2017 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement

Formative Evaluation of a Tool for Managing Software Quality

Liliana Guzmán*, Anna Maria Vollmer*, Marcus Ciolkowski†, and Michael Gillmann‡
* Fraunhofer IESE
Fraunhofer Platz 1, 67663, Kaiserslautern, Germany
Email: {liliana.guzman, anna-maria.vollmer}@iese.fraunhofer.de
† QAware GmbH
Aschauer Str. 32, 81549, München, Germany
Email: marcus.ciolkowski@qaware.de
‡ Insiders Technologies GmbH
Brüsseler Str. 1., 67657, Kaiserslautern, Germany
Email: m.gillmann@insiders-technologies.de

***Abstract*—*Context/Background:* To achieve high software quality, particularly in the context of agile software development, organizations need tools to continuously analyze software quality. Several quality management (QM) tools have been developed in recent years. However, there is a lack of evidence regarding the quality of QM tools, standardized definitions of such quality, and reliable instruments for measuring it. This, in turn, impedes proper selection and improvement of QM tools. *Goals*: We aimed at operationalizing the quality of a research QM tool, namely the ProDebt prototype, and evaluating its quality. The goal of the ProDebt prototype is to provide practitioners with support for managing software quality and technical debt. *Method:* We performed interviews, workshops, and a mapping study to operationalize the quality of the ProDebt prototype and to identify reliable instruments to measure it. We designed a mixed-method study aimed at formative evaluation, i.e., at assessing the quality of the ProDebt prototype and providing guidance for its further development. Eleven practitioners from two German companies evaluated the ProDebt prototype. *Results:* The participants assessed the information provided by the ProDebt prototype as understandable and relevant. They considered the ProDebt prototype's functionalities as easy to use but of limited usability. They identified improvement needs, e.g., that the analysis results should be linked to other information sources. *Conclusions:* The evaluation design was of practical value for evaluating the ProDebt prototype considering the limited resources such as the practitioners time. The evaluation results provided the developers of the ProDebt prototype with guidance for its further development. We conclude that it can be used and tailored for replication or evaluation of other QM tools.**

***Index Terms*—software quality, software measurement, software engineering**

1. Introduction

An important principle in software engineering projects is to avoid "broken windows" of software quality [1] – whether for software maintenance or renovation projects, or for developing new systems. This principle has become particularly prominent with the advent of agile software development (ASD) [2]. ASD facilitates flexible, rapid, and continuous development by providing practitioners with iterative methods that rely on extensive collaboration. To succeed in ASD, organizations need tools for continuously analyzing software quality along short-time releases [3]. Otherwise, developers tend to focus on functional requirements, neglecting quality requirements [4]. Thus, continuous measurement of quality deficits is crucial for ensuring software quality.

Several quality management (QM) tools have been developed in recent years [5]. However, there is very little evidence on their quality, and a standardized definition and instruments for measuring it are missing [6], [7]. Evaluating the quality of QM tools using reliable measurement instruments is important to guide research and practice in continuously improving QM tools over time and to allow comparing and selecting QM tools.

In this paper, we contribute to the formative evaluation of the quality of a QM tool. We will summarize the results of interviews, workshops, and a mapping study aiming at operationalizing the quality of a QM tool and identifying reliable instruments for measuring it (Section 2). Then, we will describe a formative evaluation design (Section 3) to evaluate a single QM tool, namely the ProDebt prototype. We will also summarize the results of the formative evaluation of the ProDebt prototype (Section 3). Eight developers and three managers assessed the information provided by the ProDebt prototype as understandable and relevant for analyzing quality deficits. They perceived the ProDebt prototype's functionalities as easy to use but of little use. They claimed there is a need to visualize the analysis results of the ProDebt prototype together with the related source code and product backlog. Finally, we will discuss the lessons learned in evaluating a QM tool (Section 5) and the implications of our research (Section 6).

