Iterative and Incremental Refinement of Microservice-based Architectures and SLOs

Sandro Speth[1], Sarah Stieß[1], Sebastian Frank[1], and Steffen Becker[1]

Institute of Software Engineering, University of Stuttgart, Stuttgart, Germany
{sandro.speth,sarah.stiess,sebastian.frank,steffen.becker}@iste.uni-stuttgart.de

Abstract. In an agile development process, non-functional requirements and software architectures continuously change, making it hard to extract and manage appropriate Service Level Objectives (SLO). We propose an interactive toolchain to support the architect in iteratively and incrementally adapting SLOs and architecture alongside each other. Our toolchain utilises the results of experiments and SLO violation impact analyses to guide the architect's adaptation decisions, reflecting continuously changing and imprecise requirements. Prototypes of the presented tools already exist. In this paper, we envision their interconnection and required extensions.

Keywords: Microservices · Issue Management · Requirements · SLO

In the following, we describe our toolchain and process as depicted in fig. 1 in more detail. It consists of seven steps and utilizes three tools. Gropius [4] is a tool to model component-based architectures and manage issues of each component together with the architecture in a component-diagram-like way [3]. RESIRIO [6] is a tool with a chatbot UI that helps architects in interactive requirements identification. SoLOMON [2] models and monitors SLOs for components in Gropius. The Impact Analyser [2, 5] calculates and explains the impact of SLO violations on dependent components and the business process.

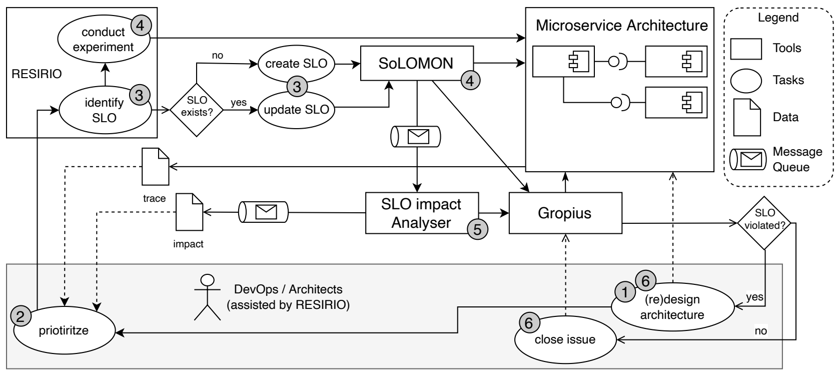

Fig. 1. Overview of Ecosystem.

The process is as follows: **(1) Model and Deploy System:** First, the architect models the initial architecture in Gropius. While each component manages its issues in a separate issue management system, Gropius acts as a wrapper above them to store and synchronize impact analysis issues.

(2) Prioritize Traces: The architect prioritizes business use cases and imports runtime data (traces) for the most critical use case into RESIRIO. Further traces can be imported in later iterations when new critical use cases arise, no requirements are elicited, or resources are underused. In future work, we aim to add the capability to manage multiple traces, connect RESIRIO to the monitoring tooling, and leverage domain-driven design techniques for prioritization.

(3) SLO Elicitation: For each trace, RESIRIO visualizes the relevant microservices and their dependencies and identifies potential SLO violations, e.g., response time spikes. The architect specifies a scenario consisting of a violation's cause, e.g., a service failure, and the expected system behavior, e.g., service restart within a minute, by interacting with RESIRIO's chatbot interface. The architect derives SLOs from the hypothetical scenarios, and registers them — manually or with the help of the chatbot — at SoLOMON.

(4) Conduct Experiment: When no data for the specified scenarios are available, experiments are derived and run [1]. Currently, experiment generation is done by the architect but is intended to be automated in the future. During the experiment, SoLOMON monitors the SLOs.

(5) Reporting and Analysis of SLO Violations: Upon identifying an SLO violation, SoLOMON publishes an event to the Analyzer. The violation and impact are reported to Gropius, and the Analyzer publishes an event, such that the next iteration's prioritization considers its results, e.g., the impact's severity.

(6) Adapt Architecture and SLOs: The architect, assisted by the chatbot, either updates the SLOs to match the architecture or the architecture to adhere to the SLOs, e.g., through the use of resilience patterns [1]. Then, they continue at *Prioritize Traces* with additional traces from the current experiment. Furthermore, the architect can close resolved issues in Gropius.

References

1. Frank, S., Hakamian, A., Wagner, L., Kesim, D., von Kistowski, J., van Hoorn, A.: Scenario-based Resilience Evaluation and Improvement of Microservice Architectures: An Experience Report. In: Proceedings of 5th FAACS@ECSA (2021)
2. Speth, S.: Semi-automated Cross-Component Issue Management and Impact Analysis. In: Proceedings of 2021 36th ASE. pp. 1090–1094. IEEE (2021)
3. Speth, S., Becker, S., Breitenbücher, U.: Cross-Component Issue Metamodel and Modelling Language. In: Proceedings of 11th CLOSER 2021. SciTePress (2021)
4. Speth, S., Breitenbücher, U., Becker, S.: Gropius — A Tool for Managing Cross-component Issues. In: Proceedings of the 2020 14th ECSA. vol. 1269. Springer (2020)

[0] This paper was accepted for the Agility with Microservices Programming workshop, 2022.

5. Stieß, S.: Tracing the Impact of SLO Violations on Business Processes across a Microservice Architecture with the Saga Pattern. Master's thesis, University of Stuttgart, Germany (2021)
6. Zorn, C.: Interactive Elicitation of Resilience Scenarios in Microservice Architectures. Master's thesis, University of Stuttgart, Germany (2021)