Organizing the TD Management Landscape for Requirements and Requirements Documentation Debt

Larissa Barbosa[1], Sávio Freire[1,2], Nicolli Rios[3], Robert Ramač[4], Nebojša Taušan[4], Boris Pérez[5], Camilo Castellanos[6], Darío Correal[6], Alexia Pacheco[7], Gustavo López[7], Vladimir Mandić[4], Rita S. P Maciel[1], Manoel Mendonça[1], Davide Falessi[8], Clemente Izurieta[9], Carolyn Seaman[10], and Rodrigo Spínola[11, 12]

[1] Federal University of Bahia, Brazil
[2] Federal Institute of Ceará, Brazil
[3] Federal University of Rio de Janeiro, Brazil
[4] University of Novi Sad, Serbia
[5] Francisco de Paula Santander University, Colombia
[6] University of Los Andes, Colombia
[7] University of Costa Rica, Costa Rica
[8] University of Rome Tor Vergata, Italy
[9] Montana State University and Idaho National Laboratories, United States
[10] University of Maryland Baltimore Country, United States
[11] Salvador University, Brazil
[12] Virginia Commonwealth University, United States
larissa.leoncio@ufba.br, savio.freire@ifce.edu.br, nicolli@cos.ufrj.br, ramac.robert@uns.ac.rs, nebojsa.tausan@ef.uns.ac.rs, {br.perez41, cc.castellanos87, dcorreal}@uniandes.edu.co, {alexia.pacheco, gustavo.lopez_h}@ucr.ac.cr, vladman@uns.ac.rs, {rita.suzana, manoel.mendonca}@ufba.br, d.falessi@gmail.com, clemente.izurieta@montana.edu, cseaman@umbc.edu, spinolaro@vcu.edu

Abstract. [*Context*:] Requirements and requirements documentation debt (R2DD) indicate shortcuts taken in software development projects, resulting in requirements partially implemented and with outdated documentation, respectively. Knowing the causes and effects of R2DD can support software teams in defining actions to prevent the occurrence of these items and aid in the prioritization for eliminating them, respectively. Besides, having information on how practitioners deal with R2DD items can support developing new strategies and artifacts for managing these items. However, little is known on the state of the practice of R2DD. [*Aims*:] To investigate the state of the practice of R2DD, revealing its causes, effects, and practices and practice avoidance reasons (PARs) considered for its prevention and repayment. [*Method*:] We analyzed quantitatively and qualitatively a corpus of responses from a survey with software practitioners on R2DD and its elements (causes, effects, prevention, and repayment). [*Results*:] We identified 55 causes, 33 effects, 26 prevention practices, three PARs related to nonprevention, 18 repayment practices, and 16 PARs associated with nonrepayment of R2DD items. [*Conclusion*:] We organized those practices into a conceptual map. Software practitioners can use the map to start or improve their initiatives for dealing with R2DD items.

Keywords: Requirements debt, Requirements documentation debt, Causes, Effects, Technical debt prevention, Technical debt repayment.

1 Introduction

Technical debt (TD) contextualizes the problem of pending software development tasks as a type of debt that brings a short-term benefit to the project, often in terms of increased development speed or shortened time to market. As TD may have to be paid with interest later in the development process [1], successful TD management is about reaching a balance between the benefits of incurring it and later impacts of its presence [2, 3]. Beyond making decisions related to whether a debt item should be repaid, TD management includes preventive actions, as preventing debt items in earlier stages of software development can reduce the chances of those items impacting development activities later on [4, 5].

TD items can affect different artifacts and phases of the software development. It is particularly important to discuss the management of TD in the context of requirements engineering (RE) activities because they are inherently complex, reflect a system purpose aligning different viewpoints of the system's stakeholder, and impact several software development phases (e.g., coding, test planning) [6]. There are two types of debt directly related to RE: requirements and documentation debt. Requirements debt refers to the distance between the optimal requirements specification and the actual system implementation (e.g., requirements that are only partially implemented), while documentation debt is associated with problems found in software project documentation (e.g. missing, inadequate or outdated documentation) [7]. A type of documentation debt is requirements documentation debt, which affects requirements specifications, causing a mismatch between the stakeholder's needs and the software implementation.

Previous works have discussed a set of practices for preventing and repaying debt [4, 5, 8-10]. Research on TD management related to understanding the causes that lead development teams to incur debt items in their projects and their effects have also been conducted [11-20]. For example, the study performed by Bano et al. [20] identified the causes of requirements debt by analyzing requirements elicitation interviews conducted between student analysts and a business owner. Martini and Bosh conducted a multiple-case case study comprehending nine sites at six soft-ware companies to investigate the effects of architectural TD [15].

