Design Support for Performance-aware Cloud Application (Re-)Distribution

Santiago Gómez Sáez and Frank Leymann

IAAS, University of Stuttgart
Universitätsstr. 38, 70569 Stuttgart, Germany
{gomez-saez,leymann}@iaas.uni-stuttgart.de

Abstract. The Cloud computing paradigm emerged by establishing innovative resources provisioning and consumption models. Together with the improvement of resource management techniques, these models have contributed to an increase in the number of application developers that are strong supporters of partially or completely migrating their application to a highly scalable and pay-per-use infrastructure. However, due to the continuous growth of Cloud providers and Cloud offerings, Cloud application developers nowadays must face additional application design challenges related to the efficient selection of such offerings to optimally distribute the application in a Cloud infrastructure. Focusing on the performance aspects of the application, additional challenges arise, as application workloads fluctuate over time, and therefore produce a variation of the infrastructure resources demands. In this research work we aim to define and realize the underpinning concepts towards supporting the optimal (re-)distribution of an application in the Cloud in order to handle fluctuating over time workloads.

Keywords: Cloud application distribution; application performance; application workload evolution; Cloud application topology

1 Introduction

In the last years the Cloud computing paradigm has successfully emerged and its model has been largely adopted by industry and research domains. Together with the exponential increase of available Cloud services, application developers can decide between partially or completely deploying their applications in a Cloud infrastructure. For example, with the successful introduction of DBaaS solutions, it became possible to host only some of the application components off-premise (in the Cloud), e.g. its database, while the remaining of the application remains on-premise.

Standards like TOSCA[1] allow for the modeling and management of application topology models in an interoperable and dynamic manner, further supporting the application distribution capabilities, potentially even in a multi-Cloud environment. However, such technological approaches lack support for guiding the application developer in tasks related to efficiently selecting the Cloud offerings to distribute the

[1] Topology and Orchestration Specification for Cloud Applications (TOSCA) Version 1.0: http://docs.oasis-open.org/tosca/TOSCA/v1.0/TOSCA-v1.0.html

application. In this work, we aim to leverage the opportunities provided by such a technological landscape and develop the means to allow the dynamic deployment and re-deployment of application components across multiple services, in order to cope with performance demands which evolve during the life of the application. There are two fundamental observations in this effort that are going to be discussed in more length during the rest of the paper. Firstly, the distribution of the application topology in the Cloud has a severe effect on the performance of the application — however it is not always obvious whether it is beneficial or detrimental. Secondly, a realistic application workload fluctuates over time, and its topology may have to be adapted to address the workload evolution.

Approaches such as MADCAT focus on providing a methodological approach targeting the design and creation of structured native applications [10]. The CloudMig approach builds on an initial topology [7] of the application which is then adapted through model transformation based on existing cloud offerings. The Palladio Component Model[2] aims at predicting the performance of model-driven software architectures, and is used in [12] towards optimizing for availability and operational expenses. Similar model-driven approaches focusing on multi-cloud management environments are investigated in the MODAClouds[3] project. The MOCCA framework focuses on optimizing the application topology by introducing variability points in the topology model and using optimization techniques to find the most suitable cloud offerings [11]. However, such approaches either do not explicitly focus on the performance-aware aspects of the application, or provide decision support mechanisms during the design phase of the application. In this work we aim at going a step further by *providing the decision support mechanisms to Cloud application developers to efficiently (re-)distribute their applications to cope with evolving application workload behaviors, and performance and resource demands.*

2 Motivation & Problem Statement

The deployment of an application in the Cloud often requires Cloud application developers to specify its underlying resources, calculate its cost, or analyze its expected performance, etc. Such tasks are supported as part of several approaches such as CloudML [5] or TOSCA. For example, the Policy4TOSCA approach enables a policy-based description of application non-functional aspects [13]. In Figure 1 we depict the topology model of a simple web shop application, which is constituted by its front-end and logic, and a back-end database. In a first step, the Cloud application developer may consider running the complete application stack in an on-premise virtualized infrastructure. Consequently, the required infrastructure must be provisioned in order to satisfy the functional and non-functional aspects of the application. However, when deciding to partially or completely distribute the application among an off-premise Cloud infrastructure, Cloud application developers must face further challenges related to deciding which Cloud provider and Cloud offering to use to partially

[2] Palladio Component Model: `http://sdqweb.ipd.kit.edu/wiki/Palladio_Component_Model`

[3] MODAClouds: `http://www.modaclouds.eu/`

or completely run the application. For example, the application database can be deployed in the AWS RDS DBaaS infrastructure[4] and the application logic in an AWS EC2[5] *m1.medium* instance or in an AWS Elastic Beanstalk[6] container.

