carefully undertaken.

Considering the specific system proposed in this work, it may be used in multiple other applications without making modifications. Rural environments can take advantage of this aerial node for many tasks, such as crop care or harvesting, because enables the monitoring for outbreaks and evolution of dangerous natural events. It can also be used to search for wild animals in protected natural parks, because of the low impact of an UAV in those environments. For urban areas, our system can also be used to control traffic jams or crowded areas and make these zones safer.

This communication solution can achieve better results considering the improvements in more advanced technologies and infrastructures. Virtualization technology in this system may also have some variations due to recent developments in multiple standards. In fact, charms can be used alongside Kubernetes in order to implement virtual services, providing dynamic code for the deployment of Docker containers. Furthermore, it is possible to create a nested infrastructure with Kubernetes orchestrating OpenStack virtual machines using these charms, providing an alternative system for service deployments.

We intend to work on all of these aspects to improve the solution proposed in this work, taking into account the developing technologies and standards that may be beneficial for this system.

REFERENCES

[1] ITU-International Telecommunication Union. (Jun. 2019). *H.264: Advanced Video Coding for Generic Audiovisual Services*. Accessed: Nov. 2019. [Online]. Available: https://www.itu.int/rec/T-REC-H.264-201906-I/en

[2] Statista. (Aug. 2019). *74% of United States Homes Have A Video Streaming Service*. Accessed: Nov. 2019. [Online]. Available: https://www.statista.com/chart/19171/share-of-households-with-access-to-a-subscription-video-on-demand-service

[3] Statista. (Aug. 2019). *Streaming Dominates Music Consumption in the U.S.* Accessed: Nov. 2019. [Online]. Available: https://www.statista.com/chart/10185/music-consumption-in-the-us

[4] Statista. (Aug. 2019). *Video Streaming Still Has Room to Grow*. Accessed: Nov. 2019. [Online]. Available: https://www.statista.com/chart/17464/video-streaming-forecast/

[5] Statista. *Video Streaming (SVoD) Worldwide*. Accessed: Dec. 2019. [Online]. Available: https://www.statista.com/outlook/206/100/video-streaming-svod-/worldwide

[6] NBC News. *Ham Radio Operators to the Rescue After Katrina*. Accessed: Nov. 2019. [Online]. Available: http://www.nbcnews.com/id/9228945/ns/technology_and_science-wireless/t/ham-radio-operators-rescue-after-katrina

[7] EMCOM Spain. (Jun. 2017). *Radioaficionados en las Inundaciones de Sri Lanka de 2017*. Accessed: Nov. 2019. [Online]. Available: https://emergencias.ure.es/radioaficionados-en-las-inundaciones-de-sri-lanka-de-2017/

[8] *Electromagnetic Compatibility and Radio Spectrum Matters (ERM): Digital Mobile Radio (DMR) Systems Part 3: DMR Data Protocol*, document ETSI TS 102 361-3 V1.1.7 (2007-12), ETSI, Dec. 2007.

[9] S. Bakaric, M. Borzic, D. Bratkovic, and V. Grga, "TETRA (terrestrial trunked radio)–technical features and application of professional communication technologies in mobile digital radio networks for special purpose services," in *Proc. 47th Int. Symp. (ELMAR)*, 2005, pp. 307–310.

[10] T. Onali, M. Sole, and D. D. Giusto, "DMR networks for health emergency management: A case study," in *Proc. 7th Int. Wireless Commun. Mobile Comput. Conf.*, Jul. 2011, pp. 2151–2156.

[11] N. D. Grechanik, A. V. Arzhakov, and O. V. Tarakanov, "Research in radio frequency signals using APCO P25 standart in moscow," in *Proc. IEEE Conf. Russian Young Res. Electr. Electron. Eng. (EIConRus)*, Feb. 2017, pp. 146–150.

[12] I. Ganchev, Z. Ji, and M. O'Droma, "Designing a low-cost DMR module for use in M2M/IoT applications," in *Proc. 2nd URSI Atlantic Radio Sci. Meeting (AT-RASC)*, May 2018, pp. 1–2.

