PPTAM: Production and Performance Testing Based Application Monitoring

Alberto Avritzer
eSulab Solutions, Princeton, NJ

Daniel Menasché, Vilc Rufino
Federal University of Rio de Janeiro, Brazil

Barbara Russo, Andrea Janes
Free University of Bozen-Bolzano, Italy

Vincenzo Ferme, André van Hoorn, Henning Schulz
University of Stuttgart, Germany

ABSTRACT

It is mandatory to continuously assess software systems during development and operation, e.g., through testing and monitoring, to make sure that they meet their required level of performance. In our previous work, we have developed an approach to assess the degree to which configurations of a software system meet performance criteria based on a domain metric that is obtained by considering operational profiles and results from load test experiments. This paper presents our PPTAM tooling infrastructure that automates our approach and provides a dashboard visualization of the results.

ACM Reference Format:
Alberto Avritzer, Daniel Menasché, Vilc Rufino, Barbara Russo, Andrea Janes, and Vincenzo Ferme, André van Hoorn, Henning Schulz. 2019. PPTAM: Production and Performance Testing Based Application Monitoring. In *Tenth ACM/SPEC International Conference on Performance Engineering Companion (ICPE '19 Companion), April 7–11, 2019, Mumbai, India.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3302541.3311961

1 INTRODUCTION

The operational profile, i.e., the statistical representation of the way a system is used in production [2], is one of the major influence factors of performance and related properties such as reliability, scalability, and elasticity. Stakeholders must be aware of the expected and the actual operational profile when designing and operating their system, e.g., to provide sufficient capacity. Continuous performance monitoring technologies are in common use to collect relevant data [4]. In our previous work [1], we have presented an approach to combine expected production usage and testing results to produce a scalability metric that reflects scalability testing results that were based on i) the used input domain partition (operational profile) [6] and ii) different system deployment configurations to be quantitatively assessed and compared w.r.t. the input domain.

After providing a brief summary of our domain-based approach and metric visualization [1], this paper gives an overview of the PPTAM tooling infrastructure that we developed to implement the approach and to integrate it with production systems and devices of stakeholders for continuous performance assessment and visualization. Our PPTAM tool including a demo is publicly available.[1]

[1]https://github.com/pptam

ICPE '19 Companion, April 7–11, 2019, Mumbai, India

ACM ISBN 978-1-4503-6286-3/19/04.
https://doi.org/10.1145/3302541.3311961

2 APPROACH AND METRIC VISUALIZATION

The key steps of the process for the domain-based assessment of software system configurations — as depicted in Figure 2 and detailed in our previous work [1] — are: (0) collection of operational data, (1) analysis of operational data, (2) load test experiment generation, (3) load test experiment execution, and (4) calculation of the domain-based metric, for each system configuration. The approach is used to quantitatively assess and compare testing results of performance-related properties of different system configurations — for instance with different deployments or parameters — based on expected workload situations (e.g., number of users).

The results can be visualized, as shown in Figure 1, depicting the degree to which a given system configuration satisfies the fail/pass criteria, e.g., response time requirements. In the example, the best theoretical relative mass (shown as the outer polygon in light blue) represents the fraction of the domain metric that would be accrued if all tests were successful. The values correspond to the occurrence of workload situations according to the operational profile. We have shaded in Figure 1 the area under the curve for i) the best theoretical relative mass (outer polygon in light blue), ii) the best measured configuration (*A*, light green), and iii) the configuration (*B*, purple) that equals the best measured relative mass for the largest number of users (150). The domain metric for a configuration is the sum of the relative masses [1]—with the theoretical maximum of 1.0. The plots in Figure 1 also show gaps between the best relative mass and the obtained measurements, for different workload situations and system configurations. These gaps represent the impact of the measured performance degradation on the domain metric. Therefore, customers using the proposed approach can carefully engineer their systems to maximize the return on investment on their (cloud) computing infrastructure.

