TestDCat: Catalog of Test Debt Subtypes and Management Activities

Bruno S. Aragão, Rossana Andrade, Ismayle S. Santos, Rute Castro, Valéria Lelli, Ticianne Darin

TestDCat: Catalog of Test Debt Subtypes and Management Activities

Bruno S. Aragão *, Rossana M. C. Andrade **, Ismayle S. Santos, Rute N. S. Castro ***, Valéria Lelli, and Ticianne G. R. Darin

Group of Computer Networks, Software Engineering, and Systems (GREat),
Federal University of Ceará (UFC)
Fortaleza, Ceará, Brazil
{bruno,rossana,ismaylesantos,rute,valerialelli,
ticiannedarin}@great.ufc.br

Abstract. When deadlines and resources of software projects become scarce, testing is usually in the first row to have its activities aborted or reduced. If defects cannot be found, products quality can be affected. In a software development process, aborted or reduced activities that can bring short-term benefits, but can be harmful to the project in a long run, are considered Technical Debts (TD). When TDs impact testing activities, they are called *Test Debt*. There are several studies dealing with *Test Debt*, however, current solutions often deal with specific types of tests (*e.g.,* exploratory and automated tests) and do not address the whole software testing process. Aiming to fill these gaps, this paper proposes a *Test Debt Catalog* with subtypes of *Test Debts* and technical debt management activities. This catalog was built based on semi-structured interviews conducted with practitioners who perform testing activities in five projects from industry. With our catalog, we intend to help the management of test debts during the execution of software testing processes.

Keywords: Technical Debt · Test Debt · Testing Process · TD management activity.

1 Introduction

Software Testing is one of the most commonly used approaches to evaluate software quality [14]. Moreover, one way to mitigate the risk of projects failing to perform tasks related to software testing is the use of a well-defined testing process. By using a process, the software development team can monitor and control the activities, as well as adjust them as required[14].

However, when deadlines or resources become scarce, organizations tend to reduce tasks and practices related to software testing [23]. Besides this, even

* Researcher scholarship - Master student, sponsored by FCPC
** Researcher scholarship - DT Level 2, sponsored by CNPq
*** PhD student Scholarship, sponsored by CAPES

using a testing process, members of a project may not perform (intentionally or unintentionally) some activities to achieve faster delivery and gain some competitive advantage [23]. In the context of a Test Factory - independent organizations that can offer high-quality testing services at a lower cost [4] [17] - these decisions are even more critical as they can directly affect the quality of testing services offered to customers.

This kind of technical commitments generated in software projects that may bring short-term benefits, but which in the long-term may be detrimental to project quality, are defined as *Technical Debts* (TDs) [13]. This concept was first used by Cunningham [7], who related the characterization of TD to problems in the code and the need for refactoring to pay the debts acquired. Other studies have addressed TDs in other activities of the software development process (*e.g.,* tests, requirements, documentation) and provided solutions to manage them in software projects [13][1]. For example, *Technical Debts* that concern the software testing are known as *Test Debts*. They arise when inadequate decisions regarding testing activities (*e.g.,* lack of tests, test estimation errors) are made [16].

Our problem identification came from our experience in a successful long-term partnership with the industry [2] in Research, Development and Innovation (R&D&I) projects. In these kinds of projects, the GREat[1] test factory team has followed a testing process[4].

We have conducted an empirical study[2] to identify the main issues faced on the GREat Test Factory. The objects of the study are five tools (two mobile and three web) we have developed in a partnership with industry. The web tools deal with critical information about the company's internal processes such as schedules, activity monitoring and resource allocation. The mobile applications are based on Android technology and are continuously updated with new features and changes in the interaction flow to cover usability issues. They currently have together over 3 millions active users, so they must be tested in several Android versions running on different target devices to ensure they work as users expected. As a result of the study, we identified several problems related to incomplete test specification such as lack of test procedures to execute the test or incorrect preconditions of test cases. Also, we observed that several releases are launched without any tests in 2019. In addition, some types of tests that were initially planned were postponed.

The problems mentioned before were identified even with the GREat test factory team using a well-defined testing process. This occurred because, under delivery pressure or by decision of the customer, the team failed to perform some steps of the testing process (intentionally or unintentionally) and, with that, they don't do it or leave some test artifacts immature[3]. With the problem identified,

[1] Group of Computer Networks, Software Engineering, and Systems. The GREat Research Group works on research and development software projects, developing web and mobile tools that are constantly being tested by the GREat Test Factory team.

[2] The details of the study are available on: `https://great-ufc.github.io/TestDCat/empirical/study.html`

[3] Immature artifact means any artifact that is not fully developed

we decided to formulate it as a test debt. Thus, the approaches and techniques used to manage this type of problem were studied in order to propose a solution to assist professionals deal with test debt. However, in spite of works dealing with *Test Debts* [16][20][18][22], they do not present in a consolidated manner the possible causes of *Test Debts* and how they can be identified and managed.

So, this work proposes a catalog, called *TestDCat*, of *Technical Debts* related to software testing as well as ways to manage them. We intend to give software engineers a broad view of subtypes of *Test Debts* that could occur in their projects and how to support their management. These subtypes were gathered from a systematic mapping study on *Technical Debts* and its management [13] and interviews with practitioners from five industry projects.

