http://doi.org/10.35784/iapgos.6786

received: 02.11.2024 | revised: 13.03.2025 | accepted: 16.03.2025 | available online: 31.03.2025

REVIEW OF OPERATING SYSTEMS USED IN UNMANNED AERIAL VEHICLES

Viktor Ivashko[1], Oleh Krulikovskyi[2], Serhii Haliuk[2], Andrii Samila[2]
[1]Yuriy Fedkovych Chernivtsi National University, Department of Computer Sciences, Chernivtsi, Ukraine, [2]Yuriy Fedkovych Chernivtsi National University, Department of Radioengineering and Information Security, Chernivtsi, Ukraine

Abstract. *Operating systems (OS) play a major role in the functionality and performance of unmanned aerial vehicles, serving as their central nervous system to manage various components and functions. This article provides a comprehensive overview of embedded operating systems (EOS), real-time operating systems (RTOS), and cloud operating systems (Cloud OS) intended for unmanned aerial vehicles (UAVs). In particular, from the perspective of practical use, both the strengths and weaknesses of the following operating systems were analyzed: PX4 Autopilot, ArduPilot, NuttX, Robot Operating System (ROS), FreeRTOS, MicroPython, and ChibiOS/RT. A general overview of the potential practical applications of Cloud OS is also presented. Therefore, one can gain insights into the criteria for selecting operating systems, as well as their strengths and limitations. It is important to understand that the role of an operating system in UAV development is crucial for optimizing performance, safety, and efficiency across various applications, from agricultural monitoring to security surveillance.*

Keywords: unmanned aerial vehicles, operating system, embedded operating systems, real-time operating systems, Cloud OS

PRZEGLĄD SYSTEMÓW OPERACYJNYCH STOSOWANYCH W BEZZAŁOGOWYCH STATKACH POWIETRZNYCH

Streszczenie. *Systemy operacyjne (OS) odgrywają kluczową rolę w funkcjonowaniu i wydajności bezzałogowych statków powietrznych, stanowiąc ich centralny układ nerwowy i zarządzając różnymi komponentami i funkcjami. W artykule tym zaprezentowano kompleksowy przegląd systemów operacyjnych wbudowanych (EOS), systemów operacyjnych czasu rzeczywistego (RTOS) i systemów operacyjnych w chmurze (Cloud OS) przeznaczonych dla bezzałogowych statków powietrznych (UAV). W szczególności, z perspektywy praktycznego zastosowania, przeanalizowano mocne i słabe strony następujących systemów operacyjnych: PX4 Autopilot, ArduPilot, NuttX, Robot Operating System (ROS), FreeRTOS, MicroPython i ChibiOS/RT. Przedstawiono również ogólny przegląd potencjalnych praktycznych zastosowań Cloud OS. Dzięki temu można poznać kryteria wyboru systemów operacyjnych, a także ich mocne i słabe strony. Ważne jest, aby zrozumieć, że rola systemu operacyjnego w rozwoju bezzałogowych statków powietrznych jest kluczowa dla optymalizacji wydajności, bezpieczeństwa i efektywności w różnych zastosowaniach, od monitorowania rolnictwa po nadzór bezpieczeństwa.*

Słowa kluczowe: bezzałogowe statki powietrzne, system operacyjny, wbudowane systemy operacyjne, systemy operacyjne czasu rzeczywistego, Cloud OS

Introduction

An important element of any control system is the software used to implement it. In this aspect, an operating system is the software, the main task of which lies in the management of the computer hardware, an "environment" that provides the basis for other applications to function proper and interact with each other [23, 27].

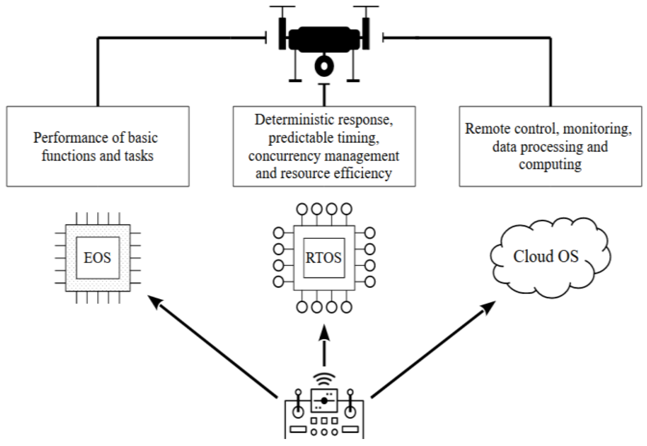

Performance of basic functions and tasks
Deterministic response, predictable timing, concurrency management and resource efficiency
Remote control, monitoring, data processing and computing
EOS
RTOS
Cloud OS

Fig. 1. General classification of operating system types used in UAVs

Unmanned aerial vehicle (UAV) or simply drone is an autonomous, mobile apparatus constructed and programmed to perform tasks which are difficult or dangerous to be carried out by humans [9]. Operating system for drones serves as it the central nervous system, managing various components and functions to enable safe, reliable, and efficient flight operations. General classification of operating systems types for UAVs is presented in Fig. 1.