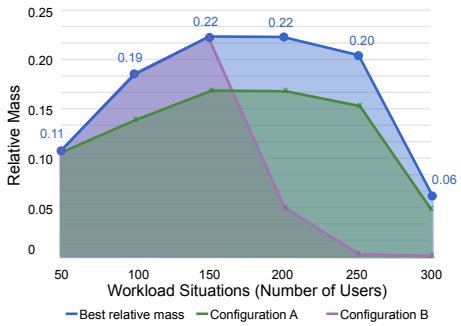

Figure 1: Visualization of best and actual test masses

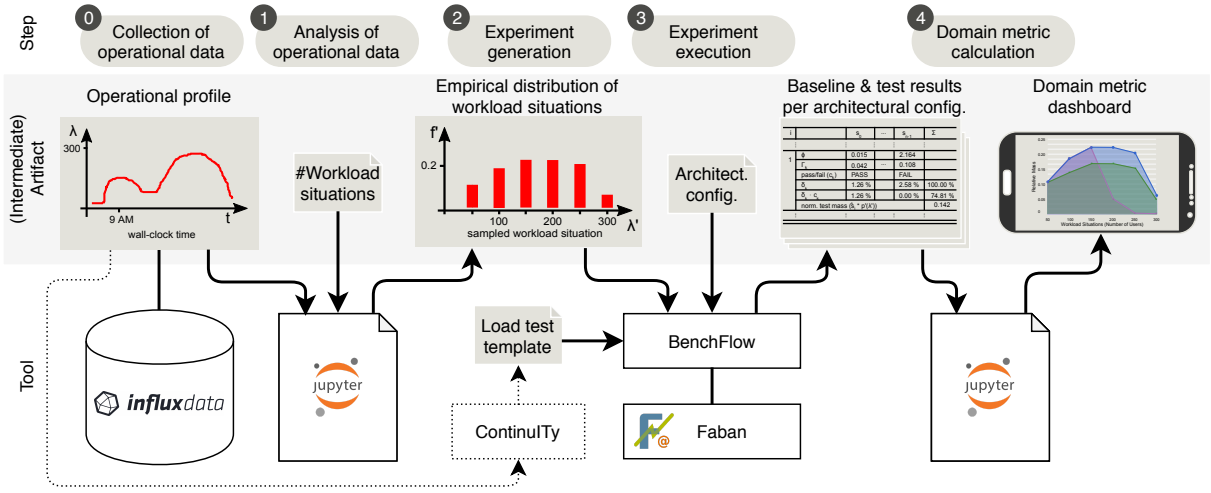

Figure 2: Overview of the PPTAM approach and tooling infrastructure

3 TOOL OVERVIEW

In addition to the process steps and artifacts, Figure 2 includes details on the tool ecosystem. It comprises the following parts: (i) an analysis component that gathers the operational profile from a production system and computes the baseline probability of finding the system at a given state (workload situation), (ii) an infrastructure that generates and executes load test experiments with different architectural configurations to collect system performance against a baseline (fail/pass criteria), and (iii) a graphical user interface that calculates and visualizes the current performance metric in a report and on a smartphone user interface (UI).

For gathering the operational profile, we utilize Application Performance Monitoring (APM) [4] tools, e.g., an open-source tool from the OpenAPM initiative.[2] These tools commonly utilize time series databases, e.g., InfluxData, for storing the monitored operational data. Our tool (packaged as a Jupiter Notebook[3]) connects to an InfluxData, retrieves the raw operational data, and generates the empirical distribution of the workload situations (best test masses) defined in the script.

The infrastructure for load testing uses the open-source BenchFlow tool [3] to automate the deployment of the defined experiments (building on container-based virtualization using Docker) for the defined workload situations and system configurations. The Faban load driver framework is used to run the experiments, to collect the performance measures, and to automate the analysis of the testing results. Load test templates are either manually defined or extracted automatically (e.g., using ContinuITy [5]).

R scripts (as a Jupyter Notebook) are used to evaluate the performance of the test against a baseline, compute the performance for each test at each state of the system (Relative Mass), and generate the plots of relative masses against the previously obtained best test masses. React-native, a Javascript framework for building native mobile apps, is used to create the UI[4].

[2]https://openapm.io/
[3]https://jupyter.org/
[4]https://facebook.github.io/react-native

4 CONCLUSIONS

Traditional performance testing approaches usually require long-running tests to be executed according to operational profile specifications [6]. In continuous integration environments, there is a need to continuously run tests to assess scalability and other performance-related properties of new software versions under changing workload situations and system configurations. In this paper, we have presented the tooling infrastructure for our previously introduced approach for continuous assessment and comparison of system configurations. APM tools are used to derive the best test masses from production data. Based on this, load tests are generated and executed to continuously assess the performance of new software versions in the test environment based on the operational profile. The results are visualized in reports and on smartphone-based user interfaces.

Acknowledgments This work has been partly supported by eSulab-Solutions, Inc., the German Federal Ministry of Education and Research (ContinuITy project, grant 01IS17010), and the Italian Ministry of Education, Universities and Research (GAUSS project, grant 2015KWREMX).

REFERENCES

[1] Alberto Avritzer, Vincenzo Ferme, Andrea Janes, Barbara Russo, Henning Schulz, and André van Hoorn. 2018. A Quantitative Approach for the Assessment of Microservice Architecture Deployment Alternatives by Automated Performance Testing. In *Proc. ECSA 2018*. 159–174.
[2] Maria Carla Calzarossa, Luisa Massari, and Daniele Tessera. 2016. Workload Characterization: A Survey Revisited. *ACM Comput. Surv.* 48, 3 (2016), 48:1–48:43.
[3] Vincenzo Ferme and Cesare Pautasso. 2018. A Declarative Approach for Performance Tests Execution in Continuous Software Development Environments. In *Proc. ACM/SPEC ICPE 2018*. 261–272.
[4] Christoph Heger, André van Hoorn, Mario Mann, and Dusan Okanovic. 2017. Application Performance Management: State of the Art and Challenges for the Future. In *Proc. ICPE 2017*. 429–432.
[5] Henning Schulz, Dusan Okanovic, André van Hoorn, Vincenzo Ferme, and Cesare Pautasso. 2019. Behavior-driven Load Testing Using Contextual Knowledge—Approach and Experiences. ACM. To appear.
[6] E. J. Weyuker and A. Avritzer. 2002. A Metric for Predicting the Performance of an Application Under a Growing Workload. *IBM Syst. J.* 41, 1 (Jan. 2002), 45–54.