On the Perceived Harmfulness of Requirement Smells: An Empirical Study

Valentina Lenarduzzi
LUT University
Lahti, Finland
valentina.lenarduzzi@lut.fi

Davide Fucci and Daniel Mendéz
Blekinge Institute of Technology
Karlskrona, Sweden
davide.fucci;daniel.mendez@bth.se

Abstract

Technical debt is considered to have negative effects to the long term success of software projects. However, how the debt metaphor applies to requirements engineering is yet not significantly explored. Previously, we proposed a framework to identify Requirements Debt (ReD) in three stages of the software development lifecycle. One of these stages is the formalization of stakeholder needs into natural language requirement specifications. In this work, we propose a live study aiming at surveying requirements engineering experts to gain further insights on the issues taking place at this stage and how they fit in our definition of ReD.

1 Introduction

Cunningham defines Technical Debt (TD) as *"the debt incurred through the speeding up of software project development which results in a number of deficiencies ending up in high maintenance overheads"* [Cun92]. TD implies sub-optimal design or implementation solutions giving a short-term benefit while making changes more costly or even impossible in the medium and long term. Unpredictable business and environmental forces, internal or external to a company, can result in TD which needs to be managed [MBC15, BML+18].

The TD metaphor was initially concerned with software implementation (i.e., at code level), but it has been gradually extended to software architecture, design, documentation, testing, and requirements [BCG+10]. Li et al. [LAL15] conducted a systematic mapping study on understanding and managing TD drawing an overview on the current state of research. They proposed a classification of nine types of TD: Requirements, Architectural, Design, Code, Test, Build, Documentation, Infrastructure, and Versioning.

A first definition of requirements debt by Ernst was "*the distance between the optimal requirements specification and the actual system implementation, under domain assumptions and constraints*" [Ern12]. Despite the importance of requirements engineering activities during software development process [Sch13, ARC+14] and the definition of the minimum viable product (MVP) [LT16], there is still no consensus in research whether ReD should be considered as a type of technical debt or not [ARC+14]. Different processes could led to different requirement decomposition and accumulate different debt [TLJ+17]. We believe the reason is the lack of formalization of requirement debt in the literature [LAL15], [LBT+19].

In: M. Sabetzadeh, A. Vogelsang, S. Abualhaija, M. Borg, F. Dalpiaz, M. Daneva, N. Condori-Fernández, X. Franch, D. Fucci, V. Gervasi, E. Groen, R. Guizzardi, A. Herrmann, J. Horkoff, L. Mich, A. Perini, A. Susi (eds.): Joint Proceedings of REFSQ-2020 Workshops, Doctoral Symposium, Live Studies Track, and Poster Track, Pisa, Italy, 24-03-2020, published at http://ceur-ws.org

We extended the definition of requirements debt by Ernst to include upstream activities involving the elicitation of requirements (particularly in user-centered requirements engineering [MNJR15]) and their translation into specifications [LF19]. Moreover, we outlined the future assessments to conceptualize and define the decision framework [LF19]. Within ReD, we proposed three definitions of debt based on *Incomplete Users' needs* (ReD type 0), *Requirement smells* (ReD type 1), and *Mismatch implementation* (ReD type 2), defining how to detect, how to quantify, and how to pay back each of ReD type. Our vision of ReD will be empirically evaluated in a series of studies with industry partners and individual stakeholders.

In this study at hands, we focus on ReD Type 1 to understand and compare the perceived harmfulness of requirement smells from a theoretical and practical perspective—i.e., an indicator for a quality violation of a requirements artifact [FWE17]. We focus on requirement specifications written in natural language. Based on the results obtained in the study presented in here, we will design and conduct a large-scale surveys involving companies in order to monitor the requirement elicitation process.

2 Plan and Design

Research Problem. We aim at understanding how the ReD Type 1 and the associated idea of code smells are perceived by requirements engineering experts.

By means of a cross-sectional questionnaire survey, we plan to elaborate on Type 1 ReD as there is little evidence of harmfulness of requirement smells when following the definition and the detection approach proposed by Femmer et al. [FWE17]. We will follow the approaches widely adopted to assess the harmfulness of code smells on different software qualities [OCS10], [SYA$^+$13], [HZBS14], [PBP$^+$18].

In a second, later phase, will plan to triangulate the results of the survey with quantitative data from issue trackers and requirements repositories platforms [LT19] and qualitative data from case studies involving requirements engineers, business analysts, and software developers [TJL17].

Type of study. We design our live study as a cross-sectional survey based on *Surveys*.

Research goal. The goal of the survey is to understand and compare the theoretical and practical perceived harmfulness of requirement smells. To this end, we survey experts on their understanding and assessment of the relevance of the requirements smells for their context and their daily activities including different context factors. Moreover, the participants feedback will help us to fulfill the additional goal of laying ground for future follow-up studies.