Although several works have investigated the state-of-the-practice on TD concerning its causes, effects, and management [11,21-22], the current literature has not approached the topic under the perspective of requirements and requirements documentation debt (R2DD). Addressing this perspective can drive software teams to change their mindset to increase the quality of the requirements and their documentation. Also, knowing R2DD TD causes can support development teams in defining TD prevention actions. Having information on R2DD TD effects can aid 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.

This work investigates the state of the practice of R2DD, revealing its causes, effects, and practices used for its prevention and repayment. To this end, we use a subset of the

data collected through the *InsighTD* Project (http://www.td-survey.com), a globally distributed family of industrial surveys on TD. In total, the survey has 78 answers on R2DD collected from practitioners from Brazil, Chile, Colombia, Costa Rica, the United States, and Serbia.

We found 55 causes of R2DD. Among them, *deadline*, *not effective project management*, and *change of requirements* are commonly mentioned by the participants. Regarding the effects, from 33 identified, the participants usually recognized *delivery delay*, *rework*, and *financial loss*. About TD prevention, we identified 26 prevention practices and the most cited were *well-defined requirements*, *following the project planning*, and *following well-defined project process*. The practices avoidance reasons (PARs) which justify the non-prevention of R2DD are *lack of qualified professionals*, *non-update documentation*, and *short deadline*. Lastly, we identified 18 repayment practices for eliminating R2DD items. Among them, *code refactoring*, *monitoring and controlling project activities*, and *design refactoring* were commonly considered. *Focusing on short term goals*, *lack of organizational interest*, and *lack of resources* were the leading PARs used to justify the R2DD non-repayment. We organize the set of R2DD elements into a conceptual map. Software practitioners can use the map to start or improve their initiatives for dealing with R2DD items.

Besides this introduction, this paper is organized in five other sections. Section 2 presents the research method. Then, Section 3 presents the results of the survey concerning R2DD management elements. Section 4 discusses the main findings. Section 5 presents the threats to the study validity. Lastly, Section 6 presents our final remarks and the next steps of this work

2 Method

This section presents the *InsighTD* Project, the research questions posed in this work, and the data collection and analysis procedures

2.1 The *InsighTD* Project

InsighTD Project is a globally distributed family of industrial surveys on TD causes, effects, and management. Its goal is to investigate the causes that lead to TD occurrence, the effects of its existence, and how software development teams react when they are aware of the presence of debt items in their project.

So far, several results from the project have been disseminated as shown at http://www.td-survey.com/publication-map/. We have reported the general list of TD management elements, such as, causes [11], effects [11], practices and PARs in prevention [5] and repayment [23] context. Also, we have investigated the TD management elements related to documentation debt [24], and the relationship between TD management and process models [12]. Although these results reveal evidence on how software practitioners face these TD management elements in their projects, there is still a lack of information on the management of R2DD. In this work, we fill this gap by investigating the causes that lead to R2DD items, their effects, and practices and PARs considered to prevent and repay these items.

2.2 Research Questions

We intend to answer the following research questions (RQ): **RQ1**: What are the primary causes that lead software practitioners to incur R2DD items?, **RQ2**: What are the leading effects felt by software practitioners due to the presence of R2DD items?, **RQ3**: What are the primary practices used by software practitioners for preventing R2DD?, **RQ4**: What are the leading reasons considered by software practitioners for explaining the non-prevention of R2DD items?, **RQ5**: What are the primary practices used by software practitioners for repaying R2DD items?, and **RQ6**: What are the leading reasons considered by software practitioners for explaining the non-repayment of R2DD items?.

2.3 Data Collection

We use data collected by six replications of the *InsighTD* questionnaire. Its survey is composed of 28 questions, as described in [11]. However, in this work, we only use the subset shown in Table 1 because it is related to the TD management elements we are investigating. The table reports the questions along with their type and the RQ they are associated with.

Table 1. Subset of the *InsighTD* survey's questions on TD management elements (adapted from [11]).

RQ	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
-	**Q10**	In your words, how would you define TD?	Open
-	**Q13**	Please give an example of TD that had a significant impact on the project that you have chosen to tell us about:	Open
RQ1	**Q16**	What was the immediate, or precipitating, cause of the example of TD you just described?	Open
RQ1	**Q17**	What other cause or factor contributed to the immediate cause you described above?	Open
RQ1	**Q18**	What other causes contributed either directly or indirectly to the occurrence of the TD example?	Open
RQ2	**Q20**	Considering the TD item you described in question 13, what were the impacts felt in the project?	Open
RQ3-4	**Q22**	Do you think it would be possible to prevent the type of debt you described in question 13?	Closed
RQ3-4	**Q23**	If yes, how? If not, why?	Open
RQ5-6	**Q26**	Has the debt item been paid off (eliminated) from the project?	Closed
RQ5-6	**Q27**	If yes, how? If not, why?	Open

