UNIVERSITÄT PADERBORN
Die Universität der Informationsgesellschaft
Faculty for Computer Science, Electrical Engineering and Mathematics
Department of Computer Science
Research Group Software Engineering

Master Thesis

Submitted to the Software Engineering Research Group
in Partial Fullfilment of the Requirements for the Degree of

Master of Science

Integration of Multiple Static Analysis Tools in a Single Interface

by
Sandeep Varma Ganaraju

Thesis Supervisors:
Prof. Dr. Eric Bodden
Dr.-Ing. Ben Hermann

Paderborn, October 2, 2019

Erklärung

Ich versichere, dass ich die Arbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen worden ist. Alle Ausführungen, die wörtlich oder sinngemäß übernommen worden sind, sind als solche gekennzeichnet.

Ort, Datum	Unterschrift

Abstract. Software is everywhere in every walk of our life with new advancements such as 'Internet of Things'. The reports show there has been a trillion of dollars loss recently because of the presence of bugs in software. Static analysis plays a major role in software development to find bugs and any vulnerabilities in code. There are different static analysis tools available in the market. However, it is found out in different surveys about why the static analysis tools are not as efficient as expected by software developers. The problems found out to be bad warning messages, complex user interface etc. Although there are solutions in industries to workaround while using a single tool but results in new issues when using multiple tools. In recent research, it is found out that in a typical software development organisation, they use multiple tools including legacy tools used in nightly builds as an example. On the other hand with ongoing research trends in using multiple static analysis tools namely Tricorder, Parfait etc. shows the opportunity and importance of developing a single interface for multiple tools. This thesis aims to address the scenario where a developer works with different tools and how adaptive could be the user interface. The novel ideas including approaches adapted from different software engineering disciplines are evaluated through user experience design cycle. Designs are made with assimilated ideas and prototypes are build using a wireframe tool. The usability aspect of the proposed ideas is considered during the evaluation phase. The target users for this evaluation are experienced software developers which ensures the applicability of this thesis work.

Keywords: Static Analysis, Usability, Wireframe, User Experience Design

Contents

1

Introduction

2

Background

3

Literature Review

4 Motivation

5

Related Work

6

Research Methodology

7

Objectives

8 Approaches

9

User Experience Cycle 1

The UX 1 plan and results discussed here.

10

User Experience Cycle 2

The UX 2 plan and results discussed here.

11

User Experience Cycle 3

The UX 3 plan and results discussed here.

12 Limitations

13

Future Work

The future work ideas discussed here.

14 Conclusion

Bibliography

[1] *Software Fail Watch*. URL: https://www.tricentis.com/news/software-fail-watch-says-1-1-trillion-in-assets-affected-by-software-bugs-in-2016/.

[2] Maria Christakis and Christian Bird. "What developers want and need from program analysis: an empirical study". In: *Automated Software Engineering (ASE), 2016 31st IEEE/ACM International Conference*. IEEE. 2016, pp. 332–343.

[3] Brittany Johnson et al. "Why don't software developers use static analysis tools to find bugs?" In: *Proceedings of the 2013 International Conference on Software Engineering*. IEEE Press. 2013, pp. 672–681.

[4] *Usability Engineering : Book by Jakob Nielsen*. URL: https://www.nngroup.com/books/usability-engineering/.

[5] *How to Change Your Career from Graphic Design to UX Design*. URL: https://www.interaction-design.org/literature/article/how-to-change-your-career-from-graphic-design-to-ux-design.

[6] *Observe, Test, Iterate, and Learn (Don Norman) (Video)*. URL: https://www.nngroup.com/videos/observe-test-iterate-and-learn-don-norman/.

[7] *Designing code analyses for Large Software Systems (DECA)*. URL: https://www.hni.uni-paderborn.de/swt/lehre/deca/.

[8] *Sample Of Covered Software Vulnerabilities (OWASP Top 10 and more)*. URL: https://www.checkmarx.com/technology/vulnerability-coverage/.