Research questions. We formulated the following research question.
RQ1. How harmful are the different requirement smells perceived by practitioners after only reading their definitions?

In order to answer to our RQ, we will provide the description and an example of each requirement smell and we will collect the perceived harmfulness of a requirement smell from the practitioners' point of view. In this RQ we do not evaluate whether practitioners know a specific requirement smell but only whether they consider it as a threat in the requirement elicitation process.

Population of interest The target population of the survey includes roles interacting with a requirement specification artifact—in particular, for this study, a natural language formalization of a user need. Such roles include business analysts, product owners, team leads&developers. Moreover, we consider academic people in case they have conducted research and/or taught on requirement and/or technical debt fields. For this study, we are not interested in a sample of the above roles attached to any specific domain.

Participants get deeper insights into the notion of requirements debt and smells in particular, thus, strengthening their learning curve.

Study Design. The survey will be based on a questionnaire organized into three sections. In order to provide the requirement smells' knowledge to the participants, we will include the list of smells in the questionnaire. Moreover, we will provide some examples of selected requirement smells.

1) Personal information. We aim to collect the profile of the practitioners, considering age, country, gender, predominant roles, and working experience in requirement engineering. Moreover, we will collect the organization size via the number of employees and the common application domain.

2) Knowledge of Requirement Smells. We aim to understand whether the participants are familiar with Requirement Smells and whether they already consider the removal of Requirement Smells in their requirement

elicitation process. This section of the questionnaire is useful for understanding whether the answers provided are based on personal experience and previous knowledge of Requirement Smells or only on the reading of the description we provided in the next question.

Requirements smells. We considered the Requirements Smells defined by [FWE17], based on the ISO 29148 requirements engineering standard [1].

Table 1: Requirement Smells [FWE17]

Requirement Smells	**Description**	**Detection Strategy**
Subjective Language	"Subjective Language refers to words of which the semantics is not objectively defined, such as user friendly, easy to use, cost effective"	Dictionary
Ambiguous Adverbs and Adjectives	"Ambiguous Adverbs and Adjectives refer to certain adverbs and adjectives that are unspecific by nature, such as almost always, significant and minimal"	Dictionary
Loopholes	"Loopholes refer to phrases that express that the following requirement must be fulfilled only to a certain, imprecisely defined extent"	Dictionary
Open-ended, non-verifiable terms	"Open-ended, non-verifiable terms are hard to verify as they offer a choice of possibilities, e.g. for the developers"	Dictionary
Superlatives	"Superlatives refer to requirements that express a relation of the system to all other systems"	Morphological Analysis, POS tagging
Comparatives	"Comparatives are used in requirements that express a relation of the system to specific other systems or previous situations"	Morphological Analysis, POS tagging
Negative Statements	"Negative Statements are statements of system capability not to be provided. Some argue that negative statements can lead to under specification, such as lack of explaining the system's reaction on such a case"	Dictionary, POS tagging
Vague Pronouns	"Vague Pronouns are unclear relations of a pronoun"	POS tagging

3) Perceived Harmfulness of Requirement Smells. We aim to capture the general criticality perception of from our respondents. We will ask to rate how concerned they are about a smells in general and about the requirement smells reported in Table 1.

Study Methods and Procedures. We scheduled 30 minutes for the survey and other 30 minutes for introduction and questions. The surveys will be carried out by means of a questionnaire based on open-ended questions and ordinal 4-points Likert scale, where 1 corresponded to not concerned at all and 4 to very concerned. Moreover, at the end of the survey, we left space for further comments. We will provide the questionnaire also by online version.

Feedback session. The last step will be a semi-structured feedback session. First, we will ask the participants to reflect on the form and contents of each survey questions to get improvement ideas. On top of that, we will discuss with the participants further improvement for the survey (e.g., additional questions which would better help us measuring one of the relevant constructs).

Equipment and infrastructure needed for performing the live study We require a computer with access to the Internet in case the participants will not have access to their own laptops/wifi. Moreover, we require a whiteboard and post-it notes for collecting and aggregating feedback.

Data Collection. We will conduct the survey by administering the questionnaire to the participants during the dedicated sessions.

Participants will have the chance to enter their data in an anonymous form or to report their email for the followup studies.

The questionnaire will be GDPR-compliant. We will add a section describing the goal of the questionnaire, how we collect the data, how we threat the data and giving the chance to the participants to withdraw their participation also after the data collection.

