Ninth International Workshop on Managing Technical Debt

Report on the MTD 2017 Workshop

Francesca Arcelli Fontana
Università degli Studi di Milano-Bicocca
Milano, Italy
arcelli@disco.unimib.it

Wolfgang Trumler
Corporate Technology, Siemens AG
Erlangen, Germany
wolfgang.trumler@siemens.com

Clemente Izurieta
Montana State University
Bozeman, MT, US
clemente.izurieta@montana.edu

Robert L. Nord
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA, US
rn@sei.cmu.edu

ABSTRACT

Welcome to the Ninth International Workshop on Managing Technical Debt, collocated with the 18th International Conference on Agile Software Development (XP 2017) in Cologne. The technical debt research community continues to expand through collaborations of industry, tool vendors, and academia. The main topic of this year's workshop was on the impact of agile development approaches towards the management of technical debt.

CCS CONCEPTS

• **Software and its engineering → Software evolution**; **Maintaining software**; *Software architectures*; *Extra-functional properties*; *Software maintenance tools*; • **General and reference →** *Measurement*; *Metrics*;

KEYWORDS

Technical debt, software economics, software quality, software evolution, software analytics, agile development

ACM Reference format:
Francesca Arcelli Fontana, Wolfgang Trumler, Clemente Izurieta, and Robert L. Nord. 2017. Ninth International Workshop on Managing Technical Debt . In *Proceedings of XP '17 Workshops, Cologne, Germany, May 22-26, 2017,* 3 pages.
https://doi.org/.1145/3120459.3120461

1 INTRODUCTION

The Ninth International Workshop on Managing Technical Debt (MTD 2017) began with a short introduction by Francesca Arcelli and Wolfgang Trumler on managing technical debt. Researchers have met regularly since 2010, in the workshop series on Managing Technical Debt (MTD), to further study and better define the concept and its applicability to software development. In addition, research focusing on technical debt has started to be part of the main tracks of major software engineering conferences. In 2016 there was a Dagstuhl seminar [4] on advancing research on managing technical debt.

Previous workshops had made progress by creating an initial landscape for scoping technical debt [2], establishing key principles of what constitutes technical debt and what it is not [3], establishing how different lines of research in software engineering form the basis of technical debt research and starting the beginnings of a research roadmap [1].

This ninth workshop focused on topics related to technical debt management and addressing technical debt on the code level under special consideration of agile approaches. One of the key questions was whether agile approaches support the management of technical debt or if they foster their introduction due to a misaligned understanding of agile development.

The workshop had a keynote given by Eltjo Poort of CGI[1], Netherlands, with a talk on "Selling the Business Case for Architectural Debt Reduction". He presented relevant points about the importance of quantifying technical debt by identifying the risk exposure related to certain functionality, which helps the business stakeholders to understand the need for regularly planned refactoring.

The workshop continued with two sessions presenting papers accepted for publication, one in the morning on Technical Debt Management chaired by Jean-Louis Letouzey and one in the afternoon on Technical Debt at the Code Level chaired by Klaus Schmidt.

2 WORKING SESSIONS

The MTD workshop series has traditionally reserved the last portion of the meeting to lively discussions that engage all attendees. This year three topics were identified: Architectural technical debt assessment, Agility: solution to technical debt or its root cause?, and Selling the business case of technical debt management.

XP '17 Workshops, May 22-26, 2017, Cologne, Germany

ACM ISBN 978-1-4503-5264-2/17/05…$15.00
https://doi.org/.1145/3120459.3120461

[1] https://www.cgi.com/

2.1 Architectural technical debt assessment

The discussion chaired by Alexandros Chatzigeorgiou was centered on the question whether technical debt, in particular architectural technical debt (ATD), can be assessed by means of methods and tools which eventually can support decision making of software architects. The participants related that tools such as static code analyzers are unable to suggest meaningful opportunities for improving code quality with respect to forthcoming changes and in providing hints to developers and architects so as to assist them in facilitating further software evolution (e.g., the transition to a new database, adoption of a new framework or library). Tools for the identification and quantification of technical debt cannot rely on static source code analysis only. Multiple indicators should be exploited by analysing all sources of information around a software project (including issues, change history, etc.). It would be valuable to identify commonly recurring instances of ATD and to classify them into well-defined and identifiable categories. Setting up a repository of such architectural features as architectural smells and decisions is of utmost importance to drive the development of efficient tools that can identify and quantify ATD. The discussants considered also that identifying technical debt items without anticipating future scenarios for the evolution of a software system might render technical debt management useless. The identification and quantification of technical debt should be elevated from the level of extracting long lists of low-level code smells to the level of suggesting architectural opportunities that have the power to facilitate future change scenarios.