We evaluate the catalog in two steps. The first one was through a survey with the same interview participants and had the objective to identify if the catalog was in accordance with what they reported in the interviews. The second evaluation was through a focus group and the participants analyzed the entire catalog in detail and made observations and suggested improvements.

The following sections are organized as follows. Section 2 discusses related work. Section 3 presents the catalog design. Section 4 introduces our test debt catalog and Section 5 details its evaluation performed with experts. Section 6 discusses the results and, finally, Section 7 concludes the paper, presenting also perspectives of future work.

2 Related Work

The literature addresses different *Technical Debts* and their management activities. In this section, we focus the discussion on work related to *Test Debts* and management activities regarding to them.

Samarthyam et al. [16] present an overview of *Test Debts*, factors that contribute to this type of debts, and strategies for repayment of acquired *Test Debts*. These authors also classify *Test Debts* into: (i) unit testing; (ii) exploratory testing; (iii) manual testing; and (iv) automated testing. For each type of test, they present possible factors that may generate debts.

Aiming to support the repayment of TDs, Samarthyam et al. [16] propose a process with three macro activities: (i) Quantify the test debt, get the permission of the high administration, and execute the refund; (ii) Repay debts periodically; and (iii) Avoid the *Test Debts* from accumulating. They also present strategies for the payment of *Test Debts* that involve the application of good practices of test codification and in the accomplishment of the activities of software testing. Besides that, these authors describe two case studies in industry, in which they report experiences with *Test Debts*.

Although Samarthyam et al. [16] present a process with macro activities for the management of TD and identify good practices to prevent and repay TDs, this process does not detail all activities for TD management.

Sousa [20] presents a set of 22 TDs collected from literature review. The author describes its causes, indicators and possible solutions related to the software

testing process. To evaluate the TDs, a survey was performed with test professionals. A map was also prepared to support professionals in the management of TDs that may occur during the execution of the testing process. This map was evaluated applying a questionnaire with software testing professionals, but the map was not applied in software organizations. Besides that, this work does not related, explicitly, the *Test Debts* with the TDs management activities.

Shah et al. [18] performed a systematic review to answer the following questions: (i) "Is the exploratory testing an example of a practice that induces technical debt?" and (ii) "Should the debt be repaid later in the software life cycle?". In this review, the authors present how the exploratory testing influences the test activities and the related *Technical Debts*. Thus, they conclude that: (i) The lack of definition of test cases makes it difficult to perform regression tests and may cause residual defects; (ii) High human dependence, the missing of results evaluation, and lack of test planning may cause residual defects; (iii) Lack of documentation may lead to a poor understanding of the functionalities, generating rework and causing a wrong effort planning.

Therefore, the Shah et al.'s review provides an overview of *Technical Debts* regarding exploratory testing, but it does not cover TDs management nor other types of TDs that can occur during the test process.

Wiklund et al. [22] outline which factors contribute to the accumulation of *Technical Debts* in automated testing and assess the awareness of these debts in organizations that use automated testing. They identified these factors by performing semi-structured interviews with software designers.

Like the study of Shah et al., the work of Wiklund et al. is limited to one type of *Test Debts* - regarding automated testing - not addressing the wide range of debts generated during the testing process.

Based on the studies analyzed and presented in this section, we identified there is a need of presenting a consolidated view of possible causes of *Test Debts* and ways of managing them. We also identify this need from practical knowledge, inside the projects activities of the GREat Research Group.

3 Catalog Design

Information and know-how that comes from practitioners and that can be organized like a body of knowledge can be arranged in a catalog [5]. Based on this definition and the gaps identified in the literature, we decided to build a catalog to assist practitioners in managing *Test Debts*.

The design to build the catalog follows the methodology presented in Figure 1. This methodology is partially based on the Gorschek technology transfer model [9].

The model on which our methodology is based favours mutual cooperation between academia and industry and can be beneficial to both. Researchers can study relevant industry issues and validate their results in a real environment. In return, professionals receive knowledge about new technologies that can, for example, optimize their processes.

