Towards Sustainable Deployment of Microservices over the Cloud-Edge Continuum, with FREEDA

Jacopo Soldani
University of Pisa
Pisa, Italy
jacopo.soldani@unipi.it

Roberto Amadini
University of Bologna, Bologna, Italy
OPTIMA ARC, Melbourne, Australia
roberto.amadini@unibo.it

Antonio Brogi
University of Pisa
Pisa, Italy
antonio.brogi@unipi.it

Stefano Forti
University of Pisa
Pisa, Italy
stefano.forti@unipi.it

Saverio Giallorenzo
University of Bologna
Bologna, Italy
saverio.giallorenzo2@unibo.it

Pierluigi Plebani
Politecnico di Milano
Milano, Italy
pierluigi.plebani@polimi.it

Monica Vitali
Politecnico di Milano
Milano, Italy
monica.vitali@polimi.it

Gianluigi Zavattaro
University of Bologna
Bologna, Italy
gianluigi.zavattaro@unibo.it

ABSTRACT

This position paper introduces FREEDA, a research project aimed at supporting DevOps engineers in achieving failure-resilient and sustainable deployments of microservice-based applications over the Cloud-IoT computing continuum. After providing the context, rationale, and motivations of FREEDA, we introduce the main research objectives of FREEDA and its concept, namely how FREEDA will realize its research objectives.

CCS CONCEPTS

• **Computer systems organization** → *Distributed architectures*; • **Social and professional topics** → *Sustainability*; • **Software and its engineering** → Software fault tolerance; • **Applied computing** → Multi-criterion optimization and decision-making.

KEYWORDS

Sustainabile IT, Energy, Cloud-Edge, Application Deployment

ACM Reference Format:
Jacopo Soldani, Roberto Amadini, Antonio Brogi, Stefano Forti, Saverio Giallorenzo, Pierluigi Plebani, Monica Vitali, and Gianluigi Zavattaro. 2024. Towards Sustainable Deployment of Microservices over the Cloud-Edge Continuum, with FREEDA. In *Workshop on Flexible Resource and Application Management on the Edge (FRAME '24), June 3–7, 2024, Pisa, Italy*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3659994.3660311

1 INTRODUCTION

The ever-increasing capabilities present on smart connected IoT devices call for an evolution of Cloud computing into large-scale, pervasive distributed environments that can avoid unnecessary latencies, and fully exploit accessible computing capabilities at the Edge of the network [13]. The distributed infrastructure supporting the Cloud-Edge computing continuum is going to be highly heterogeneous and variable. Heterogeneity comes from various devices, featuring different computing and storage capabilities, supporting many software and deployment technologies, and communicating via diverse protocols. Variability is due to dynamicity (e.g., nodes joining/leaving the infrastructure, changing workloads) and uncertainty (e.g., unstable end-to-end connectivity, software, and hardware faults). The high heterogeneity and variability of the distributed Cloud-Edge continuum exacerbate the need to face possible QoS-degradations and faults [17].

At the same time, microservices are pervading in the delivery of modern enterprise applications, therefore calling for support in their deployment over the Cloud-Edge continuum. Microservice-based Applications (MSAs) integrate multiple heterogeneous services, each having different deployment requirements. These include the affordable cost of renting Cloud-Edge resources, the hardware, software, and security to be featured by the infrastructure node used to run a service, and its QoS requirements, e.g., latency, bandwidth, and availability. Given the complexity of both MSAs and Cloud-Edge infrastructures, and the volatility of services and infrastructure nodes at the Edge, failures are first-class citizens when distributing MSAs over a Cloud-Edge infrastructure, and must necessarily be taken into account for ensuring that a deployed MSA delivers the desired QoS. DevOps engineers must hence account for the possible failures of a service and of its hosting node, as well as for the failure cascades, where failing infrastructure nodes/services cause the failure of other nodes/services [35].

The above concerns come together with the EU strategy of "building a climate-neutral, green, fair, and social Europe" [11], which targets the environmentally sustainable growth of EU industries, including IT [12]. Therefore, we need to consider environmental sustainability when deploying MSAs over Cloud-Edge infrastructures, e.g., by reducing the brown energy consumed by an MSA deployed over a Cloud-Edge infrastructure.

