PaaS and IaaS layers optimization for energy-adaptive applications

Corentin Dupont, Mehdi Sheikhalishahi, Federico Facca
Create-Net
Trento, Italy
{*cdupont, msheikhalishahi, ffacca*}*@create-net.org*

***Abstract*—**

Data centres are powerful and power-hungry facilities which aim at hosting ICT services. In the last years, the research trend was to reduce the overall consumption of a data centre so as to reduce its energy footprint. Some of those research results were integrated inside the current data centre management frameworks. This had a positive impact, probably together with other external factors: a recent study showed that the energy consumption of data centres is actually less than what was previously predicted. However, from a research perspective, the field is now already well-studied. This calls for a change in the research direction: instead of aiming at consuming less in data centres, we argue that a new research goal should be to prioritize the consumption of green energies; i.e. "consume better". In this work we present techniques to augment the ratio of renewable energies consumed in data centres. We introduce the concept of Service Flexibility Agreement (SFA), an extension of the traditional SLA able to qualify the flexibility of applications available in a PaaS environment. We then detail preliminary models able to exploit this flexibility in order to increase the ratio of renewable energy consumed. Finally, we show how PaaS and IaaS layers can collaborate to achieve this result.

***Keywords*-Platform as a Service, Energy Management, Flexibility, Scalability**

I. Introduction

A recent study [9] showed that, while still growing, the energy consumption of data centres is actually less than previously excepted. Electricity used in US data centres in 2010 was significantly lower than predicted by the EPAs 2007 report to Congress on data centres. There is a combination of factors explaining this slow down, among which the application of new energy policies in data centres.

This is why we think the attention of researchers in the field of energy management in Cloud Computing should shift from the paradigm of "consume less" to the paradigm of "consume better". As important energy consumers, it is important that the energy management and policies of data centres prioritize the consumption of renewable energies over brown energies. However, the main problem with the utilization of renewable energies is that they are very variable in time. To adapt to such energies, we need to adapt and shift the workload of applications. This means reducing the workload when there is less renewable energies available, for example.

In this paper we analyse how a PaaS on top of IaaS may bring new opportunities in energy management. We think that a PaaS layer brings great opportunities for a better energy management in data centres, because it centralizes and standardizes the scalable deployment of applications. Typically, an application manager will issue a command to a PaaS framework to scale up or down an application, according to predefined patterns. Some frameworks allow to automatize this process: Cloudify[1], for example, provides a language for auto-scaling. This language defines Key Performance Indicators (KPIs) and thresholds that will trigger the scaling operations. For example, in the case of a 3-tier Web server application, it is possible to describe that if the latency in serving the pages goes over a certain threshold, a new front-end VM should be launched.

However, when it comes to the adaptation of applications workload to the availability of the renewable energies, we think that a piece is missing. Indeed, currently the PaaS framework has no way to lower or postpone workload in a reasonable way when there is no renewable energy, for instance. We believe that the PaaS paradigm should be enhanced to better describe the flexibility of applications in a centralized and standardized way. This would allow to perform optimizations at data centre level, such as increasing the renewable energy usage.

In this paper we propose the concept of Service Flexibility Agreement (SFA). The SFA is an extended Service Level Agreement (SLA): it includes a description of the flexibility of an application. On the contrary, the SLA usually defines only minimum levels of resources that an application should be guaranteed to have. However, in the context of flexible applications, this is not enough: some applications can accept to have a temporarily reduced performance or shifted activities. Similarly, some applications would benefit from a temporary increase of allocation of resources when renewable energies are available. The SFA defines a simple interface to describe this flexibility between an application and the PaaS framework. This will allow an energy-aware PaaS framework to make better decisions, for instance for applications consolidation and scaling. Finally, our enhanced PaaS framework is installed on top of a IaaS framework: we show how an improved communication between the two can provide better energy usage.

[1] http://www.gigaspaces.com/cloudify-cloud-orchestration/overview

II. Service Flexibility Agreement

In current PaaS architectures, the framework grants a certain flexibility to applications. For example, an application can ask the PaaS framework to spawn more or less VMs or Linux containers[2] according to its needs. However, the flexibility is entirely controlled by the application and/or the application owner. The intuition behind SFA is to delegate some of the flexibility control to the PaaS framework while still guaranteeing the end user satisfaction. With respect to a traditional SLA, the SFA adds a few new dimensions: the possibility for the required resources to vary in time, plus the possibility to qualify violations of the required performance.

