How Experience Impacts Practitioners' Perception of Causes and Effects of Technical Debt

Sávio Freire
Federal University of Bahia and Federal Institute of Ceará
Brazil
savio.freire@ifce.edu.br

Nicolli Rios
Federal University of Rio de Janeiro
Brazil
nicolli@cos.ufrj.br

Boris Pérez
University of Los Andes and Francisco de Paula S/der University
Colombia
br.perez41@uniandes.edu.co

Camilo Castellanos
University of Los Andes
Colombia
cc.castellanos87@uniandes.edu.co

Darío Correal
University of Los Andes
Colombia
dcorreal@uniandes.edu.co

Robert Ramač
University of Novi Sad
Serbia
ramac.robert@uns.ac.rs

Vladimir Mandić
University of Novi Sad
Serbia
vladman@uns.ac.rs

Nebojša Taušan
Chamber of Commerce and Industry
Serbia
nebojsa.tausan@infora.rs

Gustavo López
University of Costa Rica
Costa Rica
gustavo.lopez_h@ucr.ac.cr

Alexia Pacheco
University of Costa Rica
Costa Rica
alexia.pacheco@ucr.ac.cr

Davide Falessi
University of Rome "Tor Vergata"
Italy
d.falessi@gmail.com

Manoel Mendonça
Federal University of Bahia
Brazil
manoel.mendonca@ufba.br

Clemente Izurieta
Montana State University and Idaho National Laboratories
United States
clemente.izurieta@montana.edu

Carolyn Seaman
University of Maryland Baltimore County
United States
cseaman@umbc.edu

Rodrigo Spínola
Salvador University and State University of Bahia
Brazil
rodrigo.spinola@unifacs.br

***Abstract*— Context: The technical debt (TD) metaphor helps to conceptualize the pending issues and trade-offs made during software development. Knowing TD causes can support in defining preventive actions and having information about effects aids in the prioritization of TD payment. Goal: To investigate the impact of the experience level on how practitioners perceive the most likely causes that lead to TD and the effects of TD that have the highest impacts on software projects. Method: We approach this topic by surveying 227 practitioners. Results: While experienced software developers focus on human factors as TD causes and external quality attributes as TD effects, low experienced developers seem to concentrate on technical issues as causes and internal quality issues and increased project effort as effects. Missing any of these types of causes could lead a team to miss the identification of important TD, or miss opportunities to preempt TD. On the other hand, missing important effects could hamper effective planning or erode the effectiveness of decisions about prioritizing TD items. Conclusion: Having software development teams composed of practitioners with a homogeneous experience level can erode the team's ability to effectively manage TD.**

Keywords—technical debt, technical debt causes, technical debt effects, level of experience, InsighTD

I. INTRODUCTION

Software development is a social and knowledge-intensive task, and human-centered factors such as personality, communication, and interaction patterns have been found to affect software projects [1, 2, 3]. One important aspect is team diversity, which is categorized into three types [4]: informational (or knowledge), social, and value diversity. In this article, we focus on knowledge diversity, which refers to the variation in the knowledge base (e.g., experience level, educational degree) that members bring to the software team.

Development teams whose members have heterogeneous experience levels seem to have better performance because the diversity of experience facilitates information exchange and communication due to different viewpoints [5, 6]. We also know that the technical debt (TD) metaphor can also facilitate communication in software development teams because it helps to close the communication gap between technical and non-technical individuals and teams [7].

TD contextualizes the problem of taking design shortcuts in pending development tasks as a type of debt that brings a short-term benefit to the project, usually in terms of increased development speed or shortened time to market, but which may have to be paid with interest later on in the software development process [7]. Discussions about causes and effects of TD are particularly useful in software projects. Knowing the causes for TD can support development teams in defining TD prevention actions. Having information about potential TD effects aids in the prioritization of TD items to pay off, by supporting a more precise impact analysis and the identification of corrective actions to minimize possible negative consequences for the project.

In technical literature, several researchers have reported evidence on causes and effects of TD [8-23]. For example, Martini and Bosh [16] investigated the effects of architectural debt in eight large software development companies and Ernst et al. [20] run an industrial survey and interviews for identifying the amount of debt that causes of TD can bring for a project. However, little is known about the impact of the experience level on how practitioners perceive the most likely causes that lead to TD and the effects of TD that have the highest impacts on software projects. Investigating this topic can provide valuable information for decision-makers. It helps in understanding the benefits of having diverse teams in terms of experience. And, it helps in creating fruitful scenarios for discussing project issues under the perspective of TD. Thus, in this article we seek answers for two research questions (RQs):

- **RQ1:** Are the causes most likely to lead to TD perceived differently by low and high-experienced practitioners?
- **RQ2:** Are the effects of TD that have a bigger impact on software projects perceived differently by low and high-experienced practitioners?

We approach both RQs by using a subset of the data collected from the *InsighTD* Project (http://www.td-survey.com/), a global family of surveys on causes, effects, and management of TD [8]. In total, 227 practitioners from Brazil, Colombia, Chile, Costa Rica, Servia, and the United States responded the survey. These answers are analyzed using quantitative and qualitative procedures. Initially, we analyze demographics data. Then, we investigate the relationship between the level of experience of the participants and the TD causes and effects that they reported.

In total, 76 TD causes were reported. *Deadline* and *inappropriate planning* are the most commonly cited by both low and high-experienced practitioners. Although the difference between the practitioners' perception about the causes most likely to lead to TD is small, it can vary according to the level of experience. For example, the cause *non-adoption of good practices* is in the top 10 of the most cited causes only for the low-experienced practitioners, while *lack of qualified professional* is present only in the high-experienced practitioners' list. Overall, experienced practitioners focus on human factors as causes of TD and low experienced developers seem to concentrate on technical issues as causes.

Regarding the effects, 68 were reported. We identified bigger differences on how practitioners with different levels of experience perceive the most impactful effects of TD. High-experienced practitioners see external quality attributes as the most impacted by the presence of debt items, while low-experienced practitioners see internal quality issues and increased project effort as the most impactful effects. For example, the effect *low external quality* is ranked first in the list of the most impactful effects for high-experience practitioners, while it only appears in the 7th position for low-experienced ones. On the other hand, the effects *low maintainability*, *rework*, *bad code*, *need of refactoring*, and *increased effort* only appear or are better positioned in the ranked list of low-experienced practitioners.

This work has implications for practitioners and researchers. Practitioners can use the list of causes and effects of TD by level of experience as a starting point for understanding that team diversity is important to TD management. Different levels of experience can bring a more complete view on the causes and effects, supporting the decision making for curbing the presence of TD or minimizing its effects. For researchers, our findings can motivate the development of new TD management strategies considering the fact that practitioners with different levels of experience can perceive the presence of TD in their projects differently.

This paper is organized in eight other sections. Section II discusses the related work on causes and effects of TD and Section III presents the *InsighTD* project. Then, Section IV presents our research method. Section V presents the results. Next, Section VI discusses the results and presents their implications for researchers and practitioners. Section VII presents the threats to validity. Lastly, Section VIII discusses final remarks.

