Summary: Architecture Design Evaluation of PaaS Cloud Applications using Generated Prototypes

David Gesvindr[1], Ondrej Gasior[1] and Barbora Buhnova[1]

[1]*Faculty of Informatics, Masaryk University, Brno, Czech Republic*

Abstract

Platform as a Service (PaaS) cloud brings great benefits of an elastic platform with many prefabricated services. At the same time, however, it challenges software architects who need to navigate a rich set of PaaS services, variability of PaaS cloud environment and quality conflicts in existing design tactics, which makes it very hard to foresee the impact of architectural design decisions on the overall application quality. To support the architects in the design of PaaS cloud applications, we propose a design-time quality evaluation approach for PaaS cloud applications based on automatically generated prototypes, which are deployed to the cloud and evaluated in the context of multiple quality attributes and environment configurations. In this paper, we outline the approach and its implementation in a prototype generation and evaluation tool, referred to as PaaSArch Cloud Prototyper[1].

Keywords

Cloud Computing, PaaS Cloud, Software Architecture Design, Prototype Generation, Quality Evaluation

1. Introduction

PaaS cloud has over time became a popular platform for hosting software applications, which however need to be architected consciously to take advantage of the Paas cloud services [2, 3]. Unfortunately, existing architectural patterns and quality assessment methods used for on-premise applications provide very little guidance to software architects during this endeavour. The existing model-based quality assessment methods struggle in the PaaS cloud context namely due to missing information about the inner architecture of the PaaS cloud services, which is in majority of cases not published by the cloud provider. Moreover, most analytical tools work well in an environment with known architecture and amount of available resources (CPU, memory, I/O operations, network bandwidth) but experience problems in the PaaS cloud environment, as these required inputs are not available. A tool capable of quality (e.g., performance) predictions of modelled application in the PaaS cloud has to deal with the following challenges as described in [1]: *hidden complexity of the platform*, *multi-tenant environment* and *frequent updates of the environment*. These aspects that make it practically impossible to estimate PaaS cloud application performance and other characteristics without its actual deployment to the cloud. Thus, software architects are often relying on costly manual implementation of application prototypes, profiled after deployed in the PaaS cloud target environment. That is however very tedious and costly.

[1]Use the original publication when citing this work [1]

ECSA 2021

gesvindr@mail.muni.cz (D. Gesvindr); 448273@mail.muni.cz (O. Gasior); buhnova@mail.muni.cz (B. Buhnova)

To address this, we propose an approach and tool set, called PaaSArch Cloud Prototyper[1], supporting automated prototyping of PaaS cloud applications, enabling quick assessment of various architectural options and integrating different PaaS cloud services. The tool receives a model of the application architecture (together with some details about the inner behaviour of the application and its usage of cloud services), and translates it via an automated process into a source code of a fully functional application prototype, which is together with automatically generated sample data deployed to the cloud and benchmarked.

2. Related Work

Model-driven quality prediction approaches are a popular way to support quality-driven design of software systems. To this end, the application is modeled with UML or other domain specific modelling notation, e.g. Palladio Component Model (PCM) [4]. Then, the model of the architecture is automatically transformed into a predictive model (e.g. Markov Chain or Layered Queuing Network) [5] and evaluated. On top of the model, other tools are sometimes built, like SPACE4CLOUD [6] or SimuLizar [7]. However, in Paas cloud, these approaches suffer from the unpredictability of the PaaS cloud environment itself, as discussed above.

Another way to assess the quality of the PaaS applications is to develop a simplified version of the application, deploy the application and run a set of benchmarks. The benefit of this option is that the system resources are closer to the final execution environment [8]. The StressCloud [8] simplifies the process (of full application development) with the generation of synthetic workload based on modeled CPU, memory and IO utilization using deployed agents in virtual machines (IaaS). However, such support is not yet available for PaaS cloud.

Instead of synthetic workload, a better option is to generate prototypes of the application from its model. This strategy is used by ProtoCom [9], which generates prototypes of Java applications. Initial attempts to support a variety of platforms [10] remained at conceptual level [9]. In this work, we complement the state of the art by proposing and implementing prototype-generation solution for PaaS cloud.