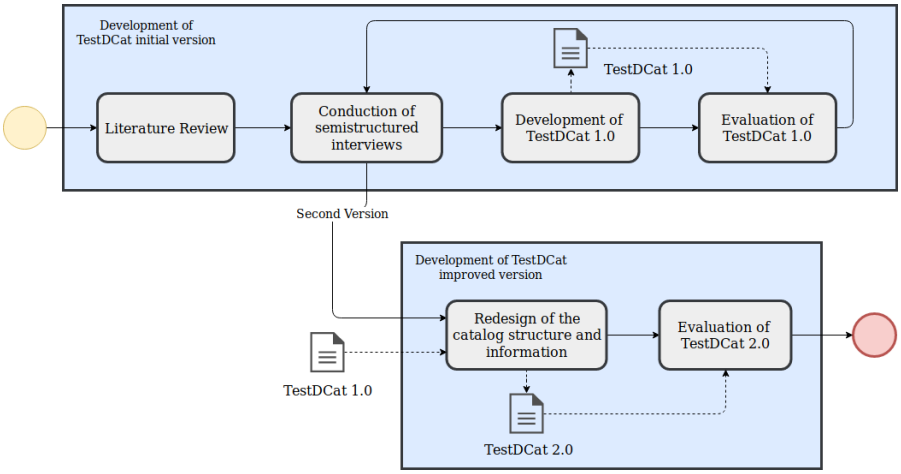

Fig. 1. Methodology used to build the *TestDCat* Catalog

The methodology is organized in two stages. At the first stage, we performed the activities: *Literature Review*; *Conduction of semistructured interviews*; *Development of TestDCat 1.0*; and *Evaluation of TestDCat 1.0*. At the second stage, we developed a new version of our catalog. Initially, we performed the *Redesign of the catalog structure and information* and evolved to the version *TestDCat* 2.0. Next, we conducted the *Evaluation of TestDCat 2.0*.

In the next sections, we detail the activities of the methodology.

3.1 Literature review

Once problems were identified from the empirical observation[4], the first step that we conducted was a *Literature Review* in the main databases of Computer Science (*e.g.,* Scopus, IEEE Xplore Digital Library and Science Direct) to identify how these problems could be formulated and what solutions are commonly used in the literature. Based on the results of our research, we realized that these problems could be formulated using the concept of *Technical Debts* (TD), more specifically, *Test Debts*. We also identified that most of the work dealing with *Test Debts* did not provide an overview of how to manage these debts throughout the test process. Thus, we realized the need to create a catalog for managing *Test Debts*.

In order to collect the information for the creation of the catalog, we started the second stage of the methodology: *Conduction of semistructured interviews*.

[4] The details of the study are available on: `https://great-ufc.github.io/TestDCat/empirical/study.html`

3.2 Conduction of semistructured interviews

Seeking to investigate practitioners' perspective about test debt, as well as to identify their strategies to deal with them, we conducted ten semi-structured interviews, in two different moments: initially focusing on test analysts' point of view, then on developers' standpoint. This method is adequate to gather in-depth data about a certain phenomenon and to tap into the expert knowledge of an individual [10], which allowed us to understand practitioners' approaches, concerns and needs regarding tests debts. Two researchers (head and auxiliary/observer) conducted and recorded the interviews, which were later transcribed and analyzed. Each session took about two hours on average and explored questions about practitioners' experience, subtypes of *Test Debts*, and TD management activities.

As a result of this step, we obtained information on which subtypes of test debt were acquired in their practical experience and actions taken to manage the debts identified. For each identified debt, we collected the actions used to: identification, measurement, prioritization, communication, monitoring, repayment, documentation and prevention of the debt. Such information was the basis for the definition of the actions to be carried out in each of the subtypes of *Test Debts*, which were later organized as a catalog.

Further details about interviews planning and conduction are explained in the remainder of this subsection, organized according to the four activities we conducted in this step: 1) Interview planning; 2) Participants selection; 3) Interviews conduction; 4) Data analysis and consolidation.

Interview planning Initially, we narrowed down some areas and topics to elicit conversation with the practitioners. Then, we organized the topics in a script format with an opening statement, a set of general questions to be explored by each topic, and additional questions designed to probe for information, in cases it did not come up[5]. To guide interviews, we developed a protocol which was refined with the consultation from experts in the field and also experts in qualitative research to provide us with feedback and guidance. We also piloted the interview guide to help improve its instrumentation [15]. Other than helping us to pay close attention to the relationship between the questions asked and the content produced during the interviews, the protocol also included statements of confidentiality, consent, options to withdraw, and intended use of the results. We formulated the topics and questions following the good practices presented in [10]. The protocol established the following steps to conduct the interview sessions: 1. *Profile Identification*, which included questions about the participants demographics, background, and experience in the field; 2. *Questions about* Test Debts to identify evidence of their presence in projects participants were currently working; 3. *Questions about the TDs management activities* selected from [13], these questions are intended to identify whether respondents execute any form of management among those identified *Test Debts*; and 4. *Slowdown*,

[5] The interview script is available on: `https://bit.ly/2JPy1CD`

used for final considerations, with questions about the impact of lack of TD management activities and the possibility of using a catalog that could assist the process of managing *Test Debts*.

Participants selection We selected participants according to the technique classified as selective or purposeful sampling [6], since selection was according to the availability of practitioners at GREat laboratory and GREat Test Factory. All participants work in research and development projects conducted in partnership with industry at the GREat Research Group. Currently, such projects encompass the development of six web tools and two mobile applications, which are periodically tested by the GREat Test Factory. Table 1 summarizes the participants profile.

Interviews occurred in two different moments. First, we conducted five interview sessions focusing on test analysts' point of view. Hence the participants included three test analysts, a test leader, and a software testing researcher who also works on projects with industry. On average, they had about three years of experience working in software tests field, including functional and non-functional testing. Later, we conducted five more interviews to endeavor to address software developers perspective on testing. For these interviews, five more participants were selected: four system analysts and a technical leader. Besides their vast experience in software development activities, all of them also worked with functional tests.