In questions Q1 to Q8, participants characterize themselves and their projects. In question Q10, participants provide their definition of TD, and, in Q13, participants specify an example of a TD item that occurred in their projects. Considering this example, participants specify the causes of TD in questions Q16 thru Q18 and the effects of TD in Q20. Also, participants indicate if the TD item was prevented (Q22) and repaid (Q26) and justify their responses in Q23 and Q27, respectively. We use the answers given to those questions for answering our research questions (RQ1: Q16 to Q18; RQ2: Q20; RQ3-4: Q22 and Q23; RQ5-6: Q26 and Q27).

For inviting practitioners from the Brazilian, Chilean, Colombian, Costa Rican, North American, and Serbian software industries, we used LinkedIn, industry-affiliated member groups, and industry partners.

All collected answers were validated following the acceptance criteria:

- *The participants should consider a TD perspective to answer the questions*. We capture the TD definition provided by the participant in Q10. If this definition is aligned with the definition used in the *InsighTD* project [11], then, we moved to the following acceptance criterion.
- *The participants should provide a valid example of a R2DD item*. In Q13, the participants provided a TD item example. First, if the example is aligned with the definition of TD used in the *InsighTD* project [11], we conclude that the participant answers the other questions considering a TD perspective. Next, we only considered answers that the TD item example was related to R2DD, as reported in the answer: "lack of updating of requirements documentation".
- *The participants should provide valid answers in questions on TD causes, effects, prevention, and repayment*. We analyzed the answers given to Q16-17, Q20, Q23, and Q27 to identify causes, effects, preventive practices, PARs to non-prevention, repayment practices, and PARs to non-repayment. As we did not find invalid answers, we concluded that the participants did not misunderstand these questions.

2.4 Data Analysis Procedures

As the survey is composed of closed and open-ended questions, we run different data analysis procedures. For closed questions, we calculated the number of participants choosing an option. Afterwards, we summarize the participants' characterization.

For open-ended questions, we applied manual open coding process [25]. In answers given to Q13, we identified the type of TD associated with the example provided by the participant. For this, we used the list of indicators reported in [7] and the types of TD found in [26]. As a result, we identified the subset of answers that the participants refer to R2DD. Considering this subset, we identified the TD management elements. Following the process described in [11], we found a set of causes and effects and their respective number of occurrences from answers given to Q16-18 and Q20. In answers given to Q23, we applied the same process previously used by [5]. From this process, we identified practices for TD prevention when Q22 received a positive response, otherwise, we recognized PARs related to TD non-prevention. We followed the process

described in [23] to code the answers given to Q27. Similarly, when Q26 received a positive answer, we identified TD repayment practices, otherwise, we identified PARs associated with TD non-repayment.

An example of this process is as follows: a participant described the following example of TD item: "development of new software features without proper architectural design documents update, user manuals, or test cases". Analyzing the example, we notice that it reflects problems in the requirement documentation, i.e., it is related to R2DD. About the TD management elements, three participants cited the following TD causes: "constant relocation of analysts by the IT manager", "changes in the people who were initially working on the project", and "many developers worked on the code and were later replaced by others". First, we extracted the codes *relocation of analysts*, *turnover of the team*, and *replace of developers*, respectively. Analyzing these codes, we realized that they had different nomenclatures, but shared the same meaning; then, we standardized them as *high turnover of the team*.

Lastly, we realized that many of the TD management elements (causes, effects, practices, and PARs) were related to each other. Thus, we grouped them into categories following the axial coding [25]. To name the categories, we use the list previously defined by [21]. For example, we used the category *development issues* to group the effects *requirements changes* and *design changes*.

At least two researchers from each replication team performed the coding and axial procedures. The initial list of causes, effects, practices for TD prevention, PARs for TD non-prevention, practices for TD repayment, PARs for TD non-repayment and their respective categories were codified by the Brazilian replication team. These lists and categories were, then, sent to the other replication teams to drive the coding and axial processes and standardize the nomenclature. To guarantee consistency in this process, the Brazilian replication team verifies the final lists and categories.

3 Results

In total, the survey received 653 answers from Brazil, Chile, Colombia, Costa Rica, United States, and Serbia. However, 78 of them are in the context of R2DD. At the moment of the writing of this paper, we finished the analyses on causes, effects, and repayment of R2DD items considering all answers, and a subset of them (32 answers from Brazil and the United States) for R2DD prevention.