FRAME '24, June 3–7, 2024, Pisa, Italy

ACM ISBN 979-8-4007-0641-7/24/06
https://doi.org/10.1145/3659994.3660311

Unfortunately, environmental sustainability can conflict with other deployment requirements, for instance, the failure resilience requirements discussed earlier. For example, the brown energy consumed by a deployed MSA can be reduced by deploying only one instance of each of its services on infrastructure nodes close to one another, possibly on the same node, but negatively impacting failure resilience. Conversely, a higher failure resilience can be ensured by heavily replicating each service over multiple different nodes, which could result in more brown energy being consumed. The resulting complexity calls for support in the deployment of MSAs over a large-scale, heterogeneous, and variable Cloud-Edge infrastructure. Such support should suitably find a trade-off among the multiple – and possibly conflicting – deployment requirements of to-be-deployed MSAs. In this process, the variability of the available Cloud-Edge infrastructure conditions over time (e.g., with new nodes joining the infrastructure, while others are leaving, crashing, or getting overloaded) should be considered, also for already deployed MSAs, whose services are often continuously delivered through CI/CD.

The research project FREEDA, introduced in this position paper, aims at enabling a sustainable and failure-resilient deployment of MSAs over existing Cloud-Edge infrastructures. FREEDA will identify and specify the deployment and configuration of software components constituting an MSA across a set of nodes in the cloud continuum to meet the deployment requirements of the application, encompassing sustainability and resilience against failures.

The project targets four main objectives:

- (O_1) *Holistic MSA Deployment over the Cloud-Edge continuum*: FREEDA will develop novel techniques to determine a suitable trade-off among the MSA's deployment requirements, therein included cost, hardware, software, security, failure resilience, and sustainability requirements.
- (O_2) *Continuous Reasoning for Adaptive MSA Deployment over the Cloud-Edge continuum*: FREEDA will develop continous reasoning-like [15, 30] to timely adapt the deployment of an MSA over a Cloud-Edge infrastructure when changes in the MSA, infrastructure, or deployment requirements occur.
- (O_3) *Explainable Enhancement of MSA Deployments' Failure Resilience*: FREEDA will develop explainable techniques to analyze and possibly avoid cascading failures in deployed MSAs, e.g., by adding circuit breakers
- (O_4) *Explainable Reduction of MSA Deployments' Environmental Impact*: FREEDA will develop explainable techniques for reducing brown energy consumption when deploying MSAs over Cloud-Edge infrastructures.

This paper is organized as follows. Sections 2 and 3 position FREEDA with respect to the state-of-the-art and introduce its approach, respectively. Finally, Section 4 draws some concluding remarks.

2 STATE OF THE ART

To the best of our knowledge, there is no solution for holistically determining the deployment of MSAs over a Cloud-Edge infrastructure, which is explainable and considers multiple deployment requirements, including failure resilience and evironmental sustainability [17]. Therefore, we hereafter separately report on existing solutions for constraint-aware deployment, failure resilience, and environmental sustainability.

Constraint-aware Deployment. Constraint reasoning was first used to optimally deploy multiservice applications on Cloud resources in [1, 9, 14], with [14] focusing on service dependencies, and [1, 9] focusing on services' hardware, software, and availability requirements. More recently, constraint reasoning has also been exploited to generate containerized MSA deployments, with [5, 6] adapting [1] to work with MSAs, while [24] enabling to schedule containers running on Kubernetes based on their QoS requirements. [10] instead focuses on deploying MSAs on Cloud VMs, encoding their services' hardware/software requirements as constraints, and aiming to minimize the overall deployment cost. In summary, existing solutions focus on single aspects of the MSA deployment problem (e.g., services' hardware/software requirements, service dependencies, or deployment cost) and they always consider a whole deployment, even when a change in the deployment context would require adapting only a portion of the deployment itself. FREEDA aims at fulfilling those gaps by targeting its objectives O_1 and O_2.

Failure Resilience. Existing solutions for enforcing failure resilience in MSAs typically provide guidelines to drive their design and development [20, 22], or to configure their deployment scripts to self-heal failing services, e.g., by restarting their hosting container [7]. There is, however, no solution for analyzing an existing MSA and the available Cloud-Edge infrastructure or for enforcing the failure-resilient deployment of such MSA over such infrastructure. Existing techniques for analyzing failures in MSAs focus on detecting such failures and identifying root causes [3, 25, 26, 39]. Whilst they can automatically determine the possible root causes for an observed failure, they mainly focus on returning *only* such possible root causes, then relying on DevOps engineers to manually inspect logs or monitored metrics to understand how failures propagated up to that observed [35]. The only attempt in this direction is our previous work [34], in which we provide explanations on failure propagation in MSAs to DevOps engineers, however relying on them to manually specify how each service therein behaves. In summary, to the best of our knowledge, there is currently no technique allowing us to analyze the deployment of an MSA over a Cloud-Edge infrastructure to enforce failure resilience of such deployment. FREEDA aims to fulfill this gap by targeting its objective O_3.