It is also responsible for managing the following:

1. Flight control: stabilizing the drone, interpreting user commands, and adjusting motor speeds to maintain balance and stability in flight.
2. Sensor integration: facilitating the integration and interpretation of data from various sensors to assist in flight control and navigation.
3. Communication: handling communication protocols, ensuring reliable data exchange, and enabling features like remote control and telemetry.
4. Autonomous operation: providing the framework for implementing autonomous behaviors, including waypoint navigation, obstacle avoidance, and mission planning.
5. Battery management: monitoring battery levels, regulating power usage, and implementing features such as low-battery warnings and automatic return-to-home functionality to ensure safe operation and prevent mid-air shutdowns due to depleted batteries.
6. Payload control: managing the interaction with payloads, including capturing and processing data from sensors or activating/deactivating payload components as needed.

After the manufacturing process practically all UAVs come with embedded operating systems. Systems that are intended for embedded computer systems (EOS) to increase reliability and functionality of task resolving. For drones they vary basing on the specific requirements of their hardware, intended use and the preferences of the developers. Here are some commonly used embedded operating systems for UAVs.

PX4 Autopilot is an open-source autopilot system aimed at autonomous aircraft. It provides a comprehensive set of features for controlling unmanned vehicles, including fixed-wing aircraft, multirotor helicopters, and VTOL (Vertical Take-Off and Landing) vehicles. PX4 is highly customizable and supports a wide range of hardware platforms [36, 37]. ArduPilot is another open-source autopilot system widely used in the drone community. It offers similar features to PX4 and supports various types of vehicles, including drones, planes, rovers, and boats. It is known for its flexibility and extensive community support [29, 30]. NuttX is an operating system designed for embedded systems, including drones. It offers a small footprint, low latency, and deterministic behavior, making it suitable for applications where precise timing is crucial. NuttX supports a variety of processor architectures and features a POSIX-like API for easy application development [34, 35]. While primarily used in robotics applications, robot operating system (ROS) is also utilized in drone development for tasks such as navigation, perception, and control. ROS provides a framework for building modular and reusable software components, facilitating rapid development and experimentation [39, 40]. These are just a few examples of embedded operating systems used in drone development. Depending on the specific requirements of the project, one may choose one of these systems. More detailed review of the different type of EOSs used in UAVs is given in the following section.

Using a real-time operating system offers several advantages over a basic operating system: deterministic response; predictable timing; concurrency management; resource efficiency; fault tolerance. RTOSs prioritize tasks based on urgency, provide deterministic timing behavior, manage concurrency efficiently, and enhance fault tolerance, making them essential for reliable and safe drone operations, especially in dynamic and unpredictable environments [4]. Thus, the analysis of RTOSs along with EOS is an important issue.

Cloud operating systems (Cloud OS) for UAVs represent a paradigm shift in how UAVs are operated and managed. These systems leverage cloud computing infrastructure and services to enable remote control, monitoring, data processing, and collaboration for UAV missions.

Therefore, this study is dedicated to a comprehensive analysis of various operating systems used in drones. Such an analysis helps identify their strengths and weaknesses, enhance performance, and improve security measures. Additionally, the results of this investigation can aid in optimizing drone operations across different applications (agriculture, surveillance, delivery services, etc.) and contribute to the development of safer, more reliable, and more efficient autonomous systems.

1. Embedded operating systems (EOS)

EOS are specialized operating systems designed to run on embedded systems, which are typically small, resource-constrained computing devices that perform dedicated functions or tasks. These systems can range from simple microcontrollers to more complex devices like industrial control systems, IoT devices, consumer electronics, and automotive control units. They play a crucial role in maintaining a wide range of embedded systems, providing the necessary software infrastructure to support their functionality, performance and reliability in various application domains (Fig. 2).

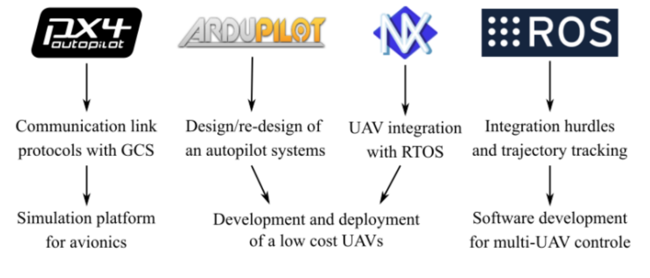

px4 autopilot
ARDUPILOT
NX
ROS
Communication link protocols with GCS
Design/re-design of an autopilot systems
UAV integration with RTOS
Integration hurdles and trajectory tracking
Simulation platform for avionics
Development and deployment of a low cost UAVs
Software development for multi-UAV controle