978-1-5090-4039-1/17 $31.00 © 2017 IEEE
DOI 10.1109/ESEM.2017.43

IEEE computer society

Authorized licensed use limited to: Idaho State University. Downloaded on July 29,2020 at 03:31:47 UTC from IEEE Xplore. Restrictions apply.

2. Related work

2.1. Quality Management Tools

In the context of this paper, we define a QM tool as a tool that provides software practitioners with automated support to assess the quality of a software system during its development based on measured properties. The quality characteristics of the software depends on context factors such as type of software, application domain, or end-users. Therefore, QM tools typically need to address multiple quality aspects. An overview of QM tools is given in [5], [6]. Current QM tools vary from research tools like Quamoco and Squale to commercial tools like SonarQube or CAST's Application Intelligence Platform [5].

In [7], we reported a mapping study to characterize empirical evaluations regarding QM tools considering technical debt. We found a lack of evidence on the quality of such QM tools: In 2016, there were 14 published empirical studies evaluating QM tools, but only two reported sufficient information about the evaluation to allow proper interpretation of their results. We also observed a lack of common quality criteria and reliable instruments to measure the quality of QM tools. These results are consistent with the findings of Li et al. [6], who performed a comprehensive systematic review regarding – amongst others – tools for QM and technical debt. They observed that only very few tools for QM and technical debt have been evaluated empirically. Evaluating QM tools will help to understand their suitability, and support their further development, continuous improvement, comparison and selection.

2.2. The ProDebt Prototype

ProDebt is a research project[1] funded by the German Federal Ministry of Education and Research. It aims at developing a method and a prototype for proactively managing software quality and technical debt in the context of ASD. The core elements of the ProDebt prototype are a tailorable quality model with quality profiles, which allows differentiated estimation of the quality cost gap and technical debt, as well as a cockpit, that visualizes both the quality model and the estimation for different end-users (c.f. Figure 1).

The ProDebt prototype aims at providing developers and managers with the following functionalities: (F1) evaluating the quality of a software system at one point in time, (F2) analyzing the quality of a software system over time at different levels such as overall system, subsystem, and files, (F3) drill-down into potential causes of quality deficiencies to allow impact analysis and planning of specific improvements, and (F4) estimating technical debt.

The ProDebt prototype is being implemented following an iterative and incremental approach. Its development has also included a formative evaluation aimied at understanding the quality of the ProDebt prototype and supporting its further development. At the time of this publication, the ProDebt prototype provides practitioners with support for the first three functionalities mentioned above (F1 to F3). A summative evaluation of the ProDebt prototype is also planned at the end of the project.

2.3. Quality of Quality Management Tools

To be able to evaluate the quality of the ProDebt prototype, we first aimed at systematically operationalizing the quality of a QM tool.

Methodology. We operationalized the quality of a QM tool as follows:

1) *Interviews.* We interviewed ten end-users of QM tools, namely four developers and six managers. The interview questionnaire included five open questions on the expected quality of a QM tool. We designed the questions taking into consideration the quality aspects extracted from the definitions used in the empirical evaluations of QM tools found in [6], [7]. Each interview lasted up to 45 minutes.
2) *Workshops.* We performed two workshops with two experts in QM and three experienced end-users of QM tools. These participants had not been interviewed previously. In the first workshop, we performed a group interview and asked the participants to define the quality of a QM tool. In the second workshop, we presented, related, and integrated the results of the interviews and the first workshop. We elicited the following quality aspects: completeness, understandability, transparency, efficiency, applicability, reliability, awareness, novelty of findings, defect proneness, trust in technology, fitness for purpose, and acceptance.
3) *Mapping study.* We looked for reliable definitions and operationalizations of each elicited quality aspect. That is, we searched for reliable test instruments. We used Kitchenham's guidelines for systematic reviews [8] to derive search strings and analyze results. We performed the search using the Inter-Nomological Network (INN)[2] tool and the Scopus database. INN aims at facilitating the search of test instruments in behavioral sciences, including information systems. We found 31 papers providing a total of 50 test instruments within our scope. An overview of our results is given in [7]. We also observed that the quality aspects of interest to us are often grouped together into two main quality criteria: information and system quality. McKinney et al. [9] and Nelson et al. [10] showed that considering this differentiation enables a more powerful and precise evaluation of a system. Thus, we chose to keep this differentiation.

