Empowering Efficient Drone Monitoring with Low-Latency Edge-Cloud Continuum Platforms

Loris Belcastro*, Cristian Cosentino*, Fabrizio Marozzo*, Aleandro Presta*, and Paolo Trunfio*

* University of Calabria, Rende, Italy
*{lbelcastro, ccosentino, fmarozzo, aleandro.presta, trunfio}@dimes.unical.it

***Abstract*—Drones used for activities such as environmental monitoring and infrastructure inspection generate vast amounts of data, requiring dedicated infrastructure for efficient management. While the cloud is a widely adopted solution, it often faces limitations such as latency, bandwidth constraints, and scalability challenges. To this scope, this paper presents a novel framework that leverages edge-cloud continuum platforms to overcome these issues. By combining the immediacy of edge computing with the computational power of the cloud, the framework processes data close to its source for real-time responsiveness and efficiently distributes tasks across multiple infrastructure layers, from edge devices to regional data centers and centralized clouds. This hybrid approach enhances scalability, efficiency, and responsiveness, addressing the demands of modern monitoring systems. The paper also addresses the lack of standardized protocols in edge-cloud configurations, a key obstacle to seamless interoperability. The proposed framework supports developers in designing and deploying applications across the edge-cloud continuum in a platform-independent manner, optimizing deployment configurations and services to meet strict quality of service (QoS) requirements. A case study on fire monitoring validates the framework, demonstrating substantial improvements in latency and scalability for critical applications such as disaster management and environmental conservation. By enabling scalable, adaptable, and cross-platform applications, the framework provides a robust solution for the complex needs of real-time, mission-critical scenarios.**

***Index Terms*—Edge-cloud continuum, Service composition, Abstract design, Drone monitoring, Requirements analysis, Platform interoperability**

I. Introduction

The rapid expansion of Internet of Things (IoT) devices is revolutionizing data collection, enabling the efficient acquisition of information from hard-to-reach locations (e.g., disaster-affected areas, agricultural fields, large infrastructures), thereby creating unprecedented opportunities in real-time monitoring. In this context, drones represent an innovative solution for monitoring critical areas, utilizing advanced sensors and IoT connectivity to detect environmental parameters, monitor critical infrastructures, identify anomalies, and transmit data continuously for immediate analysis or archiving. However, the volume of data to be processed and the speed at which it is generated create significant challenges for real-time processing and latency-sensitive applications (e.g., fire detection or flood monitoring). These challenges are further intensified by the increasing number of IoT devices, underscoring the critical need for efficient data collection, transmission, and processing solutions. [3], [21].

In response, the edge-cloud continuum has emerged as a computing model that processes data closer to its source in real-time, while using the cloud's vast computational resources for demanding tasks [5]. However, implementing edge-cloud continuum infrastructures involves substantial economic investment and a shift from traditional client-server architectures to more complex multi-layered models. These models integrate centralized data centers, regional facilities, 5G wireless stations, on-premise servers, and edge devices, working together to provide low-latency and highly responsive services. While major cloud providers like Amazon, Google, and Microsoft are extending their platforms to support edge computing, the absence of universal standards remains a critical barrier, hindering interoperability and limiting the seamless integration of distributed applications. Some open-source solutions, such as OpenStack[1] and OpenNebula[2], are advancing to provide versatile and accessible frameworks for supporting the edge-cloud continuum, but the lack of shared standards limits interoperability between different providers and forces developers to invest additional time and resources into customizing solutions.

To address these interoperability challenges, this paper introduces a new platform-independent modeling framework that enables developers to design and deploy scalable applications across the edge-cloud continuum with greater flexibility and efficiency. Specifically, the framework allows developers to define applications as compositions of abstract services that are independent of both the deployment layer (e.g., edge or cloud) and specific cloud providers, such as Amazon Web Services (AWS), Google Cloud, and Microsoft Azure. The framework facilitates the identification of optimal deployment

This work was supported by the research project "INSIDER: INtelligent ServIce Deployment for advanced cloud-Edge integRation" granted by the Italian Ministry of University and Research (MUR) within the PRIN 2022 program and European Union - Next Generation EU (grant n. 2022WWSCRR, CUP H53D23003670006) and by the European Union under the Italian National Recovery and Resilience Plan (NRRP) of NextGenerationEU, partnership on "Telecommunications of the Future" (PE00000001 - program "RESTART"). We also acknowledge financial support from "National Centre for HPC, Big Data and Quantum Computing", CN00000013 - CUP H23C22000360005

[1]https://www.openstack.org/
[2]https://opennebula.io

configurations and the selection of concrete services to deliver the required functionality, optimizing application deployment in accordance with the constraints and quality of service (QoS) requirements.

