2023 IEEE Symposium on Computers and Communications (ISCC): 28th IEEE Symposium on Computers and Communications - 3rd International Workshop on Distributed Intelligent Systems

TEMA: Event Driven Serverless Workflows Platform for Natural Disaster Management

Christian Sicari[*][†], Alessio Catalfamo[*], Lorenzo Carnevale[*], Antonino Galletta[*], Daniel Balouek-Thomert[†][‡], Manish Parashar[†], and Massimo Villari[*],
[*]Department of Mathematics, Physics, Informatic and Eart Science, University of Messina, Messina Via Stagno D'Alcontres 98166, Italy
[†]Scientific Computing and Imaging Institute, University of Utah, USA
[‡]IMT Atlantique, Nantes Université, École Centrale Nantes
CNRS, Inria, LS2N, UMR 6004, F-44000 Nantes, France

***Abstract*—TEMA project is a Horizon Europe funded project that aims at addressing Natural Disaster Management by the use of sophisticated Cloud-Edge Continuum infrastructures by means of data analysis algorithms wrapped in Serverless functions deployed on a distributed infrastructure according to a Federated Learning scheduler that constantly monitors the infrastructure in search of the best way to satisfy required QoS constraints. In this paper, we discuss the advantages of Serverless workflow and how they can be used and monitored to natively trigger complex algorithm pipelines in the continuum, dynamically placing and relocating them taking into account incoming IoT data, QoS constraints, and the current status of the continuum infrastructure. Therefore we presented the Urgent Function Enabler (UFE) platform, a fully distributed architecture able to define, spread, and manage FaaS functions, using local IOT data managed using the Fiware ecosystem and a computing infrastructure composed of mobile and stable nodes.**

***Index Terms*—faas, event-driven workflows, natural disaster management, ambient intelligence, ndm**

I. Introduction

In recent years we have assisted in the rise of the Internet Of Things (IoT) as one of the fastest-growing sources of data. The IoT has been applied in many different fields, like smart cities industries (Industry 4.0) and especially in environmental monitoring, with the purpose to keep under control different parameters, such as noise, air quality or in more sophisticated infrastructures, the rise of some kind of environmental disasters like fires, earthquakes or floods (Natural Disaster Management, **NDM**).

Independently by the application use case, IoT infrastructures are just data collectors, and they are not in charge of permanently storing, analyzing and reacting to the data. When those latter actions are critical or time-sensitive we fall in the area of Urgent Computing, which refers to the use of high-performance computing (HPC) resources to address critical and time-sensitive problems that require rapid response. These problems can include urgent scientific research, emergency response planning, or real-time decision making in high-pressure situations.

The way followed to build HPC systems changed over the years, moving from the adoption of high-performance cloud infrastructures, towards the use of distributed cloud edge cooperative infrastructures called Cloud-Edge Continuum or just *Continuum*.

The characteristics of a continuum infrastructure perfectly fit the needs of many time-critical scenarios, but, some challenging issues are still open, especially when applied to dynamic use cases like environment monitoring.

The TEMA project is a Horizon Europe (HE) project that addresses NDM needs by developing automated means for precise semantic area mapping and phenomenon evolution predictions for NDM in (near-)real-time. Potential end-users are mainly Civil Protection Agencies (CPAs), but also First Responders (FRs). To address this problem, TEMA proposes to design and develop an efficient continuum platform able to: (i) Increase responsiveness/speed of extreme data analysis algorithms; (ii) optimize the computation, dynamically migrating it accordingly with the just collected data, the historical information and the Quality of Service (QoS) needed; (iii) drive the computation considering the events that are measured by the IoT infrastructure.

In this position paper, we want to present the preliminary work carried on inside the TEMA project, proposing a continuum native, event-driven workflow architecture based on the Function as a Service (FaaS) paradigm. The goals of this architecture are the following:

- spreading functions at any continuum tier, letting the system use the best available infrastructure to run a NDM workflow;
- monitoring and controlling the FaaS infrastructures in order to build an up-to-date dataset from which to learn where to compute based on the expected QoS and the forecast one;
- dynamically connecting functions on the continuum to trigger complex distributed workflows able to properly analyze an incoming event.