II. Related Work on Causes and Effects of TD

Several works have investigated the causes and effects of TD [14-23]. Codabux and Williams [14] investigated how TD affects the adoption of agile software development. From an industrial case study with 28 practitioners, the authors identified causes and effects of TD. Martini et al. [15] identified a set of causes of architectural debt incurred in seven large software organizations. In Martini and Bosh [16], the authors went further and investigated the effects of architectural debt. These authors [17] also identified the effects of TD felt in TD prioritization activity in four companies.

In another study in the area, Yli-Huumo et al. [18] performed 12 interviews with practitioners for understanding the relationship between TD causes, effects, and management. These authors [19] also investigated the causes and effects of workarounds in 17 organizations.

By conducting an industrial survey and follow-up interviews in three organizations, Ernst et al. [20] identified that immature choices done in architecture are the primary cause of TD. From results of a systematic mapping study, Li et al. [21] recognized that causes of TD are related to quality aspects. Avgeriou et al. [22] reported causes and effects of TD from discussions with researchers and practitioners in the Dagstuhl Seminar 16162. More recently, Besker et al. [23] conducted a systematic review on architectural debt, reporting its major negative effects.

None of the existing studies investigates the influence of the level of experience on how practitioners perceive the causes and effects of TD. It is precisely the topic we addressed in this work, which is performed in the context of the *InsighTD Project*.

III. The InsighTD Project

To investigate the state of practice and industrial trends in the TD area, the *InsighTD project* was designed to run a family of industrial surveys in different countries [8]. The project intends to recognize the causes that lead to TD items, their effects, and how software practitioners deal with these items in their projects. Currently, researchers from Argentina, Brazil, Chile, Colombia, Costa Rica, Finland, Indian, Italy, Norway, Saudi Arabia, Serbia, and the United States have joined the project. Several results from *InsighTD* have been disseminated in the technical literature, reporting on causes and effects of TD [8-13], TD prevention [24], TD payment [25,26], and the relationship between effects and TD payment [27].

Concerning the causes and effects, Rios et al. [8] and Rios et al. [9] presented the design of the project and discussed the list of causes and effects of TD collected from the Brazilian replication. A set of cross-company probabilistic cause and effect diagrams was proposed by Rios et al. [10] for organizing the causes and effects of TD. Replications of the survey were also reported by the Chile [11] and Serbia [12] replication teams. Besides presenting the list of causes and effects, these works also compared their findings to the ones reported in [9]. Finally, triangulation to the results from Rios et al. [9] was conducted to build-up evidence on documentation [13].

Although these analyses draw a comprehensive view on causes and effects of TD, none of the previous works investigated the impact of the level of experience in the practitioners' perception on the causes most likely to lead to

TD and the effects that have a bigger impact on software projects. In this work, we bridge this gap using data collected from six *InsighTD* replications.

For additional information on the overarching *InsighTD* Project, the reader can visit http://www.td-survey.com/publication-map/.

IV. Research Method

In this section, we present the data collection and analysis procedures we ran to answer the research questions posed in this work.

A. Data Collection

We use a subset of the available data from 10 questions, presented in Table I, of the *InsighTD* questionnaire. The characterization of the survey participants and their workspace are captured in Q1 thru Q8. In Q19 and Q21, the participants reported up to five causes which are the most likely to lead to TD and up to five effects with bigger impact, ordered by the likelihood of causing TD and impact level, respectively. The answers given for these questions (Q19 and Q21) along with the level of experience reported in Q7 are used for answering RQ1 and RQ2.

For collecting data, the survey was sent only to software practitioners from the Brazilian, Colombian, Chilean, Costa Rican, Serbian, and North American software industries. We used LinkedIn, industry-affiliated member groups, and industry partners as invitation channels.

TABLE I. Subset of the InsighTD survey's questions. Adapted from [8].

No.	Question (Q) Description	Type
Q1	What is the size of your company?	Closed
Q2	In which country are you currently working?	Closed
Q3	What is the size of the system being developed in that project? (LOC)	Closed
Q4	What is the total number of people of this project?	Closed
Q5	What is the age of this system up to now or to when your involvement ended?	Closed
Q6	To which project role are you assigned in this project?	Closed
Q7	How do you rate your experience in this role?	Closed
Q8	Which of the following most closely describes the development process model you follow on this project?	Closed
Q19	Considering all the cases of TD you've encountered in different projects, and the causes of those TD cases, which causes would you say are the most likely to lead to TD (ordered by likelihood of causing TD, with most likely listed first)? Please list up to 5 causes.	Open
Q21	Considering all the cases of TD you've encountered in different projects and the effects of that TD that you have personally experienced, which 5 effects would you classify as the effects that have a bigger impact (ordered by their level of impact, with bigger impact listed first)?	Open

B. Data Analysis

As the questionnaire is composed of closed and open-ended questions, we performed different data analysis procedures. For closed questions, we calculated the quantity of respondents choosing an option, supporting the characterization of the survey participants.

In answers given for Q7, the participants report their level of experience in the role, choosing one of the following options: (i) *novice* (minimal or "textbook" knowledge without connecting it to practice), (ii) *beginner* (working knowledge of key aspects of practice, (iii) *competent* (good working and background knowledge of area of practice), (iv) *proficient* (depth of understanding of discipline and area of practice), and (v) *expert* (authoritative knowledge of a discipline and deep tacit understanding across areas of practice). We grouped these options into three levels of experience: **low-level of experience** (novice + beginner), middle level of experience (competent + proficient), and **high-level of experience** (expert). We excluded participants that characterized themselves as having middle-level experience. This allowed us to investigate a greater contrast among participants related to experience.

For the open-ended questions, the answers were qualitatively coded. The coding process was based on Seaman [28] and Strauss and Corbin [29], and it was previously reported in [8]. The coding was performed by, at least, two researchers from each replication team. Additionally, the first codified list of causes and effects (Brazil: coded by the authors NR and RS) was sent to the other replication teams (Chile and Colombia: BP, CC, and DC; Costa Rica: AH and GL; Serbia: RR, VM, and NT; and the United States: NR, CI, DF, and CS), so they can use it to standardize the nomenclature of causes and effects found in their results. When each replication team finished coding, its list was sent to SF, NR, or RS review to seek consistency in the use of the adopted nomenclature. In the end, we put all the lists together to create the final list of TD causes and effects considered in our study.