We assessed our approach through a use case on drone-based monitoring systems, demonstrating its potential to enhance disaster management by reducing latency and improving scalability in real-world scenarios. These systems are applicable in various contexts, including environmental monitoring, disaster management, and resource tracking, all of which depend on real-time data collection and processing. For instance, drones monitoring environmental conditions can quickly process data to respond to emergencies such as fires, floods, or wildlife tracking. Specifically, we illustrate how drone data generated during fire monitoring tasks can be processed and analyzed in real-time using machine learning algorithms within the compute continuum. The abstract modeling approach of our framework ensures seamless integration across the edge-cloud continuum, optimizing resource allocation for real-time processing while remaining agnostic to specific service providers. Once the application is modeled, the framework allows to identify suitable deployment configurations for specific service providers. In our case, we illustrate the implementation using AWS, but at the same time emphasize that the platform-independent modeling approach enables development across other platforms.

The remainder of the paper is structured as follows. Section II provides a brief review of related works in the domain of modeling frameworks for distributed applications. Section III discusses the reference edge-cloud continuum architecture and explores its implications for deploying distributed applications. Section IV presents the proposed framework for modeling edge-cloud continuum applications. Section V presents the use case used to evaluate the proposed modeling approach. Finally, Section VI presents our conclusions and outlines avenues for future research.

II. Related work

The explosion of IoT devices has driven significant research into overcoming latency and scalability challenges in data processing within the edge-cloud continuum. This paradigm must also address the heterogeneity of devices, which introduces several challenges, such as managing diverse computational capabilities, ensuring interoperability, and optimizing resource allocation across distributed infrastructures.

One notable application area within the edge-cloud continuum is drone-based systems, which leverage advanced sensors and AI to transform environmental monitoring and disaster management by enabling real-time detection and rapid emergency response. Different systems have been developed to address these needs, using both lightweight deep learning [14], [19], [20], [22] and machine learning models [9], [16], [23] for fire monitoring and real-time forecasting.

Edge computing plays a pivotal role in drone-based applications by enabling real-time data processing and reducing latency. For instance, Callegaro and Levorato [8] applied deep reinforcement learning to optimize task processing between drones and edge servers, while Alam et al. [1] presented a system for abnormal event detection using low-cost drones. Beyond disaster management, the edge-cloud continuum represents an effective solution for enhancing urban mobility [10] and smart city applications [4]–[6], [15].

Designing applications that fully leverage the complex edge-cloud continuum architecture presents significant challenges for developers, particularly in determining which services to use and how to allocate them across the various layers to meet application requirements effectively. To tackle these challenges, studies have proposed innovative modeling frameworks within the edge-cloud continuum paradigm, such ITEMa [12], SEM [11], and IADev [18]. While effective in designing IoT-based ecosystems and smart environments, these frameworks lack explicit tools for modeling communication workflows and QoS constraints, which are critical for optimizing edge-cloud applications. Similarly, MDSC [2] is a framework designed to facilitate the modeling of distributed stream-based applications. However, it does not account for service-specific features, such as internal configurations and libraries, and lacks mechanisms to guide the optimal placement of processing units across the multiple layers of the edge-cloud continuum. In contrast to these works, our framework offers a versatile tool for designing and developing edge-cloud applications. It addresses the limitations of existing approaches by enabling clear separation of concerns, identification of QoS requirements, and selection of optimal providers and services for deployment, thereby enhancing the development and optimization process. Moreover, existing frameworks lack platform abstraction, leading to platform-dependent solutions that face persistent challenges in ensuring scalability, interoperability, and seamless integration across heterogeneous infrastructures [7].

III. Edge-Cloud Continuum Architecture

The edge-cloud continuum architecture consists of several layers, each designed to optimize data processing and analysis based on proximity to the data source and the extent of processing required. Figure 1 illustrates the edge-cloud continuum architecture, where each layer plays a distinct role in data processing and analysis. These layers are classified based on the proximity of computational resources to the data source and the extent of processing performed at each level [13].

The ***cloud*** layer comprises centralized computing infrastructures located in remote data centers, offering on-demand resources such as virtual machines and storage over the Internet. This layer is ideal for intensive data processing due to its scalability and cost efficiency. Positioned closer to the cloud is the ***near edge***, which consists of computing resources like mini data centers or regional clouds. These facilities act as intermediaries, enabling more immediate data processing and reducing latency for devices located several hundred kilometers away. Even closer to the devices is the ***far edge***, where computing nodes are deployed near mobile phone towers or industrial facilities. This proximity ensures faster

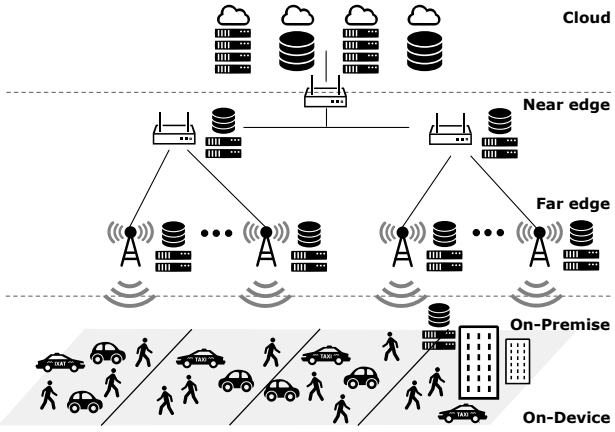

Fig. 1. Extended edge-cloud architecture

data aggregation and preliminary processing, enhancing the responsiveness of applications. The ***on-premise*** layer includes data processing nodes located within local facilities or at end-user sites, such as farms or stadiums. This setup provides a balance between proximity and autonomy, delivering fast and stable connectivity to business systems. Closest to the data source is the ***on-device*** layer, covering edge devices like IoT sensors and smartphones. These devices generate data and perform local processing, reducing latency and bandwidth usage, which is a critical feature for real-time applications like autonomous vehicles and smart industries.

