Developing a theory based on the causes of technical debt injection into software projects in Colombia

Developing a theory based on the causes of technical debt injection into software projects in Colombia

B Perez[1,2], C Castellanos[2], and D Correal[2]

[1] Grupo de Investigación en Inteligencia Artificial, Universidad Francisco de Paula Santander, San José de Cúcuta, Colombia

[2] Grupo de Investigación en Tecnologías de Información y Construcción de Software, Universidad de los Andes, Bogotá, Colombia

E-mail: borisperezg@ufps.edu.co, cc.castellanos87@uniandes.edu.co

Abstract. Engineering, like sciences such as physics, seeks to ensure that the conclusions reached can be verified through experiments, and that subsequent theories allow predictions to be made in future observations. This procedure is applied in this article, whose objective is the identification of the causes that favor the injection or presence of technical debt, and its application in software projects in Colombia, to build a theory that allows explaining this behavior. In turn, these causes could be also extrapolated to physical research projects. Results showed that "focus on producing more at the expense of quality" is the most cited cause in general. Conversely, "planning and management" is the most cited category of causes of technical debt. The concept of technical debt is well understood by software practitioners and that most efforts need to be invested by researchers on offering strategies and tools to support technical debt management.

1. Introduction

Companies related to software development have an increasing pressure to deliver a high quality solution capable of keeping functional in the long term and, at the same time, improving their effectiveness in each new project, by reducing time or resources [1]. This leads software teams to take decisions to accomplish short-term goals but possibly affecting negatively the maintainability of the system [2]. These kind of decisions are named technical debt (TD).

The presence of technical debt is inevitable because it is not always possible to have a full grasp of the totality of the problem from the beginning of the project [3]. Once the debt is incurred, it is necessary to understand how practitioners manage it. Avoid TD from the beginning is almost impossible, it is more realistic to deal with the TD and manage it, than try to eliminate it [4]. Difficulties arise because there is a lack of empirical studies concerning the management of TD in the software industry [5] making it hard to understand its current gaps.

Besides eliminating the debt presence in software projects, it is also important to understand why and how this debt was first injected in these projects. Much research have be done to understand the causes and effects of TD on software projects [6–9]. However, the sample sizes tend to be quite small, and limited to a small number of different organizations [10]. Further, half of the relevant studies focused on architectural TD, just one type of debt [10, 11].

The goal of this paper is to understand the software industry in Colombia about the status quo of TD and its causes. We focus our analysis in two software development roles: software

architects and developers. These results are part of InsighTD, a globally distributed family of industrial surveys focused on TD investigation [10]. This work is aligned with a growing concern of the software engineering community: the replication of empirical studies. It contributes to the generation of knowledge in a given topic through the accumulation of evidence about the findings, thereby increasing the level of confidence in results [12]. Following we describe our methodology in section 2. The results are presented in section 3 and discussed in section 4. Then, we conclude the paper in ection 5.

2. Methodology

In the following, we discuss our research questions, and present the target population, the questionnaire itself, and the data analysis procedure.

2.1. Research questions

This study aims to present the results of how TD is perceived by software architects and developers. We considered these roles relevant because many decisions in software implementation are taken by them. Our research questions (RQ) are presented below:

- RQ1: How familiar are software teams with the concept of TD?
- RQ2: From the point of view of software architects and developers, what causes lead software development teams to incur TD?

2.2. Survey design

This survey collects two types of data: subjective (individual's opinions, attitudes, and preferences) and objective data (demographic information). Also, this survey can be classified as both exploratory and descriptive research because (i) it is focused in the discovery of insights, (ii) it was pre-planned and structured, and (iii) the information collected can be statistically inferred over a population.

2.3. Population

This survey replication was online from January 1th to March 30th, 2019. We sent the invitation to potential targets in LinkedIn. We searched for software practitioners that work (or worked) in several software development areas (*e.g.*, testing, coding, documentation, requirements, and management). In total, we sent the survey invitation to about 1230 professionals and only 160 of them completed the questionnaire. This represents an approximate response rate of 13%. This is an interesting number of respondents considering that no benefits were given for answering the survey. Thirteen participants who answered the questionnaire were excluded from the final dataset because (i) they were not working in Colombia at that time (3) or (ii) did not provide a valid example of a TD item according to McConnell's TD definition in question 13. From this population, we later applied filters by the required roles.