Environmental Sustainability. Various existing solutions focus on improving the energy efficiency of Cloud data centers [19, 37, 38]. Best practices to improve data centers' energy efficiency are provided in [2, 33]. Computing infrastructures are however getting distributed over the Cloud-Edge continuum, which is inherently composed of heterogeneous nodes. As a result, the power usage effectiveness significantly fluctuates [23], hence making the energy-aware allocation of Cloud-Edge resources an ongoing challenge [41, 43]. In addition, when distributing applications over the Cloud-Edge continuum, application data must be stored and managed in synergy with applications themselves, to further reduce latency and guarantee security constraints [32]. However, current approaches for improving the efficiency of application deployments typically focus only on the target infrastructure, at most also considering the temporal/geographical distribution of applications deployed thereon [28, 31, 42]. Other approaches focus mainly on the application level [4, 8, 27], supporting the design from scratch of energy

sustainable applications, with a limited suitability to already existing applications. At the same time, the deployed applications' energy demand increased with the widespread adoption of the Cloud. This is expected to occur in Cloud-Edge infrastructures as well, therefore calling for supporting energy-aware MSA deployments thereon [16]. This issue is targeted by objective O_4 of FREEDA.

3 FREEDA'S APPROACH

FREEDA considers the deployment of an MSA A over a Cloud-Edge infrastructure I, based on a set R of deployment requirements (Fig. 1). The infrastructure description in I will include information on the costs for utilizing a resource, on nodes' hardware and software capabilities, and their load/currently available resources. The deployment requirements in R will instead include hardware/software services' requirements, their network QoS, security, and failure resilience requirements. R can also specify the affordable *budget* for A's deployment, which comprises the *monetary budget*, i.e., the maximum cost that can be paid, and the *sustainability budget*, i.e., the maximum amount of brown energy that can be consumed.

The FREEDA toolchain will plan the deployment D of A over I by holistically determining a trade-off among the deployment requirements in R. This will be done in two subsequent steps (Fig. 1), the first being devoted to enriching the failure resilience and environmental sustainability of the deployed MSA, based on historical data (H) from the current and previously enacted deployments of A (e.g., logs) and formerly monitored on I (e.g., nodes' availability or load). The extended application specification A+ and deployment requirements R+ will then be processed by the tools composing the trade-off step, which will return a deployment D and an explanation E of why/how such deployment constitutes the best trade-off among the multiple – and possibly conflicting – deployment requirements of A. Notably, E will also include explanations on why the enrichment step extended A and R into A+ and R+, meaning that the DevOps engineer will be informed on why and how the MSA and its deployment requirements were updated.

Enrichment. This step is composed of two tools, i.e., the *failure analyzer* and the *energy analyzer* (Fig. 1). The failure analyzer will implement the analysis techniques resulting from objective O_3. It will first identify causal failure relationships among the services in A and the nodes/links in I, also based on the historical data H coming from the current and previously enacted deployments, when available. Based on the identified causal relationships, the failure analyzer will include sidecar integration components (e.g., circuit breakers) in A to avoid the failure of a service to propagate to others, based on the identified failure causalities. The failure analyzer will also generate novel requirements in R for enforcing failure resilience when deploying a service on a node in I, e.g., avoiding the deployment of services on nodes that are known to fail if subject to a given load, or whose failure is predicted to occur soon based on the available historical data.

The energy analyzer will instead implement the analysis techniques resulting from objective O_4. It will extend the MSA specification A and the deployment requirements R to reduce the environmental impact when deploying A over the Cloud-Edge infrastructure I. The energy analyzer will also consider the historical data H coming from the current and previously enacted deployments, when available. Location and data awareness will play a key role in the energy analyzer, e.g., to reduce of the energy consumed due to interactions between components.

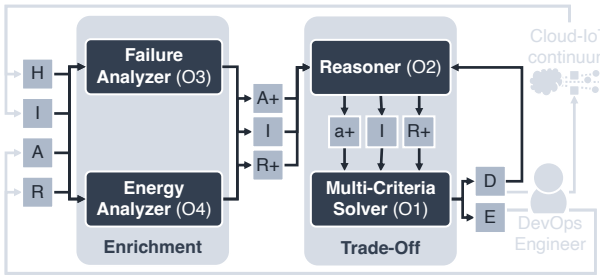

Figure 1: A bird's-eye view of FREEDA's approach.