Data Analysis. We will partition our responses into more homogeneous sub-groups based on demographic information and compare the responses obtained from all the participants with the different subgroups. For questions measuring the association between categorical variables, we will use a Chi-square test. Ordinal data, such as Likert scales, will not be converted into numerical equivalents since using a conversion from ordinal to numerical data entails the risk that subsequent analysis will give misleading results if the equidistance between the

[1] https://standards.ieee.org/standard/29148-2011.html

values cannot be guaranteed. Moreover, analyzing each value of the scale allows us to better identify the possible distribution of the answers [MPM19]. Open questions will be analyzed via open and selective coding [Bee00]. We will extract codes from the answers provided by the participants and answers group them into different code smells. The authors will independently conduct the data analysis; the final set of themes will be constructed iteratively based on the authors agreement.

3 Relevance and Feasibility of the study

Relevance of study for research and/or for practice. Requirement elicitation is one of the most important activities in the software development lifecycle. In case of issues during this phase, problems will be very expansive to be fixed. Therefore, a clear and validated approach to reduce issues during requirement elicitation is to avoid to introduce requirement smells. Thanks to this study we will validate the difference between theoretical and practical perceived harmfulness of requirement smells.

Benefits to the subjects of participating in the study. Participants will have the chance to get in touch with requirement smells. We will provide an appendix to the questionnaire that practitioners will keep for them reporting a clear summary of the different smells, that can be used as guideline during their daily work.

Plan to make publicity of the study if selected, and to attract respondents at the conference. We will advertise the survey by social networks, especially using the channels adopted by the conference. During the conference, the authors of this proposal will be present and will discuss with potential participants.

Sharing of preliminary and summary results with attendees during the conference. During the closing session, we will publish in social media the summary of the results of the closed-answer questions. Open-questions will require more time to be analyzed.

Dissemination of the results. Results will be submitted to international journals. Moreover, we will create a blog post for practitioners in order to ease the accessibility of the results itself.

4 Threats to Validity

We identified some threats to validity to our study. Since we designed the survey as a questionnaire, the participants cannot ask clarification regarding the questions. We will ask experts in empirical software engineering to review the questionnaire to improve its comprehensibility.

The survey design, its execution, and the quantitative analysis will follow a strict protocol to ease its replication. The qualitative analysis of the open questions—which is more subjective to some extent—will be documented in both process (e.g., how conflicts between annotators are resolved) and output (i.e., each code will be documented).

One limitation is that surveys can only reveal the perceptions of the respondents which might not fully represent reality. The responses will be analyzed and quality-checked by a team of four researchers.

Given the settings in which we will run the survey, we foresee two main threats due to sampling, self-selection —i.e., as the participation in the survey will be voluntary, the characteristic of the people who selected themselves to be part of the group can have an impact on the results—and sampling frame bias—i.e., the method used to select participants, which in our case is limited to the conference attendants. Nevertheless, the goal of the survey is not to achieve generalization but rather validating the constructs (e.g., the ReD stages) and obtain early feedback on the survey tool itself.

5 About the researchers

The study will be conducted by two senior researchers (Davide Fucci and Valentina Lenarduzzi) and one associate professor (Daniel Mendez) who have strong and solid experience in empirical studies.

Valentina Lenarduzzi is a Researcher in Software Engineering at LUT University. Her research is on Empirical Software Engineering with a particular focus on Technical Debt. She has experience in conducting empirical studies and especially these type of studies during practitioners and academic conferences ([TJL17], [TL18], [TLP17]). Further information on her activities, projects, and publications are available at `http://www.valentinalenarduzzi.it`.

Davide Fucci is an Assistant Professor with the department of Software Engineering at the Blekinge Institute of Technology, Sweden. His research interests lies in Empirical Software Engineering applied to data-driven requirements engineering activities and the application of natural language processing techniques to software engineering problems. Further information are available at `http://www.dfucci.co`.

Daniel Méndez is an Associate Professor in Software Engineering at the Blekinge Institute of Technology, Sweden, and Senior Scientist at fortiss, the research institute of the Free State of Bavaria for software-intensive systems and services. His research is on Empirical Software Engineering with a particular focus on interdisciplinary, qualitative research in Requirements Engineering and its quality improvement. Further information on his activities, projects, and publications are available at `http://www.mendezfe.org`.

References

[ARC$^+$14] N. S. R. Alves, L. F. Ribeiro, V. Caires, T. S. Mendes, and R. O. Spínola. Towards an ontology of terms on technical debt. In *International Workshop on Managing Technical Debt*, pages 1–7, 2014.

[BCG$^+$10] N. Brown, Y. Cai, Y. Guo, R. Kazman, M. Kim, P. Kruchten, E. Lim, A. MacCormack, R. Nord, I. Ozkaya, R. Sangwan, C. Seaman, K. Sullivan, and N. Zazworka. Managing technical debt in software-reliant systems. In *Workshop on Future of Software Engineering Research*, pages 47–52, 2010.