In questions Q19 and Q21, the respondents cited causes ordered by the likelihood of leading to TD and effects ordered by their impact level. We assigned a weight to each position in the rank, ranging from 1 (least likely/impactful) to 5 (most likely/impactful) [30]. We also grouped the answers into two subsets based on the level of experience of each participant: low and high-level of experience. For each cause/effect of each subset, we calculated the cause exposure (C_E) and the effect exposure (E_E) values. The C_E is an indicator of how much a cause leads to the occurrence of TD, while the E_E indicates how much an effect impacts a project. Both C_E and E_E are based on the concept of risk exposure defined by Amland [31], which states that risk exposure is a product of the probability of fault occurrence and the cost if the fault occurs in the production. Hence, the cause (effect) exposure is the product of the likelihood of a cause (effect) to be in the top five rank of each participant and the impact level of that cause (effect). Thus, we have:

$$C_E = C_L * C_{IL} \quad (1)$$

where C_E is the cause exposure, C_L is the cause likelihood, and C_{IL} is the impact level of the cause. C_L is defined as

$$C_L = F_c / T_{FC} \quad (2)$$

where F_C is the frequency of occurrence of a *cause*, and T_{FC} is the frequency of occurrence of all *causes* in a same subset. C_{IL} is defined as

$$C_{IL} = S_{WC} / T_{SWC} \quad (3)$$

where S_{WC} is the sum of weights of a *cause* and T_{SWC} is the sum of weight of all *causes* in a same subset.

Similarly, for effects, we have:

$$E_E = E_L * E_{IL} \quad (4)$$

where E_E is the effect expose, E_L is the effect likelihood, and E_{IL} is the effect impact level. E_L is defined as

$$E_L = F_E / T_{FE} \quad (5)$$

where F_E is the frequency of occurrence of an *effect*, and T_{FE} is the frequency of occurrence of all *effects* in a same subset. E_{IL} is defined as

$$E_{IL} = S_{WE} / T_{SWE} \quad (6)$$

where S_{WE} is the sum of weights of an *effect* and T_{SWE} is the sum of weights of all *effects* in a same subset.

For example, in the dataset of highly experienced practitioners, the T_{FC} of all causes is 488, meaning that a total of 488 (non-unique) causes appear in the top five lists (Q19) of all participants in this subset. But the cause *lack of traceability of bugs* appears only twice (F_C) in these top five lists, so its C_L is 2/488 = 0.004. By the same token, this cause was cited in the second and fifth positions by the two participants, receiving weights 4 and 1, respectively. Thus, its S_{WC} is 5. The value of T_{SWC} is 1677, which is the sum of the weights of all causes in all the participants' top five lists, so the value of C_{IL} is 5/1677 = 0.0030. As a result, the C_E of the cause *lack of traceability of bugs* is $122x10^{-6}$. The higher the C_E value, the higher is the chance of a specific cause to lead to TD in comparison to the others. E_E is calculated in a similar manner, except that it uses the top five lists from the answers to Q21. A high E_E value indicates that the effect is very likely to have a big impact on a project in comparison to the other effects.

After all the calculations, we produced the ranked lists of causes (based on C_E) and effects (based on E_E) for low and high-experienced practitioners. To investigate whether there are differences between the two subsets, we adopted the similarity measure for rankings called RBO (rank-biased overlap) [32], which quantitatively measures how similar the ranked lists are. RBO gives a value ranging from 0 to 1. The closer this value is to 1, the greater the similarity between the lists. As RBO supports top-weighted ranked lists, the first elements of a list have more impact on the similarity index than the last ones. We can configure what elements will be compared by setting the *p*-value, which, differently than the *p* statistic, refers to a level of overlapping and the degree of top-weightedness. In our analysis, we chose p-value ranging from 0.5 (only the very initial elements of a rank are considered) to 0.9 (almost all elements are considered).

V. RESULTS

At the time of our analysis, data (653 responses) was available from six of the replications of the *InsighTD* project. The analyzed sample, totaling 227 survey responses, includes data from Brazil (34), Chile (34), Colombia (43), Costa Rica (37), Serbia (28), and the United States (51).

A. Demographics

In the sample, 60% of the survey participants identified themselves as having a high-level of experience, which means that *they have authoritative knowledge of a discipline and deep tacit understanding across areas of practice*. The remaining 40% characterized themselves as practitioners with a low-level of experience, i.e., *they have a minimal or "textbook" knowledge without connecting it to practice or working knowledge of key aspects of practice*. The majority of the participants identified themselves as developers (~47%), followed by project leaders or managers (~17%), software architects (~14%), or performing other roles (~22%).

Most of the participants work in medium-sized companies (~37%), followed by large (~32%) and small ones (~31%). Most of the time, they adopt hybrid process (~47%), followed by agile (~42%) and traditional (~11%) ones. Also, they work in teams composed of less than 10 people (~53%), 10 to 30 people (~28%), and more than 30 people (~19%).

The most common system size was less than 100 KLOC (~48%), followed by systems with 100KLOC to 1 million LOC (~28%), and more than 1 million LOC (~24%). The system age mentioned by the participants was typically less than 2 years old (~38%), but we also found systems with 2 to 5 years old (~36%), and more than 5 years old (~26%).

Overall, the collected data is a good representation of the software industry heterogeneity, reaching (i) several participants' roles, (ii) organizations of different sizes, and (iii) projects of different age, size, team size, and process models.

B. RQ1: Are the causes most likely to lead to TD perceived differently by low and high-experienced practitioners?

We identified 76 causes of TD. Fig. 1 shows the number of causes by level of experience using a Venn diagram. We can observe that 47 of those causes are cited by practitioners with high and low-level of experience as, for example, *deadline*, *inappropriate planning*, and *lack of experience*. Also, we found 24 causes cited only by practitioners with high-level of experience, for instance, *adoption of contour solutions as definitive*, *external component dependency*, and *be responsible for code from others*. On the other hand, five causes were reported only by practitioners with low-level of experience: *customer does not listen to project team*, *lack of change control*, *lack of information*, *lack of prioritization*, and *version incompatibility*. The complete list of causes per level of experience is available at http://bit.ly/3cUM3yP.

Fig. 1. Number of causes by level of experience.

Although the data represented in the Venn diagram indicates that the lists have specific causes (24 exclusive causes for high-experienced and 5 for low-experienced practitioners), we cannot conclude that both groups perceive the TD causes differently. The diagram does not represent the perception of the practitioners on how decisive each cause is for the occurrence of TD. This is investigated as follows:

1) RBO analysis

Fig. 2 shows the results of the comparison between the ranked lists of causes of each level of experience. The RBO analysis indicates that the lists are quite similar, with little variations when more causes are compared (*p-value* increases). The similarity level is about 80-90% between the two lists.

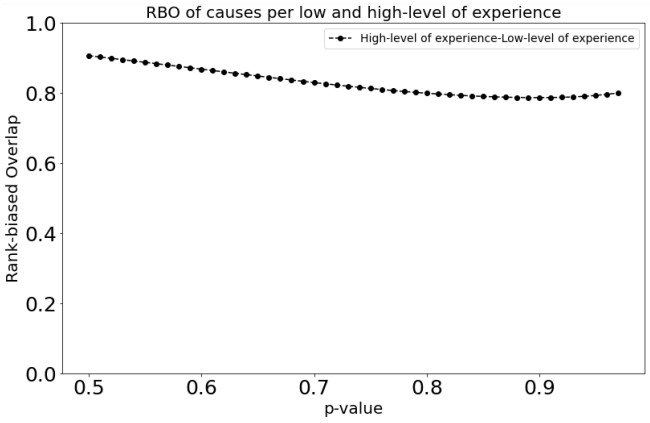

Fig. 2. RBO of TD causes rank by level of experience.