The rest of the paper is organized as follows: section II presents the current state of the art in the field of urgent

The paper is partially supported both by the Horizon Europe "Trusted Extremely Precise Mapping and Prediction for Emergency Management" project (Grant Agreement, project 101093003 - TEMA) and the Italian Ministry of Health Piano Operativo Salute (POS) trajectory 2 "eHealth, diagnostica avanzata, medical device e mini invasività" through the project "Rete eHealth: AI e strumenti ICT Innovativi orientati alla Diagnostica Digitale (RAIDD)"(CUP J43C22000380001)

computing, continuum computing, and event-driven processing; III discusses the infrastructure designed inside the TEMA project to achieve the described goals; IV summarizes work and discusses the next direction in the project.

II. State of the Art

Since IoT has risen, many different use cases appear with the aim of measuring and preventing some possible disasters [1]. In some industries, IoT can be used for the extraction of oil and gas, which is a well-known system prone to incidents. [2]; while in critical areas, IoT is used to detect possible flooding and earthquakes, eventually triggering local alarms [3], [4]; or also to detect tsunamis [5] in oceanic coast areas. The use of IoT is often associated with the implementation of Machine Learning algorithms used to analyze those data, extract knowledge, and then forecast something about the next possible events [6]–[8].

Unfortunately, working on time-sensitive use cases using machine learning algorithms can often be tricky due to algorithm heaviness, or distance from the data location [9], [10].

Urgent computing was born with the idea of providing the best resources possible as soon as they are needed to absolve time-critical events like environmental disasters or human health monitoring [11]. In literature, there exist many directions adopted to realize urgent computing. Cloud Computing, of course, thanks to its flexibility in providing any kind of resource has been immediately adopted to realize HPC system to accomplish a job sooner [12], [13], but as often highlighted, even less powerful system, but closer to the data can better accomplish a time critical job [14]. The advantages of the use of Edge Computing in fact are mostly related to the proximity to the data [15] and the possibility to work even in the presence of partial network partitions [16], thanks to these advantages some authors proposed Edge as a valid infrastructure to run time-critical analysis [17], [18].

More recently, Cloud Computing and Edge Computing, sometimes together with Fog computing, have no longer considered independent infrastructures, but cooperative ones [19], which any application can take advantage of to satisfy different QoS levels; this cooperative infrastructure is now called continuum [20].

Unfortunately, native cloud or edge applications cannot be easily migrated to the continuous infrastructure, they need to be redesigned from scratch [21]. From this assumption, many authors proposed their own native continuum paradigm, frameworks, and applications to take advantage of all the tiers that make up the continuum.

Osmotic Computing [22], was one of the first paradigms used to balance time-critical applications on cloud and edge, it does that by transforming application constraints to environment constraints using a bio-inspired approach. In [23], the authors propose an edge framework, adapted for the continuum, to spread the computation across the continuum using user-defined dynamic rules that can be defined even taking into account the urgency of the computation.

In [24] the authors collect a list of jobs to be scheduled, taking into account the emergency of each of them, then apply a federated learning algorithm to assign them to a specific node in the continuum federation, using the previous data sets (Qos Required, Qos reached, node, task) as a source of data from which learn. Using machine learning to optimize the QoS of a given task has been further used [25], but as even the authors have highlighted, knowing the nature of the task and then forecasting its behavior is not easy, and prediction risks being wrong.

With the advent of containerization, researchers have seen in orchestrators, especially Kubernetes a great tool to federate heterogeneous environments like the continuum, and then customize it to deploy containerized applications on a node that can satisfy a time-based QoS constraint [26]; taking in example [27], authors use Kubernetes to federate the continuum, then they provide an internal opensource custom scheduler to deploy pods on a node that can satisfy network latency constraints, a factor that became crucial in real-time processing.

All the previous works have the assumption that any containerized application can be run *anywhere* and that it is possible to profile it, to forecast its behavior, and both considerations are not strictly true, until the introduction of Serverless Computing and FaaS.

FaaS, often used as synonymous with Serverless Computing, let spread functions using containers on most modern orchestrators, triggering them when explicitly invoked or in reaction to some external events. Most of the open-source Faas providers automatically support multiarchitecture distribution [28], and this lets the function be spread in the cloud as well in the edge [29], the authors in [30], built an Osmotic Computing based Serverless Workflow Engine able to spread and connect functions at any continuum tiers, Furthermore, functions belong to well-defined domain functions, which makes profiling functions possible and optimize their schedule on the continuum, taking into account QoS [31] application parameters and the behavior of the function on any continuum's tier [32].