Definition. Table 1 summarizes the unified definition of the quality of a QM tool and the elicited quality aspects after

[1] http://www.prodebt.de

[2] http://inn.theorizeit.org

Authorized licensed use limited to: Idaho State University. Downloaded on July 29,2020 at 03:31:47 UTC from IEEE Xplore. Restrictions apply.

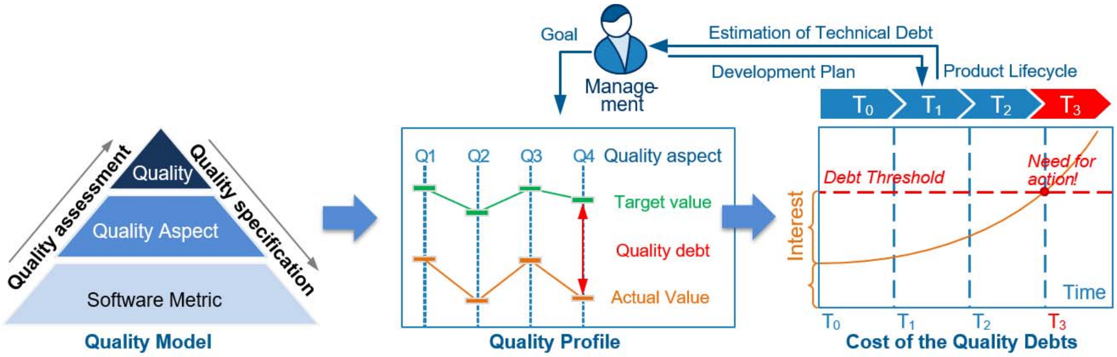

Figure 1. Core Elements of the ProDebt Approach

comparing, relating and aggregating the operationalizations found in the literature. It also provides an overview of the related references and evidence on the reliability of the related test instruments. We defined the quality of a QM tool according to two main quality criteria:

1) ***Information quality*** refers to the quality of the data stored and produced within a system. A system with high information quality comprises information that properly supports users' tasks.
   Several operationalizations of information quality exist in the literature. McKinney et al. decomposed it into understandability, reliability, and usefulness of the information encompassed in a system [9]. In contrast, Lee and Strong broke down information quality into accuracy, completeness, timeliness, relevancy, and accessibility of the provided information [11]. Nelson et al. described accuracy, completeness, currency, and format of the information as parts of information quality [10].
   After integrating the above definitions of information quality, we defined the information quality of a QM tool as the degree to which its data – stored or generated – are perceived by end-users as complete, current, relevant, reliable, understandable, and useful.
2) ***System quality*** refers to the system's functionalities and the user experience in interacting with the system to perform specific tasks.
   In the literature, system quality has been defined by Nelson et al. [10] and McKinney et al. [9]. Nelson et al. decomposed it into accessibility, reliability, response time, flexibility, and integration as quality aspects of the system's functionalities. McKinney et al. proposed another model in which system quality is broken down into accessibility, usability, and navigation of the system's functionalities.
   After analyzing both operationalizations, we defined the system quality of a QM tool as the degree to which it is likely to achieve acceptance and to which end-users perceived its functionalities and features as efficient, easy to navigate, relevant, reliable, and enjoyable. We also considered the quality of the visualizations as a part of the system quality.

3. Evaluation Design

3.1. Evaluation Goals and Questions

We planned a formative evaluation – i.e, an accompanying evaluation – to allow early reflection on the quality of the ProDebt prototype and contribute to its further development. We aimed at characterizing the information and system quality of the ProDebt prototype from the perspective of developers and managers in the context of ASD. We also aimed at identifying needs for improvements. In view of this, we defined he following evaluation questions:

Q1 *Information Quality* – What is the information quality of the ProDebt prototype?