To further investigate this, we performed two complementary analyses considering the top 10 causes reported by low and high-experienced practitioners. In the first comparison (**rank-based comparison**), we consider the order in which each cause appears in each list to identify the similarities and differences existing between the low and high-experienced practitioners' point of view. In the second comparison (**exposure-based comparison**), we investigate the variation of C_E value (ΔC_E). For each cause, we calculate the modulus of the difference between its C_E value of each list. The variation indicates how different the point of view of practitioners with low and high experience is for a specific cause.

The results of the **rank-based comparison** are shown in Table II, indicating how the ranks for low and high-experienced practitioners compare to each other using the notation: ● same, ▲ higher, and ▼ lower positions in the rank, or ■ a cause that appears in only one group. The values in parentheses together with each cause represent the C_E (cause exposure) and the percentage of the number of citations of each cause by level of experience. In Appendix Table IV, we present the causes along with their meaning and examples of citation.

From the point of view of low-experienced practitioners, *deadline*, *inappropriate planning*, and *lack of experience* are the main causes that lead to TD occurrence and they impact 30%, 24%, and 20% of the software projects, respectively. On the other hand, the high-experienced practitioners pointed out that *deadline*, *inappropriate planning*, and *not effective project management* are the primary causes perceived by them, impacting 24%, 22%, and 20% of the software projects, respectively. Thus, regardless of their experience level, participants see *deadline* and *inappropriate planning* as the causes most likely to lead to TD. Only the causes *non-adoption of good practices*, *requirements elicitation issues*, *lack of qualified professionals* and *lack of commitment* are not shared between the two subsets.

Fig. 3 shows the results from the **exposure-based comparison**. We notice that the lines follow a uniform trend with few crossings (only for the causes *not effective project management*, *lack of qualified professional*, and *lack of commitment*) between them. This behavior is an indication that, although practitioners from both lists perceive the causes very similarly, there are some small differences. For example, the graph indicates that *deadline* and *lack of qualified professional* has the greatest ΔC_E. This means that low-experienced practitioners perceive *deadline* as a cause of TD more than high-experienced ones. On the other hand, *lack of qualified professional* is more likely to lead TD items in the opinion of high-experienced practitioners than for low-experienced ones.

In summary, we found greater similarities between the causes reported by low and high-experienced practitioners from the RBO analysis. However, there are slight differences that were analyzed through the rank- and the exposure-based comparison.

C. *RQ2: Are the effects of TD that have a bigger impact on software projects perceived differently by low and high-experienced practitioners?*

We identified 68 effects of TD. Fig. 4 shows the number of effects by level of experience. We can observe that 53 of them are felt by both practitioners with high and low-level of experience such as, for example, *delivery delay*, *low maintainability*, and *financial loss*. Also, we found nine effects felt only by practitioners with high-level of experience, for instance, *increase in risks*, *lack of training*, *legal issues due to non-compliance with contracts*. On the other hand, six effects were reported only by practitioners with low-level of experience: *architecture issues*, *constant need of retest*, *lack of commitment of users*, *lack of domain knowledge*, *loss of market competitiveness*, and *need for skilled professionals to solve problems*. The complete list of effects by level of experience is available at http://bit.ly/3cUM3yP.

TABLE II. TOP 10 TD CAUSES BY THE LEVEL OF EXPERIENCE.

	Low-level of experience	High-level of experience
Cause		
1st	● Deadline (0.00806; 30%)*	● Deadline (0.00528; 24%)
2nd	● Inappropriate planning (0.00491; 24%)	● Inappropriate planning (0.00415; 22%)
3rd	▲ Lack of experience (0.00311; 20%)	▲ Not effective project management (0.00273; 20%)
4th	▲ Inappropriate/poorly planned/poorly executed test (0.00243; 19%)	■ Lack of qualified professional (0.00241; 19%)
5th	▲ Lack of a well-defined process (0.00225; 18%)	▼ Lack of experience (0.00205; 19%)
6th	▲ Lack of technical knowledge (0.00205; 18%)	▼ Inappropriate/poorly planned/poorly executed test (0.00177; 19%)
7th	■ Non-adoption of good practices (0.00200; 19%)	▼ Lack of a well-defined process (0.00156; 15%)
8th	▼ Not effective project management (0.00185; 18%)	▼ Lack of technical knowledge (0.00141; 14%)
9th	▲ Pressure (0.00144; 13%)	■ Lack of commitment (0.00132; 14%)
10th	■ Requirements elicitation issues (0.00134; 14%)	▼ Pressure (0.00128; 11%)

Caption:
● Cause in the same position in both levels of experience
▲ Cause in a higher position in relation to ones of the other level of experience
▼ Cause in a lower position in relation to ones of the other level of experience
■ Cause appears in only one level of experience
* The values in parentheses represent CE and the percentage of the number of citations of each cause by experience level