2.2 Agile approaches and their impact on technical debt management

The guiding question of the discussion chaired by Wolfgang Trumler was on how agile approaches influence technical debt and whether they support removal or rather introduction of technical debt. Three important topics from agile approaches have been identified, which have a significant impact on technical debt mangament.

The first topic is the potentially misunderstood notion of value in agile approaches, fostering the introduction of technical debt when focusing on end-customer's value only. However, there are more stakeholders of a software product like the operations team, testers, and so on, who have additional requirements valuable for them and their work.

The second topic was about implementation-related aspects. Following the Build-Measure-Act-cycle [5] described by Eric Ries, development teams might be tempted to quickly implement features to prove whether their assumption regarding customer value is correct or not. Unfortunately, many teams do not plan effort to be spent to clean-up and refactor to reduce the technical debt taken consciously in order to be fast.

We discussed a model we call GMC-cycle (Go fast, Measure, Clean-up), which could lead to a guiding principle or even as a business model. Tell the customer that the implementation of a feature will be done with the least investment and as fast as possible to validate the impact and value it is intended to create. Only if the customer is happy with the solution would he or she have to spend additional money for clean-up and refactoring activities before moving on with the next feature. Approaching agile development in this way inherently includes the reduction of technical debt as an explicit step with a maximum of information reagrding the feature and functionality to be added.

The third topic discussed was about the development practices often used in industry. As the process-related parts of agile methodologies like SCRUM and practices like continuous integration are easy to train and practice, they are adopted widely. The hard parts like test-driven development (TDD) and Pair Programming imply a change in behavior and mind set for the development team and are therefore often left out. However, the latter practices are those used to address quality and to reduce technical debt, while the former help to increase transparency about the development pace and project status.

2.3 Selling the business case of technical debt management

The group chaired by Paris Avgeriou was tasked with discussing how to sell the business case of managing technical debt. The group picked up the topic of the key note, discussed these ideas and elaborated further based on industrial experiences of the participants. The group also discussed the current state of practice in technical debt management in the different companies that participants represented.

The participants confirmed that presenting imminent risks is a very effective way to argue the case of technical debt. Risks are quantifiable providing figures to base decisions. They are also commonly used in stakeholder meetings, particularly those involving both technical and business stakeholders.

Another option is to ask the development team to score the "pain felt" on a given scale, for example, from 1 to 5. It might be enough to gauge the amount of technical debt and the urgency to resolve it.

Yet another way is to use the language business stakeholders understand. Terms like debt and interest can be more persuasive especially if they come from technical stakeholders with certain credibility in the organization. However, it is equally important to find a sponsor in the organization willing to support investments in technical debt management.

Arguing the case for technical debt management can be supported by source code analysis tools, which measure technical debt and related qualities. On the one hand, such figures can be easily expressed in a monetary way. On the other hand, some of them are highly customizable which makes the calculation process less credible as tweaking parameters can have drastic impact on reducing or increasing the calculated technical debt.

3 CONCLUSION

By holding this workshop together with XP, the workshop received a great deal of attention from leading software researchers and practioners interested in exploring theoretical and practical techniques that manage technical debt within iterative and agile software development environments. The discussions during the working sessions generated interesting points for future investigation.

The workshop closed with the announcement that the MTD workshop will evolve into a two day working conference focusing on technical debt, collocated with the International Conference on Software Engineering in Gothenburg, Sweden in May 2018 (techdebtconf.org).

4 ACKNOWLEDGEMENTS

Our thanks to the organizers of XP 2017, XP Workshop Chairs, and the keynote speaker Eltjo Poort. We would like to thank our program committee for their thorough and timely reviews of the submissions. The attendance at the MTD 2017 workshop shapes the future directions of the research in this field and its pragmatic applications in industry.

REFERENCES

[1] Clemente Izurieta, Ipek Ozkaya, Carolyn Seaman, and Will Snipes. 2017. Technical Debt: A Research Roadmap, Report on the Eighth Workshop on Managing Technical Debt (MTD 2016). *ACM SIGSOFT Software Engineering Notes* 42, 1 (2017), 28–31.

[2] Philippe Kruchten, Robert L. Nord, and Ipek Ozkaya. 2012. Technical Debt: From Metaphor to Theory and Practice. *IEEE Software* 29, 6 (2012), 18–21.

[3] Philippe Kruchten, Robert L. Nord, Ipek Ozkaya, and Davide Falessi. 2013. Technical debt: Towards a Crisper Definition, Report on the 4th International Workshop on Managing Technical Debt. *ACM SIGSOFT Software Engineering Notes* 38, 5 (2013), 51–54.

[4] Dagstuhl Reports. 2016. Managing Technical Debt in Software Engineering. *Dagstuhl Reports* 6 (April 17-22 2016). http://www.dagstuhl.de/16162

[5] Eric Ries. 2011. *The Lean Startup: How Constant Innovation Creates Radically Successful Businesses.* Portfolio Penguin.