Q2 *System Quality* – What is the system quality of the ProDebt prototype?

Q3 *Needs for improvements* – What needs to be improved to increase the information and system quality of the ProDebt prototype?

We specified the information and system quality of the ProDebt prototype based on the definitions introduced in Section 2. We also operationalized them by selecting the most reliable set of items for each quality aspect (cf. Table 1). Then we instantiated the selected items according to the purpose and functionality of the current ProDebt prototype. At the time of this evaluation, the ProDebt prototype provides practitioners with support for managing software quality, i.e., for: (1) evaluating the quality of a software system at one point in time, (2) analyzing the quality of a software system over time at different levels, (3) drilling down into potential causes of quality deficiencies. Moreover, it included data from at least one year, but not from the last development month. So, it was not possible to evaluate the currency, completeness, and reliability of its information. The information quality of the ProDebt prototype here refers to the degree to which end-users perceive the underlying quality model as understandable, relevant and useful for evaluating software quality (c.f. Table 2). The system quality of the ProDebt prototype refers to the degree to which it

Authorized licensed use limited to: Idaho State University. Downloaded on July 29,2020 at 03:31:47 UTC from IEEE Xplore. Restrictions apply.

TABLE 1. Quality Aspects mapped to Information Quality and System Quality

	Quality Aspect	**Definition**	**Reference: Reliability**^{a}
		Degree to which end-users perceive the...	
Information Quality	Completeness	...information as complete for providing enough breadth and depth for their tasks	[12]: 0.84, [11]: 0.84, [10]: >0.87
	Currency	...information as up-to-date for their tasks	[12]: 0.84, [11]: 0.88, [10]: 0.93
	Relevance	...information as applicable and helpful for their tasks	[11]: 0.94
	Reliability	...information as correct, accurate and trustworthy	[11]: 0.91, [9]: 0.97
	Understandability	...information as clear and understandable	[9]: 0.95
	Usefulness	...information as informative, valuable and useful for their tasks	[9]: 0.95
System Quality	Acceptance	...system as acceptable for their tasks	[13]: >0.75
	Efficiency	...system as efficient for their tasks	[14]: >0.88, [15]: 0.96
	Navigation	...system navigation as easy	[9]: 0.86
	Relevance	...system's functionalities, features, and overall usage as relevant for their tasks	[16]: 0.90 [13]: 0.83
	Reliability	...system as accurate, dependable, and consistent	[12]: 0.71 [16]: 0.9, [10]: >0.88
	Enjoyment	...experience using the system as satisfactory	[9]: 0.98, [10]: >0.9, [13]: 0.89
	Visualization	...system visualizations as useful	[17]: 0.95^b, [10]: >0.86

a. Cronbach's alpha
b. Composite reliability

is likely to achieve acceptance and to which its end-users perceive it as relevant, efficient, reliable, easy to navigate and enjoyable (c.f. Table 3).

3.2. Evaluation Design

We chose an embedded mixed methods design for our evaluation design. Mixed method design is a research methodology that *"involves collecting, analyzing, and integrating quantitative and qualitative research in a single study"* [18]. It is useful when combining quantitative and qualitative research approaches provides a better understanding of a phenomenon than each research approach alone. An embedded mixed method design is useful when one research approach (and related data set) provides support to explain the findings generated by the other. For example, using observations (qualitative approach) to better understand participants' productivity during a controlled experiment (quantitative approach).

In particular, we aimed at evaluating the quality of the ProDebt prototype using reliable test instruments in a controlled environment (quantitative approach). Performing the evaluation in a controlled environment aimed at increasing internal validity. Using test instruments was intended to increase measurement reliability and allow future comparison of the results of iterative evaluations of the ProDebt prototype. Moreover, we wanted to identify suggestions for improvement by gathering qualitative data using observations (qualitative approach). We did this to understand the quantitative evaluation of the ProDebt prototype and guide its further development.