[13] University Of Maryland. *Advanced RISC Machines—An Introduction to the ARM Architecture*. Accessed: Nov. 2019. [Online]. Available: https://www.cs.umd.edu/~meesh/cmsc411/website/proj01/arm/

[14] X.-L. Xie, P. Wang, and Q. Wang, "The performance analysis of docker and rkt based on kubernetes," in *Proc. 13th Int. Conf. Natural Comput., Fuzzy Syst. Knowl. Discovery (ICNC-FSKD)*, Jul. 2017, pp. 2137–2141.

[15] T. Adufu, J. Choi, and Y. Kim, "Is container-based technology a winner for high performance scientific applications?" in *Proc. 17th Asia–Pacific Netw. Operations Manage. Symp. (APNOMS)*, Aug. 2015, pp. 507–510.

[16] V. Medel, O. Rana, J. A. Banares, and U. Arronategui, "Modelling performance resource management in kubernetes," in *Proc. IEEE/ACM 9th Int. Conf. Utility Cloud Comput. (UCC)*, Dec. 2016, pp. 257–262.

[17] G. M. Yilma, F. Z. Yousaf, V. Sciancalepore, and X. Costa-Pérez, "On the challenges and kpis for benchmarking open-source NFV MANO systems: OSM vs ONAP," 2019, *arXiv:1904.10697*. [Online]. Available: http://arxiv.org/abs/1904.10697

[18] B. Van den Bergh, A. Chiumento, and S. Pollin, "Ultra-reliable IEEE 802.11 for UAV video streaming: From network to application," in *Advances in Ubiquitous Networking 2* (Lecture Notes in Electrical Engineering), vol. 397, R. ElAzouzi, D. S. Menasche, E. Sabir, and F. DePellegrini, and M. Benjillali, Eds. Berlin, Germany: Springer, 2017, pp. 637–647.

[19] C. Tipantuña, X. Hesselbach, V. Sánchez-Aguero, F. Valera, I. Vidal, and B. Nogales, "An NFV-based energy scheduling algorithm for a 5G enabled fleet of programmable unmanned aerial vehicles," *Wireless Commun. Mobile Comput.*, vol. 2019, pp. 1–20, Feb. 2019.

[20] G. Gao, Y. Zhang, J. Qian, and Y. Sun, "The lightweight system for UAV cooperative mission planning simulation," in *Proc. CyberC*, Oct. 2019, pp. 371–374.

[21] G. K. Xilouris, M. C. Batistatos, G. E. Athanasiadou, G. Tsoulos, H. B. Pervaiz, and C. C. Zarakovitis, "UAV-assisted 5G network architecture with slicing and virtualization," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2018, pp. 1–7.

[22] B. Nogales, V. Sanchez-Aguero, I. Vidal, and F. Valera, "Adaptable and automated small UAV deployments via virtualization," *Sensors*, vol. 18, no. 12, p. 4116, Nov. 2018.

[23] E. S. S. Juan, J. A. Soler, R. Cuadrado, C. Barrado, and E. Pastor, "Virtualizing super-computation on-board UAS," *ISPRS Int. Arch. Photogramm., Remote Sens. Spatial Inf. Sci.*, vols. XL-7/W3, pp. 1291–1298, Apr. 2015.

[24] A. Fernandez-Pacheco, L. Rodriguez, M. Price, A. Perez, N. P. Alonso, and M. Pardo, "Drones at the service for training on mass casualty incident: A simulation study," *Medicine*, vol. 96, p. e7159, Jun. 2017.

[25] M. Pardo and N. Perez Alonso, "Utilidad de los vehículos aéreos no tripulados en la búsqueda y triaje de personas en situaciones de catástrofe," *Emergencias*, vol. 28, pp. 109–113, Feb. 2016.

[26] Brandmeister. *Brandmeister Network*. Accessed: Nov. 2019. [Online]. Available: https://brandmeister.network/

[27] DMR. *DMR Talkgroups*. Accessed: Nov. 2019. [Online]. Available: https://www.dmrfordummies.com/talkgroups/

[28] Linux-Projects. *UV4L—User Space Video4Linux*. Accessed: Nov. 2019. [Online]. Available: https://www.linux-projects.org/uv4l/

[29] sUAS News. (Jul. 2019). *Fixed Wing Drones vs. Quadcopters: A Project Comparison*. Accessed: Nov. 2019. [Online]. Available: https://www.suasnews.com/2019/07/fixed-wing-drones-vs-quadcopters-a-project-comparison/

[30] X. Song and S. Hu, "2D path planning with dubins-path-based a algorithm for a fixed-wing UAV," in *Proc. 3rd IEEE Int. Conf. Control Sci. Syst. Eng. (ICCSSE)*, Aug. 2017, pp. 69–73.