Fig. 2. Possible application domains of commonly used EOS for UAVs [30, 35, 37, 39]

1.1. PX4 Autopilot

A good example of an embedded operating system is PX4 Autopilot, which is a popular open-source flight control software suite designed for drones and other unmanned vehicles. Allouch et al. in paper [1] successfully demonstrated the use of the PX4 together with MAVLink Protocol. A protocol which is responsible for establishing the link between the drone and ground control station (GCS) and provides stable and quick telemetry, status, commands, controls data transfer. Another good example of PX4 application is reported by Jing et al. in work [12]. In accordance with their work, a disturbance-observer-based nonlinear surface controller was designed and successfully validated with the help of simulated quadcopter based on PX4. García and Molina in paper [7] stated that a PX4 also can be a very good platform for simulating the real conditions of navigation and obstacle avoidance. They present an approach to integrate so called LIDAR (light identification, detection and ranging) sensor to improve navigation of UAV in simulated situations based on a real PX4 platform. Thus, PX4 Autopilot is a very good candidate for UAV which prefer to use a specific communication link protocols with GCS. It can also be used as a very good testing or validation tool of different types of controllers with an aim to expand the functionality of UAV and as a platform for simulation of the real navigation conditions of UAVs. It is always good practice to conduct computer modeling and testing before performing a direct practical experiment. Good arguments and examples are provided by the authors in work [26]. In summary, PX4 Autopilot offers robust and flexible solutions for UAV control and simulation, with strong community support and integration capabilities.

1.2. ArduPilot

ArduPilot is a good basis for autopilot system of the remotely operated vehicle (ROV). In the paper reported by Luo et al. [18] it was successfully re-designed and implemented the proposed autopilot system. New functions were added to the original with the aim to create a new flight mode, simultaneously maintaining the original integrity of core functions. As a result, with the help of PC software it was possible to transfer control commands (depth and altitude hold) to the ROV together with analysis of control performance. ArduPilot can also be successfully used as basis for adaptive autopilots of fixed-wing UAVs. Baldi et al. in work [2] proposed an adaptive method for ArduPilot-based autopilots. They augmented the PID (proportional–integral–derivative) controller loops embedded inside ArduPilot with a model-free adaptive control method. The adaptive augmentation was used for attitude and energy control without requiring a real UAV model. The output efficiency was measured in terms of tracking errors and total energy control loops. As a result of their experiments, the proposed augmentations have significantly improved the UAV performance (all payloads and wind conditions), the drone also exhibited more than 70% improved tracking, had 7% reduced control effort and less affected by wind. ArduPilot is a good choice for UAVs used in weeds, pest and disease detection. According to the paper [10] reported by Shankar et al., ArduPilot based UAV with the help of python programming language and image processing gave the user the simplest interface for monitoring. The proposed UAV system helped to locate regions that are affected by diseases and pesticides and to take actions (chemicals) only in affected areas. In general, the entire system proved to be cost effective compared to other systems. Thus, ArduPilot is a good candidate for design/re-design of an autopilot systems for ROV and more economical (in terms of production and application) UAVs for agriculture monitoring, etc. In general, ArduPilot offers a versatile and cost-effective solution for a range of UAV applications, with notable advantages in control capabilities and user interface integration.

1.3. NuttX

NuttX is an open-source real-time operating system designed for embedded systems that offers a rich set of features, including a POSIX-compatible API, making it suitable for various applications from small microcontrollers to more complex embedded systems. Gill and D'Andrea in paper [8] reported a successful use of NuttX EOS in a vertical takeoff and landing (VTOL) UAV of a quadrotor design with an annular wing (that provides lift in forward flight) for propulsion and attitude stabilization. An autonomous control system was designed based on model inversion, which took into account the aerodynamics of the wing. Software simulation was done based on NuttX that is POSIX (portable operating system interface for UNIX) compliant RTOS. Experimental results showed that the control system provided accurate transition maneuvers and high-speed trajectory tracking. NuttX is a good system for integration the UAV with RTOS based on UAVCAN (Unmanned Aerial Vehicle Controller Area Network) [22]. This provides the UAV system with the ability to use thread protocol which enables RTOS to receive data wireless, without the internet. NuttX is a good EOS for low cost and flexible UAVs. For instance, Sørensen et al. in work [25] turned a commercial drone into an extensible airborne sensor platform due to the functionality expansion of the drone system components. It was the basis for the RTOS that was responsible for deterministic behavior and scheduling of the timing of critical paths in the flight control software (attitude control etc.) Thus, if considering a development and deployment of low cost UAVs (even with specific design) for different type of task managing a NuttX software is good choice. In summary, NuttX offers a powerful and flexible real-time operating system with strong POSIX compatibility, real-time capabilities, and proven performance in UAV applications.