2.4. Questionnaire

Survey questions are defined within the InsighTD replication package. The survey instrument was made up of 28 questions using Google forms: fourteen closed-ended questions and fourteen open-ended. Some of the closed questions include a free text option (*e.g.*, other) so that the participants can express their opinion more appropriately.

Demographics questions (question 1 to 8) ask participants about, for example, the size of his/her company, size of the system (in terms of lines of code) he/she is working on, number of people involved in that project, participant's role, and her/his level of experience in that role. Questions 9 to 15 seek information about how familiar the respondent is with the TD concept.

Questions 16 to 19 support the identification of the causes that lead development teams to insert debt items into their projects. Questions 20 and 21 look to identify effects of the presence of TD in software projects. Finally, Questions 22 to 28, were used to provide an understanding on how TD has been managed in practice, in particular with respect to prevention, repayment, and monitoring. The full questionnaire was previously presented in [10].

In the context of this work, we considered for analysis the characterization (questions 1 to 8), familiarity with the concept of TD (questions 9 and 11) and causes of TD (questions 16 to 19).

2.5. Data analysis

For closed-ended questions, we used descriptive statistics to get a better understanding of the data. Answers for open-ended questions were codified using a code schema provided with the InsighTD replication package. This schema is composed of a list of causes identified in the Brazilian round of the survey and eight categories of causes derived from it: lack of knowledge, external factors, infrastructure, methodology, organizational, people, planning and management, and development issues.

Qualitative analysis was performed through the following steps: (i) Two researchers analyzed each participant's answer, (ii) causes were identified following the InsighTD's code schema, (iii) a comparison of the causes was performed between researchers' analysis to get the final causes, (iv) final causes were grouped into a category and, (iv) the number of times that each cause/category cited was counted.

3. Results

Most participants work as developer (35.4%), followed by project manager (21.8%), software architect (20.4%), and tester (8.2%). When asked about their level of experience in their role, most participants indicated they are competent (34%), followed by proficient (32%), expert (24.5%), beginners (8.2%), and novice (1.4%). Organizations of different sizes are represented in the dataset. They are well distributed among small (43.5%), mid (28.6%), and large size (27.9%) companies. This shows that we have representatives from companies of all sizes.

Related to the process model used, most projects were described as hybrid (49%), followed by agile (37.4%), and traditional process (13.6%). Project teams usually consisted of 5 to 9 people (33.3%), followed by teams with 10 to 20 people (25.9%), 30 or more people (18.4%), 5 or less people (11.6%), and, 21 to 30 people (10.9%), indicating that we have representatives from teams of all sizes. Thus, overall, the collected data seems to be a good representation of the diversity of the Colombian software industry. In the following, we will present the results for our research questions: section 3.1 presents results for research question 1, and section 3.2 presents results for research question 2.

3.1. Familiarity with technical debt concept

We can see in Figure 1 (left side) that a large portion of the participants (80.9%) reported that they are somewhat familiar with the concept of TD. Then, after being presented to a TD definition adapted from McConnell [13], most participants (87.7%) indicated that their understanding is close to or very close to the McConnell's TD definition (Figure 1 right side). These results suggest that TD is highly known in the Colombian software industry, but the term could still be expanded [14].

Comparing these results with the ones of the replicated study in Brazil [10], we found one difference and one similarity: (i) 40% of participants in Brazil reported that they had never heard of the TD concept (19% in Colombia) and (ii) 80% of participants indicated that their understanding is close to or very close to the McConnell's TD definition (87.7% in Colombia).

This similarity could be interpreted as a cultural issue. However, further analysis is needed to corroborate whether these results could be generalizable.

