Extended abstract: Test them all, is it worth it? Assessing configuration sampling on the JHipster Web development stack

Halin, Axel; Nuttinck, Alexandre; Acher, Mathieu; Devroey, Xavier; Perrouin, Gilles; Baudry, Benoit

Citation (APA)
Halin, A., Nuttinck, A., Acher, M., Devroey, X., Perrouin, G., & Baudry, B. (2020). Extended abstract: Test them all, is it worth it? Assessing configuration sampling on the JHipster Web development stack. In *Proceedings of the 24th ACM International Systems and Software Product Line Conference (SPLC '20)* (Vol. A). ACM DL. https://doi.org/10.1145/3382025.3414985

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Extended abstract: Test them all, is it worth it? Assessing configuration sampling on the JHipster Web development stack

Axel Halin
PReCISE, NaDI,
Faculty of Computer Science,
University of Namur
Namur, Belgium

Alexandre Nuttinck
alexandre.nuttinck@cetic.be
CETIC
Charleroi, Belgium

Mathieu Acher
mathieu.acher@irisa.fr
IRISA, University of Rennes I
Rennes, France

Xavier Devroey
x.d.m.devroey@tudelft.nl
Delft University of Technology
Delft, The Netherlands

Gilles Perrouin
gilles.perrouin@unamur.be
PReCISE, NaDI,
Faculty of Computer Science,
University of Namur
Namur, Belgium

Benoit Baudry
baudry@kth.se
KTH Royal Institute of Technology
Stockholm, Sweden

ABSTRACT

This is an extended abstract of the article: Axel Halin, Alexandre Nuttinck, Mathieu Acher, Xavier Devroey, Gilles Perrouin, and Benoit Baudry. 2018. Test them all, is it worth it? Assessing configuration sampling on the JHipster Web development stack. In *Empirical Software Engineering* (17 Jul 2018). https://doi.org/10.1007/s10664-018-9635-4.

CCS CONCEPTS

• **Software and its engineering** → **Software testing and debugging**; **Software product lines**.

KEYWORDS

Configuration sampling, variability-intensive systems, software testing, JHipster, case study

The assumption that it is impossible to test all configurations of a highly configurable software system motivates the development of many testing approaches. Such approaches rely on variability-aware abstractions and sampling techniques to cope with large configuration spaces. Yet, there is no theoretical barrier that prevents the exhaustive testing of all configurations by simply enumerating them if the effort required to do so remains acceptable.

In this case study, we report on the first ever endeavor to test all possible configurations of the industry-strength, open source configurable software system: JHipster, a popular code generator for web applications.

In addition to providing a quantitative assessment of sampling techniques on *all* 26,257 configurations, we present numerous insights regarding the testing infrastructure and compare them with JHipster developers' practice: (1) a cost assessment and qualitative insights of engineering an infrastructure able to automatically test all configurations. This infrastructure is itself a configurable system and requires a substantial, error-prone, and iterative effort (8 man*month); (2) a computational cost assessment of testing all configurations using a cluster of distributed machines. Despite some optimizations, 4,376 hours (~182 days) CPU time and 5.2 terabytes of available disk space are needed to execute 26,257 configurations; (3) a quantitative and qualitative analysis of failures and faults. We found that 35.70% of all configurations fail: they either do not compile, cannot be built or fail to run. Six feature interactions (up to 4-wise) explain this high percentage; (4) an assessment of sampling techniques. Dissimilarity and *t*-wise sampling techniques are effective to find faults that cause a lot of failures while requiring small samples of configurations. Studying both fault and failure efficiencies provides a more nuanced perspective on sampling techniques; (5) a retrospective analysis of JHipster practice. The 12 configurations used in the continuous integration for testing JHipster were not able to find the defects. It took weeks for the community to discover and fix the 6 faults; (6) a discussion on the future of JHipster testing based on collected evidence and feedback from JHipster's lead developers; (7) a feature model for JHipster v3.6.1 and a dataset to perform ground truth comparison of configuration sampling techniques, both available at https://doi.org/10.5281/zenodo.3766690.

Our work is the first endeavor to gather the ground truth of all possible configurations' failures of an industrial-strength open source project. Configuration failures represent one of the most common types of software failures; we believe our insights and data can support a much needed research in this direction.