1.4. Robot Operating System (ROS)

ROS is an effective system for trajectory tracking of UAV, as reported Kamel et al in chapter [13]. Authors presented an overview of many model-based control strategies for multiple classes of UAV. The trajectory tracking for multi-rotor system was achieve due to it integration with ROS. The integration strategy for the ROS lied in the creation of a ROS node to link the corresponding controller to ROS environment. The controller and other components are implemented as C++ shared libraries that get linked to the node at compilation time. These controllers are available as open source ROS package on https://github.com/ethz-asl/mav_control_fw for fixed wing MAVs (micro air vehicles) and https://github.com/ethz-asl/mav_control_rw for rotary wing MAVs. ROS can be used for development and testing security frameworks concerning secure UAV communications. Lee et al. in paper [16] proposed a ROS based framework to focus on security issues essential for flight missions. The unmanned aircraft systems (UAS) were operated in a native and non-native ROS environment. The performance of the proposed framework was verified through experiments. According to corresponding experiments showed that the UAS fails to function normally due to the cyberattacks (injection of abnormal data into UAV flight) which was possible through vulnerability of the ROS. But, by applying proposed security framework the system was defended against attacks and functioned normally throughout the experiments. As reported by Lamping et al. [15] ROS is a good choice in multi-UAV control software development. It was successfully used in development of so-called "FlyMaster" software for multi-UAV supervision and control. The use of ROS shown to reduce the complexity of the software development time, helped in the integration hurdles, and increased scalability, flexibility and support. Thus, ROS is a good system for trajectory tracking, secure use and software development for multi-UAV control. In general, ROS offers powerful capabilities for UAV trajectory tracking, security framework development, and multi-UAV control with strong community support and flexibility.

A comparative analysis of PX4 Autopilot, ArduPilot, NuttX, and ROS EOS is provided in Table 1.

Table 1. Advantages and disadvantages of the PX4 Autopilot, ArduPilot, NuttX, and ROS EOS

PX4 Autopilot	ArduPilot	NuttX	ROS
Advantages			
Integration with MAVLink protocol	Versatile autopilot system	Real-time capabilities	Trajectory tracking
Versatile applications	Adaptive control	POSIX compliance	Security frameworks
Testing and validation	User-friendly interface	Integration with UAVCAN	Multi-UAV control
Open-source	Cost-effective	Cost-effective and flexible	Open source and community support
Disadvantages			
Hardware dependency	Limited hardware support	Hardware compatibility	Security vulnerabilities
Resource intensive	Complexity in customization	Development resources	Development overhead
Integration challenges	Development and tuning efforts	Integration complexity	Complex integration
Ongoing development	Potential for overhead	Community and support	Steep learning curve

2. Real-Time Operating System (RTOS)

In the previous chapter we often mentioned RTOS, let's talk about them in more detail. The main feature of RTOS lies in the ability to make precise time constraints of tasks execution compared to generally used operating systems. The precise time constraints are based on the task scheduler, which assigns priority to tasks ensuring in such way higher priority tasks execution. This enables the ability of operating system always provide the same output for a repeating input. RTOS have lower probability of a crash due to their focus on a narrower set of tasks. Their support multi-core and multi-threaded processors and Intel or PowerPC architectures. There are two types of ROTS – soft and hard. Hard ones are more consistent with the time needed to complete a task. Soft RTOS have more variability and can provide a late result, whereas this cannot occur with hard ones [38]. Let’s review some commonly used RTOSs in UAVs.

Kangunde et al. in their work [14] reported an overview about the real-time control use in UAV. According to their report, essential component for UAV real-time application is embedded RTOS. As the implementation of real-time control for UAVs requires a vell tasks definition. RTOS handles these tasks with the help of such features as scheduling, priority assignment and multi-threading. These features support real-time response of the UAV control system to feedback from IMU (Inertial Measurement Unit) and GPS (Global Positioning System).

The UAV control system subsequent applies the corresponding motor speeds to obtain the desired movement of the drone. Scheduling, priority assignment of tasks ensures that at any time tasks of critical importance will receive necessary computational resources of the microprocessor. For example, obstacle avoidance is a critical task. Multi-threading facilitates real-time response of the drone. It enables such tasks as path-planning, parallel control implementation to run, feedback for position and orientation.

We have already discussed in the previous chapter the typical representatives of such embedded operating systems, these are PX4 Autopilot and NuttX. However, besides them there are distinguished and used such operating systems as FreeRTOS, MicroPython and ChibiOS/RT [31–33].