[Bee00] N. Beech. Basics of qualitative research: Techniques and procedures for developing grounded theory. In *2nd edn. Management Learning*, 2000.

[BML$^+$18] T. Besker, A. Martini, R. Edirisooriya Lokuge, K. Blincoe, and J. Bosch. Embracing technical debt, from a startup company perspective. In *International Conference on Software Maintenance and Evolution (ICSME)*, pages 415–425, Sep. 2018.

[Cun92] W. Cunningham. The wycash portfolio management system. In *OOPSLA '92*, 1992.

[Ern12] N.A. Ernst. On the role of requirements in understanding and managing technical debt. In *International Workshop on Managing Technical Debt*, MTD '12, pages 61–64, 2012.

[FWE17] H. Femmer D. Méndez Fernández, S. Wagner, and S. Eder. Rapid quality assurance with requirements smells. *Journal of Systems and Software*, 123:190–213, 2017.

[HZBS14] T. Hall, M. Zhang, D. Bowes, and Y. Sun. Some code smells have a significant but small effect on faults. *ACM Trans. Softw. Eng. Methodol.*, 23(4):33:1–33:39, September 2014.

[LAL15] Z. Li, P. Avgeriou, and P. Liang. A systematic mapping study on technical debt and its management. *Journal of Systems and Software*, 101:193 – 220, 2015.

[LBT$^+$19] V. Lenarduzzi, T. Besker, D. Taibi, A. Martini, and F. Arcelli Fontana. Technical debt prioritization: State of the art. a systematic literature review, 2019.

[LF19] V. Lenarduzzi and D. Fucci. Towards a holistic definition of requirements debt. *International Symposium on Empirical Software Engineering and Measurement (ESEM)*, 2019.

[LT16] V. Lenarduzzi and D. Taibi. Mvp explained: A systematic mapping study on the definitions of minimal viable product. In *42th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 112–119, Aug 2016.

[LT19] V. Lenarduzzi, , and N. Saarimaki D. Taibi. The technical debt dataset. In *The Fifteenth International Conference on Predictive Models and Data Analytics in Software Engineering (PROMISE'19)*, 2019.

[MBC15] A. Martini, J. Bosch, and M. Chaudron. Investigating architectural technical debt accumulation and refactoring over time: A multiple-case study. *Information and Software Technology*, 67:237 – 253, 2015.

[MNJR15] W. Maalej, M. Nayebi, T. Johann, and G. Ruhe. Toward data-driven requirements engineering. *IEEE Software*, 33(1):48–54, 2015.

[MPM19] J. Seide Molléri, K. Petersen, and E. Mendes. An Empirically Evaluated Checklist for Surveys in Software Engineering. *Information and Software Technology*, page 1 33, 2019.

[OCS10] M.S. Olbrich, D. Cruzes, and D. Sjøberg. Are all code smells harmful? a study of god classes and brain classes in the evolution of three open source systems. In *IEEE International Conference on Software Maintenance, ICSM*, pages 1–10, 09 2010.

[PBP+18] F. Palomba, G. Bavota, M. Di Penta, F. Fasano, R. Oliveto, and A. De Lucia. On the diffuseness and the impact on maintainability of code smells: A large scale empirical investigation. *Empirical Softw. Engg.*, 23(3):1188–1221, June 2018.

[Sch13] K. Schmid. On the limits of the technical debt metaphor some guidance on going beyond. In *4th International Workshop on Managing Technical Debt (MTD)*, pages 63–66, 2013.

[SYA+13] D. Sjøberg, A. Yamashita, B. Anda, A. Mockus T., and Dybå. Quantifying the effect of code smells on maintenance effort. 39:1144–1156, 08 2013.

[TJL17] D. Taibi, A. Janes, and V. Lenarduzzi. How developers perceive smells in source code: A replicated study. *Information and Software Technology*, 92:223 – 235, 2017.

[TL18] D. Taibi and V. Lenarduzzi. On the definition of microservice bad smells. *IEEE Software*, 35(3):56–62, 2018.

[TLJ+17] D. Taibi, V. Lenarduzzi, A. Janes, K. Liukkunen, and M.O. Ahmad. Comparing requirements decomposition within the scrum, scrum with kanban, xp, and banana development processes. In *Agile Processes in Software Engineering and Extreme Programming*, pages 68–83. Springer International Publishing, 2017.

[TLP17] D. Taibi, V. Lenarduzzi, and C. Pahl. Processes, motivations, and issues for migrating to microservices architectures: An empirical investigation. *IEEE Cloud Computing*, 4(5):22–32, 2017.