Trade-off. This step will process the enriched MSA specification A+ and deployment requirements R+, and the available infrastructure I. The processing starts with a *reasoner* component (Fig. 1), which implements the continuous reasoning-like techniques resulting from objective O_2. The reasoner will check whether the current deployment D (initially empty) deploys all components in A+ on nodes that are available in I, while also satisfying all deployment requirements in R+. If this is the case, there is no need to adapt the deployment D. Otherwise, the reasoner will determine the portion of D containing services of A+ and/or nodes of I that are not satisfying the requirements in R+. This information will be used by the reasoner to identify the portion a+ of the MSA (initially, the whole A+) whose components are to be re-deployed.

The obtained set a+ of to-be-deployed services will be processed, together with I and R+, by a *multi-criteria solver*. The latter will determine the deployment of the components in a+ over the nodes in I while finding a trade-off between the requirements in R+. The multi-criteria solver will also be configurable to privilege certain requirements when all requirements cannot be satisfied, e.g., to privilege environmental sustainability instead of failure resilience, or vice-versa. The multi-criteria solver will finally return a new deployment D (which differs from the former only for the deployment of the components in a+) and an explanation E on why/how D constitutes the best trade-off among the deployment requirements in R+, e.g., what was added in the enrichment phase, which requirements were not satisfied (if any), and why this happened.

DevOps engineers will then be able to exploit both the deployment D and the explanation E. The deployment D will specify the placement and configuration of the services forming an MSA with existing declarative modeling, e.g., TOSCA [29] or EDMM [40], to enable concretely enacting the specified deployment by relying on existing, multi-platform deployment tools. The information collected on the enacted deployment (e.g., services' logs and metrics monitored the available infrastructure nodes) will update the input I and H, enabling to re-plan the deployment when changes occur. The explanation E will instead enable determining whether/how to refactor her application or its deployment requirements to resolve potential issues. For instance, suppose that E indicates that D includes a new component (e.g., a circuit breaker) alongside a service to let it tolerate the failures of the services it interacts with, causing

the consumption of additional energy. The DevOps engineer may save this energy by adapting the service's source code to natively tolerate the failures of the services it invokes.

4 CONCLUSIONS

In this position paper, we have introduced the research project FREEDA, by showing how it will support DevOps engineers in achieving failure-resilient and sustainable deployments of MSAs over the Cloud-Edge computing continuum. Future work will not only include the implementation of the FREEDA approach but also the assessment of its effectiveness by exploiting the FREEDA toolchain to determine/adapt the deployment of a benchmarking MSA on an existing Cloud-Edge testbed, e.g., [18, 21, 36].

ACKNOWLEDGMENTS

This work was partly supported by the following projects: *FREEDA* (CUP: I53D23003550006, funded by the frameworks PRIN (MUR, Italy) and Next Generation EU); *WPNRR-M4C2* (Investimento 1.3, Partenariato Esteso PE00000013, *FAIR - Future Artificial Intelligence Research*, Spoke 8 *Pervasive AI*); *TEADAL* (EU Horizon Framework grant agreement 101070186).