2.1. FreeRTOS

FreeRTOS is a popular open-source real-time operating system kernel for embedded devices, including UAVs. It offers a robust and scalable platform for developing applications requiring real-time responsiveness and multitasking capabilities. It possesses such characteristics as: real-time capabilities; task management; resource management; low overhead; portability; community support; open-source license. Successful application of FreeRTOS in adaptive control of flapping-wing MAV, was reported by Mou et al. in paper [20]. The embedded software was completely developed on the FreeRTOS platform. The flight results show that such MAV could maintain hovering flight and ensured the convergence of the adaptive parameters even in situation when the unilateral thrust of the MAV was not enough (due to the manufacturing errors). According to survey conducted by Ebeid et al. [3] FreeRTOS is also a good software basis for UAV's open-source flight controller platforms, used in academic research. As Ebeid reported, a hardware of FPGA (field-programmable gate array) based platforms enables the use of algorithms of deep neural network and computer vision. It worked on FreeRTOS that contained a built-in multi-task scheduling and on ROS for hardware resources management and intelligent algorithms. Overall, due to its real-time capabilities, resource management features, portability and community support, FreeRTOS is a popular choice for UAV developers seeking to build responsive and efficient flight control systems and mission-critical applications. In summary, FreeRTOS offers strong real-time performance, efficient task and resource management, and portability, making it a popular choice for embedded applications, including UAV flight control systems.

2.2. MicroPython

MicroPython is a lean and efficient implementation of the Python 3 programming language that is optimized to run on microcontrollers and embedded systems. It brings the simplicity and flexibility of Python to the world of embedded development, including applications in UAVs. Key features of MicroPython are: Python syntax; interactive development; hardware abstraction layer (HAL); extensibility; low footprint; open-source community. MicroPython was successfully applied in the development of an interface that enables the connection of different type environmental parameters measurement devices directly to UAV, as reported by Formanek et al. in work [5]. In particular, it was used for the development and implementation of a control algorithm, to evaluate and define the level of air quality in the monitored space. In addition, a MicroPython environment was used for hardware in UAV for collision avoidance measures as reported by Minucci et al. in paper [19]. According to the experimental results UAV could safely execute the avoidance collision maneuver at a distances up to 900 m. Due to its simplicity, flexibility and extensibility, MicroPython is well-suited for prototyping, development, and deployment in UAV systems where rapid iteration and reliable performance are essential. In general, MicroPython offers a user-friendly, flexible, and efficient solution for embedded development with a focus on rapid prototyping and ease of use, thanks to its Python-based syntax and interactive features.

2.3. ChibiOS/RT

ChibiOS/RT is a compact and efficient RTOS designed for embedded systems, including those used in unmanned aerial vehicles. It provides a lightweight and flexible framework for developing embedded applications requiring real-time responsiveness. Main characteristics of ChibiOS/RT are: real-time capabilities; small footprint; preemptive multithreading; modularity; hardware abstraction layer; low power consumption; open-source license. Zhang et al. in paper [28] presented an analysis and comparison of ChibiOS/RT with NuttX. In accordance to their report both OS are good candidates for drones based on embedded systems, but ChibiOS/RT exceeds NuttX, due to the fact that NuttX has a vital defect in the implementation of priority inheritance. ChibiOS/RT also is a good choice for adaptive estimation of position, attitude and parameters of any type UAV. As reported by Fresk et al. in work [6], UAV's OS based on a ChibiOS/RT kernel, handled the communication with the hardware, data processing synchronization and the mutual exclusions for the communication buses. In result, the complexity of inertial navigation system was sufficiently reduced. ChibiOS/RT provides a robust and scalable platform for developing real-time embedded applications, making it a popular choice for UAVs. In summary, ChibiOS/RT is a compact and efficient real-time operating system that excels in providing real-time responsiveness, low power consumption, and modularity, making it well-suited for embedded applications and UAV systems. Its advantages include a small footprint, preemptive multithreading, and a robust HAL (hardware abstraction layer).

A comparative analysis of FreeRTOS, MicroPython, and ChibiOS/RT RTOS, is presented in Table 2.

Table 2. Advantages and disadvantages of the FreeRTOS, MicroPython and ChibiOS/RT RTOS

FreeRTOS	MicroPython	ChibiOS/RT
	Advantages	
Task and resource management	Interactive development	Preemptive multithreading
Portability	Hardware abstraction layer	Hardware abstraction layer
Successful applications	Successful applications	Modularity
Open-source license	Open-source community	Open-source license
	Disadvantages	
Limited built-in features	Limited libraries	Limited ecosystem
Scalability constraints	Real-Time constraints	Feature trade-offs
Integration complexity	Hardware dependency	Integration challenges
Learning curve	Learning curve	Learning curve

3. Cloud Operating Systems (Cloud OS)

Cloud OS for UAVs offer a scalable, flexible and collaborative platform for managing and controlling UAVs missions. By leveraging cloud computing technologies and services, these systems enable efficient operation, data management and decision-making for UAV operations in diverse applications and industries. Main features of Cloud OS are: remote control and monitoring; scalability and flexibility; data storage and analysis; collaboration and coordination; security and compliance; integration with IoT and edge computing; APIs and third-party integrations.