3. Our approach

Our approach allows software architects to evaluate performance related quality metrics of a proposed PaaS cloud application's architecture in early design stages by leveraging automatically generated prototypes of the application. In this way, the architects can evaluate different variants of the proposed architecture with significantly lower effort than manual implementation of prototypes. In effect, they have better continuous control over the quality of the application and can prevent later costly reimplementation. As the software architecture of the application evolves, the model can be updated and performance of newly generated prototype can be compared with previous versions of the architecture.

The individual steps of our approach (done by the software architect as well as the prototyping tool) are depicted in *Figure 1*.

[1]Project homepage: https://lasaris.fi.muni.cz/research/software-architecture-optimization-in-paas-cloud-applications/paasarch-cloud-prototyper-tool

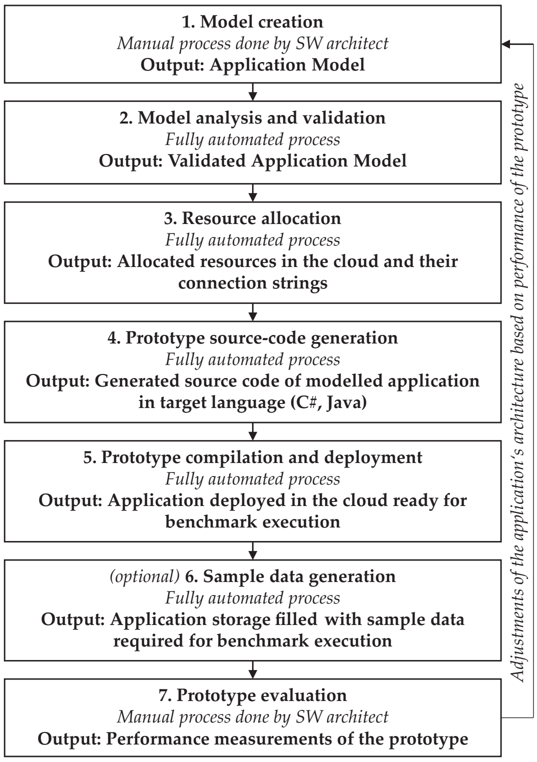

Figure 1: Prototyping process

1. **Model creation** – The software architect creates a model of the designed application that describes all important aspects that significantly influence evaluated quality metrics. Simple sample model is in *Figure 2*.
2. **Model analysis and validation** – The model is loaded by the tool and is validated against our meta-model to determine if the prototype can be correctly generated.
3. **Resource allocation** – One of the advantages of our tool is that it automatically takes care of resource allocation in the cloud environment. Resources need to be allocated prior to code generation as connection strings need to be incorporated into the prototype.
4. **Prototype source-code generation** – Based on the description of application's behavior in the model, the source code of executable prototype is generated.
5. **Prototype compilation and deployment** – Generated source code is compiled and the executable application is deployed to the target cloud hosting environment.
6. **Sample data generation** – Another advantage of our tool is that it can automatically generates sample data based on the description of their volume and complexity in the model.
7. **Prototype evaluation** – When the prototype is ready for evaluation, our tool generates a list of URL endpoints that can be passed to current state-of-the-art web benchmark tools.

```json
{ "$type": "Prototype",
  "Applications": [
    { "$type": "RestApiApplication", "Name": "SimpleSampleApplication",
      "Platform": "DotNet46",
      "Actions": [
        { "$type": "CallableAction", "Name": "GetProducts",
          "Operation": {
            "$type": "LoadEntitiesFromEntityStorage",
            "EntityName": "Product", "EntitySetName": "Products",
            "EntityStorageName": "SampleDB",
            "Filter": { "$type": "FilterCondition", "UseKey": true,
              "OnAttribute": "Id", "NumberOfResults": 30 }}
        }]}],
  "Entities": [
    { "$type": "Entity", "Name": "Product",
      "Properties": [
        { "$type": "PropertyInfo", "Name": "Id", "Type": "Int32" },
        { "$type": "PropertyInfo", "Name": "Name", "Type": "String" },
        { "$type": "PropertyInfo", "Name": "Price", "Type": "Decimal" }
      ]}],
  "Resources": [
    { "$type": "AzureSQLDatabase", "PerformanceTier": "standard",
      "ServiceObjective": "S0", "Name": "SampleDB",
      "EntitySets": [
        { "$type": "EntitySet", "EntityName": "Product",
          "Name": "Products", "Count": 25000 }]},
    { "$type": "AzureAppService", "Name": "SampleWebHosting",
      "PerformanceTier": "StandardS3",
      "WithApplication": "SimpleSampleApplication" }]
}
```

Figure 2: Valid simple sample model serialized in JSON format.