III. Architecture

In this section, we present the theoretical architecture we aim at including inside the TEMA project to enable Urgent Computing at the Continuum in the NDM context. This platform shorted by the name Urgent Function Enabler (UFE) is a distributed architecture composed of six main units: 1) the Infrastructure unit (IIA); 2) the IoT unit (IA); 3) the computation unit (CU); 4) the monitoring unit (MU); 5) the scheduler unit (SU); 6) the workflow unit (WU);

A. Infrastructure unit

The infrastructure unit is strictly related to the CU, and is basically made up of all the nodes that are able to deal in any way with the data. In turn, we distinguish two main roles in the IIU which are the sensors and the workers nodes. The sensors are at the lowest level, and they are basically able just to measure environmental parameters and provide them as system data.

The workers do not measure data, but they are able to receive them and then apply some kind of computation like a transformation, or data analysis, eventually the result of that can be reintegrated into the system in order to be collected and analyzed by more workers.

The workers can belong to two main classes: 1) static workers; 2) mobile workers.

Static workers compose the most traditional computing infrastructure that basically is composed of cloud nodes deployed on remote data centers; edge nodes close to the data sources, realized using Raspberrys, Intel Nuc devices, or other micro computers and fog nodes, usually implemented with workstations or small servers racks distributed along the path from the edge to the cloud.

Mobile workers, it is a kind of novelty in this field. We consider mobile any device that can change physically change its position, we include in this group devices such as robots and drones.

Drones and robots have already been used in the field of NDM; some real examples are in [33] where drones were used to monitor wildfires in California. The drones were able to fly over the fires and collect data on their size and intensity. This information was then used to help firefighters make decisions about how to fight the fires; and in [34] drones were used to track poachers in Africa. The drones were able to fly over the camps of poachers and identify them. The information was then used by law enforcement to arrest the poachers.

Mobile workers, as well as static ones, can be used by the other units to run algorithms based on the data they can easily reach; to do that, the computation unit has to be used.

B. IoT Unit

The IA is the lowest infrastructure in UFE, it consists of all sensors used to monitor the environment, and it is managed by a Fiware infrastructure that allows the device to be authenticated and authorized, but also provides API to send data and receive commands from and to the upper level [35]. The main components of this architecture are the Orion Context Broker, the IoT Agent (IOTA), the P2P-IDM [16], and the PEP Proxy.

- The P2P IDM is a distributed eventually consistent unit that stores all service accounts used by IOT devices; it provides the API to obtain and verify Oauth2 tokens. The advantage of using a P2P IDM with respect to a centralized IDM is the possibility to keep the service up in the presence of network partitions or disconnections, and this is crucial when the infrastructure is composed of edge and mobile nodes.
- The IOTA is a middleware that receives raw data from the IoT, transforms it in NGSI format, and then sends them to the Orion Context Broker.
- The PEP proxy is an authorization proxy, placed in front of the IOTA, that verifies the device authorization according to the Keyrock policies and then forwards or denies the request to the IOTA.
- The Orion Context Broker is the core of the IOT Unit. It receives the IoT data from the IOTA NGSI format, and then offers them to third clients using an advanced Pub/Sub model; the subscription will be used to trigger faas workflows on the continuum;

C. The Computation Unit

The CU is responsible for executing the algorithms designed to compute the data collected from the IoT unit. All algorithms are encapsulated in functions since this unit is strictly based on the FaaS paradigm. In brief, FaaS is able to encapsulate a stateless function inside the container, letting external clients invoke this function using typically HTTP or pub/sub-patterns.

In UFE, the CU is not a unique infrastructure, rather it is the logical union of all the CU installed in all the nodes that compose the continuum infrastructure.

Finally, the main component of the CU is OpenFaas. OpenFaas is the most stared opensource engine on GitHub [1], it is composed of a gateway that lets us invoke any function using HTTP; a NATS server associated with a Queue-Worker, which lets us invoke the function using pub/sub model, sending back the result using webhooks, and a faas-cli, which is used to interact with OpenFaas, building multiarchitecture function natively, and of course, deploying them. OpenFaas is a centralized Function Registry to store the functions available in all the CU; this registry is even cached locally to avoid the cold start problem, typical in any containerized infrastructure.

As anticipated previously, the CU works integrated with the IIU; in particular, the FaaS stack is supposed to run in the workers' nodes, in fact, the massive use of low-level containerization together with a light system such as OpenFaaS, makes it possible to run functions in most of the traditional mobile units as well as a static unit we want to integrate. At the end then, we will be able to run algorithms on a drone, a robot as well as in a cloud virtual machine.

