University of Groningen

Introduction to the special issue on technical debt in software systems

Falessi, Davide; Kruchten, Philippe; Avgeriou, Paris

Published in:
Journal of Systems and Software

DOI:
[10.1016/j.jss.2016.06.069](https://doi.org/10.1016/j.jss.2016.06.069)

IMPORTANT NOTE: You are advised to consult the publisher's version (publisher's PDF) if you wish to cite from it. Please check the document version below.

Document Version
Publisher's PDF, also known as Version of record

Publication date:
2016

[Link to publication in University of Groningen/UMCG research database](#)

Citation for published version (APA):

Falessi, D., Kruchten, P., & Avgeriou, P. (2016). Introduction to the special issue on technical debt in software systems. *Journal of Systems and Software*, 120, 154-155.
<https://doi.org/10.1016/j.jss.2016.06.069>

Copyright

Other than for strictly personal use, it is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license (like Creative Commons).

The publication may also be distributed here under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license. More information can be found on the University of Groningen website: <https://www.rug.nl/library/open-access/self-archiving-pure/taverne-amendment>.

Take-down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from the University of Groningen/UMCG research database (Pure): <http://www.rug.nl/research/portal>. For technical reasons the number of authors shown on this cover page is limited to 10 maximum.

Editorial

Introduction to the special issue on technical debt in software systems

1. Introduction

Starting from a definition given by McConnell (2013), *technical debt* in software-intensive systems is defined as “consisting of design or implementation constructs that are expedient in the short term, but set up a technical context that can make a future change more costly or impossible. Technical debt is a contingent liability whose impact is limited to internal system qualities, primarily maintainability and evolvability”. Market opportunities bring economic benefits that help achieve project success. However, there are many examples of software development projects in which taking shortcuts to meet market opportunities have resulted in large cost overruns, severe quality issues, inability to add new features without breaking existing features, and even the premature loss of a system (Avgeriou et al., 2016). All these issues have a significant impact on profitability (Kruchten et al., 2012). On the other hand, our industry experience (Falessi et al., 2013) shows that the effort required for removing all the debt can be higher than the cost of the entire project, even in the case of a healthy and successful project. Therefore, technical debt should be monitored and decreased in a controlled manner; it is not realistic to completely eliminate it (Falessi et al., 2013, Falessi et al., 2014, Lim et al., 2012, Li et al., 2015). The rationale is that in some situations the effort required to improve the internal quality of the product can be better spent in other more valuable activities, such as reducing time-to-market. Therefore, development teams need to choose wisely whether to invest in repaying technical debt or in achieving business goals.

Over the last decade the theme of technical debt has grown in interest to practitioners and academia (Li et al., 2015). The theme has been discussed at more than eight international workshops, at several conference and in several journal papers as well as a past special issue of the *IEEE Software* magazine (Kruchten et al., 2012).

2. Overview of the issue

The present special issue comprises five papers.

The first paper, written by Yuepu Guo, Carolyn Seaman and Fabio Silva, reports on the cost and obstacles encountered when managing technical debt in industry. Regarding cost, the authors discovered several patterns including the fact that TD identification accounted for the majority of the total TD management cost. Regarding obstacles, the learning curve of understanding TD and

the technical difficulty of TD identification constitute the most important ones.

The second paper, written by Alan Maccormack and Daniel Sturtevant, reports on an analysis on the relationship between system architecture and maintenance effort. Specifically, the authors used design structure matrices (DSM) to measure software system architecture and they analyzed the relationship between coupling and maintenance costs, measured as the number of patches made to files to fix defects. Their result show that the most tightly coupled components proved to be much more expensive to maintain.

The third paper, written by Michael Mohan, Des Greer and Paul McMullan, reports a comparison of four refactoring approaches based on different fitness functions to reduce technical debt. Three of these functions relate to common properties of an object (i.e., coupling, inheritance, abstraction) whereas a fourth function is proposed by the authors and it represents the technical debt by combining different metrics. The results based on six open source projects show that the proposed fitness function significantly outperforms the other functions.

