Extracting Software Architectures from Traces for the Simulation of Microservice-based Architectures [Extended Abstract]

Tim Thüring
University of Stuttgart, Stuttgart

Gabriel Glaser
University of Stuttgart, Stuttgart

Marcel Hafner
University of Stuttgart, Stuttgart

Abel Gitzing
University of Stuttgart, Stuttgart

Sebastian Frank
sebastian.frank@uni-hamburg.de
University of Hamburg, Hamburg

Alireza Hakamian
mir-alireza.hakamian@iste.uni-stuttgart.de
University of Stuttgart, Stuttgart

André van Hoorn
andre.van.hoorn@uni-hamburg.de
University of Hamburg, Hamburg

1 Context and Problem

Architecture-based quality evaluation of software systems is well-established [4]. When initial architectural models do not exist, it is a common technique to extract the architecture of software systems from trace data. A suitable view to represent microservice-based software architectures is the component and connector (C&C) view, which can be used to describe how provided services' operations interact. Such representations are already used in (1) Resirio for the analysis and visualization in the context of resilience requirements elicitation [7] and (2) MiSim for resilience assessment through simulation of microservice-based architectures [6]. The (semi-)automated extraction of input models for these approaches would benefit resilience engineers by reducing their effort in (re)modeling and calibrating, especially considering the dynamic nature of microservice-based software systems.

However, existing approaches for extracting architectures that follow a microservice-based style, such as MicroART [5], do not consider (1) identification of resilience patterns and mechanisms from application traces, (2) resource demand estimation of operation calls, and (3) the integration into the Resirio and MiSim approaches.

2 Objective

This study aims to extract the C&C view of microservice-based software systems at runtime and provide capabilities for (1) resource demand estimation of operation calls, (2) detection of resilience mechanisms and patterns — namely, retry, circuit breaker, and load balancers — and (3) support for various state-of-the-art application trace formats. The high-level goal is to minimize manual modeling overhead for simulation-based resilience assessment of microservice-based architecture in MiSim.

3 Method

We follow the design science approach with a practical problem [2]. In the solution space, we designed a technique for detecting resilience patterns, estimating resource demand, and supporting multiple trace formats. Moreover, we provide tooling that implements our proposed technique. In particular, we use state-of-the-art approaches for resource demand estimation implemented in the library LibReDE [1]. We apply heuristics to detect resilience patterns, e.g., curve fitting to identify linear and exponential retry strategies. Finally, we use the intermediate trace format OPEN.xtrace [3] to reach compatibility with various application trace formats.

In order to validate that our tooling does solve the problem at hand, we evaluated our tooling by (i) detecting round-robin load balancing of a HAProxy [8] and an exponential retry implementation of Resilience4j [9] from application traces in a controlled environment, and (ii) extracting the architecture of an industrial software system.

4 Result

Our tooling is able to correctly identify both resilience mechanisms, including their configuration based on the application traces. Furthermore, we successfully extracted the architecture of the industrial software system. However, since our approach is partially built on heuristics, the evaluation also revealed limitations and the need for a limited amount of manual adjustment by the resilience engineer.

5 Talk Outline and Additional Resources

In our talk, we will present our approach, focusing on the resilience mechanism detection. In addition, we will present selected evaluation results, our lessons learned during the development of the approach, and give a short demonstration of the tool.

6 Acknowledgments

The authors thank the German Federal Ministry of Education and Research (dqualizer project and Software Campus 2.0 — Microproject: DiSpel, FKZ: 01IS17051) for supporting this work.

References

[1] S. Spinner et al. "Librede: A library for resource demand estimation". In: *Proceedings of the 5th ACM/SPEC international conference on Performance engineering*. 2014, pp. 227–228.

[2] R. J. Wieringa. *Design Science Methodology for Information Systems and Software Engineering*. Springer, 2014.

[3] D. Okanović et al. "Towards performance tooling interoperability: An open format for representing execution traces". In: *European Workshop on Performance Engineering*. Springer. 2016, pp. 94–108.

[4] R. H. Reussner et al. *Modeling and simulating software architectures: The Palladio approach*. MIT Press, 2016.

[5] G. Granchelli et al. "Towards Recovering the Software Architecture of Microservice-Based Systems". In: *2017 IEEE International Conference on Software Architecture Workshops, ICSA Workshops 2017, Gothenburg, Sweden, April 5-7, 2017*. IEEE Computer Society, 2017, pp. 46–53.

[6] L. Wagner et al. "MiSim - A Lightweight and Extensible Simulator for a Scenario-Based Resilience Evaluation of Microservice Architectures (Poster)". In: *Short Paper Proceedings of Symposium on Software Performance 2021*. Vol. 3043. CEUR Workshop Proceedings. CEUR-WS.org, 2021.

[7] S. Frank et al. "Interactive Elicitation of Resilience Scenarios Based on Hazard Analysis Techniques". In: *Lecture Notes in Computer Science*. Vol. 13365. (in press). Springer, 2022.

[8] HAProxy document. *HAProxy*. `https://www.haproxy.org/`. Accessed: 2022-August. 2022.

[9] Resilience4j. *Resilience4j document*. `https://github.com/resilience4j/resilience4j`. Accessed: 2022-August. 2022.