REFERENCES

[1] E. Ábrahám et al. 2016. Zephyrus2: On the Fly Deployment Optimization Using SMT and CP Technologies. In *SETTA 2016 (LNCS, Vol. 9984)*, M. Fränzle et al. (Eds.). 229–245. https://doi.org/10.1007/978-3-319-47677-3_15
[2] M. Acton et al. 2021. Best Practice Guidelines for the EU Code of Conduct on Data Centre Energy Efficiency. JRC Technical Notes, JRC123653.
[3] P. Aggarwal et al. 2020. Localization of Operational Faults in Cloud Applications by Mining Causal Dependencies in Logs Using Golden Signals. In *ICSOC 2020 Workshops (LNCS, Vol. 12632)*, Hakim Hacid et al. (Eds.). Springer, 137–149. https://doi.org/10.1007/978-3-030-76352-7_17
[4] F.G. Alvares De Oliveira et al. 2010. Self-optimisation of the energy footprint in service-oriented architectures. In *GCM 2010*. ACM, 4–9. https://doi.org/10.1145/1925013.1925014
[5] L. Bacchiani et al. 2021. Microservice Dynamic Architecture-Level Deployment Orchestration. In *COORDINATION 2021 (LNCS, Vol. 12717)*, F. Damiani et al. (Eds.). Springer, 257–275. https://doi.org/10.1007/978-3-030-78142-2_16
[6] M. Bravetti et al. 2019. Optimal and Automated Deployment for Microservices. In *FASE 2019 (LNCS, Vol. 11424)*, Rainer Hähnle et al. (Eds.). Springer, 351–368. https://doi.org/10.1007/978-3-030-16722-6_21
[7] A. Brogi et al. 2022. Self-healing trans-cloud applications. *Computing* 104, 4 (2022), 809–833. https://doi.org/10.1007/S00607-021-00977-Z
[8] C. Cappiello et al. 2011. Business process co-design for energy-aware adaptation. In *ICCP 2011*. IEEE, 463–470. https://doi.org/10.1109/ICCP.2011.6047917
[9] R. Di Cosmo et al. 2014. Automated synthesis and deployment of cloud applications. In *ASE 2014*, Ivica Crnkovic et al. (Eds.). ACM, 211–222. https://doi.org/10.1145/2642937.2642980
[10] M. Erascu et al. 2021. Scalable optimal deployment in the cloud of component-based applications using optimization modulo theory, mathematical programming and symmetry breaking. *J. Log. Algebraic Methods Program.* 121 (2021), 100664. https://doi.org/10.1016/J.JLAMP.2021.100664
[11] EU. 2019. A new strategic agenda for the EU.
[12] EU. 2022. EU Industrial Policy.
[13] A. J. Ferrer et al. 2019. Towards the Decentralised Cloud: Survey on Approaches and Challenges for Mobile, Ad hoc, and Edge Computing. *ACM Comput. Surv.* 51, 6 (2019), 111:1–111:36. https://doi.org/10.1145/3243929
[14] J. Fischer et al. 2012. Engage: a deployment management system. In *PLDI 2012*, J. Vitek et al. (Eds.). ACM, 263–274. https://doi.org/10.1145/2254064.2254096
[15] S. Forti et al. 2022. Declarative continuous reasoning in the cloud-IoT continuum. *J. Log. Comput.* 32, 2 (2022), 206–232. https://doi.org/10.1093/LOGCOM/EXAB083
[16] Stefano Forti and Antonio Brogi. 2022. Green Application Placement in the Cloud-IoT Continuum. In *PADL 2022 (LNCS, Vol. 13165)*, James Cheney et al. (Eds.). Springer, 208–217. https://doi.org/10.1007/978-3-030-94479-7_14
[17] M. Gaglianese et al. 2023. Green Orchestration of Cloud-Edge Applications: State of the Art and Open Challenges. In *SOSE 2023*. IEEE, 250–261. https://doi.org/10.1109/SOSE58276.2023.00036
[18] Y. Gan et al. 2019. An Open-Source Benchmark Suite for Microservices and Their Hardware-Software Implications for Cloud & Edge Systems. In *ASPLOS 2019*. ACM, 3–18. https://doi.org/10.1145/3297858.3304013
[19] N. Gholipour et al. 2020. A novel energy-aware resource management technique using joint VM and container consolidation approach for green computing in cloud data centers. *Simul. Model. Pract. Theory* 104 (2020), 102127. https://doi.org/10.1016/J.SIMPAT.2020.102127
[20] V. Giedrimas et al. 2018. The Aspect of Resilience in Microservices-Based Software Design. In *STAF 2018 Collocated Workshops (LNCS, Vol. 11176)*, M. Mazzara et al. (Eds.). Springer, 589–595. https://doi.org/10.1007/978-3-030-04771-9_44
[21] H. Gupta et al. 2017. iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments. *Softw. Pract. Exp.* 47, 9 (2017), 1275–1296. https://doi.org/10.1002/SPE.2509
[22] V. Heorhiadi et al. 2016. Gremlin: Systematic Resilience Testing of Microservices. In *ICDCS 2016*. IEEE Computer Society, 57–66. https://doi.org/10.1109/ICDCS.2016.11