3.1 Demographics

Fig. 1 summarizes the participants' characterization, indicating their country, company and team size, system age and size, role and experience level performing the role, and the process model adopted in their company. We can notice that our data set is composed of a large variety of software development contexts, depicting in 78 answers collected by six *InsighTD*'s replication teams.

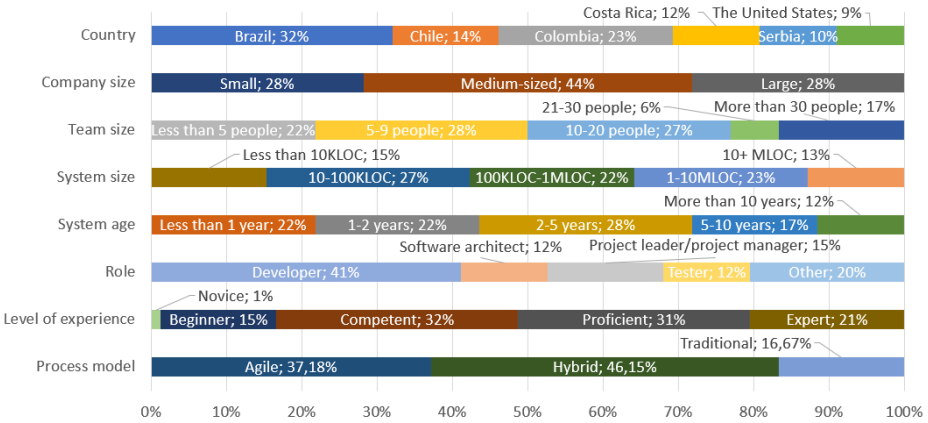

Fig. 1. Participant's characterization.

3.2 RQ1: Causes of R2DD

We identified 55 causes that lead software teams to incur R2DD. Fig. 2 presents the top 5 most commonly cited causes along with their number of citations and a percentage revealing how frequently each cause was considered in the represented software projects. The causes are organized in a map along with other R2DD elements (effects, practices, and PARs) that will be discussed in the following sections.

Analyzing Fig. 2, we notice that *deadline* is the most cited cause and was considered in 25% of the projects. The causes *not effective project management*, *change of requirements*, *inappropriate planning*, and *high turnover of the team* are also common. We also observed that the causes are related to each other in categories that reflect their main concern in a software project:

- **Development issues**: group causes that occur during the development phase in the project, for example, *change of requirements*, *inaccurate or complex requirement*, and *requirements elicitation issues*;
- **External factors**: encompass causes that are external to the development team and organization, such as *pressure* and *customer does not know his need*;
- **Lack of knowledge**: has causes related to the team's lack of knowledge to develop the project, such as, *lack of knowledge* and *lack of experience*;
- **Methodology**: encompasses causes related to processes and methodologies, for instance, *lack of well-defined process*, *inappropriate/ poorly planned/ poor executed test*, and *lack of requirements analysis*;
- **Organizational**: includes causes at the organizational level. *High turnover of the team* and *lack of qualified professional* are causes composing this category;
- **People**: groups causes directly related to members of a software team. Examples are *lack of commitment* and *lack of team communication;*
- **Planning and management**: includes causes related to the project planning and management as, for example *deadline* and *not effective project management*.

3.3 RQ2: Effects of R2DD

We identified 33 effects of R2DD. Fig. 2 presents the five most commonly cited effects. *Delivery delay* is the most cited effect, being felt in 41% of the projects. The effects *rework*, *financial loss*, *low external quality*, and *low maintainability* are also commonly experienced in projects. As expected, the presence of R2DD brings several consequences to software projects across different areas:

- **Development issues**: groups effects related to the project development activities, for example, *design changes*, *inadequate documentation*, and *constant need for retest*;
- **External quality**: has effects related to the quality of artifacts. Examples are *low external quality* and *project not completed*;
- **Internal quality issues**: encompasses effects associated with internal quality issues, such as, *low maintainability*, *need for refactoring*, and *bad code*;
- **Organizational**: includes effects at the organizational level like *financial loss* and *impaired company image*;
- **People**: groups effects related to the development team. Examples are *stress with stakeholders*, *team demotivation*, and *stakeholder dissatisfaction*;
- **Planning and management**: groups effects related to the project planning and management, such as, *delivery delay*, *rework*, and *increased effort*.

3.4 RQ3: Practices for preventing R2DD

We found 26 practices for preventing R2DD items. Fig. 2 presents the five most commonly cited practices. The practice *well-defined requirements* is the most cited, being used in 37% of the projects. The practices *following the project planning*, *following well-defined process*, *well-defined scope statement*, and *good allocation of resources in the team* complete the top 5. We organized the whole set of prevention practices into the following categories:

- **Development issues**: *adoption of good practices*, *project design*, and *understanding the technology in use*;
- **Methodology**: *focusing on agile delivery*, *requirements changes tracking*, and *well-defined documentation*;
- **Organizational**: *organizational support* and *training*;
- **People**: *discipline*, *focus*, and *readiness of team*;
- **Planning and management**: *appropriate task allocation*, *following the project planning*, and *well-planned deadlines*.