[31] L. Yang, Z. Liu, and X. Wang, "A new image-based visual servo control algorithm for target tracking problem of fixed-wing unmanned aerial vehicle," in *Proc. Chin. Control Conf. (CCC)*, Jul. 2019, pp. 8142–8147.

[32] FrSKY. *Horus X10*. Accessed: Nov. 2019. [Online]. Available: https://www.frsky-rc.com/product/x10/

[33] B. Dab, I. Fajjari, M. Rohon, C. Auboin, and A. Diquélou, "An efficient traffic steering for cloud-native service function chaining," in *Proc. 23rd Conf. Innov. Clouds, Internet Netw. Workshops (ICIN)*, Feb. 2020, pp. 71–78.

ASIER ATUTXA received the B.Sc. and M.Sc. degrees in telecommunication engineering from the University of the Basque Country (UPV/EHU) in 2018 and 2020, respectively. He is currently pursuing the Ph.D. student in mobile network information and communication technologies with the UPV/EHU. He joined as a Researcher with the I2T Research Laboratory (Engineering and Research on Telematics), Department of Communications Engineering, UPV/EHU, in 2018. His research interests include software-defined networking and virtualization.

JASONE ASTORGA received the B.Sc. and M.Sc. degrees in telecommunication engineering from the University of the Basque Country (UPV/EHU) in 2004 and the Ph.D. degree from the same university in 2013. From 2004 to 2007, she was with Nextel S. A., a telecommunications enterprise. In 2007, she joined as a Lecturer and as a Researcher with the I2T Research Laboratory, UPV/EHU, where she is currently an Assistant Professor. She has participated in several research projects at the local, national, and European levels. She has supervised two Ph.D. theses. Her research interests include software-defined networking and network function virtualization, IP-enabled wireless sensor networks, and cybersecurity.

MAIDER HUARTE received the Ph.D. degree in i-voting systems with the I2T Research Group in 2009. Her Ph.D. thesis was on designing and validating cryptographic communication protocols. Since 2002, she has been a Lecturer with the University of the Basque Country, where she started teaching in computer science subjects, such as programming languages and computer communications. She currently teaches advanced Java EE programming and networking. She is also member of the I2T Research Group, where she has been researching cryptographic security, distributed systems, and software fault tolerance. She is also researching software-defined networks and OpenFlow.

EDUARDO JACOB (Senior Member, IEEE) received the B.Sc. degree in industrial engineering and the M.Sc. degree in industrial communications and electronics in 1987 and 1991, respectively, and the Ph.D. degree in communications engineering from the University of the Basque Country (UPV/EHU) in 2001. He spent in a public research and development telecommunications enterprise (currently Tecnalia) for two years. He spent several years as the IT Director in the private sector. From 2012 to 2016, he was with the Faculty of Engineering in Bilbao, UPV/EHU, where he was the Head of the Department of Communications Engineering. He is currently a Full Professor with the Faculty of Engineering in Bilbao. He also leads the I2T (Engineering and Research on Telematics) Research Laboratory. He also directed several Ph.D. theses and managed several research projects at the local, national, and European levels. His research interests include applying software-defined networks to industrial communications, cybersecurity in distributed systems, software-defined wireless sensor networks, and in-network processing.

JUANJO UNZILLA received the B.S. and M.S. degrees in electrical engineering and the Ph.D. degree in communications engineering with the Department of Communications Engineering, UPV/EHU, in 1990 and 1999, respectively. He is a currently a Professor with the Department of Communications Engineering, UPV/EHU, where he teaches subjects related to telecommunications networks and services. He is part of the 12T Research Laboratory, where he participates in several national and European research and development projects. His research interests include SDN and NFV, network security, and techno-economic models for access networks. His other research interests are models and metrics for knowledge transfer from universities to enterprises.