[9] *A Survey of Static Program Analysis Techniques*. URL: https://www.ics.uci.edu/~lopes/teaching/inf212W12/readings/Woegerer-progr-analysis.pdf (visited on).

[10] *Usability 101: Introduction to Usability*. URL: https://www.nngroup.com/articles/usability-101-introduction-to-usability/.

[11] *Balsamiq. Rapid, effective and fun wireframing software. | Balsamiq*. URL: https://balsamiq.com/.

[12] *The Definition of User Experience (UX)*. URL: https://www.nngroup.com/articles/definition-user-experience/.

[13] Alan Dix et al. "Spreadsheets as User Interfaces". In: *Proceedings of the International Working Conference on Advanced Visual Interfaces - AVI '16*. Ed. by Maria Francesca Costabile et al. New York, New York, USA: ACM Press, 2016, pp. 192–195. ISBN: 9781450341318. DOI: 10.1145/2909132.2909271.

[14] Garima Gaur, Sumit Kalra, and Arnab Bhattacharya. “Patterns for Indexing Large Datasets”. In: *Proceedings of the 23rd European Conference on Pattern Languages of Programs - EuroPLoP ’18*. Ed. by Unknown. New York, New York, USA: ACM Press, 2018, pp. 1–6. ISBN: 9781450363877. DOI: 10.1145/3282308.3282314.

[15] James J Horning. “What the compiler should tell the user”. In: *Compiler Construction*. Springer. 1974, pp. 525–548.

[16] dustinca. *Proceedings of the 2nd Workshop on Refactoring Tools*. New York, NY: ACM, 2008. ISBN: 9781605583396. URL: http://dl.acm.org/citation.cfm?id=1636642.

[17] *Gamification | Coursera*. URL: https://www.coursera.org/learn/gamification.

[18] Shinpei Hayashi et al. “Historef: A tool for edit history refactoring”. In: *2015 IEEE 22nd International Conference on Software Analysis, Evolution, and Reengineering (SANER)*. IEEE, 2/03/2015 - 06/03/2015, pp. 469–473. ISBN: 978-1-4799-8469-5. DOI: 10.1109/SANER.2015.7081858.

[19] Erica Mealy et al. “Improving Usability of Software Refactoring Tools”. In: *2007 Australian Software Engineering Conference (ASWEC’07)*. IEEE, 10/04/2007 - 13/04/2007, pp. 307–318. ISBN: 0-7695-2778-7. DOI: 10.1109/ASWEC.2007.24.

[20] Olga Baysal, Reid Holmes, and Michael W. Godfrey. “No issue left behind: reducing information overload in issue tracking”. In: *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering - FSE 2014*. Ed. by Shing-Chi Cheung, Alessandro Orso, and Margaret-Anne Storey. New York, New York, USA: ACM Press, 2014, pp. 666–677. ISBN: 9781450330565. DOI: 10.1145/2635868.2635887.

[21] Shaowei Wang, David Lo, and Lingxiao Jiang. “An empirical study on developer interactions in StackOverflow”. In: *Proceedings of the 28th Annual ACM Symposium on Applied Computing*. ACM. 2013, pp. 1019–1024.

[22] Christoph Treude, Ohad Barzilay, and Margaret-Anne Storey. “How do programmers ask and answer questions on the web?” In: *Proceeding of the 33rd international conference on Software engineering - ICSE ’11*. Ed. by Richard N. Taylor, Harald Gall, and Nenad Medvidović. New York, New York, USA: ACM Press, 2011, p. 804. ISBN: 9781450304450. DOI: 10.1145/1985793.1985907.

[23] *Where Developers Learn, Share, and Build Careers*. URL: https://stackoverflow.com/.

[24] Daniel Plakosh et al. *Improving the Automated Detection and Analysis of Secure Coding Violations*. Tech. rep. CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST, 2014.

[25] *Checkmarx – Application Security Testing and Static Code Analysis*. URL: https://www.checkmarx.com/.

[26] Al Bessey et al. “A few billion lines of code later: using static analysis to find bugs in the real world”. In: *Communications of the ACM* 53.2 (2010), pp. 66–75.