[23] N. Jones. 2018. How to stop data centres from gobbling up the world's electricity. *Nature* 561 (2018), 163–166. https://doi.org/10.1038/d41586-018-06610-y
[24] T. Lebesbye et al. 2021. Boreas - A Service Scheduler for Optimal Kubernetes Deployment. In *ICSOC 2021 (LNCS, Vol. 13121)*, H. Hacid et al. (Eds.). Springer, 221–237. https://doi.org/10.1007/978-3-030-91431-8_14
[25] P. Liu et al. 2020. Unsupervised Detection of Microservice Trace Anomalies through Service-Level Deep Bayesian Networks. In *ISSRE 202*, M. Vieira et al. (Eds.). IEEE, 48–58. https://doi.org/10.1109/ISSRE5003.2020.00014
[26] Y. Meng et al. 2020. Localizing Failure Root Causes in a Microservice through Causality Inference. In *IWQoS 2020*. IEEE, 1–10. https://doi.org/10.1109/IWQOS49365.2020.9213058
[27] A. Nowak et al. 2014. Automating Green Patterns to Compensate CO2 Emissions of Cloud-based Business Processes. In *ADVCOMP 2014*. XPS, 132–139.
[28] T. Nylander et al. 2018. Brownout^{CC}: Cascaded Control for Bounding the Response Times of Cloud Applications. In *ACC 2018*. IEEE, 3354–3361. https://doi.org/10.23919/ACC.2018.8431282
[29] OASIS. 2020. TOSCA Simple Profile in YAML Version 1.3. OASIS Standard.
[30] P.W. O'Hearn. 2018. Continuous Reasoning: Scaling the impact of formal methods. In *LICS 2018*. ACM, 13–25. https://doi.org/10.1145/3209108.3209109
[31] A.V. Papadopoulos et al. 2017. Power-aware cloud brownout: Response time and power consumption control. In *CDC*. IEEE, 2686–2691. https://doi.org/10.1109/CDC.2017.8264049
[32] P. Plebani et al. 2018. Fog Computing and Data as a Service: A Goal-Based Modeling Approach to Enable Effective Data Movements. In *CAiSE 2018 (LNCS, Vol. 10816)*, J. Krogstie et al. (Eds.). Springer, 203–219. https://doi.org/10.1007/978-3-319-91563-0_13
[33] H. Singh. 2011. Data Center Maturity Model. The Green Grid.
[34] J. Soldani et al. 2021. What Went Wrong? Explaining Cascading Failures in Microservice-Based Applications. In *SummerSoc 2021 (CCIS, Vol. 1429)*, J. Barzen (Ed.). Springer, 133–153. https://doi.org/10.1007/978-3-030-87568-8_9
[35] J. Soldani and A. Brogi. 2023. Anomaly Detection and Failure Root Cause Analysis in (Micro) Service-Based Cloud Applications: A Survey. *ACM Comput. Surv.* 55, 3 (2023), 59:1–59:39. https://doi.org/10.1145/3501297
[36] Akshitha Sriraman and Thomas F. Wenisch. 2018. μ Suite: A Benchmark Suite for Microservices. In *IISWC 2018*. IEEE, 1–12. https://doi.org/10.1109/IISWC.2018.8573515
[37] M. Vitali et al. 2015. Learning a goal-oriented model for energy efficient adaptive applications in data centers. *Inf. Sci.* 319 (2015), 152–170. https://doi.org/10.1016/J.INS.2015.01.023
[38] U. Wajid et al. 2016. On Achieving Energy Efficiency and Reducing CO_2 Footprint in Cloud Computing. *IEEE Trans. Cloud Comput.* 4, 2 (2016), 138–151. https://doi.org/10.1109/TCC.2015.2453988
[39] L. Wu et al. 2020. MicroRCA: Root Cause Localization of Performance Issues in Microservices. In *NOMS 2020*. IEEE, 1–9. https://doi.org/10.1109/NOMS47738.2020.9110353
[40] M. Wurster et al. 2020. The essential deployment metamodel: a systematic review of deployment automation technologies. *SICS Softw.-Intensive Cyber Phys. Syst.* 35, 1-2 (2020), 63–75. https://doi.org/10.1007/S00450-019-00412-X
[41] Y. Xiao and M. Krunz. 2017. QoE and power efficiency tradeoff for fog computing networks with fog node cooperation. In *INFOCOM 2017*. IEEE, 1–9. https://doi.org/10.1109/INFOCOM.2017.8057196
[42] M. Xu et al. 2021. A Self-Adaptive Approach for Managing Applications and Harnessing Renewable Energy for Sustainable Cloud Computing. *IEEE Trans. Sustain. Comput.* 6, 4 (2021), 544–558. https://doi.org/10.1109/TSUSC.2020.3014943
[43] W. Zhang et al. 2020. Energy-efficient Workload Allocation and Computation Resource Configuration in Distributed Cloud/Edge Computing Systems With Stochastic Workloads. *IEEE Journal on Selected Areas in Communications* 38, 6 (2020), 1118–1132. https://doi.org/10.1109/JSAC.2020.2986614

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009