Towards a self-adaptive architecture for energy-efficient cloud applications

Henrique Medeiros*, Sophie Chabridon*, Denisse Muñante†
*SAMOVAR, Télécom SudParis, Institut Polytechnique de Paris, 91120 Palaiseau, France
† ENSIIE & SAMOVAR, 91025 Évry, France
{henrique.de_medeiros, sophie.chabridon}@telecom-sudparis.eu, denisse.munantearzapalo@ensiie.fr

***Index Terms*—energy efficiency, energy awareness, cloud computing, self-adaptive systems, adaptation, microservices**

Cloud applications have many benefits, such as high availability, low cost of maintenance, and elasticity [1], with the last one focusing on scalability, cost efficiency, and time efficiency. These characteristics, aligned with hardware provisioning, allow to configure an application according the available resources. These resources are allocated to answer different goals of cloud applications. However, they are not directly accessible to clients but managed through cloud providers in Virtual Machines (VMs) [2].

Self-adaptive systems (SAS) are conceived as a way to avoid and correct, under certain constraints, the degradation of quality of service (QoS) of software during its execution, which can be due to changes in environmental conditions. SAS is used in the context of cloud-based applications to allow their elasticity by applying various configurations, thus avoiding the degradation of their QoS. Recent works mainly focus on the analysis of the CPU usage and resource allocation of cloud applications, **neglecting the analysis of their impact on energy consumption** [3].

Energy-awareness and energy efficiency are two QoS in our research. *Energy efficiency* means consuming less energy to perform a given task, whilst *energy-awareness* refers to the knowledge on the energy consumed when performing a given task. Energy-awareness is expected to have an indirect positive impact on energy efficiency itself [4]. Energy and Power consumption are the metrics used for evaluating energy efficiency and energy awareness of software.

The increase of energy consumption in cloud systems turns light over the amount of hardware resources provisioned in virtual machines [5]. Researchers are recently attracted on comparing the energy consumed by cloud architectures, for instance in [6] authors compared the impact of energy consumption of six cloud patterns, to provide, at **design-time**, guidelines of building more energy efficient cloud applications. However, in SAS context, energy efficiency and energy awareness, as QoS criteria at **run-time** in cloud applications, are still **challenging** because they require a deep analysis of the resource requirements and the behavior of applications.

The energy consumption of applications is usually determined through resources analysis or using power meters, such as CPU profiling, as pointed out by [7], the consumed power by the CPU is related to CPU utilization and CPU frequency. Or machine's battery measurements, such as presented by [8] to monitor the power consumption of serverless functions. Serverless function presents the advantage that it is possible to compute the amount of energy, once the function has a start time and end time. But this analysis is made concerning the entire VM. SAS usually uses these methodologies to trigger some adaptations, such as in [9] applying at datacenters, optimizing the maximization of the use of renewable energies. In the literature, there is no work concerning the energy consumption at the application level of cloud applications.

Adaptations concerning energy efficiency at cloud applications are more challenging, the analysis must be made at **run-time**, without having to stop the application and considering the execution time of the application, and with **energy fluctuations**, considering different events of the environment over the application, e.g. different workloads and datacenter's temperature. We focus on addressing these challenges by providing a self-adaptive architecture to adapt cloud applications for optimizing the energy consumption of the application. This optimization must concern that the proposal must not have a significant impact on the energy consumption of the cloud application itself, when performing its adaptation.

Therefore, in order to address the previous challenges, in this extended abstract we aim at answering the following research questions:

- **RQ1: Which are the requirements to enable energy efficiency of cloud applications?**
- **RQ2: What analysis should be considered to reach energy efficiency in a cloud environment?**

To answer RQ1, we introduce a self-adaptive architecture to enable energy efficiency of cloud applications (see Figure 1). It extends the work of Weyns [10]. Weyns presents the architecture of SASs into four components: i) *Managed System*, which represents the application that should be adapted, ii) *Managing System*, which is responsible to decide when and how adapt the *Managed System*, iii) *Adaptation Goals*, which define what is the adaptation aims, and iv) *Environment*, which refers to the external world with which the SAS interacts. To cope with environment's changes, adaptations on the managed system could be required.

The architecture of the managing system can be based on the MAPE-K (Monitor, Analyse, Plan, Execute and Knowledge) model [11]. *Monitor* gathers information from the managed system and the environment context. Using monitored data,