[27] Aurelien Delaitre et al. “Evaluating Bug Finders–Test and Measurement of Static Code Analyzers”. In: *2015 IEEE/ACM 1st International Workshop on Complex Faults and Failures in Large Software Systems (COUFLESS)*. IEEE. 2015, pp. 14–20.

[28] Daniel Plakosh et al. *Improving the Automated Detection and Analysis of Secure Coding Violations*. Tech. rep. CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST, 2014.

[29] Lori Flynn et al. “Prioritizing alerts from multiple static analysis tools, using classification models”. In: *Proceedings of the 1st international workshop on software qualities and their dependencies*. ACM. 2018, pp. 13–20.

[30] Na Meng et al. “An approach to merge results of multiple static analysis tools (short paper)”. In: *2008 The eighth international conference on quality software*. IEEE. 2008, pp. 169–174.

[31] Caitlin Sadowski et al. “Tricorder: Building a Program Analysis Ecosystem”. In: *Proceedings of the 37th International Conference on Software Engineering - Volume 1*. ICSE ’15. Florence, Italy: IEEE Press, 2015, pp. 598–608. ISBN: 978-1-4799-1934-5. URL: http://dl.acm.org/citation.cfm?id=2818754.2818828.

[32] Cristina Cifuentes and Bernhard Scholz. “Parfait: Designing a Scalable Bug Checker”. In: *Proceedings of the 2008 Workshop on Static Analysis*. SAW ’08. Tucson, Arizona: ACM, 2008, pp. 4–11. ISBN: 978-1-59593-924-1. DOI: 10.1145/1394504.1394505. URL: http://doi.acm.org/10.1145/1394504.1394505.

[33] *FindBugs™ - Find Bugs in Java Programs*. URL: http://findbugs.sourceforge.net/.

[34] *FindBugs™ - GUI Scan Results*. URL: http://findbugs.sourceforge.net/manual/gui.html.

[35] Lisa Nguyen Quang Do and Eric Bodden. “Gamifying Static Analysis”. In: *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. ESEC/FSE 2018. Lake Buena Vista, FL, USA: ACM, 2018, pp. 714–718. ISBN: 978-1-4503-5573-5. DOI: 10.1145/3236024.3264830.

[36] John David Colleran, Gerardo Bermudez, and Vadim Gorokhovky. *Responsive user interface to manage a non-responsive application*. US Patent 6,850,257. Feb. 2005.

[37] *Response Time Limits: Article by Jakob Nielsen*. URL: https://www.nngroup.com/articles/response-times-3-important-limits/ (visited on).

[38] Lorraine Borman. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. New York, NY: ACM, 1985. ISBN: 0897911490. URL: http://dl.acm.org/citation.cfm?id=317456.

[39] Lars Heinemann, Benjamin Hummel, and Daniela Steidl. “Teamscale: Software quality control in real-time”. In: *Companion Proceedings of the 36th International Conference on Software Engineering*. ACM. 2014, pp. 592–595.

[40] *Teamscale*. URL: https://www.cqse.eu/en/products/teamscale/features/.

[41] *Why You Only Need to Test with 5 Users*. URL: https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/.

[42] Jakob Nielsen and Thomas K. Landauer. “A Mathematical Model of the Finding of Usability Problems”. In: *Proceedings of the INTERACT ’93 and CHI ’93 Conference on Human Factors in Computing Systems*. CHI ’93. Amsterdam, The Netherlands: ACM, 1993, pp. 206–213. ISBN: 0-89791-575-5. DOI: 10.1145/169059.169166. URL: http://doi.acm.org/10.1145/169059.169166.

[43] Jakob Nielsen. “Usability inspection methods”. In: *Conference companion on Human factors in computing systems*. ACM. 1994, pp. 413–414.

[44] Marilyn Hughes Blackmon et al. “Cognitive walkthrough for the web”. In: *Proceedings of the SIGCHI conference on human factors in computing systems*. ACM. 2002, pp. 463–470.