D. The Monitoring Unit

The MU is fundamental for planning a time-critical computation. The MU is installed along with the computation unit and collects all the useful metrics of all the functions that are executed in the CU. The Monitoring Unit is made up of a high-performance proxy (HPC), a Prometheus instance, and a centralized shared data lake. The HPC is posed in front of the OpenFaas gateway, then it intercepts all the function invocations, forwarding the result to the client, but before doing that it retrieves from the OpenFaas response the request id that is used asynchronously to fetch all the information from Prometheus and OpenFaas' logger. Prometheus is a popular open-source monitoring and alerting system that is used to collect and analyze metrics from various sources in real-time, but at this moment the information that the MU fetches are the following: 1) function executed; 2) continuum's node

[1]https://github.com/openfaas/faas

where the function is run; 3) start time and duration of the computation; 4) start and end time of the request; 5) CPU cycles and memory used to run that function and conclude the request. All these metrics are collected together and then sent to the Data Lake component, which will permanently store them, to be used later to apply scheduler choices.

E. The Scheduler Unit

A key component in the described architecture is related to the scheduling of created workflows among the components that belong to the Cloud-Continuum Infrastructure. In particular, the SU exploits context data to estimate the more efficient and convenient node in which the function should be deployed. A key role is played by *MU* that collects the monitored data that will actually be used to guide the scheduler in the offloading of workflow processes. The scheduling is dynamic, and it exploits Machine Learning inference to establish more appropriate nodes in which workflow can be deployed. The inference of a Machine Learning model could be considered a complex operation that can compromise a time-constrained application. For this reason, the scheduler will figure out an inference with a pre-trained model in an asynchronous fashion when a new workflow is not yet created or deployed. The model consists of a Deep Neural Network in which the inputs are all metrics collected through *MU* and the QoS score brought by the specific workflow. The model output consists of a score that ranks each node in the architecture on the basis of different parameters: time-response respect to the scheduler, used memory, used CPU, and other possible parameters collected by Prometheus instance present in *MU*. In particular, it performs, exploiting a classical *Soft Max Activation Function*, a classification of more appropriate nodes according to computed scores. It classifies the most efficient node candidates for deploying the next workflow processes. The decision of the nodes for each process is decided following the probabilities computed by the model. Moreover, the model will be updated by exploiting historical data collected through a continuous learning mechanism that takes advantage of the Federated Learning approach [36]. Indeed, each node, periodically, trains a local model that will be aggregated in the central cloud server and exploited by the scheduler's central component. The scheduler will become more precise thanks to historical data collected by MU.

F. The Workflow Unit

One of the most highlighted downsides of FaaS is the inability to deploy complex and distributed workflows that connect functions to serve a bigger and composed computation. To overcome this, most of the FaaS-based architectures propose to use a centralized microservice that invokes and synchronizes the right function when needed. This small trick might be not the best choice for any distributed time-critical environments; therefore, we integrated the WU. The WU let us define serverless workflows using a light version of OpenWolf [28]. OpenWolf is a small distributed broker that can be instantiated on Kubernetes or with just Docker for the lightest environments in this use case, we have an OpenWolf agent for each computing tier. OpenWolf [2] uses a customized version of the Serverless Workflow DSL [3] to describe how the input and output functions are connected, allowing one to define asynchronous faas DAGs. Using the DSL, we can submit a *Manifest file* to the OpenWolf agent, the manifest file will be used to declare a workflow and then it will be used to trigger the function defined inside it when a specific event occurs. Through the Manifest is even possible to decide where to deploy a function to use inside the workflow, but in this case, we modified the function allocation, in order to migrate a function location according with the QoS of an event and the suggestion coming from the SU. Manifests are even more shared with all the OpenWolf agents, in this way, any instance can run the same workflow.

G. Units Integration

The integration of the UFE unit is quite straightforward. The overall architecture is presented in figure 1, from the bottom we distinguished three different zones provided with the IoT sensors. Each device can use any of D-IDM to get an Oauth2 token, that is used to access to the IOT Agent in the IA. In the same unit, the IoT Agent, sends the data to the Context Broker, where the data live. In parallel, as soon as a Workflow manifest is sent to OpenWolf, the functions included in that file are deployed on the Computing Infrastructure according to the scheduler's choices. When all the functions are ready, the workflow endpoint is subscribed to the Context Broker which contains the data needed to trigger the workflow. When the subscription arrives at the Context Broker, data will be sent to the first function on the workflow, and in turn, the result will be forwarded to the other functions until all the functions in the workflow are not consumed. While the functions are run, the monitor system installed alongside the serverless platform records all the information related to the executions. This information will travel to the Data Lake, and then it will be continuously read by the scheduler in order to dynamically adapt the function position in the infrastructure.