.

3.5 RQ4: PARs related to R2DD prevention

We found only three PARs (practice avoidance reasons): *lack of qualified professionals* (from category *organizational*), *non-update documentation* (*methodology*), and *short deadline* (*planning and management*). Each one was considered in 50% of the projects.

3.6 RQ5: Practices for repaying R2DD

We identified 18 practices for repaying R2DD. Fig. 2 presents the ten most commonly cited ones. *Code refactoring* is the most cited, being used in 20% of the projects. *Monitoring and controlling project activities*, *design refactoring*, *investing effort on TD repayment activities*, and *changing project scope* were also commonly cited. We grouped the repayment practices into the following categories:

- **Development issues**: *changing project scope*, *update system documentation*, and *solving technical issues*;
- **External quality issues**: *bug fixing*;
- **Internal quality issues**: *design refactoring*;
- **Methodology**: *investing effort on TD repayment activities*, *implementing preventive actions*, and *improving requirement elicitation process*;
- **People**: *communicating the customer of TD items*;
- **Planning and management**: *monitoring and controlling project activities*, *increasing the project budget*, and *negotiating deadline extension*;
- **Organizational**: *changing the project management*.

3.7 RQ6: PARs related to R2DD repayment

In total, we identified 16 PARs for R2DD non-repayment. Fig. 2 presents the five most commonly cited ones. *Focusing on short term goals* is the most cited PAR, being considered in 30% of the projects. The PARs *lack of organizational interest*, *lack of resources*, *cost*, and *team overload* were also commonly cited. For repayment PARs, we have the following categories:

- **Development issues**: *complexity of the project*;
- **External factors**: *customer decision*, *TD items do not affect the user*, and *the project was discontinued*;
- **Internal quality issues**: *number of TD items;*
- **Methodology**: *lack of adoption of lessons learned* and *lack of monitoring of TD items*;
- **Organizational**: *lack of organizational interest* and *lack of resources*;
- **People**: *team overload* and *insufficient management view about TD repayment*;
- **Planning and management issues**: *focusing on short term goals*, *cost*, and *effort.*

4 Discussion

For making our results more feasible to be used by software practitioners, we organized them into a conceptual map. Fig. 2 presents a summarized version of the map, showing the top 5 of each R2DD element (causes, effects, practices, and PARs) and all the identified categories. The complete version of the map is available at https://bit.ly/39kCN9x. In the map, the different R2DD management elements are presented in solid-line rectangles. In each of them, the elements are listed in dashed-line

rectangles representing their categories. Each category and its elements are associated with a percentage. To compute it, we summed up the number of occurrences for each of them and divided the result by the number of projects in which that R2DD element was cited. For example, the effect *delivery delay* was cited by 29 participants. As we had 70 participants indicating effects of R2DD, *delivery delay* was felt by 41% of them.

Causes from the category *planning and management* are very common, being experienced in 87% of the projects. Inside that category, the cause *deadline* stands out with