The existence of multiple Cloud offerings, which are in some cases provided by several Cloud providers, builds an *alternative topologies space* depicting all possible application distribution alternatives [2]. Such application distribution posibilities are also motivated from the perspective of the multiple application partial and complete migration categories presented in [1]. Cloud application developers must analyze and evaluate the alternative topologies space towards achieving an efficient selection of Cloud offerings to distribute the application. The existence of such alternatives introduces a multi-dimensional problem related to evaluating and deciding among such alternatives. Cloud providers typically offer Cloud application developers tools for targeting one dimension, e.g. to calculate and analyze the monetary cost when using their offered services, such as the Amazon Simple Monthly Calculator[7], and provide configuration samples for different applications types and resources demands.

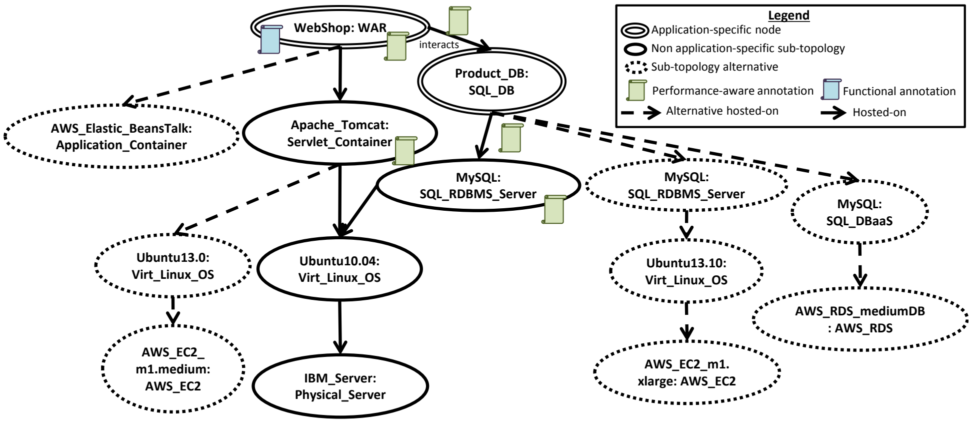
WebShop: WAR
interacts
Product_DB: SQL_DB
AWS_Elastic_BeansTalk: Application_Container
Apache_Tomcat: Servlet_Container
MySQL: SQL_RDBMS_Server
MySQL: SQL_RDBMS_Server
MySQL: SQL_DBaaS
Ubuntu13.0: Virt_Linux_OS
Ubuntu10.04: Virt_Linux_OS
Ubuntu13.10: Virt_Linux_OS
AWS_RDS_mediumDB : AWS_RDS
AWS_EC2_ m1.medium: AWS_EC2
IBM_Server: Physical_Server
AWS_EC2_m1. xlarge: AWS_EC2
Legend
Application-specific node
Non application-specific sub-topology
Sub-topology alternative
Performance-aware annotation
Functional annotation
Alternative hosted-on
Hosted-on

Fig. 1. Web Shop Application Topology Model (extended from [2])

The application performance is typically evaluated and analyzed in all phases of its life cycle, and therefore has a significant impact on the application design decisions. Moreover, its workload fluctuation and the impact on the expected performance and resources demands arises several additional challenges. For example, a web shop application workload can increase at certain time periods, e.g. before the Christmas season, and therefore would require the usage and configuration of concrete resources towards satisfying the expected performance. However, such configuration may generate unnecessary monetary costs in time periods with a lower performance

[4] AWS RDS: `http://aws.amazon.com/rds/`

[5] AWS EC2: `http://aws.amazon.com/ec2/`

[6] AWS Elastic Beanstalk: `http://aws.amazon.com/elasticbeanstalk/`

[7] Amazon Simple Monthly Calculator: `http://calculator.s3.amazonaws.com/index.html`

demand. Optimizing the application distribution towards balancing the performance-cost trade-off must be considered as a long-term collaborative task which focuses on the one hand on the evolutionary aspect of the application workload, and on the other hand ensures that the triggered resource allocations and dynamic adaptations comply with the expected service objectives. Cloud elasticity techniques aim at dynamically and automatically pulling and releasing of computational resources. However, most providers nowadays offer reactive-based elasticity features which must be configured in advance by the developer, e.g. static thresholds definition in AWS Autoscaling[8].

