MiSim — A Lightweight and Extensible Simulator for a Scenario-Based Resilience Evaluation of Microservice Architectures

Lion Wagner[1], Sebastian Frank[1], Alireza Hakamian[1] and André van Hoorn[2]

[1]University of Stuttgart, Institute of Software Engineering, Universitätsstraße 38, 70569 Stuttgart, Germany
[2]University of Hamburg, Department of Informatics, Vogt-Kölln-Straße 30, 22527 Hamburg, Germany

Context and Problem

With the growing popularity of microservice-based architectures, the need for an effective resilience assessment of these systems occurred. A resilience assessment is often done in production using so-called chaos experiments. While producing representative results, chaos experiments often (1) require a significant investment of time, (2) impact customer experiences as they stress the system under test, and (3) do not lean towards parallel running because of interference. Simulating chaos experiments is an alternative approach to running experiments in a production environment. Today, there are many simulators available for distributed and service-oriented architectures. Popular examples are SimuLizar [1], DraCeo [2], BigHouse [3], μqsim [4], PacketStorm [5], iFogSim [6] or GreenCloud [7]. Most of these focus on either performance or efficiency by simulating or solving queuing models. However, none of them satisfies the following requirements: (1) supporting many resilience patterns, (2) simulating multiple typical chaos injections such as killing a service instance, and (3) being a lightweight simulator that has low overhead in modeling and simulation.

Objective

Therefore, we developed *MiSim*, a simulator specializing in the simulation of (1) resilience patterns and (2) chaos injections. In addition, the simulator supports scenario structures introduced by the Architecture Tradeoff Analysis Method (ATAM) [8] as input. Furthermore, it was targeted to have no external requirements, such as a Platform as a Service (PaaS), submodules, or libraries.

SSP'21: Symposium on Software Performance, November 09–10, 2021, Leipzig, Germany
st148345@stud.uni-stuttgart.de (L. Wagner); sebastian.frank@iste.uni-stuttgart.de (S. Frank); mir-alireza.hakamian@iste.uni-stuttgart.de (A. Hakamian); andre.van.hoorn@uni-hamburg.de (A. v. Hoorn)
0000-0003-2567-6077 (A. v. Hoorn)

Method

We elicited 24 requirements for *MiSim* by interviewing stakeholders. In our case, the stakeholders are a group of researchers interested in simulating chaos experiments. We evaluated the simulator engine quality and usability of the simulator. For simulator engine quality, we performed a (1) code review, (2) feature comparison to other simulators, (3) performance analysis, and (4) analyzed the simulation accuracy of real-world scenarios [9].

Result

The current version of *MiSim* supports 20 out of the 24 elicited requirements. We did not implement four requirements due to time restrictions of the project. For a more detailed overview of supported features, have a look at *MiSim*'s GitHub repository[1].

As required, *MiSim* supports (1) common resilience patterns (i.e., Circuit Breaker, Rate Limiter, Retry, Autoscaler, and Self-Restarting) and (2) chaos injections (i.e., instance/service killing and latency injections). Additionally, (3) there are no external dependencies, and the compiled simulator is only about 11 MB in size. Among others, the simulator accepts LIMBO models [10, 11] as workload definitions and an ATAM scenario-based experiment format. Lastly, throughout the development process, multiple supporting tools for creating architectures or scenarios were created in related projects [12, 13, 14].

Regarding the usability of the simulator, we performed an online meeting with our stakeholders to collect experiences on installing and running chaos experiments through *MiSim*. The overall feedback was positive. However, one minor complaint was about the necessity to improve the code documentation.

The performance evaluation of *MiSim* revealed that it has a high memory-impact, that strongly relates to the number of simulated requests. This is mainly due to a faulty metric collection system. Additionally, the computational demand on the underlying simulation engine DESMO-J[1] is relatively high. Over 90% of the computation time is spent on the (re-)scheduling of events. However, even for the simulation of complex experiments, the actual computation time does not take unreasonably long.