4. The Tool

Steps 2–6 in our approach are fully automated using our tool PaaSArch Cloud Prototyper. The the tool itself is implemented as a set of libraries in the .NET framework using C# packed together with a console application, which displays the progress of prototype generation based on input model. The tool currently contains code generators, which generate final prototypes in .NET/C# and supports automated deployment and management of Microsoft Azure cloud resources. Thanks to its extensibility, new code generators can be implemented to support extensibility of the application model by adding new operations and cloud resources. Support for additional cloud providers (Amazon, Google, etc.) can be added via resource manager plug-ins. Detailed description of the tool's architecture is available in [1].

5. Evaluation

Our approach and outcomes of the implemented prototype generator were evaluated on multiple case studies as described in [1]. Numerous variants of the software architecture of designed applications were modeled and evaluated using our approach. Based on generated prototypes and their evaluation, we were able to choose the variant of the architecture design with the most desirable qualities based on the given requirements of the project. We also evaluated that both the prototype and the final implementation manifested the same degree of scalability.

6. Conclusion

In our work, we take advantage of the cloud elasticity and introduce a design-time quality evaluation technique for PaaS applications based on automatically generated application prototypes, which can be deployed to the cloud and evaluated in the context of multiple quality attributes and environment configurations. As part of this work, we discuss the implementation of the approach in terms of the PaaSArch Cloud Prototyper, which automates the approach to assist software architects in the design of cloud applications that will be effectively utilizing the PaaS cloud platform and combining the available services in an optimal way.

References

[1] D. Gesvindr, O. Gasior, B. Buhnova, Architecture design evaluation of paas cloud applications using generated prototypes: Paasarch cloud prototyper tool, Journal of Systems and Software 169 (2020) 110701. URL: https://www.sciencedirect.com/science/article/pii/S0164121220301485. doi:https://doi.org/10.1016/j.jss.2020.110701.

[2] D. Gesvindr, B. Buhnova, Performance challenges, current bad practices, and hints in paas cloud application design, SIGMETRICS Perform. Eval. Rev. 43 (2016).

[3] D. Gesvindr, B. Buhnova, Architectural tactics for the design of efficient paas cloud applications, in: 2016 13th Working IEEE/IFIP Conference on Software Architecture (WICSA), 2016.

[4] S. Becker, H. Koziolek, R. Reussner, The palladio component model for model-driven performance prediction, J. Syst. Softw. 82 (2009).

[5] H. Koziolek, Performance evaluation of component-based software systems: A survey, Performance Evaluation 67 (2010). Special Issue on Software and Performance.

[6] D. Franceschelli, D. Ardagna, M. Ciavotta, E. Di Nitto, Space4cloud: A tool for system performance and costevaluation of cloud systems, in: Proceedings of the 2013 International Workshop on Multi-cloud Applications and Federated Clouds, ACM, 2013.

[7] S. Lehrig, H. Eikerling, S. Becker, Scalability, elasticity, and efficiency in cloud computing: A systematic literature review of definitions and metrics, in: Proceedings of the 11th Int. ACM SIGSOFT Conference on Quality of Software Architectures, ACM, 2015, pp. 83–92.

[8] F. Chen, J. Grundy, J.-G. Schneider, Y. Yang, Q. He, Stresscloud: A tool for analysing performance and energy consumption of cloud applications, in: Proceedings of the 37th International Conference on Software Engineering - Volume 2, ICSE '15, IEEE Press, 2015.

[9] C. Klaussner, S. Lehrig, Using java ee protocom for sap hana cloud, in: SOSP'14 Symposium on Software Performance: Joint Descartes/Kieker/Palladio Days 2014, 2014, p. 17.

[10] S. Becker, Coupled model transformations for QoS enabled component-based software design, Ph.D. thesis, Universität Oldenburg, 2008.