Interviews conduction Each interview was conducted by two researchers. They both followed the instructions from the interview protocol detailing each procedure that must be followed during data collection. The information collected from participants was properly kept anonymous and private and was used exclusively for the analysis of test debt characteristics and actions in the scope of this investigation. Before beginning the interview session, the researcher organized a room with no distractions, and set the evaluation environment, including voice recorder so that each person's interview session could be documented. Before a participant entered the room, the recorder and all data collection instruments were already available and organized. When a participant entered the room, the researcher explained what the investigation was about and requested participants' permission to record the interviews. Then, the participant would sign a *terms of free and informed consent*, which assured data confidentiality and anonymity.

Data analysis and categorization Initially, we run a brief quantitative analysis on closed ended questions including, for example, whether practitioners felt pressured to perform test activities, and the occurrence of different sub-types of *Test Debts*. Objective answers were often also followed by comments that were later analyzed in a qualitative way.

Then, a qualitative analysis was conducted on the answers recordings from each interview. Interviews were transcribed, and critical comments were iden-

Table 1. Participants profile

ID	SPECIALTY	EXPERTISE	CURRENT POSITION	TDs REPORTED
P01	Software testing	Functional and non-functional tests	Junior Test Analyst	- Deferring testing - Lack of tests - Defects not found in tests - Expensive tests - Test estimation errors
P02			Mid-level Test Analyst / Test Leader	
P03			Junior Test Analyst	- Deferring testing - Lack of tests - Lack of tests automation - Defects not found in tests - Expensive tests - Test estimation errors
P04			Researcher	
P05			Junior Test Analyst	- Lack of tests - Expensive tests - Test estimation errors
P06	SW development and testing	SW development and functional tests	Mid-level System Analyst / Technical Leader	- Low code coverage - Deferring testing - Defects not found in tests - Expensive tests - Test estimation errors
P07			Senior Systems Analyst / Technical Leader	- Low code coverage - Deferring testing - Lack of tests - Expensive tests - Test estimation errors
P08			Mid-level System Analyst	- Deferring testing - Lack of tests - Lack of tests automation - Defects not found in tests - Expensive tests - Test estimation errors
P09			Mid-level System Analyst	- Low code coverage - Lack of tests - Defects not found in tests
P10			Researcher	- Low code coverage - Deferring testing - Lack of tests automation - Defects not found in tests - Test estimation errors

tified by using open coding approach for the categorization of transcripts and identification of goals connections among the findings they revealed [3]. Overall, the transcripts analysis took about 70 hours. We used the classification tree and *Technical Debts* management activities proposed by Li et al. [13] as basis for our data categorization. Hence, we organized the transcripts into eight well-defined categories that guided data consolidation as described in the *Test Debt Catalog* section.

3.3 Development of TestDCat 1.0

For the *Development of TestDCat 1.0*, all information obtained from the first session of interviews (with five participants) was analyzed and organized in a

matrix that represents the *TestDCat 1.0* catalog. The matrix was formed by test debt subtypes and, for each of them, a set of TD management activities was associated. Within each of these activities, there was information about "Approaches", "Points of attention in the test process" and "Good Practices". Figure 2 presents an example with the test debt subtype "Low code coverage" and the TD management activity "Identification".

Test Debt subtype	TD Management Activities
	Identification
Low code coverage	**Approaches:** **Code analysis:** Investigate the code to analyze its coverage. This analysis is usually done with a specifc tool (e.g., Eclema).
	Points of attention in the test process: **Test Planning:** Definition of the coverage targets. **Test Design:** Registration of test coverage **Test Execution:** feedback about the coverage to test monitoring **Test Monitoring:** Collection and analysis of measurements
	Good Practices: Use of continuous integration to automate the code coverage verification process

Fig. 2. *TestDCat 1.0* example

3.4 Evaluation of TestDCat 1.0

The goal of the first evaluation was to gather the participants perception about the catalog clarity, ease of use and completeness. So, the evaluation intended to get answers to the following: *Does the TestDCat catalog have clear and enough information to support the management of Test Debts?*

For this evaluation, we presented the whole catalog, printed on paper, to the five interviews' participants of the first moment and, after reading and analyzing the catalog, they answered the questions of the survey.

With regards to clarity, all participants agree, 60% strongly agree and 40% agree, that the catalog presents the information in a clear and objective way. Furthermore, most of the participants agrees, 40% strongly agree and 40% agree, that the catalog is easy of use. Just one of them chose a neutral response.

Regarding the completeness, most of the participants agrees, 20% strongly agree and 60% agree, that the catalog has enough information to support the management of *Test Debts*. On the other hand, one of them disagreed about the completeness of the catalog.

Regarding the usefulness of the catalog, all participants agree, 100% strongly agree, that it would certainly help users to manage their *Test Debts*.

In addition, the participants suggested improvements to the catalog in the open questions. For instance, they asked for more details on each approach presented, as well as more practical information that would make the use of the catalogue more precise.

3.5 Redesign of the catalog structure and information

Based on the results of the first evaluation and new information from the second interview session, we created a new version of the catalog. This new version is described in section 4.

3.6 Evaluation of TestDCat 2.0

In order to evaluate the new version of *TestDCat* produced, a new evaluation was performed. This evaluation was conducted through focus groups with five participants of different kinds of expertise in the software development process. Details about this evaluation are in Section 5.

