A Literature Survey of Software Analytics

Moritz Beller, IN4334 2018 TU Delft

2018-09-14

Contents

Chapter 1

Preamble

The book you see in front of you is the outcome of an eight week seminar run by the Software Engineering Research Group (SERG) at TU Delft. We have split up the novel area of Software Analytics into several sub topics. Every chapter addresses one such sub-topic of Software Analytics and is the outcome of a systematic literature review a laborious team of 3-4 students performed.

With this book, we hope to structure the new field of Software Analytics and show how it is related to many long existing research fields.

Moritz Beller

1.1   License

Chapter 2

A contemporary view on Software Analytics

2.1 What is Software Analytics?

2.2 A list of Software Analytics Sub-Topics

Chapter 3

Sample Sub-Topic

This is an example for the deliverable every group works on. Every group works on one independent chapter (starting as one Rmd file).

Chapter 4

Final Words

We have finished a nice book on Software Analytics.

Chapter 5

Build analytics

5.1 Background

When building a project from source code to executables everything should go smoothly. This is not always the case, a build can break for several reasons. This chapter will give an overview of research done on build scripts and continuous integration.

5.2 Research Questions

- What are causes of a broken build?
- With which goals is continuous integration applied?
- What can be used to effectively fix a broken build?

5.3 Search Strategy

Using the initial seed consisting of Bird and Zimmermann (2017), Beller et al. (2017a), Rausch et al. (2017), Beller et al. (2017b), Pinto and Rebouças (2018), Zhao et al. (2017), Widder et al. (2018) and Hilton et al. (2016) we used references and similar keywords to find new papers to analyze.

5.4 Study Selection

Through this we found the following papers

5.4.1 Vassallo et al. (2017)

Discusses the difference in failures on continuous integration between open source software (OSS) and industrial software projects. For this 349 Java OSS projects and 418 project from ING Nederland, a financial organisation.

Using cluser analysis it was observed that both kinds of projects share similar build failures, but in other cases very different patterns emerge.

5.4.2   Hassan and Wang (2018)

This paper uses TravisTorrent (Beller et al. (2017b)) to show that 22% of code commits include changes in build script files to keep the build working or to fix the build.

In the paper a tool is proposed to automatically fix build failures based on previous changes.

5.4.3   Vassallo et al. (2018)

This paper proposes a tool called BART to help developers fix build errors. This tool eliminates the need to browse error logs which can be very long by generating a summary of the failure with useful information.

5.4.4   Zampetti et al. (2017)

This paper studies the usage of static analysis tools in 20 Java open source software projects hosted on GitHub and using Travic CI as continuous integration infrastructure. There is investigated which tools are being used, what types of issues make the build fail or raise warnings and how is responded to broken builds.

Bibliography

Beller, M., Gousios, G., and Zaidman, A. (2017a). Oops, my tests broke the build: An explorative analysis of travis ci with github. In *Mining Software Repositories (MSR), 2017 IEEE/ACM 14th International Conference on*, pages 356–367. IEEE.

Beller, M., Gousios, G., and Zaidman, A. (2017b). Travistorrent: Synthesizing travis ci and github for full-stack research on continuous integration. In *Proceedings of the 14th International Conference on Mining Software Repositories*, pages 447–450. IEEE press.

Bird, C. and Zimmermann, T. (2017). Predicting software build errors. US Patent 9,542,176.

Hassan, F. and Wang, X. (2018). Hirebuild: an automatic approach to history-driven repair of build scripts. In *Proceedings of the 40th International Conference on Software Engineering*, pages 1078–1089. ACM.

Hilton, M., Tunnell, T., Huang, K., Marinov, D., and Dig, D. (2016). Usage, costs, and benefits of continuous integration in open-source projects. In *Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering*, pages 426–437. ACM.

Pinto, G. and Rebouças, F. C. R. B. M. (2018). Work practices and challenges in continuous integration: A survey with travis ci users.

Rausch, T., Hummer, W., Leitner, P., and Schulte, S. (2017). An empirical analysis of build failures in the continuous integration workflows of java-based open-source software. In *Proceedings of the 14th International Conference on Mining Software Repositories*, pages 345–355. IEEE Press.

Vassallo, C., Proksch, S., Zemp, T., and Gall, H. C. (2018). Un-break my build: Assisting developers with build repair hints.

Vassallo, C., Schermann, G., Zampetti, F., Romano, D., Leitner, P., Zaidman, A., Di Penta, M., and Panichella, S. (2017). A tale of ci build failures: An open source and a financial organization perspective. In *Software Maintenance and Evolution (ICSME), 2017 IEEE International Conference on*, pages 183–193. IEEE.

Widder, D. G., Hilton, M., Kästner, C., and Vasilescu, B. (2018). I'm leaving you, travis: A continuous integration breakup story.

Zampetti, F., Scalabrino, S., Oliveto, R., Canfora, G., and Di Penta, M. (2017). How open source projects use static code analysis tools in continuous integration pipelines. In *Mining Software Repositories (MSR), 2017 IEEE/ACM 14th International Conference on*, pages 334–344. IEEE.

Zhao, Y., Serebrenik, A., Zhou, Y., Filkov, V., and Vasilescu, B. (2017). The impact of continuous integration on other software development practices: a large-scale empirical study. In *Proceedings of the 32nd IEEE/ACM International Conference on Automated Software Engineering*, pages 60–71. IEEE Press.