Preparation. Before the evaluation, we selected one ASD project per company joining the study. For each selected project, we elicited the quality model and metrics for measuring and analyzing quality deficits as follows: First, we gathered the quality model and metrics by analyzing available documentation such as requirement specifications. Second, we complemented the quality model and metrics by performing semi-structured interviews with a sample of project members. The sample included developers, managers, the Scrum master and the product owner. Finally, we presented the quality model and the metrics to the sample of project members and asked them for feedback.

Based on each elicited quality model, we configured the ProDebt prototype and (if necessary) elicited the corresponding project data for a time period of not less than one year. Then, we invited the project team members to take part in the evaluation. We sent them an informed consent in advance. We also asked them to individually select three refactoring tasks in which they had been involved in the last six months of the corresponding project. This was intended to make it possible to analyze of the quality of the corresponding software system over time during the evaluation sessions.

Procedures and Instruments. For each participant, the evaluation session included four steps:

1) *Introduction.* We explained the study goals and procedures to the participant. Then the participant signed the informed consent and answered the demographic questionnaire. The demographic questionnaire included three questions regarding the participant's professional experience as well as his or her attitude towards QM and QM tools.
2) *Training.* We trained the participant in the ProDebt prototype. First, we explained the quality model and metrics and provided hin or her with a printed copy of them. Second, we introduced the ProDebt prototype using predefined examples to illustrate how to evaluate the overall software quality at one point in time and how to analyze it over time. We answered all questions asked by the participant regarding the ProDebt prototype. Finally, the participant answered a questionnaire including two questions regarding

Authorized licensed use limited to: Idaho State University. Downloaded on July 29,2020 at 03:31:47 UTC from IEEE Xplore. Restrictions apply.

TABLE 2. CONCEPTUALIZATION OF INFORMATION QUALITY OF THE PRODEBT PROTOTYPE

Information Quality *[Reference for selected items]*	**Definition** *Degree to which end-users perceive the information provided by the quality model as ...*	**Items** *(7-point rating scale from 1: strongly disagree to 7: strongly agree)*
Understandability [9]	...understandable for evaluating the quality a software system	The quality metrics are clear in meaning In general, they are understandable
Relevance [11]	...applicable and helpful for evaluating the quality of a software system	The quality metrics are: (1) useful to my work; (2) relevant to my work; (3) appropriate for my work; (4) applicable to my work
Usefulness [9]	...informative, valuable, and useful for evaluating the quality of a software system	The quality metrics are informative to my tasks The quality metrics are valuable for my tasks In general, they were useful for my tasks