4 Test Debt Catalog

Based on the results of the first evaluation carried out, as well as the new information taken from the new interview session conducted, we redesigned the catalog.

The *TestDCat* 2.0 catalog consists of TD management activities and, for each of them, a set of subtypes of *Test Debts* are associated. Within each of these subtypes, actions are presented using the 5W1H (Who, What, When, Where, Why, and How) model. This model is commonly used for the development of action plans [8].

The data is categorized according to TD management activities and subtypes of test debt, both mapped by Li et.al [13]. The test debt subtypes are the possible causes for this kind of debt. In addition to the subtypes identified by Li et.al, we identified two other subtypes: Inadequate equipment; and Inadequate allocation.

Figure 3 presents the new structure of the catalog, emphasizing with listed circles parts of the catalog. Circle number 1 emphasises the technical debt management activities: Identification, Measurement, Prioritization, Communication, Monitoring, Repayment, Documentation, and Prevention. By clicking on any of these activities, are presented subtypes of *Test Debts* and its related actions. In this example, the "Identification" activity was selected.

The circle number two presents the subtypes of *Test Debts*: Low code coverage, Deferring testing, Lack of tests, Lack of tests automation, Defects not found in tests, Expensive tests, Test effort estimation errors, Inadequate equipment, and Inadequate allocation. In this catalog, we present a set of actions for each subtype. For example, after clicking on the "Identification" activity, the user can choose which subtype he wants to handle. In this example, we choose "Low Code Coverage", which has two related actions.

The circle number three brings forward the actions identified through semistructured interviews conducted. They are following the 5W1H model. In this case, these are the suggested actions to help catalog users to identify *Test Debts* caused by "Low Code Coverage".

Actions can be used together or individually. Thus, in the circle number four we present the functionality in which the user can select which actions are most

related to their context. After selecting, a custom action plan will be created. Finally, it's possible to print or download this plan.

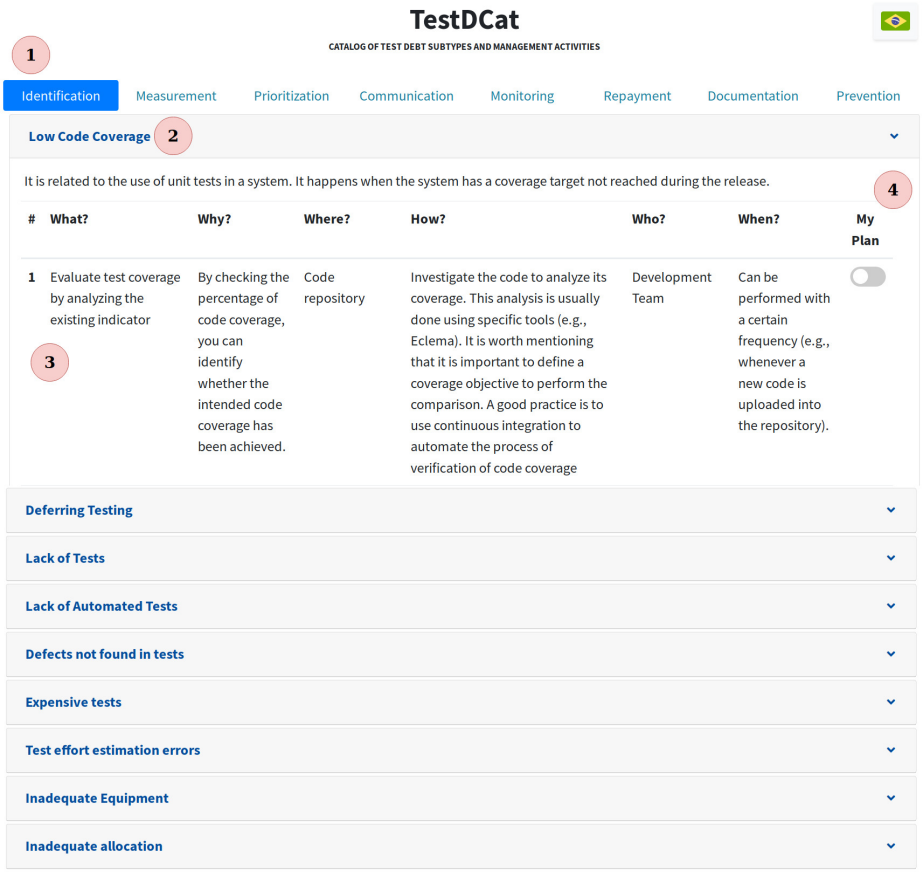

Fig. 3. *TestDCat 2.0* structure

All the catalog's information can be viewed on the *TestDCat*'s website [6]. Another aspect of this catalog website is that we inserted a form so the visitors can also suggest improvements or new actions for managing *Test Debts*.