3 Research Challenges

Providing therefore the Cloud application developers with such design support to optimally distribute and re-distribute the application to cope with fluctuating workloads and performance demands raises several challenges. Such decision support must cover the complete application life-cycle, define the underpinning concepts, and provide the required mechanisms towards targeting the analysis and evaluation of the evolutionary aspects of the application performance, e.g. its workload fluctuation. The following research challenges are identified as part of this research work:

1. How to partially or completely specify during the design phase the Cloud application topology and its performance-aware aspects.
2. Based on such specification, we must provide the techniques to derive the alternative topologies among existing Cloud offerings and prune the alternative space conforming to application performance requirements.
3. The analysis of the application workload behavior and its resource demands evolution towards
4. deriving and assessing the Cloud application developer with efficient application (re-)distribution alternatives and profitable resources configuration.

4 Work in Progress & Research Plan

As a first step of this research work, we focused on three-layered applications defined in [6]. The optimal distribution of the application in the Cloud is targeted as in [2] by proposing a technology agnostic framework for deriving the multiple topology alternatives and selecting the optimal application distribution through the usage of utility-based ranking techniques. Focusing on the application performance, in [9] we identified the need to partially or completely distribute the application layers among multiple Cloud offerings to cope with application workloads fluctuations. Two approaches for analyzing the application workload behavior and evolution were identified: *top-down* and *bottom-up*. The top-down approach comprises the application workload characterization and the application behavior model derivation, before or during the deployment of the application. However, the top-down analysis approach is restricted to handling the workload evolution over time. Bottom-up approaches address this deficiency with the help of resource consumption monitoring techniques

[8] AWS Autoscaling:`http://aws.amazon.com/autoscaling/`

and performance metrics. Both top-down and bottom-up analysis approaches can be combined over time in order to support the dynamic (re-)destribution of an application topology to cope with varying resources demand [9].

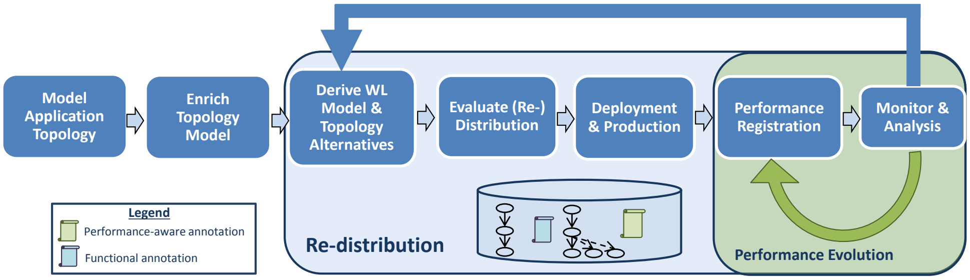

Fig. 2. Performance-aware Application (Re-)Distribution Process

Based on the consolidation of the previous analysis approaches, we then proposed an application analysis and distribution process which can be used to enable the application (re-)distribution based on dynamic analysis of the workload [9]. As depicted in Figure 2, such process consists of several tasks: (i) modeling the application topology, (ii) enriching such topology with performance awareness, e.g. expected performance or workload behavior, (iii) deriving the alternative topologies space and the workload distribution model, (iv) pruning the alternative topologies space by using utility-based evaluation techniques and based on historical knowledge or empirical results, (v) deploying the application, and (vi) registering the application performance demands and workload behavior evolution during its production phase through monitoring techniques. We proposed the *Collaborative Loop* as an approach to support the (re-)distribution of the application over time to proactively react to fluctuating workloads. Implementing the toolchain required as part of this process and creating a comprehensive framework for application distribution support is our main task in ongoing work, as a number of tools are already in place both for workload analysis and application topology management. In this respect, our focus is on integrating them, rather than developing them from scratch, except from when deemed necessary, as for example in the case of defining a performance-aware deployment language and container for the Cloud. Ongoing work focus on fleshing out the individual process tasks and connecting them with the specific techniques and tools. For example, the TOSCA specification can be used for specifying the Cloud application topology and the Policy4TOSCA for indicating the non-functional aspects of the application [13]. Application workloads characteristics can be also specified using the GT-CWSL language [3]. During the application workload analysis and generation tasks, existing tools such as the Faban Harness[9] or the Rain [4] workload generator can be integrated.