TABLE 3. CONCEPTUALIZATION OF SYSTEM QUALITY OF THE PRODEBT PROTOTYPE

System Quality *[Reference for selected items]*	**Sub-Attribute**	**Definition** *Degree to which end-users perceive ...*	**Items** *(7-point rating scale from 1: strongly disagree to 7: strongly agree)*
Acceptance [13]	Perceived usefulness	...using the ProDebt prototype to enhance their performance	Using the ProDebt prototype improves my performance in my job Using the ProDebt prototype increases productivity in my job Using the ProDebt prototype enhances my effectiveness I find the ProDebt prototype to be useful in my job
	Perceived ease of use	...using the ProDebt prototype to be free of effort	My interaction with the ProDebt prototype is understandable My interaction with the ProDebt prototype is clear Interacting with the ProDebt prototype does not require a lot of my mental effort I find the ProDebt prototype to be easy to use I find it easy to get the ProDebt prototype to do what I want it to do
	Behavioral intention	...intend to use the ProDebt prototype	Assuming I had access to the ProDebt prototype, I intended to use it Given that I had access to the ProDebt prototype, I predict that I would use it I plan to use the ProDebt prototype
Relevance [9], [13]		...the functionalities, features, and usage of the ProDebt prototype as relevant	The ProDebt prototype has the functionality that I need to do my job The ProDebt prototype has the features required for my tasks In my job, usage of the ProDebt prototype is relevant
Efficiency [15]		...the ProDebt prototype as efficient in evaluating the software quality of a software system	It did not take too long to perform It does not require a lot of time It does not require a lot of thought I did not spend a lot of effort The ProDebt prototype speeds the evaluation of software quality
Reliability [9]		...the ProDebt prototype as accurate, dependable, and consistent	The information provided by the ProDebt prototype is trustworthy The information provided by the ProDebt prototype is accurate The information provided by the ProDebt prototype is credible In general, the information was reliable for assessing quality deficits of a given software system
Awareness		...the ProDebt prototype provides them with support to be aware of quality deficits	After using the ProDebt prototype ... ...I am up-to-date of current quality deficits ...I know the relative priority of current quality deficits ...I know which are the most critical quality deficits ...I am more aware of quality deficits
Visualization [10]	Format	...the information presented by the ProDebt prototype as well formatted	The information (e.g., in form of metrics, diagrams, and tables) provided by the ProDebt prototype are ... ...well formatted ...well laid out ...clearly presented on the screen
Visualization [17]	Diagrams and Tables	...the diagrams and tables presented by the ProDebt prototype as useful	The included diagrams and labels made it easy to do the tasks
Visualization [17]	Visual Clues	...the information presented by the ProDebt prototype as useful	The textual and visual clues in the ProDebt prototype helped me to do the tasks
Navigation [9]		...the ProDebt prototype as easy to navigate regarding the links to the needed information	The ProDebt prototype is easy to go back and forth between screens/pages The ProDebt prototype provides a few clicks to locate information In general, it is easy to navigate
Enjoyment [9]		...the experience of using the ProDebt prototype as satisfactory	After using the ProDebt prototype , I am ... ...[-3: very dissatisfied; 3: very satisfied] ...[-3: very displeased; 3: very pleased] ...[-3: very terrible; 3: very delighted] ...[-3: very frustrated; 3: very contented]

Authorized licensed use limited to: Idaho State University. Downloaded on July 29,2020 at 03:31:47 UTC from IEEE Xplore. Restrictions apply.

how well prepared he or she felt to use the ProDebt prototype.

3) *Execution.* We asked the participant to solve three tasks: (1) analyze the current quality of the software system, (2) identify quality deficits, and (3) analyze the impact on the software quality of at least one refactoring task in which she or he was involved. We encouraged the participant to think aloud and mention both positive and negative aspects of the ProDebt prototype. This served to get a better understanding of the participant's insights into the ProDebt prototype.
4) *Feedback.* After solving the assigned tasks, the participant answered a feedback questionnaire on the information quality and another one on the system quality of the ProDebt prototype (c.f. Table 2 and Table 3). Moreover, we asked the participants for their open feedback on the ProDebt prototype and the evaluation. We also asked the participant not to share information or insights regarding the evaluation session with other participants until the end of all evaluation sessions in the corresponding company.

An observer documented the progress of each evaluation session using a predefined protocol. The observer kept records of the participants' comments and questions on the ProDebt prototype and deviations from the plan. This was intended to understand the quantitative evaluation of the ProDebt prototype and to facilitate later analysis of possible threats to internal validity such as experimenter bias. All evaluation sessions were conducted by the same researcher and observer using predefined guidelines.

We scheduled each evaluation session for up to 60 minutes, taking into consideration the availability of eventual participants. We confirmed that experts and novices in QM and QM tools can solve the planned tasks using the ProDebt prototype in the given time by performing a pre-evaluation.

Directly after the evaluation at each company, we documented and summarized our perceptions regarding the evaluation and insights gathered. These notes were intended to allow the identification of possible confounding factors. We stored these notes separately from the participants' observational protocols and questionnaires.

After analyzing the evaluation results, we discussed them with representatives of the participating software companies and the developers of the ProDebt prototype.