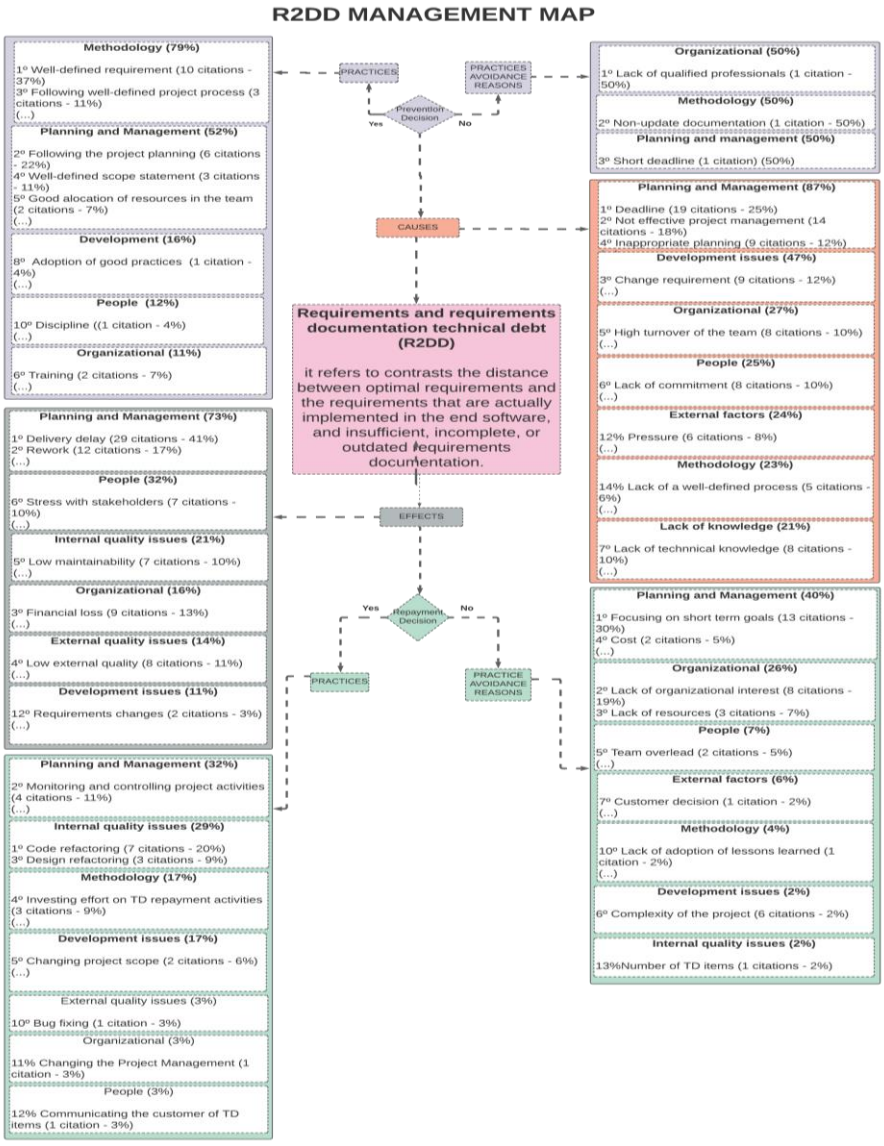

Fig. 2. A summarized version of the R2DD map.

presence in 25% of the projects. The category *planning and management* also stands out in respect to effects of R2DD, being considered in 73% of the projects. The effect *delivery delay* is the most common effect felt in 41% of the projects.

Concerning R2DD prevention, the practices from the category *methodology* are more commonly used by 79% of the projects, and its practice *well-defined requirements* stands out, being used in 37% of the projects. On the other hand, the three categories of prevention PARs have the same percentual, indicating that organizational, methodology, and planning and management issues can influence the non-prevention of R2DD items. Regarding R2DD repayment, the practices from the category *planning and management* are commonly used to eliminating R2DD items in 32% of the projects. The practice *monitoring and controlling project activities* highlights in this category, being used in 11% of the projects. On the other hand, the PARs from the category *planning and management* are commonly used to justify the non-repayment of R2DD items, being considered in 40% of the projects. The PAR *focusing on short term goals* stands out in this category, being considered in 30% of the projects.

4.1 Use of the Conceptual Map

Software practitioners can use the map to start or improve their initiatives for dealing with R2DD items. As a benchmark, software practitioners can understand how R2DD items arise in their projects (causes) and their possible effects. Also, they can identify the practices used to prevent or repay these items and the PARs considered to explain R2DD non-prevention or non-repayment. The percentage can be used as a starting point for choosing preventive or repayment practices, or identifying causes, effects, and PARs associated with non-prevention or non-repayment.

As a conceptual guide, the map can be employed to inform actions in response to perceived R2DD items, and as a comprehensive guide when assessing software development practices. It facilitates a more effective identification and acknowledgement of R2DD items associated with software development issues (represented as categories in the map) that impact or are impacted by the presence of this type of debt.

In the map, causes of R2DD items are linked to practices and PARs related to their prevention, while effects of R2DD items are linked to practices and PARs associated with their repayment. It means that causes of R2DD can be avoided using a preventive practice or a PAR can be considered to explain the non-mitigation of a cause. On the other hand, the effects of R2DD can be eliminated by applying a repayment practice or a PAR can be considered to justify the non-repayment of a R2DD item.

The map does not indicate a correlation between causes and prevention or between effects and repayment. However, the map and its represented associations can facilitate the definition of strategies for dealing with R2DD items. For example, if a software team identifies that *deadline* is a cause of R2DD, analyzing the categories of preventive practices, the team can realize that applying practices from the category *following a well-defined project process* can curb this cause. In another example, consider that a software team felt the effect *low maintainability*. Navigating through the categories of repayment practices, the team can apply practices from the category *internal quality issues* for eliminating or mitigating this effect.

We believe that the map can help to create a more favorable environment for R2DD management. By considering the lists of causes, effects, practices, and PARs, as a communication facilitator, the map can be used to support software practitioners to more effectively communicate R2DD problems to management, and for managers to make better informed decisions concerning R2DD items.