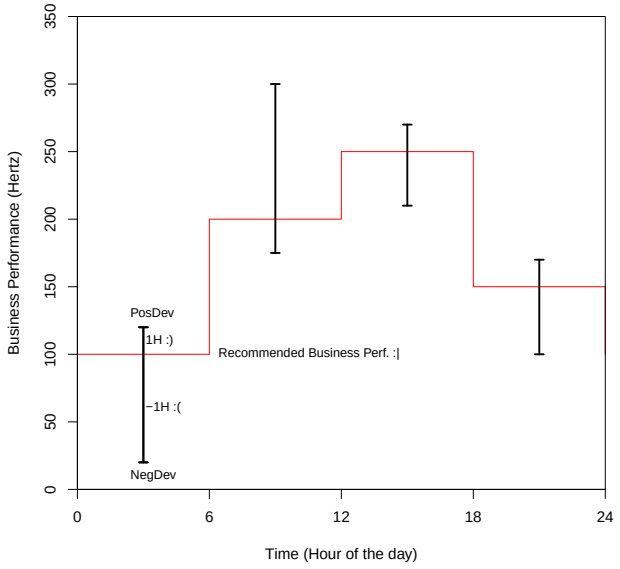

Figure 1: Service Flexibility Agreement representation

As shown in listing 1 and also represented graphically in figure 1, for each time frame of a day the SFA defines a recommended business performance (RecoBP, in red in the figure). The business performance is one of the KPIs of the application. For example, for a Web server it is the number of pages served per minutes, for a video transcoding service it will be how many videos can be transcoded per minutes.

We then define a concept called "Happy points", noted "H". This is an abstraction of the end-user satisfaction. An application having zero Happy points means that the end user is reasonably satisfied. An application being allocated exactly the number of resources corresponding to the RecoBP collects 0 Happy points. This situation corresponds to the traditional SLA threshold. The positive and negative deviations (PosDev and NegDev) declared in the SFA are then the way that each application "reacts" to being given more or less resources than the RecoBP. Indeed, some applications such as video transcoding can benefit from receiving temporarily more resources because they can process more videos and thus make their end user happier. This kind of application reacts linearly to the amount of resources it is allocated. On the other hand, an application such as a web server typically have a "turning point" in their relation between performance and resources. Indeed, giving them less that a certain level of resources (such as the number of front-end VMs) will start to augment the latency in delivering web pages, thus making the end-user unhappy. On the contrary, giving them more than that level of resources will not have any perceptible impact on the end user, as the latency is already small. This is represented by the vertical deviation bars in figure 1: the length of the bar represents the amount of Happy points that the application will win/lose when given more/less performance than the recoBP, respectively. In this example, at 10 Oclock, if the PaaS framework allocates the resource corresponding to the recommended business performance of 200Hz, the application will collect 0 Happy points. If the application gets 300Hz, it will get 1H, and one more Happy point for each 100Hz above that. Conversely, if it gets less than the recommended 200Hz, it will loose Happy point at the rate of one happy point per 25Hz.

We further define the various Working Modes (WM) of an application. A WM corresponds to a set of resources allocated by the PaaS to an application. We associate to each WM its typical power consumption and the maximum

```
SFA:
  - Start: 00:00
    End: 06:00
    RecoBP: 100 Hz
    PosDev: 100 Hz/H
    NegDev: 50 Hz/H
  - Start: 07:00
    End: 12:00
    RecoBP: 300 Hz
    PosDev: 50 Hz/H
    NegDev: 200 Hz/H

WorkingModes:
  - WMName: WM1
    actuator: 'cf scale myApp -i 3'
    defaultPower: '300 W'
    maxBusinessPerf: '100 Hz'
  - WMName: WM2
    actuator: 'cf scale myApp -i 5'
    defaultPower: '500 W'
    maxBusinessPerf: '150 Hz'
```

Listing 1: SFA example

[2]http://docs.cloudfoundry.org/concepts/architecture/warden.html

business performance it can offer. In practice, a WM corresponds to a number of VMs or Linux Containers, each running instances of the application. Using the SFA, it is now possible to compute the number of Happy points provided by each WM for each time slot.