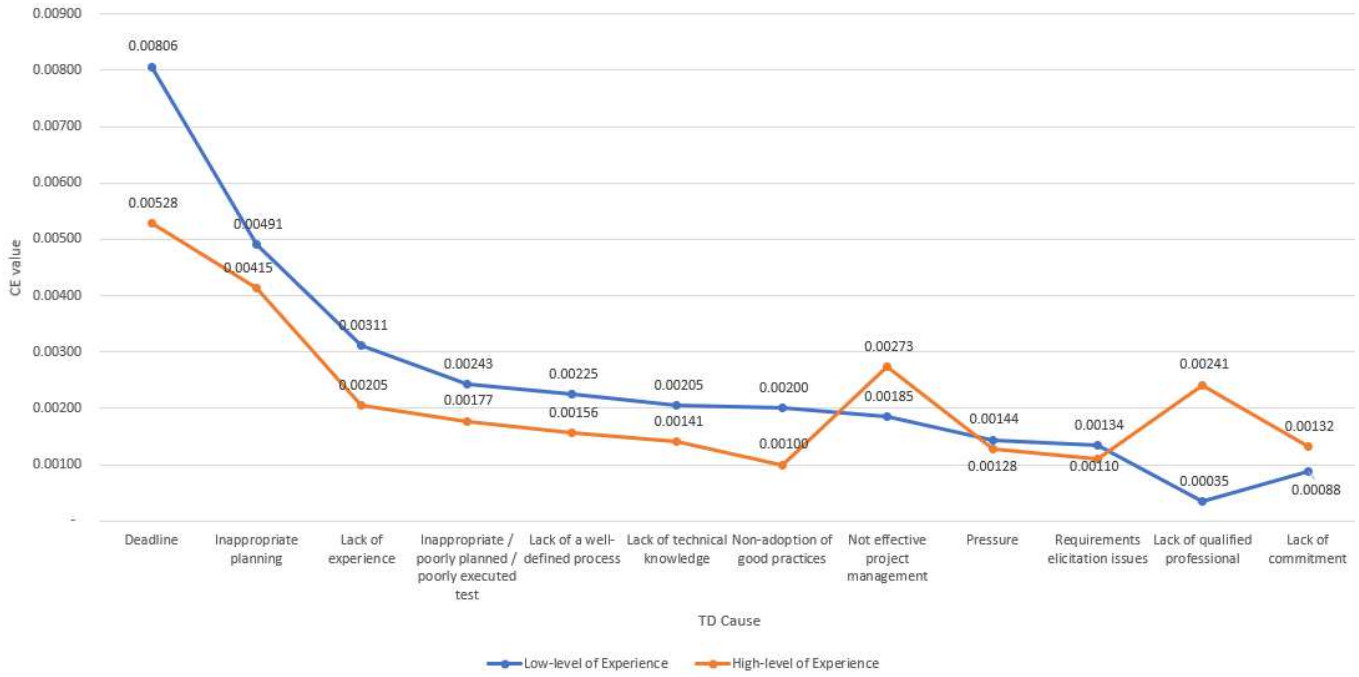

Fig. 3. Comparison between top 10 causes by level of experience.

Fig. 4. Number of effects by level of experience.

At first glance, the results presented in the Venn diagram suggest that the two subsets are quite similar, with minor differences between them. However, the two subgroups having very close lists of effects does not mean that they perceive the level of impact of each element of the lists in a similar way. This is further investigated as follows:

1) RBO analysis

Fig. 5 shows the results of the comparison between the ranked lists of effects identified for each experience level. The RBO analysis indicated a considerable difference of perception on how low and high-experienced practitioners see the most impactful effects of TD. The graph shows that the similarity level concentrated on the very initial effects of the ranks is only around 25%. This value increases as we consider more effects from both ranks. However, the difference of perceptions in the very beginning of the ranks is significant because these effects have the biggest exposure values, which means that they have a large impact as perceived by practitioners.

To analyze in detail the RBO result, we also performed two complementary analyses considering the top 10 effects reported by low and high-experienced practitioners. The **rank-based comparison** takes into consideration the order that each effect appears in each list to identify the similarities and differences. With the **exposure-based comparison**, we investigated the variation of E_E value (ΔE_E) for each effect, calculating its modulus of the difference between the E_E value in each list. This variation can reveal how different the point of view of practitioners with low and high experience is for the same effect.

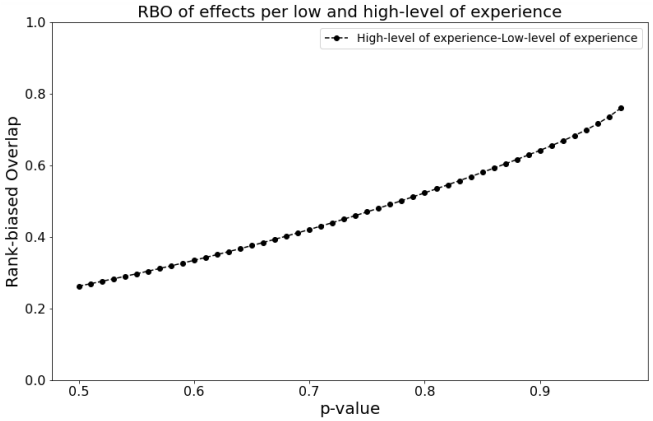

Fig. 5. RBO of TD effects rank by level of experience.

Table III presents the results of the **rank-based comparison**, showing the rank of the ten most impactful effects of TD. The values in parentheses together with each effect represent the E_E (effect exposure) and the percentage of the number of citations of each effect by level of experience. We use the following notation for indicating how the ranks for low and high-experienced practitioners compare: ● same, ▲ higher, and ▼ lower positions in the rank, or ■ an effect that appears in only one level of experience. From the point of view of the low-experienced practitioners, *delivery delay*, *low maintainability*, and *rework* are the most impactful effects of TD, affecting 37%, 23%, and 21% of the software projects, respectively. On the other hand, the high-experienced practitioners indicated that *low external quality*, *delivery delay*, and *financial loss* are the most impactful effects, impacting 33%, 27%, and 26% of the software projects, respectively. In Appendix Table V, we present the effects along with their meaning and examples of citation.

The first difference that catches our attention is the effect *low external quality*, which is seen as the most impactful and cited by 33% of the high-experienced (H) practitioners. Only 15% of the less-experienced (L) participants cited this effect,

TABLE III. TOP 10 TD EFFECTS BY LEVEL OF EXPERIENCE.

	Low-level of experience	High-level of experience
Effect		
1st	▲ Delivery delay (0.01409; 37%)*	▲ Low external quality (0.01058; 33%)
2nd	▲ Low maintainability (0.00526; 23%)	▼ Delivery delay (0.00743; 27%)
3rd	▲ Rework (0.00381; 21%)	▲ Financial loss (0.00694; 26%)
4th	▼ Financial loss (0.00360; 20%)	▼ Low maintainability (0.00662; 26%)
5th	● Stakeholder dissatisfaction (0.00320; 21%)	● Stakeholder dissatisfaction (0.00382; 20%)
6th	▲ Bad code (0.00218; 15%)	▲ Team demotivation (0.00178; 16%)
7th	▼ Low external quality (0.00187; 15%)	▼ Rework (0.00136; 11%)
8th	▼ Team demotivation (0.00128; 13%)	■ Design problems (0.00074; 8%)
9th	■ Need of refactoring (0.00114; 11%)	▼ Bad code (0.00058; 8%)
10th	■ Increased effort (0.00080; 11%)	■ Low performance (0.00041; 7%)

Caption:
● Effect in the same position in both levels of experience
▲ Effect in a higher position in relation to ones of the other level of experience
▼ Effect in a lower position in relation to ones of the other level of experience
■ Effect appears in only one level of experience
* The values in parentheses represent EE and the percentage of the number of citations of each effect by experience level

ranked in the 7th position in their list. We also highlight the results for the effects *rework* (L: 3th, 21% | H: 7th, 11%) and *bad code* (L: 6th, 15% | H: 8th, 8%), both better positioned and more commonly cited in the list of the less experienced practitioners. Lastly, we also recognized the presence of some well-known effects of TD in both lists: *delivery delay*, *low maintainability*, and *financial loss*.

Fig. 6 shows the results of the **exposure-based comparison**, indicating that *low external quality* and *delivery delay* has the greatest ΔE_E. For the effect *low external quality,* high-experienced practitioners consider it more impactful than low-experienced ones. On the other hand, *deadline* is more likely to provide a large impact in the opinion of low-experienced practitioners than for high-experienced ones. Overall, we observe many crossings between the lines in the graph, indicating different perceptions of the practitioners. While the effects *delivery delay*, *rework*, *bad code*, *need for refactoring*, and *increase effort* are more seeing as impactful by low-experienced practitioners, *low maintainability*, *financial loss*, *stakeholder dissatisfaction*, *low external quality*, *team demotivation*, *design problems*, and *low performance* have a bigger impact for practitioners with a high level of experienced.