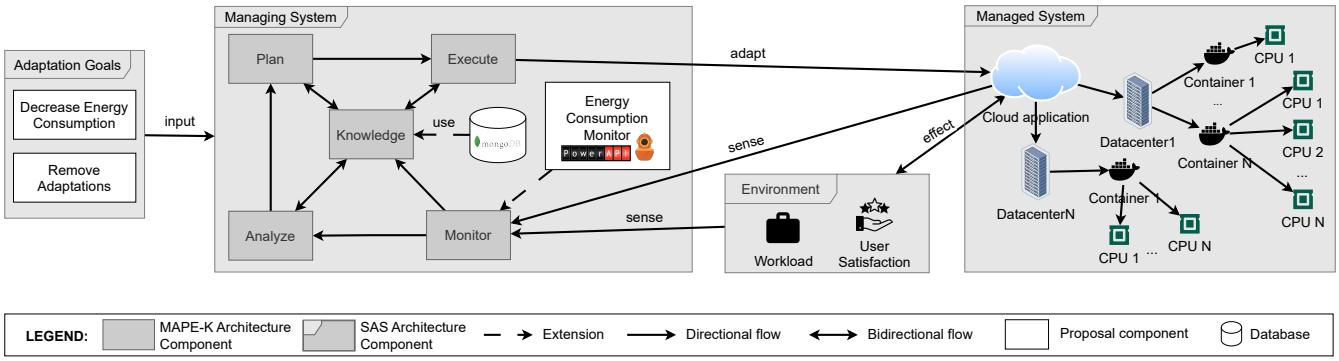

Fig. 1. Self-adaptive architecture for energy-efficient cloud applications

Analyze determines if an adaptation of the managed system is needed or not. *Plan* decides what is the adaptation strategy that will address the situation reported by the analysis. Finally, *Execute* deploys the adaptation plan into the managed system. *Knowledge* refers to all information needed for adapting the managed system, *e.g.,* model of system architecture, variants of systems, etc.

Our contribution is represented by white boxes in Figure 1. *Energy Consumption Monitor* extends the *Monitor* component, it gathers the energy and power consumed by the *managed system*, which in our research refers to cloud applications. Cloud applications are represented by a set of datacenters running different Docker[1] containers. Each container receives a quantity of available CPUs and this value can be modified at run-time. Docker CPU configuration has default value as "unlimited", it means that we can use all available machine's CPU in the cloud provider environment. However, when an adaptation is performed in the number of CPUs, Docker does not allow to re-adapt to the "unlimited" value, so it requires always a value. We can adapt the number of CPUs considering as maximum value the CPU limit that was established.

The power consumption of cloud applications is collected using PowerAPI[2] and Scaphandre[3] tools. These tools monitor instantaneous power consumption of cloud applications, using the collected power consumption we are then able to infer the energy consumed by cloud applications.

In our proposal, the power values are stored using a database. PowerAPI stores the power consumption in a MongoDB database. Scaphandre does not provide the same functionality as PowerAPI, the values are stored in memory. Based on the PowerAPI's functionality, we collect the power using Scaphandre and store them in a MongoDB database. To do that, we collect the power using Prometheus Scaphandre's option. After collect the power consumption, using both tools, a filtering process of the data is necessary to obtain the power consumption of a specific application by using its process id.

For *Analyze*, the proposal uses a rule-base approach to achieve the adaptation goals. One adaptation goal is *Decrease Energy Consumption* that focuses on identifying an increase of the energy consumed by the cloud application that is limited by a threshold. It triggers adaptations in case this threshold is violated. Moreover, once the analysis of the monitored energy consumption is less than an established value, *i.e.,* this value represents the lowest energy consumption for the application that indicates a performing energy, the SAS adapts to remove the previous adaptations achieving the goal *Remove Adaptations*. It allows to avoid working on limits of the cloud application.

Plan should analyze the variants of the cloud application and their associated energy consumption to allows the decision making of the adaptation. *Execute* should analyze the architecture and code of the cloud application to adapt it. To adapt a cloud application, it should be able to receive adaptation plans to be executed, it is made through actuators, *e.g.,* using an endpoint to be used by the *Execute*. As variants of the cloud application, the literature provides a set of ideas, *e.g.,* algorithms to make scheduling and processing of workloads or host scaling [7], [12], vertically and horizontally [13]. First experiments that we performed consider to modify the number of configured CPUs of Docker container. Another experiments that we executed consider to modify the applications them-self that were running in Docker containers.

Environment presents the main attributes that are affected by the cloud application, *e.g., Workload* and *User Satisfaction*. The energy consumption can be related directly with the *workload*, once that modifying the workload can impact in the energy consumption. And the *user satisfaction* can be impacted by the adaptations applied by the SAS.

To answer RQ2, one challenge that should be addressed is to manage the fluctuations of energy consumption when a software is monitored. These fluctuations do not allow to use instantaneous values of energy consumption without considering an error range. So, we plan to execute an empirical study to evaluate the error range of energy consumption that is collected through tools such as PowerAPI and Scaphandre. Our purpose is to determine how much this error is and how we can manage it. A second challenge is to decide what is the optimal frequency of the analysis that will not impact the energy consumption of our proposal. Finally, we

[1]https://www.docker.com/
[2]https://powerapi.org/
[3]https://github.com/hubblo-org/scaphandre

could consider other approaches to infer situations that need adaptation using historical data.

We are working on applying the proposed architecture to the TeaStore [14] benchmark microservice to validate our approach, focusing on analyzing the energy consumption of the application and of the adaptivity architecture. The benchmark is a basic web tea store, providing all the functionalities of a store and it is already used in the academia, *e.g.,* multi-objective of autoscale microservice [15] and monitoring applications' dependencies and metrics [16]. Our experimental evaluation intends to compare the energy consumption of a cloud application, with a varied workload, in scenarios where we change the quantity of CPU available in a docker environment and we make code changes in the application. The expected results should help to understand how each adaptation may affect the application and whether energy consumption reduction is obtained with each one of them.