III. Collaboration between IaaS and PaaS layers

In a typical PaaS application, the scaling is performed by launching more containers. Each container contains an instance, or worker, of the application. The containers are running in a VM, controlled by an underlying IaaS framework. To save energy, those VMs are traditionally consolidated on a part of the servers of the data centre, which permits to switch off unused servers and thus save energy. Using the SFA, it is now possible to predict the amount of resource that an application will need, together with the possible deviations. This will allow to optimize the usage of servers by placing VMs on them judiciously, and to minimize the VM movements. However to achieve this we need to enhance the collaboration between the IaaS and PaaS. There are essentially two types of information that need to be exchanged:

- VM grouping
- VM life-time

We believe that the VM grouping is an important information that should be transmitted between the PaaS and the IaaS layers. Indeed the PaaS layer has a certain degree of knowledge about the applications that are deployed. If an application is composed of several VMs, it is beneficial to keep them together on the same node as much as possible, because they will probably have the same life cycle. Those VMs will probably exchange a lot of information among them. Furthermore, they will be switched off together when the application is terminated. This justifies to keep them together on the same node or group of nodes as much as possible. Furthermore, the affinity between VMs is an information that could be transmitted to the IaaS when a PaaS manager asks for VMs creation during application deployment.

The other information that is worth transmitting from the PaaS to the IaaS is the VM life-time: an estimated duration that the VM will be kept running before being switched off. This information is important for the IaaS layer when optimizing the energy consumption of a data centre through VM consolidation: indeed to switch off a server it is necessary to migrate all VMs running on that server. However, migrating a VM is an investment, and if the VM is about to be terminated by the PaaS layer, this investment is lost. Of course, the IaaS layer shouldn't be aware of the applications that are running in the data centre, as it is not its role and would furthermore break the separation of concerns between the IaaS and the PaaS. The two proposed information transmission (VM grouping and VM life-time) does not break the separation of concerns between IaaS and PaaS, because they are expressed in terms of the IaaS VMs only: at IaaS level no knowledge of the running applications should be necessary.

IV. Design & Optimization

The SFA is used in the communication between an application and the underlying PaaS framework (while the SLA, on the contrary, is used for the communication between the data centre and its clients), as shown in figure 2.

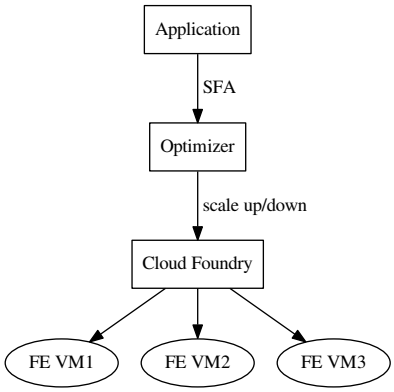

Figure 2: SFA block diagram

In practice, we add a component in the Cloud Foundry stack called the "Optimizer". This component accepts commands from the application load balancer. Instead of just scale-up/scale-down commands as it is the case currently, we include a recommended scaling level, together with a positive deviation and a negative deviation, as described in Section II. This will let the Optimizer to optimize according to multiple criteria: 1) the energy consumption in the data centre, 2) the global happiness of the applications (the sum of all happy points granted to applications) and finally 3) the usage of renewable energies.

V. Related Works

Workload consolidation is a powerful means to improve IT efficiency and reduce power consumption. VM consolidation approaches to reduce energy consumption at IaaS layer have been explored in many recent papers [2] [7] [10] [8] [11] [12] [6].

In addition, there are energy efficient solutions based on scaling operations (scale up/down application) based on applications performance metrics. Although these proposals reduce the energy footprint of applications and cloud infrastructures, they do not model the applications performance trend to finely define a trade-off between applications Quality of Service and energy footprint.

Autonomic Computing has been exploited in the design of cloud computing architectures in order to devise autonomic loops aiming at providing coordinated actions among cloud layers for efficiency measures, turning each layer of the cloud stack more autonomous, adaptable and aware of the runtime environment [1] [5] [4].

In order to reach a global and efficient state due to conflicting objectives, autonomic loops need to be synchronized. In [1], authors proposed a generic model to synchronize and coordinate autonomic loops in cloud computing stack. The feasibility and scalability of their approach is evaluated via simulation-based experiments on the interaction of several self-adaptive applications with a common self-adaptive infrastructure.

In addition to elasticity, scalability is another major advantage introduced by the cloud paradigm. In [13], automated application scalability in cloud environments are presented. Authors highlight challenges that will likely be addressed in new research efforts and present an ideal scalable cloud system.