IV. Conclusion and Future works

The work proposes a solution for Natural Disaster Management in the Horizon TEMA project exploiting Cloud-Continuum workflows, and applying them considering QoS parameters. TEMA is a Horizon Europe-funded project that wants to manage Natural disaster Management exploiting a Cloud-Edge Continuum infrastructure. For this reason, the architecture depicted in this work must be able to manage Urgent Computation applications.

The solution described here is a concrete workflow system use case that can satisfy the QoS and realize a complex algorithm pipeline exploiting IoT-collected data in a Cloud-Continuum infrastructure. Moreover, the architecture described is capable of performing dynamic scheduling of workflow components, exploiting the monitored data by each node of

[2]https://github.com/christiansicari/OpenWolf
[3]https://serverlessworkflow.io/

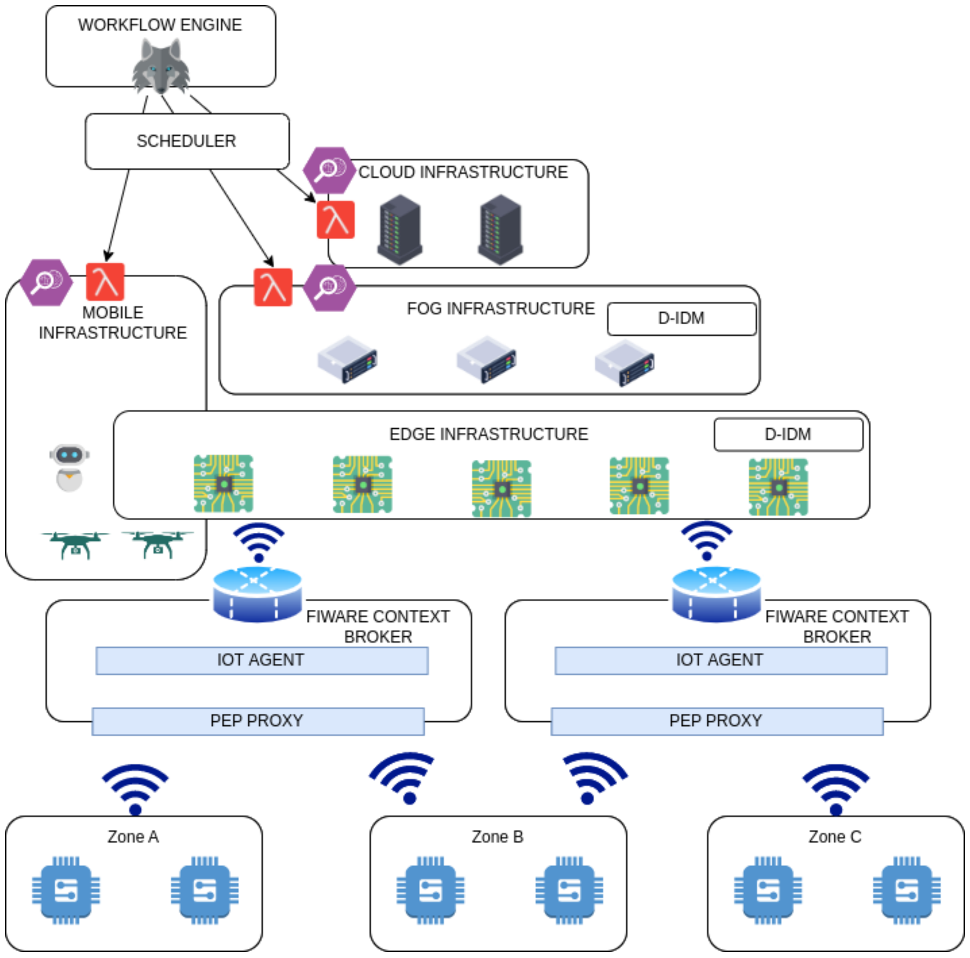

Fig. 1. TEMA Platform Architecture

infrastructure, and a sophisticated machine learning model capable of retrieving the optimal nodes. The fixed steps to perform are the practical application of the solution in a concrete use case provided by the TEMA project and the practical performance evaluation of implemented architecture. As naturally expected in future works, we need to validate the architecture we proposed, in order to understand the limits and strengths of the works.

