Representative Load Testing in Continuous Software Engineering: Automation and Maintenance Support

Henning Schulz[1], André van Hoorn[2]

Abstract: This extended abstract summarizes our work on reducing the maintenance effort for the parameterization of representative load tests using annotations, which we have published in the Journal of Software Testing, Verification & Reliability in 2019 [SHW19].

Representative load testing [JH15] can effectively test and preserve the performance before delivery by mimicking the expected workload. However, it also requires a notable amount of expertise and effort for creating, maintaining, and executing the load test. During load test creation, experts need to take care of different workload scenarios occurring in the production system, which the test needs to replay. Maintenance tasks comprise updating the test due to changing workloads and APIs of the system under test (SUT).

For easing representative load testing and reducing its effort, we aim at automatically generating tailored representative load tests [SAH18]. Based on an expert's request and continuously recorded monitoring data, our approach generates a load test tailored to a particular service and replaying a particular workload observed in production. However, while we can leverage existing approaches for extracting the workload model automatically, generating a load test also requires proper parameterization — e.g., specifying credentials for the simulated users that are valid in the test environment — typically entailing manual effort. The manual effort accumulates if the expert generates multiple load tests or if the workload or APIs change. In the context of continuous software engineering, such manual effort prevents the integration into the highly-automated build and delivery infrastructure.

Therefore, our approach reduces the manual effort for parameterization and completely automates the test generation [SHW19]. As shown in Fig. 1, instead of parameterizing a load test directly, an expert can define all parameterizations in a separate model: the input data and properties annotation (IDPA) (1). Leveraging API specifications, our approach can generate and update parts of an IDPA automatically (2). Also, we provide mechanisms for evolving an IDPA over API changes. Hence, the expert can maintain the IDPA and store it at a central place such as the code repository (3). The application running in production and being used by the end-users collects the user's requests and sessions and stores them into a measurement repository (4). Hence, when a new load test is to be generated — e.g., inside a continuous integration and delivery (CI/CD) pipeline — our approach can retrieve an up-to-date IDPA and measurement data (5). It extracts a workload model — e.g., replaying

[1] Novatec Consulting GmbH, Karlsruhe, Germany, henning.schulz@novatec-gmbh.de
[2] University of Stuttgart, Stuttgart, Germany, van.hoorn@informatik.uni-stuttgart.de

doi:10.18420/SE2020_46

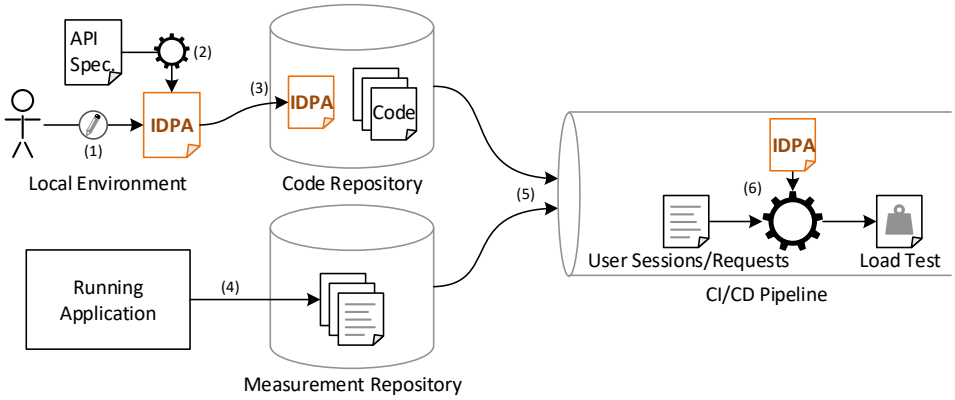

Fig. 1: Overview of the load test parameterization process using IDPAs [SHW19].

the recently typical workload — from the user sessions and requests, transforms it to a load test, and parameterizes it using the IDPA (6). Finally, the pipeline can automatically execute the generated load test without manual intervention.

Our evaluation, consisting of two experimental studies, effort estimation models, and an industrial case study, shows the applicability of our approach. Given the workload is dominated by the order and rate of the requests rather than the input data, it restores the representativeness of generated load tests while it reduces the manual effort from a quadratic to a linear function of time. Details on the evaluation are presented in the original article [SHW19]. The evaluation artifacts and a demo are provided online.[3]

Acknowledgement This work is part of the ContinuITy[4] project, which is supported by the German Federal Ministry of Education and Research (grant no. 01IS17010).

References

[JH15] Jiang, Z. M.; Hassan, A. E.: A Survey on Load Testing of Large-Scale Software Systems. IEEE Trans. on Softw. Eng. 41/11, pp. 1091–1118, 2015.

[SAH18] Schulz, H.; Angerstein, T.; van Hoorn, A.: Towards Automating Representative Load Testing in Continuous Software Engineering. In: Proc. ACM/SPEC International Conference on Performance Engineering (ICPE 2018) Companion. ACM, pp. 123–126, 2018.

[SHW19] Schulz, H.; van Hoorn, A.; Wert, A.: Reducing the Maintenance Effort for Parameterization of Representative Load Tests Using Annotations. Software Testing, Verification & Reliability/, in press (early preview status), 2019.

[3] Artifacts: `https://doi.org/10.5281/zenodo.3255389`, demo: `https://doi.org/10.5281/zenodo.2647976`
[4] ContinuITy web site: `https://continuity-project.github.io/`