The fourth paper, written by Jesse Yli-Huomo, Andrey Maglyas and Kari Smolander, reports on an explorative case study investigating how software development teams manage technical debt in a real-life industrial environment. Specifically, the authors collected and analyzed data in the case organization by interviewing a total of 25 persons in eight software development teams. They found that the technical debt management process was conducted at various levels and that it has many similarities with the capability maturity model (CMM) (Avgeriou et al., 2016). Finally, they propose a technical debt management framework that describes the management activities, stakeholders and responsibilities on three levels and approaches, practices, and tools.

The fifth and last paper, written by Claurton A. Siebra, Rebeka G. Oliveira, Carolyn B. Seaman, Fabio Q. B. Silva, and Andre L. M. Santos, reports on a theoretical conceptualization of technical debt. Specifically, the work contributes to the effort in building a formal theory about TD and it provides directions to assist the work of developers/managers who intend to identify and monitor TD items in their projects.

Acknowledgments

We are grateful to all the authors and reviewers for their excellent work. We also thank the Editor in Chief (Hans van Vliet)

and the Special Issues Editor (Wing-Kwong Chan) for their support throughout the process of preparing this special issue.

References

- Avgeriou, P., Kruchten, P., Nord, R., Ozkaya, I., Seaman, C., January/February 2016. Reducing Friction in Software Development. *IEEE Softw.* 33 (1), 66–73.
- Falessi, D., Kruchten, P., Nord, R., Ozkaya, I., 2014. Technical Debt at the Crossroads of Research and Practice: Report on the Fifth International Workshop on Managing Technical Debt. *ACM SIGSOFT Software Engineering Notes* vol. 39 (2), 1–15.
- Falessi, D., Shaw, M.A., Shull, F., Mullen, K., Stein, M., 2013. Practical Considerations, Challenges, and Requirements of Tool-Support for Managing Technical Debt. In: *Proceedings of the 4th International Workshop on Managing Technical Debt*, pp. 16–19.
- Kruchten, P., Nord, R.L., Ozkaya, I., Nov. 2012. Technical debt: from metaphor to theory and practice. *IEEE Softw.* 29 (6), 18–21.
- Lim, E., Taksande, N., Seaman, C., Nov. 2012. A balancing act: what software practitioners have to say about technical debt. *IEEE Softw.* 29 (6), 22–27.
- Li, Z., Avgeriou, P., Liang, P., March 2015. A systematic mapping study on technical debt and its management. *J. Syst. Softw.* 101, 193–220.
- McConnell, S., 2013. Managing Technical Debt. [Online]. Available: <http://www.sei.cmu.edu/community/td2013/program/upload/technicaldebt-icse.pdf>

Davide Falessi is an Associate Professor of Computer Science at the California Polytechnic State University, USA. He is the Associate Editor in Software Economics and the Multimedia Editor of *IEEE Software*. He is also a member at large of IEEE Computer Science publications board. His main research interest is in devising and empirically assessing scalable solutions for managing technical debt of software applications in industry. He received the PhD, MSc, and BSc degrees in Computer Engineering from the University of Rome “Tor Vergata”, Italy.

Philippe Kruchten is a Professor of Software Engineering at the University of British Columbia in Vancouver, Canada. His current research interest are in software architecture and the software development process, technical debt sitting nicely at the intersection of the two. He's an associate editor of *IEEE Software*. He received his degrees in mechanical engineering from Ecole Centrale de Lyon, France, and a doctorate degree in information systems for the Ecole nationale Supérieure de Télécommunications in Paris.

Paris Avgeriou is a Professor of Software Engineering at the University of Groningen, in the Netherlands. Before joining Groningen, he was a post-doctoral Fellow of the European Research Consortium for Informatics and Mathematics. He sits on the editorial board of *IEEE Software* and *Springer Transactions on Pattern Languages of Programming (TPLOP)*. His research interests lie in the area of software architecture, with strong emphasis on architecture modeling, knowledge, evolution, patterns, metrics and technical debt.

Davide Falessi*
California Polytechnic State University, USA

Philippe Kruchten
University of British Columbia, Canada

Paris Avgeriou
University of Groningen, The Netherlands

* Corresponding author.

E-mail addresses: dfalessi@calpoly.edu (D. Falessi), pbk@ece.ubc.ca (P. Kruchten), paris@cs.rug.nl (P. Avgeriou)