REFERENCES

[1] Y. Awasthi and A. S. Mohammed, "Iot- a technological boon in natural disaster prediction," in *2019 6th International Conference on Computing for Sustainable Global Development (INDIACom)*, pp. 318–322, 2019.

[2] R. F. Hussain, M. A. Salehi, A. Kovalenko, Y. Feng, and O. Semiari, "Federated edge computing for disaster management in remote smart oil fields," in *2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pp. 929–936, 2019.

[3] V. Babu and V. Rajan, "Flood and earthquake detection and rescue using iot technology," in *2019 International Conference on Communication and Electronics Systems (ICCES)*, pp. 1256–1260, 2019.

[4] D. Balouek-Thomert, P. Silva, K. Fauvel, A. Costan, G. Antoniu, and M. 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), Association for Computing Machinery, 2022.

[5] F. Løvholt, S. Lorito, J. Macias, M. Volpe, J. Selva, and S. Gibbons, "Urgent tsunami computing," in *2019 IEEE/ACM HPC for Urgent Decision Making (UrgentHPC)*, pp. 45–50, 2019.

[6] M. Mohammadi, A. Al-Fuqaha, S. Sorour, and M. Guizani, "Deep learning for iot big data and streaming analytics: A survey," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 2923–2960, 2018.

[7] Y. S. Lonkar, A. S. Bhagat, and S. A. S. Manjur, "Smart disaster management and prevention using reinforcement learning in iot environment," in *2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*, pp. 35–38, 2019.

[8] T. J. Saleem and M. A. Chishti, "Deep learning for the internet of things: Potential benefits and use-cases," *Digital Communications and Networks*, vol. 7, no. 4, pp. 526–542, 2021.

[9] M. V. K. Choda, S. V. Perla, B. Shaik, Y. T. A. Yelchuru, and P. Yalla, "A critical survey on real-time traffic sign recognition by using cnn machine learning algorithm," in *2023 International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT)*, pp. 445–450, 2023.

[10] C. Augenstein, N. Spangenberg, and B. Franczyk, "Applying machine learning to big data streams : An overview of challenges," in *2017 IEEE 4th International Conference on Soft Computing & Machine Intelligence (ISCMI)*, pp. 25–29, 2017.

[11] S. H. Leong and D. Kranzlmüller, "Towards a general definition of urgent computing," *Procedia Computer Science*, vol. 51, pp. 2337–2346, 2015. International Conference On Computational Science, ICCS 2015.

[12] B. Posey, A. Deer, W. Gorman, V. July, N. Kanhere, D. Speck, B. Wilson, and A. Apon, "On-demand urgent high performance computing utilizing

the google cloud platform," in *2019 IEEE/ACM HPC for Urgent Decision Making (UrgentHPC)*, pp. 13–23, 2019.

[13] Z. Zhao, P. Martin, J. Wang, A. Taal, A. Jones, I. Taylor, V. Stankovski, I. G. Vega, G. Suciu, A. Ulisses, and C. de Laat, "Developing and operating time critical applications in clouds: The state of the art and the switch approach," *Procedia Computer Science*, vol. 68, pp. 17–28, 2015. 1st International Conference on Cloud Forward: From Distributed to Complete Computing.

[14] S. Koulouzis, P. Martin, H. Zhou, Y. Hu, J. Wang, T. Carval, B. Grenier, J. Heikkinen, C. de Laat, and Z. Zhao, "Time-critical data management in clouds: Challenges and a dynamic real-time infrastructure planner (drip) solution," *Concurrency and Computation: Practice and Experience*, vol. 32, no. 16, p. e5269. e5269 cpe.5269.

[15] A. Catalfamo, A. Celesti, M. Fazio, G. Randazzo, and M. Villari, "A platform for federated learning on the edge: a video analysis use case," in *2022 IEEE Symposium on Computers and Communications (ISCC)*, pp. 1–7, 2022.

[16] C. Sicari, A. Catalfamo, A. Galletta, and M. Villari, "A distributed peer to peer identity and access management for the osmotic computing," in *2022 22nd IEEE International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*, pp. 775–781, 2022.