This distributed architecture enhances modern computing systems by addressing latency, privacy, and data localization requirements, enabling real-time responsiveness and scalability in applications such as drone-based monitoring.

Amazon Web Services (AWS) stands out among major cloud providers as offering the most advanced services for the edge-cloud continuum. This leadership is attributed to Amazon's extensive and highly distributed infrastructure, which enables seamless integration and support across all levels of the edge-cloud paradigm. AWS provides a comprehensive suite of tools and technologies, extending its computing capabilities from the central cloud to the edge with efficiency and scalability. For example, AWS's global network, spanning over 33 geographic regions, supports seamless deployment across the edge-cloud continuum, providing tailored solutions for every layer. At the near edge level, *Local Zones* position services closer to densely populated areas and major IT hubs, enhancing performance and reducing latency. Moreover, AWS offers various solutions to bring cloud services closer to data sources, such as *Wavelength*, which integrates computing resources at the far edge within 5G networks, and *Outposts* and *IoT Greengrass*, which enable seamless integration, local storage, and processing at the on-premise and on-device levels, respectively. While AWS leads in this space, other providers are also investing heavily in building continuous architectures for the edge-cloud, complemented by open-source initiatives like OpenStack and OpenNebula, which promote flexible and community-driven solutions for distributed computing.

IV. Framework for Modeling Edge-Cloud Applications

The proposed framework introduces an innovative, platform-independent methodology to model edge-cloud applications, decoupling application design from implementation constraints to ensure adaptability and scalability [17]. In this context, an application is defined as a set of abstract services organized in a workflow, decoupled from the deployment layer (e.g., cloud or edge) and not bound to specific services offered by cloud providers like AWS, Microsoft Azure, or open-source platforms such as OpenStack and OpenNebula. These abstract services, encompassing data collection, transformation, and processing, are mapped onto a cloud service catalog to ensure optimal deployment configurations while satisfying QoS constraints. This mapping provides guidelines on how to implement these tasks using concrete services while ensuring compliance with functional constraints.

Figure 2 illustrates the framework, which integrates abstract application components and deployment configuration tools to streamline the implementation of edge-cloud applications. The main components of the proposed framework are described below. The ***Execution Infrastructure*** component models the execution infrastructures for both cloud and edge, considering the different technologies, hardware types, and network topologies through which they can be realized. Parameters such as virtual machine sizes, the number of devices, on-premise resources, and available execution layers (cloud, far edge, near edge, on-premise, or on-device) are used to define a model capable of representing a wide range of cloud/edge technologies and infrastructures. The ***Abstract Application*** component represents the application as a composition of abstract services with associated tasks and dependencies, modeling both functional and non-functional requirements among services. Services can run on different execution layers and may be adapted, partitioned, optimized, or compressed. This flexibility allows for tailoring the application to specific needs and resources. Moreover, the ***QoS Constraints*** component defines the constraints that the application must meet, such as latency, bandwidth usage, operational costs, persistence of data storage, scalability, energy consumption, and more. These constraints can be general, applying to the application as a whole, or specific, targeting particular services or tasks within the application. In addition, the ***Cloud Service Catalog*** includes service catalogs from various cloud providers (e.g., AWS, Microsoft Azure, and Google Cloud), which are mapped to the abstract services defined in the previous step to meet application requirements. It contains both quantitative information (e.g., latency, energy consumption) and qualitative information (e.g., data persistence, supported data formats, suitable execution layers) about each service, facilitating the mapping between abstract and concrete services and aiding in selecting the most appropriate services for implementation. Furthermore, the ***Deployment Configuration Generator*** builds valid deployment configurations for the