5 Catalog Evaluation

After producing *TestDCat* version 2.0, we further conducted in-depth evaluations to assess the correctness, quality, and coverage of the catalog content - including

[6] *TestDCat*'s website: `https://great-ufc.github.io/TestDCat/index.html`

all statements, descriptions and subcategories - under the perspective of practitioners. Hence, to capture impressions of potential catalog users and elicit their suggestions, reactions, frustrations, and fears we convened three sessions of focus groups. Focus group is an inexpensive and powerful approach from the social sciences largely used in human-computer interaction experimental and empirical researches to help generating a deeper and more nuanced understanding of an issue [12], [19].

The focus group was planned, piloted and conducted by an experienced moderator, together with two assistant observers who took structured notes. *TestDCat* was used as a discussion guide, as all catalog sections and actions were systematically discussed in three focus group sessions, which lasted from 2-3 hours each. Each participant received a hard copy of *TestDCat*, some Post-its and pens, and was encouraged to make annotations and suggestions as the discussions progressed. Due to the long sessions, the typical question and answer style script was combined with prioritization and summary exercises. Additionally, refreshments were made available and regular comfort breaks were offered to the participants.

Five participants were carefully selected according to their expertise, background and availability. As summarized in Table 2, the focus group sessions gathered practitioners with different types of expertise within the software development process, as well as various levels of experience. They also had different levels of knowledge about the concept of technical debt. Our goal was to obtain insights and feedback about the utility, clarity and applicability of *TestDCat* to practitioners in different stages of their careers. The variety of participants provided us with a broad range of viewpoints and insights among peers with whom participants shared a common background: the knowledge and practical experience about software testing, and the need to manage *Technical Debts* in some level. Although opinions differ on optimal sizes for focus groups [12], smaller groups are more appropriate to produce deeper and more fruitful discussions [11].

Table 2. Profile of focus group sessions participants

ID	SPECIALTY	EXPERIENCE IN THE FIELD	CURRENT POSITION	LEVEL UNDERSTANDING ABOUT TECHNICAL DEBT
P01	Software testing	3 years	Junior Test Analyst	I'm familiar with the concept
P02	SW development and testing	13 years	Senior Systems Analyst / Technical Leader	
P03	SW development and testing	1,5 year	Junior Systems Analyst	I've applied strategies to manage TDs
P04	Software testing	2 years	Software testing researcher	I'm familiar with the concept
P05	Software development	4 years	Mid-level Systems Analyst	I'm not familiar with the concept

During the evaluation session, for each TD subtype the moderator promoted discussions on every component (5W1H) of the listed actions. Then, in a round

table, focus group participants commented on each others point of view, often challenging each others motives and actions, and relating their own experiences to the catalog descriptions. This procedure allowed participants to thoroughly analyze each action and its steps. In addition, a flip chart was available so that moderator could summarize the identified drawbacks and advantages, suggestions of improvements, and doubts that were common to all participants. The moderator would sum up important points at convenient times, making sure participants had understood them. After discussing an action, the mediator asked whether the participants indicated that the action should be kept in the catalog, undergo changes, or whether they felt it was not appropriate and should leave. Then, when all actions of a subtype category were discussed, participants individually used a 5-point Likert rating scale to assess the following criteria: applicability to the full range of intended uses, concreteness, clarity, ease of understanding, ease of use, impartiality, and relevance to the context. Such criteria were adapted from those proposed to assess the quality of evaluation checklists in a particular area [21]. The results of participants' rating scales are summarized in Figure 4. By the end of each session, the moderator and observers conducted a debriefing and led a summary exercise to gather key themes and check for further understanding on participants, moderation and observers' notes. Besides, they identified and categorized note themes, hunches, interpretations, and ideas. Then, they labeled, compared and contrasted information from field notes, and other materials.

A total of 63 actions were evaluated and analyzed. Overall, participants suggested to remove an action related to the "Communication" activity and add a new action to the "Identification" activity. Some relevant doubts arose regarding: "Test coverage and code coverage. Is it separate or is it the same thing?", "How to identify the ideal version?". The main improvements suggested were "Categorize separating tests of what is manual and automated", "Standardize the terms (especially for the columns "Who" and "Where")". Finally, some of the changes requested included: "Put a glossary with test area terms and technical debt (TD) for people who are not very experienced in testing and TD", "As it is dependent on some preconditions, it would be better to clearly separate what is precondition and what is the action in fact", "Use the term iteration instead of Sprint" and "Keep the term follow-up meetings in all actions that mention holding meetings".

6 Discussion

The second evaluation, in which the participants evaluated each action in the catalog, proved to be very useful, despite requiring a lot of effort. Several improvements were suggested and only one of the actions was considered inadequate for the proper purpose. In a nutshell, the proposed catalog got good results.

