Transitioning to the Cloud? A Model-driven Analysis and Automated Deployment Capability for Cloud Services

Faruk Caglar, Kyoungho An, Aniruddha Gokhale and Tihamer Levendovszky
Vanderbilt University, ISIS and EECS
Nashville, TN, USA
{faruk.caglar,kyoungho.an,a.gokhale,tihamer.levendovszky}@vanderbilt.edu

ABSTRACT

As cloud computing becomes increasingly popular and appealing, application and service providers increasingly face questions on whether moving to the cloud would be beneficial to their business, and how should the cloud deployment of their application be realized. Analysis techniques, such as simulations, hold promise in analyzing the benefits of moving to the cloud, and while generative mechanisms can automate the deployment of applications in the cloud. This paper describes how model-driven engineering (MDE) supports both these desired capabilities by providing intuitive and automated capabilities for driving simulations of cloud infrastructures and application services to analyze the benefits of moving the applications to the cloud, and automating the deployment of these applications in the cloud.

Categories and Subject Descriptors

I.6.3 [**Computing Methodologies**]: Simulation and Modeling—*Applications, Tools*

General Terms

Design, Performance

Keywords

model-driven analysis, simulation, deployment

1. MOTIVATION

Cloud computing [1] is increasingly becoming an attractive option for service providers to host their services without having to worry about procuring and maintaining the resources needed to host their services, or adapt to changing customer workloads. Despite this huge appeal, not every application or service is currently hosted in the cloud infrastructure. Many service providers often face a dilemma on whether it is the right time to move their services to the cloud or not. Compounding the problem further, a general lack of analysis tools makes it hard for them to make informed decisions on the benefits of moving to the cloud.

Even if the service providers were to decide to move to the cloud, they must grapple with a range of issues. First, different cloud providers provide different models for hosting customer's services/applications, which requires these customers to understand the cloud provider's hosting model. This includes making informed decisions on the type of virtual machine defined by the operating system it runs and the resources allocated to it that will suffice their needs. Second, given the distributed nature and scale of the service being moved to the cloud, the customer is required to make informed choices on the number of virtual machines to use and the topology of the interconnection as well as mapping of their service components to the virtual machine topology. Third, a lack of standardization has resulted in different cloud providers supporting proprietary APIs[1] to access cloud resources, which in turn can also involve an additional dimension of heterogeneity in terms of the programming language used (*e.g.*, Python, Ruby, Java, C++).

Thus, the key research questions that remain unresolved are: when is moving to cloud beneficial and if so, how should the deployment of an application in the cloud be realized? One approach to overcome this problem is for the service providers to resort to non-scientific, trial-and-error based approaches where the service providers can try deploying their services on the cloud and ascertain whether any benefit is derived. However, this approach requires the potential cloud user to learn the APIs and resort to trial-and-error deployments, both of which are undesirable.

Model-driven engineering (MDE) [4] holds promise in overcoming the need for trial-and-error while also significantly reducing the learning curve involved in moving to a cloud platform. MDE offers the following benefits when used in the context of cloud hosting:

- *Alleviates the learning curve impediment:* With even very little technical knowledge of the cloud provider-specific API, the deployment model in a modeling tool, such as Generic Modeling Environment (GME), could be designed with ease and the deployment script could be generated by executing the model interpreter.
- *Target language independence:* Since the model is agnostic of the target language used to write the scripts for deploying cloud-based applications, any programming language or library could be the target of the model-based automated deployment capability, which

[1]Some APIs like EC2 are becoming de facto standards.

MDHPCL '12, Oct 01 - October 05 2012, Innsbruck, Austria

is achieved by decoupling the model from the interpreter that generates the language-specific scripts.

- *Visualization:* Since visual modeling tools provide visual representation of the components of the model, generating automated deployment script by modifying the attributes and connections of the model components is intuitive, powerful and flexible.

Prior work presented in [5] provides a model-based proxy for unified Infrastructure as a Service (IaaS) management. The purpose is to manage services provided by any cloud platform from a common interface, however, this work does not provide price simulation and automated deployment as described in Section 2. EMUSIM is another simulation environment which supports the modeling, evaluation, and validation of performance of Cloud computing applications. It is built on top of Cloudsim that we have used in our research [2].