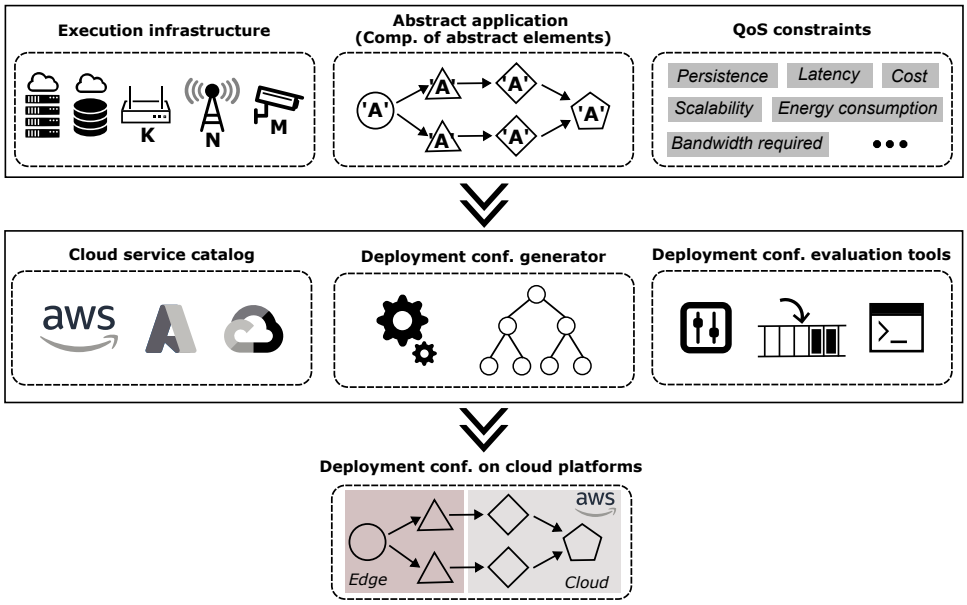

Fig. 2. An overview of the proposed modeling framework.

abstract application, given the desired QoS constraints, the real execution infrastructure, and the cloud service catalog. Specifically, it generates valid deployment configurations that consider the distribution of workloads across the different layers of the edge-cloud continuum architecture. Subsequently, the ***Deployment Configuration Evaluator*** assesses application deployment configurations by analyzing performance parameters and requirements. Linear programming systems, simulation tools, or advanced large language models automate this evaluation process, ensuring that the configurations align with performance and QoS requirements prior to implementation. Finally, the ***Deployment Configurations on Cloud Platforms*** component ensures seamless deployment, translating abstract workflows into concrete implementations that fully leverage the capabilities of the selected cloud platform.

V. Use Case: Prediction and Monitoring of Fires using a Drone Fleet

Wildfires pose a significant threat to environmental and public safety, requiring innovative solutions for early detection and efficient resource management. To validate our modeling approach, we present a use case focused on *fire risk assessment and forecasting* that integrates machine learning with edge-cloud technologies. Specifically, the focus is on predicting high-risk areas to optimize resource allocation and response strategies.

The proposed architecture leverages distributed computing resources across the edge-cloud continuum. Specifically, a fleet of drones, equipped with sensors and cameras, captures real-time video feeds and environmental data, supplemented by strategically placed ground sensors. The collected data undergoes preprocessing (cleaning, filtering, and transformation) before being fed into a machine learning model trained to predict areas at high risk of fires, thereby enabling efficient patrolling by drones.

Our framework allows applications to be defined in an abstract manner, without specifying layers or platform-specific services. This flexibility enables the application to be implemented in various ways. For example, in the *centralized approach*, drones send preprocessed data to the cloud, where a global machine learning model generates predictions and transmits them back to the drone manager to devise optimal patrolling strategies. In contrast, the *decentralized approach* leverages the edge-cloud continuum, emphasizing the importance of performing computations as close to the data source as possible. Drones rely on local machine learning models for low-latency predictions and transmit only essential updates or aggregated results to the cloud. This approach minimizes delays, reduces bandwidth usage, and optimizes computational efficiency by utilizing distributed resources effectively.

The proposed modeling approach is illustrated in Figure 3 as a workflow comprising four types of interconnected components: devices, networking, computation, and storage. This abstraction simplifies the design and deployment of scalable systems for real-time fire monitoring. In particular, devices capture and generate data, monitoring activities, events, and environmental conditions in real time. Among devices, drones serve as the main actors in the system. Equipped with sensors and cameras to monitor areas in order to detect fires autonomously, they utilize dedicated docking platforms for recharging their batteries, performing routine maintenance, and uploading collected data to ensure uninterrupted and efficient operation. Event monitors rely on ground-based sensors, such as temperature, smoke, and humidity sensors, strategically placed to detect fires in real time. Traffic monitors access street camera feeds near fire zones to identify vehicles and

pedestrians, aiding in source investigations. Official requests are intervention requests submitted by private citizens or public authorities through phone, web services, or mobile applications, validated by operators before processing. All these devices collaborate and work together, forming an integrated system that enhances the efficiency and effectiveness of fire detection and response.

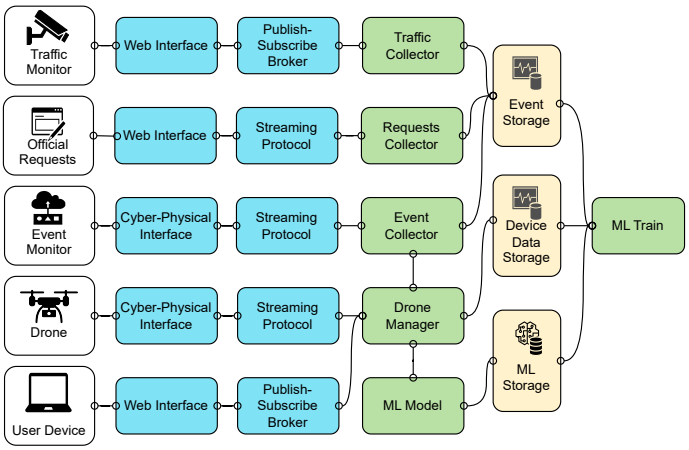

Fig. 3. Abstract workflow and corresponding services for fire risk assessment using drones. Nodes represent *devices* (white), *networking* (light blue), *computation* (green), and *storage* (yellow) elements.