Driven experiments showed significant performance improvement of the application database layer when migrating its data to IaaS or DBaaS solutions, showing

[9] Faban: `http://faban.org`

the latter to have the most improved performance for different workload characteristics [9]. In [8] we evaluated different caching strategies which can be utilized for mitigating the migration of the application data to the Cloud and its transparent access through a message-based middleware. Future work comprises the evaluation of the performance of the overall process when (re-)distributing, i.e. (re-)deploying the different application components. Utility-based techniques can be helpful to investigate the relationship between user preferences and application performance, as well as the usage monitoring and analysis tools and approaches.

Acknowledgment

This work is partially funded by the FP7 EU-FET project 600792 ALLOW Ensembles. Many thanks to Vasilios Andrikopoulos for his invaluable input.

References

1. Andrikopoulos, V., Binz, T., Leymann, F., Strauch, S.: How to Adapt Applications for the Cloud Environment. Computing 95(6), 493–535 (2013)
2. Andrikopoulos, V., Gómez Sáez, S., Leymann, F., Wettinger, J.: Optimal Distribution of Applications in the Cloud. In: Proceedings of CAiSE'14. Springer (June 2014)
3. Bahga, A., Madisetti, V.K.: Synthetic Workload Generation for Cloud Computing Applications. Journal of Software Engineering and Applications 4, 396–410 (2011)
4. Beitch, A., Liu, B., Yung, T., Griffith, R., Fox, A., Patterson, D.A.: Rain: A Workload Generation Toolkit for Cloud Computing Applications. Tech. Rep. UCB/EECS-2010-14, University of California (2010)
5. Brandtzæg, E., Mohagheghi, P., Mosser, S.: Towards a domain-specific language to deploy applications in the clouds. In: Proceedings of Cloud Computing 2012. pp. 213–218. IARIA (2012)
6. Fowler, M.: Patterns of Enterprise Application Architecture. Addison-Wesley Professional (2002)
7. Frey, S., Hasselbring, W.: The CloudMIG approach: Model-based migration of software systems to cloud-optimized applications. International Journal on Advances in Software 4(3 and 4), 342–353 (2011)
8. Gómez Sáez, S., Andrikopoulos, V., Leymann, F., Strauch, S.: Evaluating Caching Strategies for Cloud Data Access using an Enterprise Service Bus. In: Proceedings of IC2E'14 (2014)
9. Gómez Sáez, S., Andrikopoulos, V., Leymann, F., Strauch, S.: Towards Dynamic Application Distribution Support for Performance Optimization in the Cloud. In: Proceedings of CLOUD'14 (June 2014)
10. Inzinger, C., Nastic, S., Sehic, S., Voegler, M., Li, F., Dustdar, S.: MADCAT - A Methodology For Architecture And Deployment Of Cloud Application Topologies. In: Proceedings of SOSE'14 (2014)
11. Leymann, F., Fehling, C., Mietzner, R., Nowak, A., Dustdar, S.: Moving Applications to the Cloud: An Approach based on Application Model Enrichment. IJCIS 20(3), 307–356 (October 2011)
12. Miglierina, M., Gibilisco, G., Ardagna, D., Di Nitto, E.: Model based control for multi-cloud applications. In: Proceedings of MiSE'13. pp. 37–43 (2013)
13. Waizenegger, T., Wieland, M., Binz, T., Breitenbücher, U., Haupt, F., Kopp, O., Leymann, F., Mitschang, B., Nowak, A., Wagner, S.: Policy4TOSCA: A Policy-Aware Cloud Service Provisioning Approach to Enable Secure Cloud Computing. In: OTM'13