2. CONTRIBUTIONS

Our MDE solution comprises the following elements. To enable a potential cloud customer to analyze the benefits of moving to the cloud, our MDE solution makes it easier for a user to test their services/applications and obtain price/performance insights by simulating in a cloud simulator called CloudSim [3]. Results of the simulation provide feedback to user about beneficial deployment choices. The users subsequently use the deployment modeling capabilities of our MDE solution to model the deployment of their applications in the cloud. Subsequently, generative capabilities of our MDE solution synthesize the deployment scripts pertaining to the cloud platform, which automates the deployment of applications on the cloud.

The cloud hosting analysis is conducted via simulations using the CloudSim simulator. We do not require the user to be aware of the simulator capabilities. Instead, our domain-specific modeling language (DSML) requires the user to supply the price and performance bounds for hosting their service on the cloud. The modeling tool generates the simulation script by running the model interpreter. The generated simulation script is then executed by Cloudsim tool and the price results as well as expected performance for the specified price is displayed. These results provide sufficient feedback to the user on whether moving to the cloud is beneficial to their business.

If the user is satisfied with the results, they move to the next stage of modeling the deployment proposed by the simulation results. Although in our current work we expect the users to manually model this deployment, our future work is exploring automated transformation of simulation results into deployment models conforming to our cloud deployment DSML. Model interpreters associated with this DSML synthesize deployment scripts that are used to deploy the applications on the cloud. One interpreter we have currently implemented synthesizes the deployment scripts using Python-based APIs for Amazon EC2 platform.

3. CONCLUSION AND FUTURE WORK

This paper presented ongoing work on a model-based simulation and automated deployment in cloud. This MDE tooling is essential to help users analyze the benefits of moving their services to the cloud, and the expected resulting performance of their services for a given price. This analysis is conducted using simulation tools, however, the user is completely shielded from having to learn the simulator since the MDE tool generates scripts to run the simulator. Based on the results of this analysis, the MDE tooling enables the user to model the deployment of their services in the cloud. The MDE tool synthesizes deployment scripts for the cloud thereby shielding the user from having to manually write scripts using low-level APIs, which is error-prone and tedious. We used an representation example to test our hypothesis in our in-house cloud environment.

In this research, two separate models were used for cloud simulation and automated deployment. Both models are run and generate results independently. To address this limitation, the meta-models need to be combined and only a single model should be designed for price simulation and automated deployment. In our deployment model, we support only a limited number of cloud service-provider API commands for deployment purposes. In our future work, we will include additional cloud service-provider APIs in the deployment model, which will help to generate more efficient, flexible, and robust deployment scripts.

Additionally, in our current cloud simulation and automated deployment model interpreters, the scripts for Cloudsim and Amazon EC2 API are generated. The generated scripts for both simulation and deployment parts should be independent from the simulation tool and cloud service-provider API, respectively. The other simulation tools such as EMUSIM should be supported as well to be able to compare the simulation results and make better decisions. Moreover, automated deployment model interpreter should support all major cloud service-providers such as GoGrid, OpenNebula, and Rackspace.

Acknowledgments

This work was supported in part by NSF CNS SHF 0915976 and CNS CAREER 0845789. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

4. REFERENCES

[1] M. Armbrust, A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, et al. A View of Cloud Computing. *Communications of the ACM*, 53(4):50–58, 2010.

[2] R. N. Calheiros, M. A. Netto, C. A. De Rose, and R. Buyya. EMUSIM: An Integrated Emulation and Simulation Environment for Modeling, Evaluation, and Validation of performance of Cloud Computing Applications. *Software: Practice and Experience*, pages 00–00, 2012.

[3] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. D. Rose, and R. Buyya. CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms. *Software: Practice and Experience*, 41(1):23–50, jan 2011.

[4] D. C. Schmidt. Model-Driven Engineering. *IEEE Computer*, 39(2):25–31, 2006.

[5] S. Yan, B. S. Lee, and S. Singhal. A Model-Based Proxy for Unified IaaS Management. 2010.