Networking elements manage communication and data transfer between devices and computational elements. Cyber-physical interfaces connect physical devices like drones and sensors to the network, while publish-subscribe brokers route data to appropriate computational elements. Streaming protocols enable real-time data transmission for low-latency interactions, and web interfaces allow data retrieval from virtual components such as traffic monitors.

Computation elements process and analyze data, supporting both real-time monitoring and predictions. The event collector monitors data from ground sensors, adjusting drone patrol routes when anomalies such as abnormal temperatures or smoke are detected. The request collector processes validated intervention requests, which are integrated into the predictive model. The machine learning model predicts high-risk fire locations using preprocessed data, including drone mission logs, sensor readings, and official requests. Retraining of the model is performed periodically by the ML training component, incorporating new patterns and feedback while balancing the need for up-to-date predictions and real-time performance. The traffic collector gathers data from street cameras to identify potential causes of fires, enabling a comprehensive analysis.

Storage elements ensure reliable data persistence for both real-time processing and long-term analysis. ML storage preserves the machine learning model, storing its weights for retraining and predictions. Device data storage archives drone mission data, including video feeds, drone locations, and meteorological information. This historical data aids in refining predictions and improving model accuracy. Drones are equipped with advanced cameras, such as optical cameras for visual mapping, thermal cameras for detecting hotspots, and hyperspectral cameras for detailed spectral analysis.

The proposed modeling approach adopts a cloud-agnostic client-server architecture, allowing services to be flexibly allocated across devices, near-edge nodes, or cloud platforms, depending on QoS requirements. This cloud-agnostic design allows cross-platform deployment of abstract services, which can be implemented on various cloud and edge computing platforms. Requirements and QoS metrics are defined to ensure optimal system performance: devices require low latency and minimal energy consumption; networking components demand high bandwidth, low latency, and scalability; computation elements require fast response times, scalable resources, and energy-efficient processing; storage components need reliable and scalable data persistence for historical analysis and retraining. By meeting these requirements, the system validates the ability of the proposed framework to provide scalable, efficient, and real-time fire risk assessment, paving the way for broader applications in disaster management. Among all the platforms available, we chose AWS and decided to utilize its services available across all layers of the edge-cloud continuum.

A. Constraints and Optimization of Services in the Use Case

The proposed use case underscores the critical importance of selecting services that align with the stringent operational constraints of distributed applications, ensuring efficient fire prediction and monitoring. Since we chose to use AWS, Table I provides a summary of the AWS Service Catalog, mapping the functional and deployment capabilities of various services to the specific operational demands of the use case. Each entry in the table outlines key AWS products, categorized by functionality and deployment scope, providing a quick reference for aligning system requirements with available services. The *type* column categorizes each service based on its core functionality, grouping them into computation (⚙), networking (💬), and storage services (🗄). The *category* column further groups services by their functional roles, such as analytics, IoT, ML, and storage, reflecting their application scope. The *description* column highlights key functionalities, such as enabling real-time video analysis, monitoring IoT devices, or supporting machine learning workflows.

The table also outlines the compatibility of each service across different levels of the continuum, including *On-Device*, *On-Premise*, *Far-Edge*, *Near-Edge*, and *Cloud*. This provides insights into where and how these services can be deployed. In particular, to identify the services suitable for deployment at each layer, we utilized the corresponding *enabling services*: for *On-Device*, we selected from services compatible with *IoT Core* and *IoT Greengrass*; for *On-Premise*, services compatible with *Outposts*; for *Far-Edge*, services available through *Wavelength*; and for *Near-Edge*, services available in *Local Zones*.

For instance, *Kinesis Video Streams* is ideal for real-time video processing at the cloud level, with limited on-device support, making it suitable for detecting environmental anomalies.