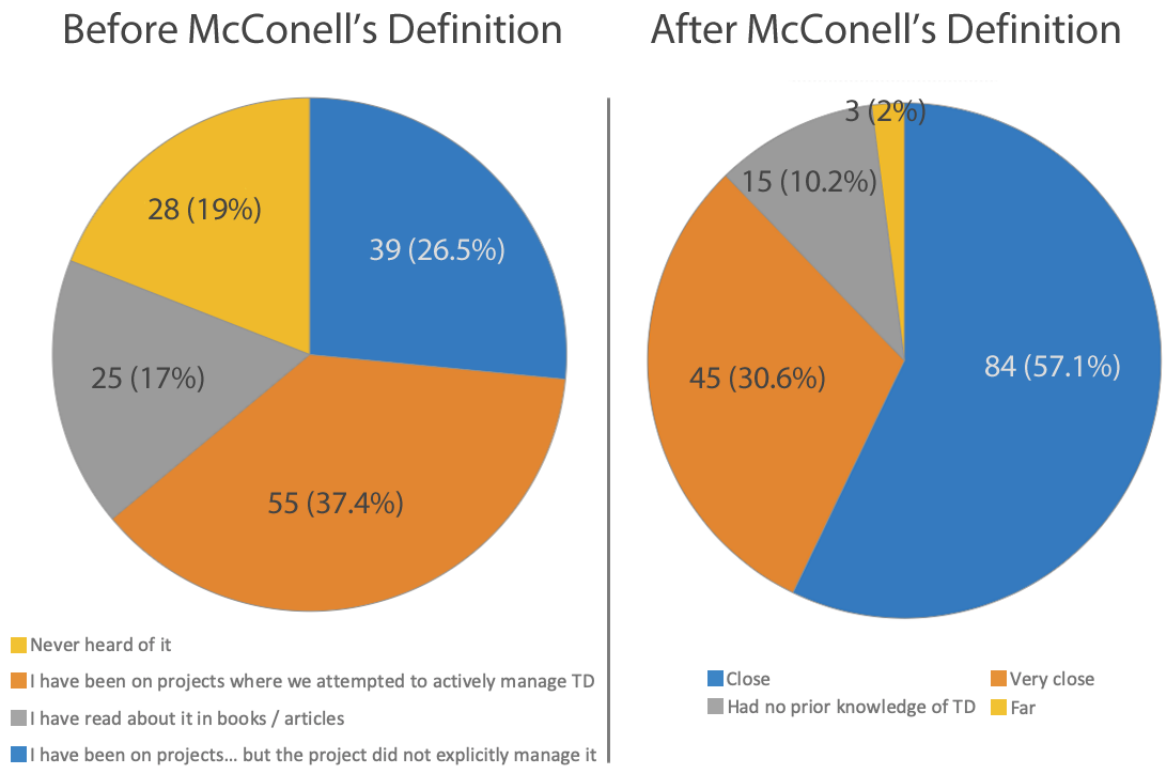

Figure 1. Participants' familiarity with technical debt concept.

3.2. Causes of technical debt from software architects and developers' points of view (RQ2)
Figure 2 presents the top 10 most cited TD causes, as informed by the 147 participants in Q16-18. From this chart we can observe that "focus on producing more at the expense of quality" is the most cited cause of TD. It in interesting to note that the first three causes are linked to "planning and management" category. Only one cause is linked to a technical issue: "change of requirements". This result indicates that non technical issues are an important cause of TD. The categories of causes are represented by colors. Five out of the ten causes presented here were listed in [10]. In general, both surveys conclude that "planning and management" category is an important source of TD.

As stated in the research questions, we are also interested in investigating what software architects and developers identify as causes of TD. Figure 3 presents the top 5 most cited TD causes from the different roles and classified them according to its category of cause. We can see that both roles share some causes, such as "innacurate time estimate", "inappropriate planning" and "inappropriate / pooly planned / poorly executed test". Also, it is relevant to note that the most cited cause by software architects is the less cited cause for developers: "inappropriate planning". "Lack of qualified professionals" is the most selected cause for developers. However, it is not clear what kind of professionals they refer to. There is a lack of qualified developers, or maybe a lack of qualified architects. This will require a deeper analysis.

Related to categories of causes, "planning and management" is the most cited category. However, this result is not aligned with previous finding from Ernst, *et al.* [6] that indicated architectural decisions as the most important source of TD. It is relevant to say that both roles could release their complaints against the manager of the software project. It could be interesting to review what project manager have to say about causes of TD.

Figure 2. Top 10 technical debt causes cited.