Sobhy et al. in paper [24] reported good results of the UAV based cloud computing performance by using Windows and Linux. Two OSs were used in comparison to each other in order to select the more efficient among them. The choice of the best OS was based on the calculating values of the throughput (bits/Sec). Experiment included six UAVs every two of which were connected to one GCS with the whole topology wirelessly connected to three servers, part of a private cloud. Four different scenarios where tested. In accordance with the results, better routing is obtained when using Linux OS with multi-processors. But in the same time, Windows with single processor showed higher throughput than single processor Linux at the expense of high delay and losses.

Incorporating cloud feature into UAV system gives one the possibility to handle big data generated by a wide range of external sensors equipped by drones. That is, to apply a cloud-based UAV system with computing capabilities of terrestrial cloud. Such approach was proposed by Luo et al. in paper [17]. According to given approach, a cloud-based UAV system with cloud computing capability for the multi-UAV systems to improve the scalability and flexibility was proposed. Such system consists of some numbers UAVs as a front-end witch acquires data from sensors, access network and general static cloud (GSC) as the back-end (entering, database, processing and output servers). The GSC permanently supplies on-demand computing services for the UAVs. The control process was considered as network control system in which the communication among different devices (sensors, controllers) and operators was conducted through a shared communication network. According to simulation results based on OPNET network simulator, the UAV front-end and GSC provided the on-demand services for all the UAVs in each cloudlet. When the maximum value of the eigenvalues of subsystems are small, the average packet arrival rate can achieve the maximum value. The increase of the number drones prolongs the response time of the UAV front-end and those of the whole system in general.

Cloud OS is a good choice for real-time flight monitoring and management for UAVs. For instance, in work [11] a cloud-based web application with real-time support for UAVs is presented. Given system automatically adjust flight path of UAV and prevents potential collisions, simultaneously alerting user about the changes. It was possible due to the development and implementation of a special UAV tracker which in accordance to its architecture, connects client with cloud server. Each UAV was also equipped with sensors like GPS, ultrasonic sensor, visual position cameras and sensors that provide status reports from the UAV's internal components (motor, battery, etc.). All flights were visualized due to the web-application dynamic map. System was tested through the computer simulation. According to the results, for a set of 20 connected UAVs the normal time of mission completion without using the redirected flight is 900 s but with it is 1075 s. The redirection algorithm adds a total of 175 s but prevents the collision. The collision likelihood increases with the increase of number of active UAVs.

Thus, cloud computing enhances the capabilities and efficiency by providing powerful computing resources, real-time analysis, remote management, scalability, connectivity, and collaborative capabilities [21]. Nevertheless, there are issues concerning the data security, privacy concerns, latency issues, and reliance on internet connection. Addressing such challenges is crucial for realizing the full potential of cloud-operated drones.

4. Conclusions

Operating systems serve as the backbone of UAVs, managing hardware and facilitating essential functions such as flight control, sensor integration, communication, autonomous operation, battery management, and payload control. They play a vital role in UAV functionality, offering tailored solutions for embedded, real-time, and cloud-based operations. Since at given moment there is no universal software that accounts for all the features and tasks performed by drones, the first and most important task will always be to make the optimal choice. The results obtained in the given research allow one to make such a choice between operating systems for drones, mainly depending on the task they need to perform. Additionally, this choice will also strongly depend on specific UAV requirements, including performance, flexibility, and scalability, to ensure safe and efficient flight operations in diverse environments.

Acknowledgment

All authors have read and approved the final manuscript. Authors state that there is no conflict of interest of any kind. The work is supported by the National Research Foundation of Ukraine project No 2023.04/0150 "Development of mobile radiowave source positioning and power measuring and visualizing system".

References

[1] Allouch A. et al.: MAVSec: Securing the MAVLink Protocol for Ardupilot/PX4 Unmanned Aerial Systems. 15th International Wireless Communications & Mobile Computing Conference (IWCMC), IEEE, Tangier, Morocco, 2019, 621–628 [https://doi.org/10.1109/IWCMC.2019.8766667].

[2] Baldi S. et al.: ArduPilot-Based Adaptive Autopilot: Architecture and Software-in-the-Loop Experiments. IEEE Transactions on Aerospace and Electronic Systems 58(5), 2022, 4473–4485 [https://doi.org/10.1109/TAES.2022.3162179].

[3] Ebeid E., Skriver M., Jin J.: A Survey on Open-Source Flight Control Platforms of Unmanned Aerial Vehicles. Euromicro Conference on Digital System Design (DSD), IEEE, Vienna, Austria, 2017, 396–402 [https://doi.org/10.1109/DSD.2017.30].

[4] Farabi M. R. A., Sintawati A.: Flood Early Warning System at Jakarta Dam Using Internet of Things (IoT)-Based Real-Time Fishbone Method to Support Industrial Revolution 4.0. Journal of Soft Computing Explorations 5(2), 2024, 99–106 [https://doi.org/10.52465/joscex.v5i2.293].