The differences between the perception of low and high-experienced practitioners on TD effects detected by RBO analysis are confirmed by the rank- and exposure-based comparison.

VI. ON THE IMPACT OF HOMOGENEOUS TEAMS ON HOW PRACTITIONERS PERCEIVE THE CAUSES AND EFFECTS OF TD

In previous sections, our analysis showed that while there are small differences concerning causes, the diversity of experience level leads to big differences when talking about the TD effects.

To understand the implications of having homogeneous teams (composed of only highly or low experienced practitioners), on how practitioners perceive the TD causes and effects, we further analyzed the differences between the two subsets. The results of this comparison are observed in Tables II and III**.** Despite the similarities in the lists of top TD causes for both high and low experienced responses, Table II shows that software development teams composed of only low-experienced practitioners might not track the causes *lack of qualified professionals* and *lack of commitment.* Both are related to human issues, which in general could be hard to identify for developers with low experience, and thus could cause harm while remaining hidden unless those with experience know to look for them.

This could affect how effectively a low-experience team would be able to address TD prevention. Developer commitment and training, if deficient, could lead to considerable TD despite other efforts at TD prevention. On the other hand, high-experienced teams might not think about the importance of the *non-adoption of good practices* and *inappropriate tests* as sources of TD, both technical issues, according to Table II. Such technical practices might be "second nature" to highly experienced developers, and thus they would be easy to miss as potential problems unless others with less experience catch them. Further, overlooking these types of causes might lead the team to ignore some fairly straightforward TD prevention activities, like instituting more systematic testing practices.

Turning attention to the effects of TD, we can observe in Table III that high-experienced software teams are more concerned with external software quality attributes, as demonstrated by the presence of effects like *low performance* (not present in the top 10 list of low experienced practitioners) and *low external quality issues* (positioned first in the rank). To the contrary, low experienced teams seem to concentrate their concerns on internal quality issues and the necessity of investing more effort on the project, as demonstrated by the presence of the effects *low maintainability* (second place in their list) and, *bad code*, *need of refactoring*, and *increased effort,* all of which are either better positioned in the top 10 rank of practitioners with lower levels of experience, or do not appear at all in the list of effects ranked by high-experience respondents.

Extrapolating from these results implies that a homogeneously highly experienced team might focus too much on the customer view of the software at the expense of longer-term productivity goals, which can be eroded by low maintainability. Low-experienced teams, on the other hand, might have less of an understanding and concern for satisfying the customer in the short term. Effective TD management is, in fact, about balancing the long-term and short-term goals of a development project. Our results indicate that the diversity (or lack thereof) of the project team in terms of experience could seriously affect the team's ability to maintain that balance.

Our study does not allow us to indicate which of the ranks (low or high) point to the most important causes of TD to be prevented or effects to be monitored, but it shows that having

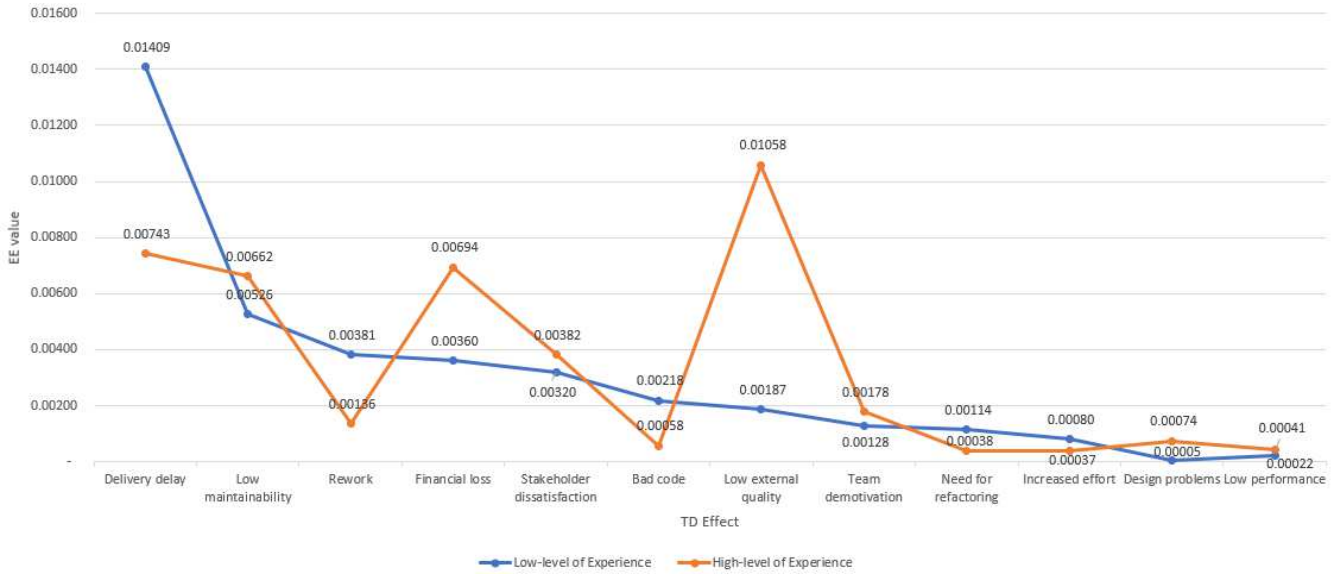

Fig. 6. Comparison between top 10 effects by level of experience.

software development teams composed of practitioners with homogeneous experience levels can erode the team's ability to effectively manage TD.

To make our results more feasible for using in practice, we organized them into a heat map (Fig. 7). The map presents, for each level of experience, the top 10 causes and effects classified by their C_E and E_E. We used colored rectangles ranging between tones of green, yellow, and red for representing C_E and E_E values. While green rectangles represent a cause or an effect with the lowest C_E or E_E values, the red ones indicate a cause or an effect with highest C_E or E_E values. Yellow rectangles show a cause or an effect that its C_E or E_E value is in the midpoint (50th percentile). Variations in these three colors demonstrate the range of C_E or E_E values between the midpoint, lowest, or highest C_E or E_E values. For example, *deadline* is the cause that mostly leads to TD in the point of view of low and high-level experienced practitioners. However, this cause is more likely to be noticed by low-level experienced practitioners (red rectangle) than by high-level experienced ones (orange rectangle, a combination of red and yellow).

When looking at the heat map, low-experienced practitioners can clearly see how their perception differs from those with higher level of experience (and vice-versa). Thus, for example, by observing the Fig. 7, a high-experienced practitioner could notice that maybe (s)he should also pay attention about issues related to *delivery delay* and *rework* as consequence of the presence of debt items. Similarly, a low-experienced practitioner could notice that *low external quality* and *team demotivation* should also be a central concern when managing the effects of TD.

VII. Threats to Validity

As in any empirical study, threats could affect the validity of our study. We identified them following the categorization defined by Wohlin et al. [33] and sought to eliminate or mitigate these threats.