[17] A. Jain and D. S. Jat, "An edge computing paradigm for time-sensitive applications," in *2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, pp. 798–803, 2020.

[18] P. R. Ovi, E. Dey, N. Roy, and A. Gangopadhyay, "Aris: A real time edge computed accident risk inference system," in *2021 IEEE International Conference on Smart Computing (SMARTCOMP)*, pp. 47–54, 2021.

[19] M. Liu, D. Li, H. Wu, F. Lyu, and X. S. Shen, "Cooperative edge-cloud caching for real-time sensing big data search in vehicular networks," in *ICC 2021 - IEEE International Conference on Communications*, pp. 1–6, 2021.

[20] S. Moreschini, F. Pecorelli, X. Li, S. Naz, D. Hästbacka, and D. Taibi, "Cloud continuum: The definition," *IEEE Access*, vol. 10, pp. 131876–131886, 2022.

[21] S. Dustdar, V. C. Pujol, and P. K. Donta, "On distributed computing continuum systems," *IEEE Transactions on Knowledge and Data Engineering*, vol. XX, pp. 1–14, 2022.

[22] C. Sicari, A. Galletta, A. Celesti, M. Fazio, and M. Villari, "An osmotic computing enabled domain naming system (oce-dns) for distributed service relocation between cloud and edge," *Computers & Electrical Engineering*, vol. 96, p. 107578, 2021.

[23] E. G. Renart, D. Balouek-Thomert, and M. Parashar, "An edge-based framework for enabling data-driven pipelines for iot systems," in *2019 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pp. 885–894, 2019.

[24] G. P. Mattia and R. Beraldi, "Leveraging reinforcement learning for online scheduling of real-time tasks in the edge/fog-to-cloud computing continuum," in *2021 IEEE 20th International Symposium on Network Computing and Applications (NCA)*, pp. 1–9, 2021.

[25] A. Jonathan, A. Chandra, and J. Weissman, "Awan: Locality-aware resource manager for geo-distributed data-intensive applications," in *2016 IEEE International Conference on Cloud Engineering (IC2E)*, pp. 32–41, 2016.

[26] D. Hass and J. Spillner, "Workload deployment and configuration reconciliation at scale in kubernetes-based edge-cloud continuums," in *2022 21st International Symposium on Parallel and Distributed Computing (ISPDC)*, pp. 121–128, 2022.

[27] F. Rossi, V. Cardellini, F. Lo Presti, and M. Nardelli, "Geo-distributed efficient deployment of containers with kubernetes," *Computer Communications*, vol. 159, pp. 161–174, 2020.

[28] C. Sicari, L. Carnevale, A. Galletta, and M. Villari, "Openwolf: A serverless workflow engine for native cloud-edge continuum," in *2022 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, pp. 1–8, 2022.

[29] T. Pfandzelter and D. Bermbach, "tinyfaas: A lightweight faas platform for edge environments," in *2020 IEEE International Conference on Fog Computing (ICFC)*, pp. 17–24, 2020.

[30] G. Morabito, C. Sicari, A. Ruggeri, A. Celesti, and L. Carnevale, "Secure-by-design serverless workflows on the edge–cloud continuum through the osmotic computing paradigm," *Internet of Things*, vol. 22, p. 100737, 2023.

[31] S. K. R and J. Lakshmi, "Qos aware faas platform," in *2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*, pp. 812–819, 2021.

[32] F. Rossi, S. Falvo, and V. Cardellini, "Gofs: Geo-distributed scheduling in openfaas," in *2021 IEEE Symposium on Computers and Communications (ISCC)*, pp. 1–6, 2021.

[33] M. A. Akhloufi, A. Couturier, and N. A. Castro, "Unmanned aerial vehicles for wildland fires: Sensing, perception, cooperation and assistance," *Drones*, vol. 5, no. 1, 2021.

[34] K. E. Doull, C. Chalmers, P. Fergus, S. Longmore, A. K. Piel, and S. A. Wich, "An evaluation of the factors affecting 'poacher' detection with drones and the efficacy of Machine-Learning for detection," *Sensors (Basel)*, vol. 21, June 2021.

[35] L. Carnevale, A. Galletta, M. Fazio, A. Celesti, and M. Villari, "Designing a fiware cloud solution for making your travel smoother: The fliware experience," in *2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC)*, pp. 392–398, 2018.

[36] "Communication-efficient learning of deep networks from decentralized data," 2017.