Lastly, simulating real-world scenarios confirmed that the implemented patterns behave as expected. Furthermore, this showed that the calibration options and accuracy of the simulation could be improved, since specifically varying workloads were sometimes poorly simulated.

Talk Outline and Additional Resources

In this talk, we will present the current state of *MiSim*. We will show an extract of its internal design and how other researchers/practitioners can extend the simulator. Further, we demonstrate how to extract architecture models from real traces. Additionally, we show examples of chaos experiments and cover our findings on performance and accuracy. For a preview, check out *MiSim*'s GitHub repository[2].

[1] http://desmoj.sourceforge.net/

[2] https://github.com/Cambio-Project/resilience-simulator

Acknowledgments

This research is funded by the Baden-Württemberg Stiftung (Orcas project) and the German Federal Ministry of Education and Research (Software Campus 2.0 — Microproject: DiSpel).

References

[1] M. Becker, S. Becker, J. Meyer, Simulizar: Design-time modeling and performance analysis of self-adaptive systems, in: S. Kowalewski, B. Rumpe (Eds.), Software Engineering 2013, Gesellschaft für Informatik e.V., Bonn, 2013, pp. 71–84.

[2] H. H. A. Valera, M. Dalmau, P. Roose, J. Larracoechea, C. Herzog, DRACeo: A smart simulator to deploy energy saving methods in microservices based networks, in: 2020 IEEE 29th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), volume 2020-Septe, IEEE, 2020, pp. 94–99.

[3] D. Meisner, J. Wu, T. F. Wenisch, BigHouse: A simulation infrastructure for data center systems, in: 2012 IEEE International Symposium on Performance Analysis of Systems & Software, IEEE, 2012, pp. 35–45.

[4] Y. Zhang, Y. Gan, C. Delimitrou, μqSim: Enabling Accurate and Scalable Simulation for Interactive Microservices, in: 2019 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), IEEE, 2019, pp. 212–222.

[5] Packetstorm Communications, Network Simulation - PacketStorm, 2018.

[6] H. Gupta, A. Vahid Dastjerdi, S. K. Ghosh, R. Buyya, iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments, in: Software - Practice and Experience, volume 47, John Wiley and Sons Ltd, 2017, pp. 1275–1296.

[7] D. Kliazovich, P. Bouvry, S. U. Khan, GreenCloud: a packet-level simulator of energy-aware cloud computing data centers, The Journal of Supercomputing 62 (2012) 1263–1283.

[8] R. Kazman, M. Klein, P. Clements, ATAM: Method for Architecture Evaluation, Technical Report, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA 15213, 2000.

[9] S. Frank, A. Hakamian, L. Wagner, D. Kesim, J. von Kistowski, A. van Hoorn, Scenario-based resilience evaluation and improvement of microservice architectures: An experience report, in: Companion of the 15th European Conference on Software Architecture (ECSA 2021), 2021. To appear.

[10] J. v. Kistowski, N. R. Herbst, S. Kounev, Modeling variations in load intensity over time, LT '14, Association for Computing Machinery, New York, NY, USA, 2014, p. 1–4.

[11] J. v. Kistowski, N. Herbst, S. Kounev, Limbo: A tool for modeling variable load intensities, in: Proceedings of the 5th ACM/SPEC International Conference on Performance Engineering, ICPE '14, Association for Computing Machinery, New York, NY, USA, 2014, p. 225–226.

[12] S. Beck, Simulation-based Evaluation of Resilience Antipatterns in Microservice Architectures, 2018. Bachelor's Thesis, University of Stuttgart.

[13] C. Zorn, Interactive elicitation of resilience scenarios in microservice architectures, 2021. Masters's Thesis, University of Stuttgart.

[14] N. Kammhoff, Algorithms for Efficient Chaos Experiment Selection in Microservice Architectures, 2019.