A threat to conclusion validity arises from the coding process performed in the open-ended questions, as this process has subjectivity and possible inconsistencies. To mitigate this threat, at least, two researchers performed this process separately, and a consensus meeting or a third researcher resolved the disagreements. Besides, three other researchers consolidated the final list of codes seeking consistency in the code nomenclature among the replication teams of *InsighTD*.

Fig. 7. Heat map of top 10 causes and effects of TD by level of experience.

Regarding internal validity, the questions of the *InsighTD* questionnaire could be misinterpreted by the participants, representing a threat in our study. To reduce it, as described by Rios et al. [8], we ran three internal validations, one external validation, and a pilot study before the first execution of the questionnaire.

Lastly, concerning conditions that limit our ability to generalize the results (external validity), we reduce this threat by achieving a diversity of participants who answered the survey. Besides, the number of participants (#227) minimizes

the chances of subsets of participants biasing the results. However, we cannot say how generalizable the results are because we are not able to estimate the representativeness of our sample given the lack of empirical data characterizing the population. We intend to continuously reduce this threat by following the design of our family of surveys conducting replications in different countries and synthesizing the results to reach a more reliable and empirically founded result.

VIII. Final Remarks

This study investigates whether the perception of causes and effects of TD is impacted by the practitioners' level of experience. The results can support researchers (i) providing the state of the practice of causes and effects of TD and (ii) indicating what causes and effects are more perceived by practitioners with low and high level of experience. This information can be used for driving the development of new artifacts or strategies that consider the diversity of level of experience in the software development process.

The next steps of this study include: (i) increasing the external validity of our findings to include more data from other *InsighTD* replications and (ii) performing interviews with less and high-experienced practitioners to triangulate the obtained results to ones reported in this study.

Acknowledgment

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior Brasil (CAPES) Finance Code 001. This research was also supported in part by funds received from the David A. Wilson Award for Excellence in Teaching and Learning, which was created by the Laureate International Universities network to support research focused on teaching and learning. For more information on the award or Laureate, please visit www.laureate.net.

Appendix

Tables IV and V show the top 10 causes and effects along with their meaning and examples of citation.

TABLE IV. Top 10 TD causes and their definition and example of citation.

Cause	Definition	Example of citations
Deadline	A certain period of time defined by team, project manager and/or customer to deliver a determined activity, feature, or product.	• "The rush of managers (customers) that want to receive something working as soon as possible".
Lack of qualified professional	Occurs when unprepared professionals performing a certain activity or lack of professionals prepared to carry it out.	• "Absence of specialist to carry out specific activities"; • "Professionals unable to work".
Lack of experience	Refers to the lack of experience, obtained through the practice in certain software development activities.	• "Lack of experience of programmers"; • "Little experience of the people involved in the team".
Non-adoption of good practices	Refers to the non-use of good practices that would facilitate the accomplishment and maintenance of activities in the project.	• "Employment of bad design practices"; • "Lack of use of good software development practices".
Not effective project management	Refers to inadequate management during project development.	• "Lack of understanding of managers".
Inappropriate planning	Refers to problems in project planning.	• "Deficiency in project planning (disorganization)".
Lack of a well-defined process	Refers to the lack of a sustainable methodology aimed at creating and maintenance of guides that would increase the productivity and development of the software team.	• "Lack of methodology, simply the bosses come with the project and we start to do it without even understanding what to do".
Inappropriate / poorly planned / poorly executed test	Refers to project that is poorly tested, or even when the tests were poorly planned or do not have good coverage.	• "Lack of testing"; • "Failed tests".
Lack of technical knowledge	Refers to the unfamiliarity with any activity or artifact of the project.	• "Lack of knowledge of the team in tests".
Lack of commitment	Nonprofessional commitment of stakeholders to fulfill the tasks assigned to them.	• "Stakeholders not engaged"; • "Little commitment of the development team".
Requirements elicitation issues	Means problems regarding the requirements elicitation and their validation.	• "Poor-elicitated requirements".
Pressure	Occurs when there is high pressure on team members to meet deadlines and speed deliveries.	• "Pressure for delivery time"; • "Customer pressure to accelerate the project".

TABLE V. Top 10 TD effects and their definition and example of citation.

Effect	Definition	Example of citations
Low external quality	Refers to any aspect that reduces the quality of an artifact (including errors and known defects that are not fixed).	• "Low quality of what was offered known and uncorrected defect".
Delivery delay	Non-fulfillment of the deadlines agreed with the customer.	• "Six months delay in project delivery".
Low maintainability	Encompasses problems that occur during software maintenance activities, such as increased effort to fix bugs as well as limitation in system evolution.	• "Extremely difficult maintenance and evolution".
Financial loss	Occurs when a company has financial losses due to issues in the software development.	• "Decrease of profitability because of the extremely high costs to keep the product in the market".
Rework	Refers to redoing something that should have been done following quality standard.	• "The increasing rework that will be needed when TD is resolved".
Bad code	Use of bad practices in coding activities (e.g., bad variables/methods names, over complex code).	• "Bad workarounds in coding".
Low maintainability	Encompasses problems that occur during software maintenance activities, such as increased effort to fix bugs as well as limitation in system evolution.	• "Extremely difficult maintenance and evolution".
Need of refactoring	Refers to the need of improving the internal structure of the code without changing its external behavior.	• "Need to refactor"; • "Pending code refactoring".
Design problems	Issues in design during the software development process.	• "Poorly designed methods / classes".
Stakeholders dissatisfaction	Occurs when stakeholders are dissatisfied with the progress of the project.	• "Customer dissatisfaction with new functionality deadlines"; • "Customer dissatisfaction with product quality".
Stress with stakeholders	Refers to the presence of friction between team members due to various factors such as pressure, deadline, accumulation of activities, etc.	• "Attrition on customer relationship, development team". • "Project team stressed".
Team demotivation	Occurs when the team is discouraged from the daily routine due to various reasons	• "High degree of stress/demotivation of the team involved in the project".
Increased effort	Refers to the increase of effort to perform activities due to the TD presence in the project.	• "Greater effort and time for understanding, maintenance and evolution of the software".
Low performance	Refers to issues in reaching performance requirements of the software (due to the degraded internal quality of the software).	• "After application growth, performance has become unsustainable and refactoring was inevitable".

REFERENCES

[1] Jiang, J.J., Klein, G. and Pick, R.A., "The impact of IS department organizational environments upon project team performances," Information & Management, vol. 40, no. 3, 2003, pp.213-20.

[2] M. Pinzger, N. Nagappan, and B. Murphy, "Can developer-module networks predict failures?," in FSE. ACM, 2008, pp. 2–12.

[3] D. A. Tamburri, P. Kruchten, P. Lago, and H. van Vliet, "Social debt in software engineering: insights from industry," J. Internet Services and Applications, vol. 6, no. 1, pp.10:1–10:17, 2015.

[4] Jehn, K.A., "Why differences make a difference: a field study of diversity, conflict, and performance in workgroups," Administrative Science Quality, vol. 44, no. 4, 1999, pp.741-63.