[5] Formanek L. et al.: Prototype for Measuring and Predicting Air Quality Using UAVs. EDULEARN23 Proceedings, IATED Academy, Palma, Spain, 2023, 6810–6814 [https://doi.org/10.21125/edulearn.2023.1794].

[6] Fresk E., Nikolakopoulos G., Gustafsson T.: A Generalized Reduced-Complexity Inertial Navigation System for Unmanned Aerial Vehicles. IEEE Transactions on Control Systems Technology 25(1), 2017, 192–207 [https://doi.org/10.1109/TCST.2016.2542022].

[7] García J., Molina J. M.: Simulation in Real Conditions of Navigation and Obstacle Avoidance with PX4/Gazebo Platform. Personal and Ubiquitous Computing 26, 2022, 1171–1191 [https://doi.org/10.1007/s00779-019-01356-4].

[8] Gill R., D'Andrea R.: An Annular Wing VTOL UAV: Flight Dynamics and Control. Drones 4(2), 2020, 14 [https://doi.org/10.3390/drones4020014].

[9] Grogan S., Pellerin R., Gamache M.: The Use of Unmanned Aerial Vehicles and Drones in Search and Rescue Operations–A Survey. Conference PROLOG, Hull, UK, 2018, 1–13.

[10] Hari Shankar R. L. et al.: Application of UAV for Pest, Weeds and Disease Detection Using Open Computer Vision. International Conference on Smart Systems and Inventive Technology (ICSSIT), IEEE, Tirunelveli, India, 2018, 287–292 [https://doi.org/10.1109/ICSSIT.2018.8748404].

[11] Itkin M., Kim M., Park Y.: Development of Cloud-Based UAV Monitoring and Management System. Sensors 16(11), 2016, 1913 [https://doi.org/10.3390/s16111913].

[12] Jing Y. et al.: PX4 Simulation Results of a Quadcopter with a Disturbance-Observer-Based and PSO-Optimized Sliding Mode Surface Controller. Drones 6(9), 2022, 261 [https://doi.org/10.3390/drones6090261].
[13] Kamel M. et al.: Model Predictive Control for Trajectory Tracking of Unmanned Aerial Vehicles Using Robot Operating System. Koubaa A. (ed.): Robot Operating System (ROS). Springer, Cham 2017, 3–39 [https://doi.org/10.1007/978-3-319-54927-9_1].
[14] Kangunde V., Jamisola R. S., Theophilus E. K.: A Review on Drones Controlled in Real-Time. International Journal of Dynamics and Control 9, 2021, 1832–1846 [https://doi.org/10.1007/s40435-020-00737-5].
[15] Lamping A. P. et al.: Multi-UAV Control and Supervision with ROS. Aviation Technology, Integration, and Operations Conference, American Institute of Aeronautics and Astronautics, Atlanta, Georgia, 2018, 4245 [https://doi.org/10.2514/6.2018-4245].
[16] Lee H. et al.: A Robot Operating System Framework for Secure UAV Communications. Sensors 21(4), 2021, 1369 [https://doi.org/10.3390/s21041369].
[17] Luo F. et al.: Stability of Cloud-Based UAV Systems Supporting Big Data Acquisition and Processing. IEEE Transactions on Cloud Computing 7(3), 2019, 866–877 [https://doi.org/10.1109/TCC.2017.2696529].
[18] Luo Z., Xiang X., Zhang Q.: Autopilot System of Remotely Operated Vehicle Based on Ardupilot. Yu H. et al. (eds.): Intelligent Robotics and Applications. Springer, Cham 2019, 206–217 [https://doi.org/10.1007/978-3-030-27535-8_19].
[19] Minucci F., Vinogradov E., Pollin S.: Avoiding Collisions at Any (Low) Cost: ADS-B Like Position Broadcast for UAVs. IEEE Access 8, 2020, 121843–121857 [https://doi.org/10.1109/ACCESS.2020.3007315].
[20] Mou J. et al.: Adaptive Control of Flapping-Wing Micro Aerial Vehicle with Coupled Dynamics and Unknown Model Parameters. Applied Sciences 12(18), 2022, 9104 [https://doi.org/10.3390/app12189104].
[21] Pandian A. P.: A Review on Future Challenges and Concerns Associated with an Internet of Things Based Automatic Health Monitoring System. Journal of Electrical Engineering and Automation 3(2), 2021, 92–109 [https://doi.org/10.36548/jeea.2021.2.003].
[22] Ravi N., El-Sharkawy M.: Integration of UAVs with Real-Time Operating Systems Using UAVCAN. 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), IEEE, New York, USA, 2019, 600–605 [https://doi.org/10.1109/UEMCON47517.2019.8993011].
[23] Silberschatz A., Galvin P. B., Gagne G.: Operating System Concepts. 10th ed. John Wiley & Sons, 2018.
[24] Sobhy A. R. et al.: UAV Cloud Operating System. 5th International Conference of Engineering Against Failure (ICEAF-V 2018), MATEC Web of Conferences, Chios, Greece, 2018, 05011 [https://doi.org/10.1051/matecconf/201818805011].
[25] Sørensen L. Y., Jacobsen L. T., Hansen J. P.: Low Cost and Flexible UAV Deployment of Sensors. Sensors 17(1), 2017, 154 [https://doi.org/10.3390/s17010154].
[26] Sushma R., Kumar J. S.: Dynamic Vehicle Modelling and Controlling Techniques for Autonomous Vehicle Systems. Journal of Electrical Engineering and Automation 4(4), 2023, 307–315 [https://doi.org/10.36548/jeea.2022.4.007].
[27] Tanenbaum A. S., Bos H.: Modern Operating Systems. 5th ed. Pearson, 2023.
[28] Zhang M. et al.: Which Is the Best Real-Time Operating System for Drones? Evaluation of the Real-Time Characteristics of NuttX and ChibiOS. International Conference on Unmanned Aircraft Systems (ICUAS), IEEE, Athens, Greece, 2021, 582–590 [https://doi.org/10.1109/ICUAS51884.2021.9476878].
[29] ArduPilot Documentation. Ardupilot [https://ardupilot.org/ardupilot/]. (Available:6 Feb. 2024).
[30] Ardupilot. Ardupilot [https://ardupilot.org/] (available: 6 Feb. 2024).
[31] ChibiOS/RT. ChibiOS [https://www.chibios.org/dokuwiki/doku.php] (available: 12 Feb. 2024).
[32] FreeRTOS. FreeRTOS [https://www.freertos.org/] (available: 12 Feb. 2024).
[33] MicroPython. MicroPython [https://micropython.org/] (available: 12 Feb. 2024).
[34] Nutt G.: NuttX Operating System User's Manual. Apache NuttX [https://cwiki.apache.org/confluence/display/NUTTX/Nuttx] (available: 6 Feb. 2024).
[35] NuttX. Apache NuttX [https://nuttx.apache.org/] (available: 6 Feb. 2024).
[36] PX4 Autopilot User Guide. PX4 [https://docs.px4.io/main/en/] (available: 6 Feb. 2024).
[37] PX4 Autopilot. PX4 [https://px4.io/] (available: 6 Feb. 2024).
[38] Real-Time Operating Systems (RTOS). Unmanned Systems Technology [https://www.unmannedsystemstechnology.com/expo/real-time-operating-systems/] (available: 8 Feb. 2024).
[39] Robot Operating System. ROS [https://www.ros.org/] (available: 6 Feb. 2024).
[40] ROS (Robot Operating System) Documentation. ROS Wiki [https://wiki.ros.org/Documentation] (Available:6 Feb. 2024).