All criteria had more than 50% agreement. The criteria "Applicability to all intended uses" obtained 77.5% agreement and "Relevance to the context"

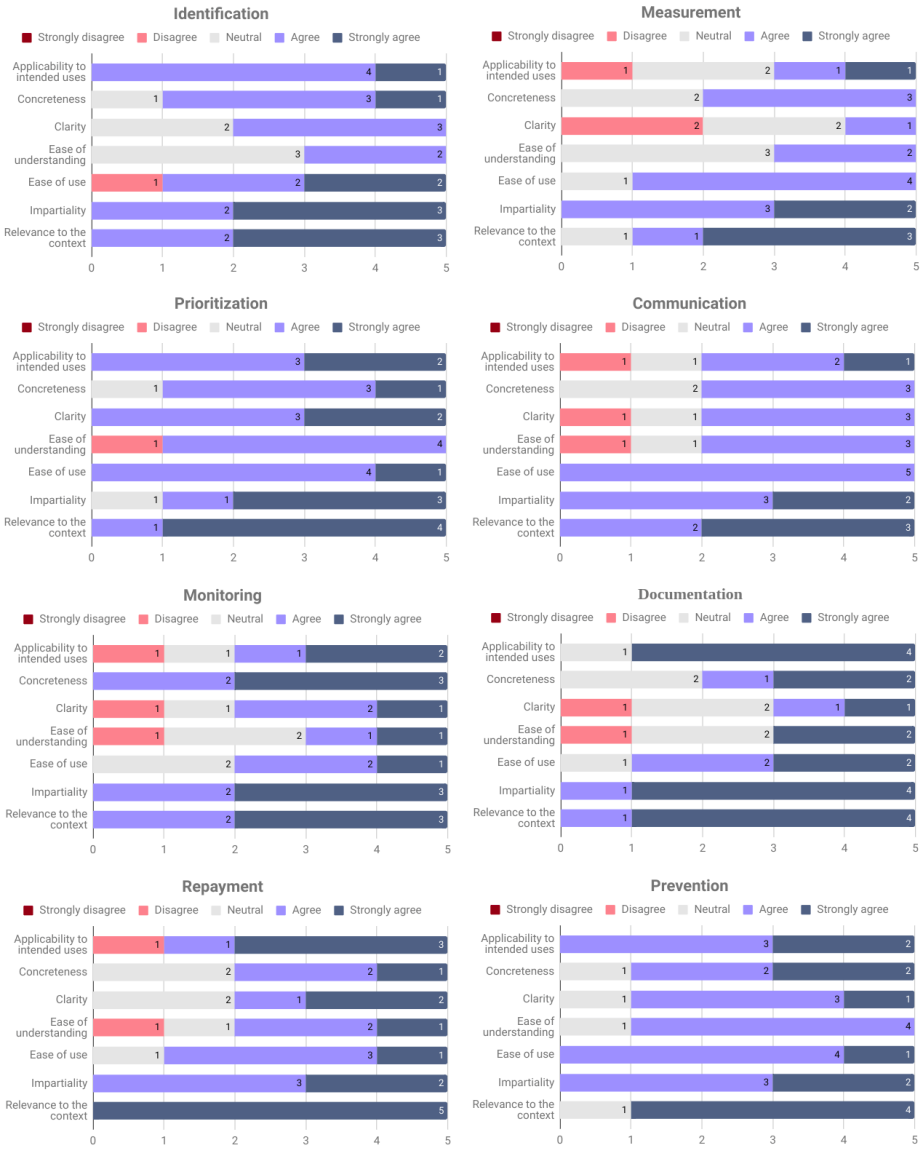

Fig. 4. Results of the evaluation

obtained more than 95%, which reflects that the participants believe that the catalog can indeed be used to assist in the management of *Test Debts*.

The criteria "Clarity" and "Ease of understanding" were the ones that had the highest rate of disagreement with 12.5%. In the individual analysis of the answers, we noticed that the majority of the participants who had these opinions had less work experience. However, as the use of the catalog must be made by all

levels of expertise, these are points of improvement that must be addressed. It is worth mentioning that during the evaluation the participants made many observations in order to improve these criteria, so we believe that when considering these new improvements, these rates tend to decrease.

It is also worth mentioning that the interview with professionals brings the benefit of gathering information about the perception of the regarding *Test Debts*. This was useful to identify new *Test Debts*. For instance, to the best of our knowledge, the *Test Debts Inadequate equipment* and *Inadequate allocation* were not previously reported in the literature.

Threats to Validity Regarding the threats to validity, we discuss threats related to *Internal Validity* and *External Validity*. According to [24], threats to *Internal Validity* are influences that can affect the independent variable with respect to causality, and threats to *External Validity*, in turn, are conditions that limit the ability to generalize the results to industrial practice.

In our case, the main threat regarding the *Internal Validity* is the selection of the subjects that was made based on convenience sampling [24]. In the case of *External Validity*, the small number of participants is our main threat. Despite these limitations, it is worth noting that the participants were professionals with great experience on software testing activities in industry. Furthermore, these professionals had experience of testing mobile and web software.

7 Conclusion and future work

Test Debts have a high impact on software quality. So, they require the use of management activities whithin the testing process to be monitored and controlled during the testing releases. Nevertheless, most of the studies focus on the management of *Technical Debts* in general or management of specific *Test Debts*.

Aiming to address this gap and support the practitioners in the management of *Test Debts*, we proposed a catalog, called TestDCat, that was created based on the information gathered from semi-structured interviews performed with professionals from industry.

TestDCat presents an overview of management activities, subtypes of *Test Debts* and actions to assist in TD management activities. To get an initial evaluation, we presented the catalog to the participants of the interviews who answered a survey regarding clarity, ease of use and completeness. In the second evaluation, we made a focus group that aimed to analyze in detail the actions of the catalog and suggest changes and improvements.