ID	Name	Type	Vendor Category	Description	On-Device (IoT Core + IoT Greengrass)	On-Premise (Outposts)	Far-Edge (Wavelength)	Near-Edge (Local zones)	Cloud	...
1	Kinesis Video Streams	⚙	Analytics	Stream video to AWS for analytics and ML	✔	✘	✘	✘	✔	...
2	IoT Greengrass ML Inference	⚙	IoT	Machine Learning inference on devices	✔	✘	✘	✘	✘	...
3	EC2	⚙	Compute	Resizable cloud compute capacity	✘	✔	✔	✔	✔	...
4	DocumentDB	🗄	Database	Managed JSON document database	✘	✘	✘	✘	✔	...
5	Timestream	🗄	Database	Secure, scalable, and high-performance DBs	✘	✘	✘	✘	✔	...
6	Amplify	⚙	Front-End	Tools to build web and mobile apps	✘	✘	✘	✘	✔	...
7	SageMaker	⚙	ML	Prepare, train, and deploy ML models quickly	✘	✘	✔	✔	✔	...
8	Route 53	💬	Network	Reliable DNS routing for Internet apps	✘	✔	✔	✔	✔	...
9	S3	🗄	Storage	Scalable and secure object storage	✘	✔	✘	✘	✔	...
10	Glue	⚙	Analytics	Simple data integration service	✘	✘	✘	✘	✔	...
11	Lambda Function	⚙	Compute	Run code without provisioning or managing servers.	✔	✔	✘	✘	✔	...
12	Device Gateway	💬	IoT	Ultra-low-latency connectivity for mobile edge applications	✔	✔	✔	✔	✔	...

TABLE I
AWS SERVICE CATALOG. A SUBSET OF AWS SERVICES CATEGORIZED AS COMPUTATION (⚙), NETWORKING (💬), STORAGE (🗄). AVAILABILITY ON A SPECIFIC LAYER IS INDICATED AS AVAILABLE (✔) OR NOT AVAILABLE (✘).

Similarly, *SageMaker* supports the preparation, training, and deployment of machine learning models, effectively functioning both at the far edge and cloud levels to deliver real-time predictions. For storage, *S3* is optimized for managing massive historical datasets, while *Timestream* specializes in handling time-series data, both within the cloud.

Once we have defined all the services available on AWS, along with their categorization and description (Table I), the next step is to define the application constraints. To this end, we analyzed all the services used in the workflow shown in Figure 3, specifying the constraints for each, as detailed in Table II for a sample of services. These constraints are expressed as <*metric*: *value*> pairs. For instance, a metric could represent computing power, storage capabilities, or latency requirements, while values could be either categorical (e.g., moderate, high, low) or numeric. The table also highlights the constraints for some services involved in the case study considered: the Drone Manager requires high computing power and bandwidth to manage fleets of drones, processing and analyzing incoming video streams; intensive tasks, such as the machine learning service (ML Train and ML Storage), demand very high computing power and large storage capabilities; and the ML Model service performs inference on data coming from devices to predict potential fire risks, which are eventually communicated to the Drone Manager to update drone patrolling routes.

B. Implementation of the use case

After defining the execution infrastructure, the abstract application, and the QoS constraints, we used the service catalog to determine optimal configurations for deploying the application on a specific cloud platform. Additionally, linear optimization techniques or simulations can be applied, following the flow defined by the modeling framework proposed in this work. By adopting this strategy and leveraging AWS services, we created the appropriate deployment setup shown in Figure 4, which distributes the application across the layers of the edge-cloud continuum (device, far edge, and cloud).

At the device level, drones equipped with advanced sensors, ground-based monitors, and traffic cameras work together to collect data for real-time fire detection and response. We assumed that these devices were equipped with specific IoT hardware, such as the Raspberry Pi, which includes sensors for data collection, communication ports for transmitting data to servers for further processing, and low energy consumption.

To facilitate interaction between these IoT devices (e.g., drones) and servers located at the other layers, we utilize *AWS IoT Core*, a managed cloud service that enables the management of device connectivity and data transfer, and *IoT Greengrass*, which provides a local device gateway and enables the local execution of *Lambda* functions and to process and aggregate data before transmitting it to the nearest

Name	Type	Vendor Category	Constraints
Traffic Collector	⚙	Compute	Computing Power: *Moderate* Latency: *Moderate*
Event Collector	⚙	Compute	Computing Power: *Very Low* Latency: *Moderate*
Request Collector	⚙	Compute	Computing Power: *Low* Latency: *High*
Drone Manager	⚙	Compute	Computing Power: *Moderate-High* Bandwidth: *High*
Publish-Subscribe Broker	💬	Network	Latency: *Moderate* Bandwidth: *Moderate*
Streaming Protocol	💬	Network	Latency: *Moderate* Bandwidth: *High*
Web Interface	💬	Network	Latency: *Moderate* Bandwidth: *Moderate*
ML Storage	🗄	Storage	Storage: *Very High*
ML Train	⚙	ML	Computing Power: *Very High*
ML Model	⚙	ML	Computing Power: *Moderate* Latency: *Low-Moderate*

TABLE II
ABSTRACT REPRESENTATION OF SERVICES WITH CONSTRAINTS (CASE STUDY: FIRE RISK ASSESSMENT USING DRONES), CATEGORIZED AS COMPUTATION (⚙), NETWORKING (💬), AND STORAGE (🗄).