Ph.D. Viktor Ivashko
e-mail: v.ivashko@chnu.edu.ua

Yuriy Fedkovych Chernivtsi National University. Ph.D. (Physics and Mathematics), assistant professor, Department of Computer Sciences.
Research interests: Computer modeling of stochastic processes in physical systems.
Author of more than 18 publications.

https://orcid.org/0000-0002-9339-4648

Ph.D. Serhii Haliuk
e-mail: s.haliuk@chnu.edu.ua

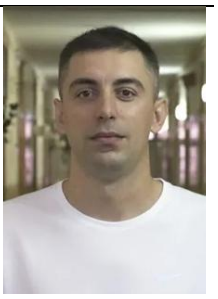
Yuriy Fedkovych Chernivtsi National University. Ph.D. (Engineering), professor, Department of Radio Engineering and Information Security.
Research interests: Radiotechnical devices and telecommunications means. Deterministic chaos for communication systems and cryptography. Generation of pseudo-random sequences based on chaotic systems. Security of cryptographic algorithms based on chaos.
Author of more than 60 publications.

https://orcid.org/0000-0003-3836-2675

Ph.D. Oleh Krulikovskyi
e-mail: o.krulikovskyi@chnu.edu.ua

Yuriy Fedkovych Chernivtsi National University. Ph.D. (Engineering), professor, Department of Radio Engineering and Information Security.
Research interests: RFSoC FPGA based systems and systems programming.
Author of more than 25 publications.

https://orcid.org/0000-0001-5995-6857



Prof. Andrii Samila
e-mail: a.samila@chnu.edu.ua

Yuriy Fedkovych Chernivtsi National University. D.Sc. (Engineering), professor, Department of Radio Engineering and Information Security.
Research interests: IoT, Microelectronics & Electronic Packaging, Signal Processing, Computer Hardware Design, Robotics, High Energy & Nuclear Physics.
Author of nearly 200 publications in this research area.

https://orcid.org/0000-0001-8279-9116