The results of these two evaluations presented evidence that the information organized in the catalog can support the management of *Test Debts*. Thus, it may help the development and testing team to monitor the current debts and to take the actions according to the identified test debt.

As future work, we intend to proceed with the improvements suggested during the second evaluation and apply the actions catalogued in *TestDCat* in real projects so that it will be possible to evaluate their practical contribution.

References

1. Alves, N.S., Mendes, T.S., de Mendonça, M.G., Spínola, R.O., Shull, F., Seaman, C.: Identification and management of technical debt: A systematic mapping study. Information and Software Technology **70**, 100–121 (2016)
2. Andrade, R.M.C., Lelli, V., Castro, R.N.S., Santos, I.S.: Fifteen years of industry and academia partnership: Lessons learned from a brazilian research group. In: 2017 IEEE/ACM 4th International Workshop on Software Engineering Research and Industrial Practice (SER IP). pp. 10–16 (May 2017). https://doi.org/10.1109/SER-IP.2017..2
3. Burnard, P.: A method of analysing interview transcripts in qualitative research. Nurse education today **11**(6), 461–466 (1991)
4. de Castro Andrade, R.M., de Sousa Santos, I., Lelli, V., de Oliveira, K.M., da Rocha, A.R.C.: Software testing process in a test factory - from ad hoc activities to an organizational standard. In: ICEIS (2017)
5. Chung, L., Nixon, B.A., Yu, E., Mylopoulos, J.: Non-functional requirements in software engineering, vol. 5. Springer Science & Business Media (2012)
6. Coyne, I.: Sampling in qualitative research. purposeful and theoretical sampling; merging or clear boundaries? Journal of advanced nursing **26**(3), 623–630 (1997)
7. Cunningham, W.: The wycash portfolio management system. SIGPLAN OOPS Mess. **4**(2), 29–30 (Dec 1992). https://doi.org/10.1145/157710.157715, `http://doi.acm.org/10.1145/157710.157715`
8. Fernandes, F., Sousa, S., Lopes, I.d.S.: On the use of quality tools: a case study. In: World Congress on Engineering 2013. vol. 1, pp. 634–639. Newswood Limited Publisher (2013)
9. Gorschek, T., Garre, P., Larsson, S., Wohlin, C.: A model for technology transfer in practice. IEEE software **23**(6), 88–95 (2006)
10. Hove, S.E., Anda, B.: Experiences from conducting semi-structured interviews in empirical software engineering research. In: Software metrics, 2005. 11th ieee international symposium. pp. 10–pp. IEEE (2005)
11. Krueger, R.A., Casey, M.A.: Designing and conducting focus group interviews (2002)
12. Lazar, J., Feng, J.H., Hochheiser, H.: Research methods in human-computer interaction. Morgan Kaufmann (2017)
13. Li, Z., Avgeriou, P., Liang, P.: A systematic mapping study on technical debt and its management. Journal of Systems and Software **101**, 193–220 (2015)
14. Orso, A., Rothermel, G.: Software testing: a research travelogue (2000–2014). In: Proceedings of the on Future of Software Engineering. pp. 117–132. ACM (2014)
15. Rogers, Y., Sharp, H., Preece, J.: Interaction design: beyond human-computer interaction. John Wiley & Sons (2011)
16. Samarthyam, G., Muralidharan, M., Anna, R.K.: Understanding test debt. In: Trends in Software Testing, pp. 1–17. Springer (2017)
17. Sanz, A., Garcia, J., Saldana, J., Amescua, A.: A proposal of a process model to create a test factory. In: Software Quality, 2009. WOSQ'09. ICSE Workshop on. pp. 65–70. IEEE (2009)
18. Shah, S.M.A., Torchiano, M., Vetro, A., Morisio, M.: Exploratory testing as a source of technical debt. IT Professional **16**(3), 44–51 (2014)
19. Shneiderman, B., Plaisant, C., Cohen, M., Jacobs, S., Elmqvist, N., Diakopoulos, N.: Designing the user interface: strategies for effective human-computer interaction. Pearson (2016)

20. Sousa, C.L.d.: Mapa de apoio à gestão de dívida técnica no processo de teste de software. Master's thesis, Universidade Federal de Pernambuco (2016)
21. Stufflebeam, D.L.: Guidelines for developing evaluation checklists: the checklists development checklist (cdc). Kalamazoo, MI: The Evaluation Center (2000)
22. Wiklund, K., Eldh, S., Sundmark, D., Lundqvist, K.: Technical debt in test automation. In: 2012 IEEE Fifth International Conference on Software Testing, Verification and Validation. pp. 887–892 (April 2012). https://doi.org/10.1109/ICST.2012.192
23. Wiklund, K., Eldh, S., Sundmark, D., Lundqvist, K.: Impediments for software test automation: A systematic literature review. Software Testing, Verification and Reliability **27**(8) (2017)
24. Wohlin, C., Runeson, P., Hst, M., Ohlsson, M.C., Regnell, B., Wessln, A.: Experimentation in Software Engineering. Springer Publishing Company, Incorporated (2012)