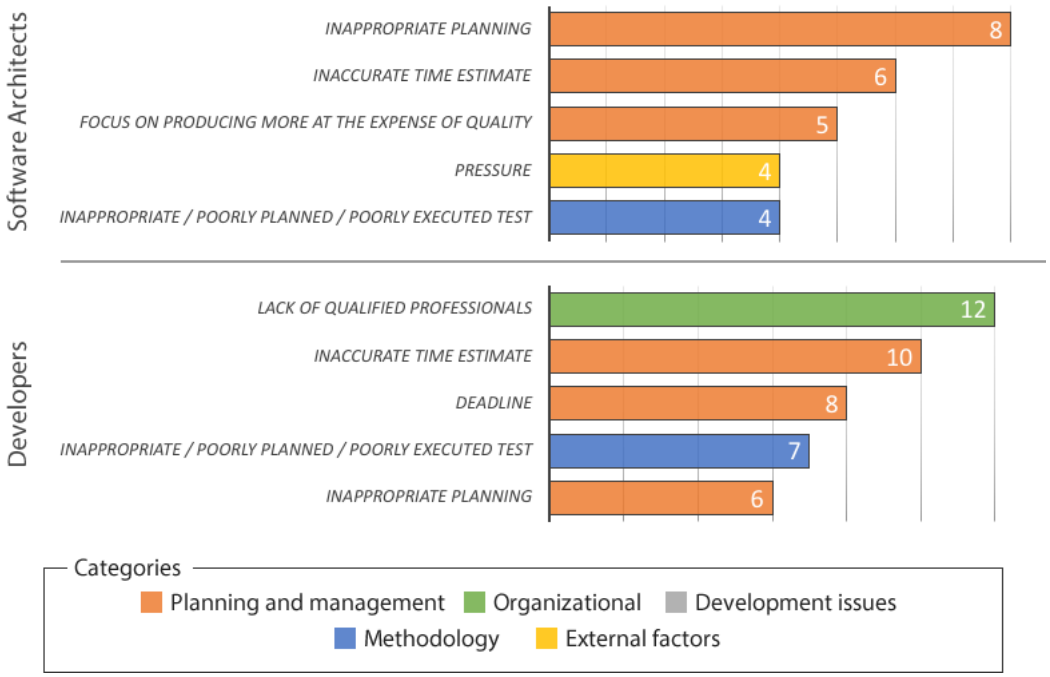

Figure 3. Top 5 technical debt causes cited by role.

4. Discussion

The combination of data from 147 participants in Colombian' software organizations provided an overall picture of the current state of practice.

The results related to the familiarity with TD concept, indicate that software development practitioners are familiar with the concept of TD (80.9%). Besides, after presenting the TD definition adapted from McConnell, it could be established that this definition is close to which practitioners are familiar with (87.7%). This is also supported in a different study by Martini, Besker and Bosch [15] who stated that 56.4% of the software engineers were aware of the TD

concept. For researchers, this is an indication that the TD concept is understood and that most efforts need to be put on offering strategies and tools to support TD management activities.

Regarding the causes of TD (Figure 2 and Figure 3), the results highlight the importance of qualified professionals in software industry. However, we could not identify what skills are required to be a qualified professional. Maybe, in the context of TD, it could be understood as lack of knowledge on the use of specific technologies or even lack of commitment with the internal quality of the product. Software professionals are not the only ones guilty of this. Companies related to software development have an increasing pressure to improve their effectiveness in each new deployment, by reducing time or the use of resources [1].

Thus, while current TD identification strategies are more commonly focused in the analysis of the source code [13], the precursor of TD occurrence is in part in the planning stage of the project. According to Rios, *et al.* [10], the proposal of TD management strategies has not necessarily led to the development of a support tool, leading to a non-integrated development of solutions often making their practical use unfeasible. This impacts both practitioners and researchers because it is a signal that new efforts need to be put on strategies and tools to support project managers in dealing with TD.

The obtained results will support practitioners to understand what is causing TD in their projects. Based on the list of causes, professionals can adopt different strategies to minimize the occurrence of TD. For researchers, results can support future investigation by providing insights into software practitioners' perspectives on TD causes and how they are currently dealing with the TD incurred in their projects.

5. Conclusions

This paper discusses the results of the replication of the InsighTD survey in Colombia. This work contributes to the generation of knowledge on TD through the accumulation of evidence about the findings, thereby increasing the level of confidence in results.

In total, 147 software practitioners from the software industry in Colombia answered the questionnaire. The collected data allowed us to understand how TD is perceived and what causes lead to TD occurrence by software teams. In general, software practitioners have a close understanding of TD. This understanding could help them in reduce the amount of technical debt items injected in their software systems. However, there is still 12% of the practitioners not having a general understanding of TD. It is important to understand why this concept is not of common knowledge in software companies. Further analysis is needed to corroborate whether these results could be generalizable.