5 Threats to validity

We identified some threats to validity affecting this work. They are removed, when possible, or mitigated. The categorization defined by Wohlin *et al*. [27] is used to identify and analyze these threats.

External validity. Although the study uses data collected from different countries, presenting a variety of participants in terms of roles, levels of experience, and workspace, we cannot generalize the results. To increase the external validity, we intend to include more data from other *InsighTD's* replications.

Internal validity. Threats affecting the internal validity can arise from the questionnaire. As it was applied remotely, the participants can misinterpret the questions. To mitigate this threat, the questionnaire was submitted to three internal and one external validations. Another threat is related to the terms TD prevention and repayment used in the questions. As the participants could misinterpret these terms, they could give invalid answers. To reduce this threat, we analyzed all answers given to Q23 and Q27 and concluded that no invalid answer was reported.

Conclusion validity. A threat arises from the qualitative analyses we carried out because they are a subjective task. To mitigate this threat, in each *InsighTD*'s replication team, the analyses were carried out separately by two researchers and the consensus was performed by an experienced researcher. Another threat comes from the survey's questions, because none of them is specific to requirements engineering or R2DD. We reduced this threat analyzing only the participant's answers who cited a valid example of R2DD in Q13, which is one of the acceptance criteria explained in Section 2.3.

Another threat affecting the conclusion validity is related to the participants' project role. Although we recognize the importance of requirements analysists in the context of this work, we did not survey only practitioners performing requirements tasks. Having a set of participants encompassing several project roles is also necessary because we can further understand the R2DD causes, effects, and managerial aspects that, although are related to requirements engineering, can affect other activities of the software development process.

6 Final Remarks

In this paper, we investigate R2DD and its elements, i.e., its causes, effects, practices used to prevent and repay R2DD items, and the PARs considered to justify its nonprevention and nonrepayment. We organize these elements into a conceptual map that can

support software teams in the management of R2DD items. Also, the map can guide new research efforts in a problem-driven way. Researchers can consider the R2DD's state of the practice in the development of new methods, strategies, and tools.

As future work, we intend to use more data from other *InsighTD*'s replications to evolve the map. Besides, we aim at assessing the map empirically concerning its effectiveness for supporting R2DD management by applying the technology acceptance model [28] that contributes to know the perception on the use of a new technology (the map).

References

1. Spínola, R.O., Zazworka, N., Vetrò, A., Shull, F., Seaman, C.: Understanding automated and human based technical debt identification approaches: a two-phase study. Journal of the Brazilian Computer Society, vol. 25(5) (2019).
2. Guo, Y., Spínola, R.O., Seaman, C.: Exploring the costs of technical debt management --- a case study. Empirical Softw. Engg., vol. 21, 1 (Feb. 2016), pp. 159-182 (2016).
3. Lim, E., Taksande, N., Seaman, C.: A balancing act: What software practitioners have to say about technical debt. IEEE Softw., vol. 29 (6), pp. 22-27 (2012).
4. Li, Z., Avgeriou, P., Liang, P.: A systematic mapping study on technical debt and its management. Journal of Systems and Software, vol. 101, pp. 193-220 (2015).
5. Freire, S., Rios, N., Mendonça, M., Falessi, D., Seaman, C., Izurieta, C., Spínola, R.O.: Actions and impediments for technical debt prevention: results from a global family of industrial surveys. In Proc. of the 35th ACM SAC, pp. 1548-1555 (2020).
6. Fernández, D.M., Wagner, S., Kalinowski, M., Felderer, M., Mafra, P., Vetrò, A., Conte, T., Christiansson, M.-T., Greer, D., Lassenius, C., Männistö, T., Nayabi, M., Oivo, M., Penzenstadler, B., Pfahl, D., Prikladnicki, R., Ruhe, G., Schekelmann, A., Sen, S., Spinola, R., Tuzcu, A., de la Vara, J. L., Wieringa, R.: Naming the pain in requirements engineering. Empir Software Eng, vol. 22, pp. 2298-2338 (2017).
7. Rios, N., Mendonça, M., Spínola, R.: A tertiary study on technical debt: Types, management strategies, research trends, and base information for practitioners. Information and Software Technology, vol. 102, pp. 117-145 (2018).
8. Aragão, B.S., Andrade, R.M.C., Santos, I.S., Castro, R.N.S., Lelli, V., Darin, T.G.R.: TestDCat 3.0: catalog of test debt subtypes and management activities. Software Quality Journal (2021).
9. Toledo, S.S., Martini, A., Przybyszewska, A., Sjøberg, D.I.K.: Architectural technical debt in microservices: a case study in a large company. In Proceedings of the TechDebt, pp. 78-87 (2019).
10. Apa, C., Solari, M., Vallespir, D., Travassos, G.H.: A Taste of the Software Industry Perception of Technical Debt and its Management in Uruguay: A survey in software industry. In Proc. of the 14th ESEM, pp. 1–9 (2020).
11. Rios, N., Spínola, R.O., Mendonça, M., Seaman, C.: 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 Softw.Engg., vol. 25, pp. 3216-3287 (2020).
12. Rios, N., Freire, S., Perez, B., Castellanos, C., Correal, D., Mendonça, M., Falessi, D., Izurieta, C., Seaman, C., Spínola, R.: On the Relationship Between Technical Debt Management and Process Models. IEEE Software, vol. 38 (5), pp. 56-64 (2021).
13. Kalinowski, M., Spínola, R., Conte, T., Prikladnicki, R., Fernández, D.M., Wagner, S.: Towards Building Knowledge on Causes of Critical Requirements Engineering Problems. In Proc. of the Twenty-Seventh International Conference on Software Engineering and Knowledge Engineering (SEKE) (2015).

