Architectural Runtime Modeling and Visualization for Quality-Aware DevOps in Cloud Applications

Robert Heinrich
Karlsruhe Institute of Technology
Email: heinrich@kit.edu

Christian Zirkelbach
Kiel University
Email: czi@informatik.uni-kiel.de

Reiner Jung
Kiel University
Email: rju@informatik.uni-kiel.de

I. Topic

In this paper, we present a tutorial on modeling and visualizing software architectures in form of architectural runtime models to support quality-aware DevOps in cloud applications. The tutorial is held in context of the 2017 14th IEEE International Conference on Software Architecture to share our findings and experiences with conference participants and give them the opportunity to expand their knowledge and skills on software architecture modeling, visualization, and analysis in development and operations.

A. Summary

Cloud-based software applications are designed to change often and rapidly during operations to provide constant quality of service. This leads to increasing blurring of the boundary between development and operations. DevOps denotes a set of practices to support communication and collaboration of developers and operators of software applications. Software architecture is the key artifact for documenting and analyzing a software application during development and operations. However, while comparing architectural models used in the development phase to those used in the operation phase we can identify several differences in terms of purpose, abstraction, and content [2]. Consequences are constrained reuse of development models during operations and limited phase-spanning consideration of the software architecture.

In this tutorial, we present approaches to address gaps between architectural modeling in development and operations and thus allow for phase-spanning usage of architectural models. The foundation is maintaining the semantic relationships between monitoring outcomes and architectural models [3]. We discuss the integration of development models, code generation, monitoring, runtime model updates, as well as adaptation candidate generation and execution. We describe the combination of descriptive and prescriptive architectural models to improve the communication and collaboration between operators and developers. The consideration of static and dynamic content in architectural models supports operation-level analysis and adaptations. Furthermore, we present different architectural runtime model visualizations, which allow to detect the above mentioned gaps for development on the one hand and for operating on the other hand.

B. Goals

In the tutorial we pursue three goals:

1. Share our knowledge and experience on architecture modeling and analysis in dynamic cloud applications with the tutorial audience.
2. Gather feedback from the tutorial audience and identify potentials for future collaborations.
3. Conduct a comprehensibility study among the tutorial audience to evaluate the applicability and usefulness of the proposed approaches and tools.

C. Tutorial Audience

The intended tutorial audience consists of PhD students, researchers, and industrial engineers with strong interests and background in software architecture modeling and analysis. The audience should have completed their bachelor degree in computer science, software engineering or a related area and should have basic knowledge in software architecture. We explicitly welcome participants with industrial background and practical experience related to software architecture. We will present essential foundations, so the participants do not require specific expertise.

D. Key Take-Away Messages

The audience should leave with three take-away messages:

1. The audience is aware of approaches and tools for software architecture modeling, analysis, and visualization during operations.
2. The audience is aware of how semantic relationships between monitoring outcomes and architectural models can be maintained.
3. The audience is aware of combining architectural models during development and operations to support DevOps activities.

II. Implementation

We apply for a full-day tutorial with the following preliminary schedule of events as shown in Table I.

The tutorial will cover presentations of approaches and demonstrations of tools for modeling and analyzing software architectures for dynamic cloud applications. This part of the tutorial will be conducted in form of a lecture with facilitated discussions and includes examples for illustration

Table I
SCHEDULE OF EVENTS

Time	Topic
09:00 – 09:10	Welcome and General Introduction
09:10 – 09:40	Study Foundations
09:40 – 10:00	Model-based Software Application Monitoring
10:00 – 10:30	Runtime Architecture Modeling and Visualization
10:30 – 11:00	Coffee Break
11:00 – 12:15	Introduction to the ExplorViz, Palladio, and iObserve Approaches with following Tool / Visualization Demos
12:15 – 12:30	Study Setup
12:30 – 14:00	Lunch
14:00 – 15:30	Comprehensibility Study
15:30 – 16:00	Coffee Break
16:00 – 16:30	Live Database Trace Visualization in Large Software Landscapes
16:30 – 17:00	Feedback and Open Discussion

purposes and short exercises for the audience. We start with an introduction of established approaches on software architecture modeling and analysis, e.g., the Palladio approach [5], which provides the basis for approaches presented in the following. Afterwards, we will present the runtime modeling approach iObserve [2] for updating development architectural models by observations made during operations. iObserve supports the integration of operation-level adaptation and development-level evolution of cloud applications [3]. ExplorViz is an approach for visualizing large software landscapes and embedded applications during development and operations [1]. Furthermore, it features two distinct visualizations, showing a monitored software landscape on the one hand, and an application level visualization on the other hand.

The lecture part of the tutorial is followed by a hands-on part, where a comprehensibility study is conducted among the audience. In the comprehensibility study we will evaluate the applicability and usefulness of the presented approaches and tools for system and program comprehension. Subject of the study is the Common Component Model Example (CoCoME) – a community case study for component-based software engineering and software evolution [4]. The audience will apply the presented approaches and tools on the subject in order to answer questions to assess their comprehension on the CoCoME application. The audience answers will allow us to draw conclusions on the applicability and usefulness of the approaches and tool for system and program comprehension. Details are given in the schedule.