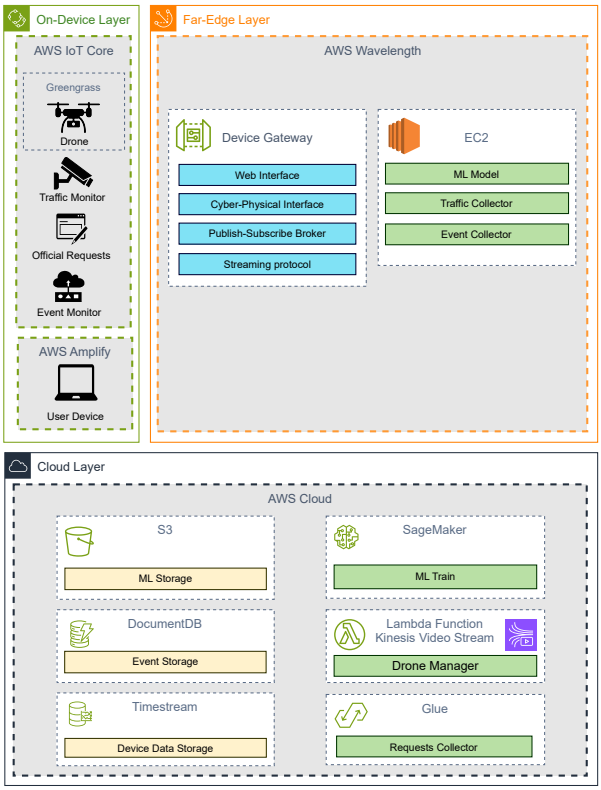

Fig. 4. Deployment of the abstract services of the workflow for fire risk assessment on AWS infrastructure.

server for further processing. At the far-edge, we leverage *Wavelength* to move computing resources closer to the data sources. *Wavelength* is a managed service that integrates AWS infrastructure directly into data centers managed by telecommunication operators, bringing compute and storage capabilities to the edge of telecoms networks. Here, we deploy services, such as Traffic and Event Collectors, dedicated to gathering data from diverse sources and systems distributed across a wide geographical area, which may belong to different organizations. At the cloud layer, *S3* is used to store models that were trained with *SageMaker* in the cloud. Such models are periodically deployed to relevant nodes within the *Wavelength* architecture after being trained on data specific to specific geographic locations. In addition, we use: *Timestream* to store data from devices that arrive at regular intervals; and *DocumentDB*, a noSQL compatible document database service, to store traffic/event/request data from external sources as JSON-like documents, collected via *Glue ETL* jobs. This configuration represents one of many possible combinations of AWS services, highlighting the flexibility of our abstract model to adapt to diverse project requirements.

VI. CONCLUSION

This paper presents an innovative modeling framework that empowers developers to design high-responsive, platform-independent applications for the edge-cloud continuum, addressing key challenges in flexibility, scalability, and service optimization. The framework introduces a set of abstractions to define services as workflows composed of computation, networking, and storage elements, tailored to meet specific application constraints and QoS metrics. This approach enables optimal service selection during deployment, enhancing scalability and adaptability across diverse cloud infrastructures. By abstracting application components, the framework supports efficient cross-platform deployment, ensuring effective service selection and adaptability to varying edge-cloud configurations. It facilitates the identification of appropriate execution platforms within the edge-cloud continuum and the selection of suitable services within those platforms.

Future research will enhance the modeling framework by providing more advanced tools for automated service selection and dynamic adaptation. This includes developing sophisticated mechanisms to define abstract service requirements and QoS criteria, enabling precise alignment with application demands. The goal is to automate the selection process, allowing users to effortlessly identify the best service provider and configuration for specific applications. This will be achieved through optimization or heuristic algorithms that identify the most effective service options and deployment strategies.

Additionally, machine learning techniques will be integrated to improve the selection process. By analyzing historical data, these techniques can predict service performance and enhance decision-making accuracy. A key focus will be the development of a dynamic adaptation mechanism to address real-time variations in service availability, workload, and user demands. Through continuous monitoring and analysis of service performance metrics, this mechanism will ensure that services consistently meet the required QoS standards.

REFERENCES

[1] Md Shahzad Alam, BV Natesha, TS Ashwin, and Ram Mohana Reddy Guddeti. Uav based cost-effective real-time abnormal event detection using edge computing. *Multimedia tools and Applications*, 78(24):35119–35134, 2019.

[2] Daniel Balouek-Thomert, Pedro Silva, Kevin Fauvel, Alexandru Costan, Gabriel Antoniu, and Manish Parashar. Mdsc: modelling distributed stream processing across the edge-to-cloud continuum. In *Proceedings of the 14th IEEE/ACM International Conference on Utility and Cloud Computing Companion*, UCC '21, New York, NY, USA, 2022. Association for Computing Machinery.

[3] Loris Belcastro, Riccardo Cantini, Fabrizio Marozzo, Alessio Orsino, Domenico Talia, and Paolo Trunfio. Programming big data analysis: Principles and solutions. *Journal of Big Data*, 9(4), 2022.

[4] Loris Belcastro, Fabrizio Marozzo, and Alessio Orsino. Hybrid edge/cloud solutions for supporting autonomous vehicles. In *Advances in Autonomous Vehicle Systems*. River Publishers, 2023.