[3] proposes a co-design of IaaS and PaaS layers for energy optimization in the cloud. The paper outlines the design of interfaces to enable cross-layer cooperation in clouds. This position paper claims that significant energy gains could be obtained by creating a cooperation API between the IaaS layer and the PaaS layer. Authors discuss two complementary approaches for establishing such cooperation: cross-layer information sharing, and cross-layer coordination.

VI. Conclusion and Future Work

The PaaS Cloud paradigm and the corresponding frameworks are currently developing at high-speed. As shown in this paper, they provide new opportunities for energy optimization. We presented the SFA, a concept able to describe the flexibility of applications. This format includes the trade-off between the amount of resources allocated to a PaaS scalable application, the resulting performance and the corresponding user experience, in a simple way. This allows the PaaS framework to perform multi-criteria optimizations such as lowering the global energy consumption of the data centre, using more renewable energies and increasing the application user satisfaction. As future work, we will design the proposed API and components within the Cloud Foundry framework.

Acknowledgments

This work has been carried out within the European Project DC4Cities (FP7-ICT-2013.6.2).

References

[1] F. Alvares de Oliveira, R. Sharrock, and T. Ledoux. Synchronization of multiple autonomic control loops: Application to cloud computing. In *Proceedings of the 14th International Conference on Coordination Models and Languages*, COORDINATION'12, pages 29–43, Berlin, Heidelberg, 2012. Springer-Verlag.

[2] M. Cardosa, M. R. Korupolu, and A. Singh. Shares and utilities based power consolidation in virtualized server environments. In *Proceedings of the 11th IFIP/IEEE Integrated Network Management (IM 2009)*, Long Island, NY, USA, June 2009.

[3] A. Carpen-Amarie, D. Dib, A.-C. . Orgerie, and G. Pierre. Towards energy-aware IaaS-PaaS co-design. In *SMARTGREENS 2014*, pages 203–208, 2014.

[4] F. de Oliveira, T. Ledoux, and R. Sharrock. A framework for the coordination of multiple autonomic managers in cloud environments. In *2013 IEEE 7th International Conference on Self-Adaptive and Self-Organizing Systems (SASO)*, pages 179–188, Sept. 2013.

[5] F. A. de Oliveira, Jr. and T. Ledoux. Self-management of cloud applications and infrastructure for energy optimization. *SIGOPS Oper. Syst. Rev.*, 46(2):10–18, July 2012.

[6] C. Dupont, F. Hermenier, T. Schulze, R. Basmadjian, A. Somov, and G. Giuliani. Plug4green: A flexible energy-aware vm manager to fit data centre particularities. *Ad Hoc Networks*, 25:505–519, 2015.

[7] P. Graubner, M. Schmidt, and B. Freisleben. Energy-efficient virtual machine consolidation. *IT Professional*, 15(2):28–34, 2013.

[8] F. Hermenier, X. Lorca, J.-M. Menaud, G. Muller, and J. Lawall. Entropy: a consolidation manager for clusters. In *Proc. of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual Execution Environments, VEE 2009*, pages 41–50. ACM, 2009.

[9] J. Koomey. Growth in data center electricity use 2005 to 2010. *Oakland, CA: Analytics Press. August*, 1:2010, 2011.

[10] K. Schröder and W. Nebel. Behavioral model for cloud aware load and power management. In *Proc. of HotTopiCS '13, 2013 international workshop on Hot topics in cloud services*, pages 19–26. ACM, May 2013.

[11] M. Sheikhalishahi, I. M. Llorente, and L. Grandinetti. Energy aware consolidation policies. In *International Conference on Parallel Computing*, pages 109–116, Belgium, Sept. 2011. IOS Press.

[12] M. Sheikhalishahi, R. M. Wallace, L. Grandinetti, J. L. Vazquez-Poletti, and F. Guerriero. A multi-capacity queuing mechanism in multi-dimensional resource scheduling. In F. Pop and M. Potop-Butucaru, editors, *Adaptive Resource Management and Scheduling for Cloud Computing*, Lecture Notes in Computer Science, pages 9–25. Springer International Publishing, Jan. 2014.

[13] L. M. Vaquero, L. Rodero-Merino, and R. Buyya. Dynamically scaling applications in the cloud. *SIGCOMM Comput. Commun. Rev.*, 41(1):45–52, Jan. 2011.