The tutorial is therefore well appropriate for an audience from academia and industry with basic knowledge in software architecture, which wants to broaden its knowledge for up-to-date dynamic cloud applications. There is no specific expertise needed as we will introduce all necessary foundations. The proposed tutorial aims at PhD students, experienced researchers, and industrial engineers and experts. Therefore, we think the ICSA participants fit the intended audience for our tutorial very well. Moreover, we are convinced that our tutorial suits the topics of ICSA very well and will enrich the conference by imparting background knowledge and presenting recent tooling. The tutorial will be conducted as a lecture with facilitated discussions and tool demonstrations followed by a study, which comprises program comprehension tasks based on our tools. The study results are collected in an online-questionnaire.

Afterwards, we present a novel and practically oriented approach, which facilitates a live database trace visualization in large software landscapes. As databases are a crucial part in almost every software system, they are embedded in the overall architecture and need to be understood as well. The described approach supports system, program, and database comprehension, extends the previous presented approaches, and aids developers and operators alike. The idea behind the approach is motivated by a previous case study we conducted on the open repository software EPrints, when we detected performance bottlenecks, which were database-related [6]. At the end of our proposal, we conduct a concluding feedback session and give the opportunity for an open discussion.

III. BACKGROUND OF THE PRESENTERS

Robert Heinrich is a senior researcher and head of the Quality-driven System Evolution research group at the Software Design and Quality chair, Karlsruhe Institute of Technology. He holds a doctoral degree from Heidelberg University. His research interests include software evolution and adaptation using model-driven monitoring, run-time architecture modeling and analysis techniques. Robert Heinrich holds the Certificate for University Didactics of the State Baden Württemberg (Germany) and is active in academic teaching since more than eight years. He presented his research at major international conferences and published in major international journals. Currently, he is involved in the iObserve research project on integrated observation and modeling to support adaptation and evolution of software systems.

Christian Zirkelbach is a junior researcher and Ph.D. student in the Software Engineering research group at the Kiel University. He is particularly interested in software and database visualization, software architecture, and empirical methods. Parallel to his studies, he worked for five years part-time as a system administrator. Prior to his academic career, he completed an apprenticeship in industry as an IT specialist for systems integration followed by working for about 2 years as an Oracle DBA at the Federal Motor Transport Authority. He is involved in academic teaching for about two years and presented his research at smaller conferences. At the moment, he is working on the ExplorViz research project and his doctoral thesis.

Reiner Jung is a senior researcher at Kiel University. He holds a doctoral degree from Kiel University. His research focus are domain-specific modeling, model-driven monitoring, run-time models and analysis, model and code generation, both at design-time and runtime. He presented his research at major international conferences, e.g., MODELS and ICMT, and is currently involved in iObserve and the application of domain-specific modeling in context of research software.

IV. Background of the Tutorial

This tutorial has been developed for ICSA 2017 exclusively. It has not been offered previously. We expect between 20 and 30 participants.

Acknowledgment

This work was partially supported by the DFG (German Research Foundation) under the Priority Programme SPP1593: Design For Future – Managed Software Evolution and the MWK (Ministry of Science, Research and the Arts Baden-Württemberg) in the funding line Research Seed Capital (RiSC).

References

[1] F. Fittkau, A. Krause, and W. Hasselbring *Software landscape and application visualization for system comprehension with ExplorViz*, In: Information and Software Technology, http://dx.doi.org/10.1016/j.infsof.2016.07.004, 2016, Elsevier.

[2] R. Heinrich, R. Jung, C. Zirkelbach, W. Hasselbring and R. Reussner *An Architectural Model-Based Approach to Quality-aware DevOps in Cloud Applications*, In: Software Architecture for Big Data and the Cloud, Elsevier, 2017, to appear.

[3] R. Heinrich *Architectural run-time models for performance and privacy analysis in dynamic cloud applications*, ACM SIGMETRICS Performance Evaluation Review, 43(4):13-22, 2016, ACM.

[4] R. Heinrich, S. Gärtner, T.M. Hesse, T. Ruhroth, R. Reussner, K. Schneider, B. Paech, and J. Jürjens *A platform for empirical research on information system evolution*, In: 27th International Conference on Software Engineering and Knowledge Engineering, 2015, pages 415-420.

[5] R.H. Reussner, S. Becker, J. Happe, R. Heinrich, A. Koziolek, H. Koziolek, M. Kramer, and K. Krogmann *Modeling and Simulating Software Architectures - The Palladio Approach*, MIT Press 2016.

[6] C. Zirkelbach, W. Hasselbring, and L. Carr *Combining Kieker with Gephi for Performance Analysis and Interactive Trace Visualization*, In: Symposium on Software Performance 2015: Joint Developer and Community Meeting of Descartes/Kieker/Palladio, 35(3):26-28, 2015, Softwaretechnik-Trends.