[5] Loris Belcastro, Fabrizio Marozzo, Alessio Orsino, Domenico Talia, and Paolo Trunfio. Edge-cloud continuum solutions for urban mobility prediction and planning. *IEEE Access*, 11:38864–38874, 2023.

[6] Loris Belcastro, Fabrizio Marozzo, Alessio Orsino, Domenico Talia, and Paolo Trunfio. Using the compute continuum for data analysis: Edge-cloud integration for urban mobility. In *2023 31st Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP)*, pages 338–344. IEEE, 2023.

[7] Loris Belcastro, Fabrizio Marozzo, Aleandro Presta, Rosa Varchera, and Andrea Vinci. Developing platform-agnostic iiot applications in edge-cloud environments. In *International Conference on Industry 4.0 Smart Manufacturing 2024 (ISM 2024)*, 2024.

[8] Davide Callegaro and Marco Levorato. Optimal edge computing for infrastructure-assisted uav systems. *IEEE Transactions on Vehicular Technology*, 70(2):1782–1792, 2021.

[9] Riccardo Cantini, Cristian Cosentino, Fabrizio Marozzo, Domenico Talia, and Paolo Trunfio. Harnessing prompt-based large language models for disaster monitoring and automated reporting from social media feedback. *Online Social Networks and Media*, 45:100295, 2025.

[10] Eugenio Cesario, Fabrizio Marozzo, Domenico Talia, and Paolo Trunfio. Sma4td: A social media analysis methodology for trajectory discovery in large-scale events. *Online Social Networks and Media*, 3-4:49–62, 2017.

[11] Franco Cicirelli, Giancarlo Fortino, Antonio Guerrieri, G. Spezzano, and Andrea Vinci. Metamodeling of smart environments: from design to implementation. *Adv. Eng. Informatics*, 33:274–284, 2017.

[12] Franco Cicirelli, Antonio Guerrieri, Alessandro Mercuri, Giandomenico Spezzano, and Andrea Vinci. Itema: A methodological approach for cognitive edge computing iot ecosystems. *Future Gener. Comput. Syst.*, 92:189–197, 2019.

[13] European Commission. European industrial technology roadmap for the next generation cloud-edge offering. Technical report, European Commission, May 2021.

[14] Mostafa M Fouda, Sadman Sakib, Zubair Md Fadlullah, Nidal Nasser, and Mohsen Guizani. A lightweight hierarchical ai model for uav-enabled edge computing with forest-fire detection use-case. *IEEE Network*, 36(6):38–45, 2022.

[15] Dragi Kimovski, Narges Mehran, Christopher Emanuel Kerth, and Radu Prodan. Mobility-aware iot application placement in the cloud–edge continuum. *IEEE Transactions on Services Computing*, 15(6):3358–3371, 2021.

[16] Yizhou Li, Zilong Wang, and Xinyan Huang. Super real-time forecast of wildland fire spread by a dual-model deep learning method. *Journal of Environmental Informatics*, 43(1):65–79, 2024.

[17] Fabrizio Marozzo and Andrea Vinci. Design of platform-independent iot applications in the edge-cloud continuum. In *20th International Conference on Distributed Computing in Smart Systems and the Internet of Things (DCOSS-IoT)*, 2024.

[18] Wajid Rafique, Xuan Zhao, Shui Yu, Ibrar Yaqoob, Muhammad Imran, and Wanchun Dou. An application development framework for internet-of-things service orchestration. *IEEE Internet of Things Journal*, 7(5):4543–4556, 2020.

[19] Montaser NA Ramadan, Tasnim Basmaji, Abdalla Gad, Hasan Hamdan, Bekir Tevfik Akgün, Mohammed AH Ali, Mohammad Alkhedher, and Mohammed Ghazal. Towards early forest fire detection and prevention using ai-powered drones and the iot. *Internet of Things*, page 101248, 2024.

[20] Arthur Sabino, Luiz Nelson Lima, Carlos Brito, Leonel Feitosa, Marcos F Caetano, Priscila Solis Barreto, and Francisco Airton Silva. Forest fire monitoring system supported by unmanned aerial vehicles and edge computing: a performance evaluation using petri nets. *Cluster Computing*, pages 1–21, 2024.

[21] Domenico Talia, Paolo Trunfio, Fabrizio Marozzo, Loris Belcastro, Riccardo Cantini, and Alessio Orsino. *Programming Big Data Applications: Scalable Tools and Frameworks for Your Needs*. World Scientific, 2024. ISBN: 978-1-80061-504-5.

[22] Md Fahim Shahoriar Titu, Mahir Afser Pavel, Goh Kah Ong Michael, Hisham Babar, Umama Aman, and Riasat Khan. Real-time fire detection: Integrating lightweight deep learning models on drones with edge computing. *Drones*, 8(9):483, 2024.

[23] Tianhang Zhang, Zilong Wang, Ho Yin Wong, Wai Cheong Tam, Xinyan Huang, and Fu Xiao. Real-time forecast of compartment fire and flashover based on deep learning. *Fire Safety Journal*, 130:103579, 2022.