Related to the causes leading to TD injection, the causes grouped in the "planning and management" category were the most cited by respondents as precursors to TD. This means that it is a generalizable concept among software practitioners that management is a big responsibility for the presence of TD in software projects. However, this goes against the fact that architectural decisions are the most common source of TD. Specially, software practicioners acknowledged that focus on producing more is the most common cause of TD. Software companies are under pressure to produce software as fast as possible, and this comes with a price.

Future works comprehend a deeper analysis of the results to include effects of TD in software projects and how it is monitored and paid off. In addition, understanding that cultural matter can influence the results, we are planning to compare and analyze the Chilean results with other results as a part of InsighTD. Currently, results from Brazil, Chile, and the United States are available.

References

[1] Martini A, Sikander E and Madlani N 2018 A semi-automated framework for the identification and estimation of architectural technical debt: A comparative case-study on the modularization of a software component *Information and Software Technology* **93** 264
[2] Besker T, Martini A and Bosch J 2018 Managing architectural technical debt: A unified model and systematic literature review *Journal of Systems and Software* **135** 1
[3] Allman E 2012 Managing technical debt *Queue* **10(3)** 10
[4] Freire S, Rios N, Gutierrez B, Torres D, Mendonça M, Izurieta C, Seaman C and Spínola R O 2020 Surveying software practitioners on technical debt payment practices and reasons for not paying off debt items *Proceedings of the Evaluation and Assessment in Software Engineering (EASE '20)* (New York: Association for Computing Machinery) pp 210–219
[5] Li Z, Avgeriou P and Liang P 2015 A systematic mapping study on technical debt and its management *Journal of Systems and Software* **101** 193
[6] Ernst N A, Bellomo S, Ozkaya I, Nord R L and Gorton I 2015 Measure it? manage it? ignore it? software practitioners and technical debt *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering (ESEC/FSE 2015)* ESEC/FSE (New York: Association for Computing Machinery) pp 50–60
[7] Yli-Huumo J, Maglyas A and Smolander K 2014 The sources and approaches to management of technical debt: A case study of two product lines in a middle-size finnish software company *Product-Focused Software Process Improvement (PROFES 2014)* (*Lecture Notes in Computer Science* vol 8892) ed Jedlitschka A, Kuvaja P, Kuhrmann M, Männistö T, Münch J and Raatikainen M (Switzerland: Springer) pp 93–107
[8] Yli-Huumo J, Maglyas A and Smolander K 2015 The benefits and consequences of workarounds in software development projects *Software Business (ICSOB 2015)* (*Lecture Notes in Business Information Processing* vol 210) ed Fernandes J, Machado R and Wnuk K (Switzerland: Springer) pp 1–16
[9] Pérez B, Brito J P, Astudillo H, Correal D, Rios N, Spínola R O, Mendonça M and Seaman C 2019 Familiarity, causes and reactions of software practitioners to the presence of technical debt: A replicated study in the chilean software industry *38th International Conference of the Chilean Computer Science Society (SCCC)* (Concepcion: IEEE) pp 1–7
[10] Rios N, Spínola R O, Mendonça M and Seaman C 2018 The most common causes and effects of technical debt: first results from a global family of industrial surveys *Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM '18)* (New York: Association for Computing Machinery) pp 1–10
[11] Pérez B, Correal D and Astudillo H 2019 A proposed model-driven approach to manage architectural technical debt life cycle *IEEE/ACM International Conference on Technical Debt (TechDebt)* (Montreal: IEEE) pp 73–77
[12] Bezerra R M M, da Silva F Q B, Santana A M, Magalhaes C V C and Santos R E S 2015 Replication of empirical studies in software engineering: An update of a systematic mapping study *ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)* (Beijing: IEEE) pp 1–4
[13] Alves N S, Mendes T S, de Mendonça M G, Spínola R O, Shull F and Seaman C 2016 Identification and management of technical debt: A systematic mapping study *Information and Software Technology* **70** 100
[14] Pérez B, Correal D and Vera-Rivera F H 2020 How do software architects perceive technical debt in colombian industry? an analysis of technical debt causes *Journal of Physics: Conference Series* **1513** 012003:1
[15] Martini A, Besker T and Bosch J 2016 The introduction of technical debt tracking in large companies *23rd Asia-Pacific Software Engineering Conference (APSEC)* (Hamilton: IEEE) pp 161–168