[5] Liang, T., Liu, C., Lin, T. and Lin, B., "Effect of team diversity on software project performance," Industrial Management & Data Systems, vol. 107, no. 5, 2007, pp.636-653.

[6] Nonaka, I. and Takeuchi, H., The Knowledge-Creating Company, Rinehart and Winston, New York, NY, 1995.

[7] C. Izurieta, A. Vetrò, N. Zazworka, Y. Cai, C. Seaman and F. Shull, "Organizing the technical debt landscape," 2012 Third International Workshop on Managing Technical Debt (MTD), Zurich, 2012, pp.23-26.

[8] N. Rios, R. O. Spínola, M. Mendonça, and C. Seaman. (2020), "The practitioners' point of view on the concept of technical debt and its causes and consequences: a design for a global family of industrial surveys and its first results from Brazil," Empirical Software Engineering.

[9] N. Rios, R.O. Spínola, M. Mendonça, and C. Seaman, "The most common causes and effects of technical debt: first results from a global family of industrial surveys," 2018 12th International Symposium on Empirical Software Engineering and Measurement (ESEM), Oulu, 2018.

[10] N. Rios, R.O. Spínola, M. Mendonça, and C. Seaman, "Supporting analysis of technical debt causes and effects with cross-company probabilistic cause-effect diagrams," 2019 IEEE/ACM International Conference on Technical Debt (TechDebt), Montreal, 2019, pp. 3-12.

[11] B. Perez, H. Astudillo, D. Correal, J.P.B. Carvajal, N. Rios, M. Mendonça, R.O. Spínola, and C. Seaman, "Familiarity, causes and reactions of software practitioners to the presence of technical debt: a replicated study in the chilean software industry," 2019 38th Int. Conf. of the Chilean Comp. Science Society, Concepción, 2019, pp. 1-7.

[12] R. Ramac, V. Mandic, N. Tausan, N. Rios, M. Mendonça, C. Seaman, and R.O. Spínola, "Common causes and effects of technical debt in Serbian IT: InsighTD survey replication," 2020 Euromicro Conference Series on Software Engineering and Advanced Applications (SEAA), Portoroz, 2020, pp. 354-361.

[13] N. Rios, L. Mendes, C. Cerdeiral, A.P.F. Magalhães, B. Perez, D. Correal, H. Astudillo, C. Seaman, C. Izurieta, G.S. Souza, and R.O. Spínola, "Hearing the voice of software practitioners on causes, effects, and practices to deal with documentation debt", 2020 26th Int. Working Conf. on Req. Eng.: Foundation for Software Quality, Pisa, 2020.

[14] Z. Codabux and B. Williams, "Managing technical debt: an industrial case study," 2013 4th international workshop on managing technical debt (MTD), San Franscisco, 2013, pp. 8-15.

[15] A. Martini, J. Bosch and M. Chaudron, "Architecture technical debt: understanding causes and a qualitative model," 2014 40th Euromicro conference on software engineering and advanced applications (SEAA '14), Washington, 2014, pp 85–92.

[16] A. Martini and J. Bosch, "On the interest of architectural technical debt: uncovering the contagious debt phenomenon," Journal of Software: Evolution and Process, 29:e1877, July 2017.

[17] A. Martini and J. Bosch, "Towards prioritizing architecture technical debt: information needs of architects and product owners," 2015 41st Euromicro conference on software engineering and advanced applications (SEAA), Funchal, 2015, pp. 422–429.

[18] J. Yli-Huumo, A. Maglyas and K. Smolander, "The sources and approaches to management of technical debt: a case study of two product lines in a middle-size finnish software company," 2014 Product-Focused Soft. Process Improvement, 2014, pp 93–107.

[19] J. Yli-Huumo, A. Maglyas and K. Smolander, "The benefits and consequences of workarounds in software development projects," 2015 Int. Conference on Software Business (ICSOB), Braga, 2015, pp 1–16.

[20] N.A. Ernst, S. Bellomo, I. Ozkaya, R.L. Nord and I. Gorton, "Measure it? Manage it? Ignore it? software practitioners and technical debt," 2015 10th joint meeting on foundations of software engineering (ESEC/FSE 2015). ACM, New York, pp 50–60.

[21] Z. Li, P. Avgeriou and P. Liang, "A systematic mapping study on technical debt and its management," Journal of Systems and Software, vol. 101, pp. 193-220, March 2015.

[22] P. Avgeriou, P. Kruchten, I. Ozkaya and C. Seaman, "Managing Technical Debt in Software Engineering," 2016 Dagstuhl Seminar 16162, Dagstuhl Reports, vol 6:4.

[23] T. Besker, A. Martini and J. Bosch, "Managing architectural technical debt: a unified model and systematic literature review," Journal of Systems and Software, v.135, pp. 1–16, January 2018.

[24] S. Freire, N. Rios, M. Mendonça, D. Falessi, C. Seaman, C. Izurieta and R.O. Spínola, "Actions and impediments for technical debt prevention: results from a global family of industrial surveys," 2020 35th ACM Symp. on Applied Computing, Brno, 2020, pp. 1548–1555.

[25] S. Freire, N. Rios, B. Gutierrez, D. Torres, M. Mendonça, C. Izurieta, C. Seaman and R.O. Spínola, "Surveying software practitioners on technical debt payment practices and reasons for not paying off debt items," 2020 Evaluation and Assessment in Software Engineering (EASE '20), Trondheim, 2020, pp. 210–219.

[26] B. Pérez, C. Castellanos, D. Correal, N. Rios, S. Freire, R. Spínola and C. Seaman, "What are the practices used by software practitioners on technical debt payment: results from an international family of surveys," 2020 3rd International Conference on Technical Debt (TechDebt '20), Seoul, 2020, pp. 103–112.

[27] S. Freire, N. Rios, B. Pérez, D. Correal, M. Mendonça, C. Izurieta, C. Seaman and R.O. Spínola, "How do technical debt payment practices relate to the effects of the presence of debt items in software projects?," to appear in 2021 International Conference on Software Analysis, Evolution and Reengineering (SANER 2021), Honolulu, 2021.

[28] Seaman C, "Qualitative methods in empirical studies of software engineering, " IEEE Transactions on Software Engineering, vol. 25(4), pp. 557-572, July-August 1999.

[29] A. Strauss and J.M. Corbin, Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory. Sage Publications, 1998.

[30] H. Srikanth, C. Hettiarachchi, and H. Do, "Requirements based test prioritization using risk factors: An industrial study," Information and Software Technology, Vol. 69, 2016.

[31] S. Amland, "Risk-based testing: Risk analysis fundamentals and metrics for software testing including a financial application case study," Journal of Systems and Software, vol. 53, no. 3, 287–295, 2000.

[32] W. Webber, A. Moffat, and J. Zobel, "A Similarity Measure for Indefinite Rankings," ACM Trans. on Information Systems, Vol. 28, no.4, 2010.

[33] C. Wohlin, P. Runeson, M. Höst, M.C. Ohlsson, B. Regnell and A. Wesslén, Experimentation in Software Engineering: An Introduction. Springer, 2012.