14. Martini, A., Bosch, J., Chaudron, M.: Architecture technical debt: understanding causes and a qualitative model. In Proc. of the 40th Euromicro conference on software engineering and advanced application, pp. 85-92 (2014).
15. Martini, A., Bosch, J.: On the interest of architectural technical debt: uncovering the contagious debt phenomen. Journal of Software: Evolution and Process, 29:e1877 (2017).
16. Yli-Huumo, J., Maglyas, A., Smoander, K.: The sources and approaches to management of technical debt: a case study of two products lines in a middle-size finnish software company. In Proc. of the Product-Focused Soft. Process Improvement, pp. 93-107 (2014).
17. Yli-Huumo, J., Maglyas, A., Smolander, K.: The benefits and consequences of workarounds in software development projects. In Proc. of the Int. Conference on Software Business (ICSOB), pp. 1-16 (2015).
18. Avgeriou, P., Kructhen, P., Ozkaya, I., Seaman, C.: Managing Technical Debt in Software Engineering. Dagstuhl Seminar 16162, Dagstuhl Reports, vol 6:4 (2016).
19. Besker, T., Martini, A., Bosch, J.: Impact of architectural technical debt on daily software engineering and work – a survey of software practitioners. In Proc. of the 43rd Euromicro Conference on Software Engineering and Advanced Applications (SEAA) (2017).
20. Bano, M., Zowghi, D., Ferrari, A., Spoletini, P., Donati, B.: Learning from mistakes: An empirical study of elicitation interviews performed by novices. In Proc. of IEEE 26th International Requirements Engineering Conference (RE), pp. 182-193 (2018).
21. Rios, N., Spínola, R.O., Mendonça, M., Seaman, C.: Supporting analysis of technical debt causes and effects with cross company probabilistic cause effect diagrams. In Proceedings of the TechDebt, pp. 3-12 (2019).
22. Rios, N., Spínola, R.O., Mendonça, M., Seaman, C.: The most common causes and effects of technical debt: first results from a global family of industrial surveys. In Proc. of the 12th ESEM (2018).
23. Freire, S., Rios, N., Gutierrez, B., Torres, D., Mendonça, M., Izurieta, C., Seaman, C., Spínola, R.O.: Surveying Software Practitioners on Technical Debt Payment Practices and Reasons for not Paying off Debt Items. In Proc. of the Evaluation and Assessment in Software Engineering (EASE), pp. 210-219 (2020).
24. Rios, N., Mendes, L., Cerdeiral, C., Magalhães, A.P.F., Perez, B., Correal, D., Astudillo, H., Seaman, C., Izurieta, C., Santos, G., Spínola, R.O. Hearing the Voice of Software Practitioners on Causes, Effects, and Practices to Deal with Documentation Debt. In: Madhavji N., Pasquale L., Ferrari A., Gnesi S. (eds) Requirements Engineering: Foundation for Software Quality (2020).
25. Strauss, A., Corbin, J.M.: Basics of qualitative research: Techniques and procedures for developing grounded theory. Sage Publications (1998).
26. Alves, N.S.R., Mendes, T.S., Mendonça, M.G., Spínola, R.O., Shull, F., Seaman, C.: Identification and management of technical debt: A systematic mapping study. Information and Software Technology, vol. 70, pp. 100-121 (2016).
27. Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., Wesslén, A.: Experimentation in software engineering: An introduction. Springer (2012).
28. Davis, F.D.: Perceived usefulness, perceived ease of use, and user acceptance of information technology. MIS Quarterly, vol. 13 (3), pp. 319-340 (1989).