As a final goal, we expect to deliver a fully usable self-adaptive architecture, helping developers deliver energy-efficient applications without requiring adaptations in their solutions.

ACKNOWLEDGMENTS

This research was produced within the framework of Energy4Climate Interdisciplinary Center (E4C) of IP Paris and Ecole des Ponts ParisTech. This research was supported by 3rd Programme d'Investissements d'Avenir [ANR-18-EUR-0006-02]. This work received funding from the France 2030 program, managed by the French National Research Agency under grant agreement No. ANR-23-PECL-0003.

REFERENCES

[1] P. Jamshidi, A. Ahmad, and C. Pahl, "Autonomic resource provisioning for cloud-based software," in *Proceedings of the 9th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, ser. SEAMS 2014. New York, NY, USA: Association for Computing Machinery, 2014, p. 95–104. [Online]. Available: https://doi.org/10.1145/2593929.2593940

[2] R. Moreno-Vozmediano, E. Huedo, R. S. Montero, and I. M. Llorente, "Latency and resource consumption analysis for serverless edge analytics," *Journal of Cloud Computing*, vol. 12, no. 1, p. 108, 2023.

[3] T. Chen, R. Bahsoon, and X. Yao, "A survey and taxonomy of self-aware and self-adaptive cloud autoscaling systems," *ACM Computing Surveys (CSUR)*, vol. 51, no. 3, pp. 1–40, 2018.

[4] M. G. Hassan, R. Hirst, C. Siemieniuch, and A. Zobaa, "The impact of energy awareness on energy efficiency," *Int. Jrnl. of Sustainable Engineering*, vol. 2, no. 4, pp. 284–297, 2009.

[5] M. Dayarathna, Y. Wen, and R. Fan, "Data center energy consumption modeling: A survey," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 732–794, 2016.

[6] F. Khomh and S. A. Abtahizadeh, "Understanding the impact of cloud patterns on performance and energy consumption," *J. Syst. Softw.*, vol. 141, pp. 151–170, 2018. [Online]. Available: https://doi.org/10.1016/j.jss.2018.03.063

[7] M. Tsenos, A. Peri, and V. Kalogeraki, "Energy efficient scheduling for serverless systems," in *2023 IEEE International Conference on Autonomic Computing and Self-Organizing Systems (ACSOS)*, 2023, pp. 27–36.

[8] A. Alhindi, K. Djemame, and F. B. Heravan, "On the power consumption of serverless functions: An evaluation of openfaas," in *2022 IEEE/ACM 15th International Conference on Utility and Cloud Computing (UCC)*, 2022, pp. 366–371.

[9] M. Xu, A. N. Toosi, and R. Buyya, "A self-adaptive approach for managing applications and harnessing renewable energy for sustainable cloud computing," *IEEE Transactions on Sustainable Computing*, vol. 6, no. 4, pp. 544–558, 2021.

[10] D. Weyns, "Software Engineering of Self-adaptive Systems," *Handbook of Software Engineering*, p. 399, 2019.

[11] IBM, "An architectural blueprint for autonomic computing," IBM, Tech. Rep., Jun. 2005.

[12] M. Xu, A. N. Toosi, and R. Buyya, "A self-adaptive approach for managing applications and harnessing renewable energy for sustainable cloud computing," *CoRR*, vol. abs/2008.13312, 2020. [Online]. Available: https://arxiv.org/abs/2008.13312

[13] T. Chen and R. Bahsoon, "Symbiotic and sensitivity-aware architecture for globally-optimal benefit in self-adaptive cloud," in *Proceedings of the 9th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, ser. SEAMS 2014. New York, NY, USA: Association for Computing Machinery, 2014, p. 85–94. [Online]. Available: https://doi.org/10.1145/2593929.2593931

[14] J. von Kistowski, S. Eismann, N. Schmitt, A. Bauer, J. Grohmann, and S. Kounev, "TeaStore: A Micro-Service Reference Application for Benchmarking, Modeling and Resource Management Research," in *26th IEEE Int. Symposium on the Modelling, Analysis, and Simulation of Computer and Telecommunication Systems*, ser. MASCOTS '18, Sep. 2018.

[15] A. Horn, H. M. Fard, and F. Wolf, "Multi-objective hybrid autoscaling of microservices in kubernetes clusters," in *Euro-Par 2022: Parallel Processing*, J. Cano and P. Trinder, Eds. Cham: Springer International Publishing, 2022, pp. 233–250.

[16] M. Elsaadawy, A. Lohner, R. Wang, J. Wang, and B. Kemme, "Dymond: dynamic application monitoring and service detection framework," in *Proceedings of the 22nd International Middleware Conference: Demos and Posters*, ser. Middleware '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 8–9. [Online]. Available: https://doi.org/10.1145/3491086.3492471