3.3. Population and Sampling

The target population included developers and managers working in ASD. We drew an extreme case sampling, i.e., a purposive sampling focusing on opposed, unusual, or deviant cases. Extreme sampling is useful to get an in-depth understanding of a phenomenon and derive lessons learned to guide research and practice [19]. We defined two extreme cases: experienced versus inexperienced developers and managers regarding QM and QM tools.

Members from two German small and medium-size enterprises – namely, QAware and Insiders Technologies – took part in the evaluation. QAware develops information systems for several application domains. Insiders Technologies develops mainly document management solutions for the public, insurance, commercial, and finance sectors. We selected one project of QAware and one of Insiders Technologies for the evaluation of the ProDebt prototype:

1) *QAware project.* The selected project has been running since 2012 using Scrum. It focuses on the development of an enterprise search web application using Java, .Net, and Objective-C for the automotive and after-sales domains.The current development team includes 22 professionals working in sprints lasting between four and six weeks. The software quality has been managed using SonarQube. The current software release has approximately 150 thousand lines of code and 32 components.
2) *Insiders Technologies project.* The chosen project has been running since 2000. It focuses on the development of a software for processing, extracting, and classifying information from any kind of business correspondence for the insurance domain using C++. The releases have been developed using Scrum since 2009. The current Scrum team includes nine developers working in two-weeks sprints. The software quality has been managed ad-hoc without any tool support. The current software release has approximately one million lines of code and 50 components.

In total, six developers and two managers of QAware and two developers and one manager of Insiders Technologies participated in the evaluation. The remaining project team members did not participate because of time constraints.

3.4. Data Analysis

We carried out within-case and cross-case analyses [20]. That is, we first analyzed the quantitative and qualitative data within each company and then we compared, related, and integrated the results between companies.

Regarding quantitative analysis, we report descriptive statistics including the sample size (*N*), median (*Mdn*), minimum (*Min*), maximum (*Max*), frequencies, and valid percent. For seven-point rating scale data, we used the One-sample Wilcoxon signed-rank test to test for significant differences from the midpoint, i.e., H_0: Mdn(x) = 4 [21]. We interpreted scores above 4 as positive answer. We report the One-sample Wilcoxon signed-rank observed value (*Z*) and the significance level (*p*). We performed all statistical tests using IBM SPSS Statistics 19 (including IBM SPSS Exact Tests for analyzing small samples) and setting the confidence interval at the 95% confidence level. We selected the One-sample Wilcoxon signed-rank test because it is appropriate for testing small samples with N>=6 considering a confidence interval at the 95% [21].

Regarding qualitative analysis, we used thematic analysis [22] to analyze the participants' feedback on the ProDebt prototype. We derived themes – i.e., suggestions for improvement – inductively by coding and interpreting all ob-

Authorized licensed use limited to: Idaho State University. Downloaded on July 29,2020 at 03:31:47 UTC from IEEE Xplore. Restrictions apply.

servation protocols. We counted as themes issues explicitly mentioned by the participants during the evaluation.

The raw data and the evaluation package are stored in a Fraunhofer IESE repository. The participants only authorized the evaluation team to share aggregated and anonymized results with other researchers or practitioners. Thus, raw data cannot be shared. Further analysis can be shared under the authorization of QAware and Insiders Technologies. To access the detailed evaluation design, instruments, and materials, please contact Fraunhofer IESE.

3.5. Execution

The evaluation of the ProDebt prototype was conducted in November 2016 onsite at QAware and Insiders Technologies. The evaluation sessions with the eight QAware participants were scheduled over two consecutive days and the one with the three Insiders Technologies participants on one day. The evaluation sessions lasted 65 minutes on average *(Min = 60 minutes, Max = 75 minutes)*. The evaluation was performed in German. Given that all questionnaires were in English, we provided German translations (previously reviewed by an certified German-English translator) whenever further explanations were needed. The questionnaires were kept in English because at the time of this evaluation, no validated translation of the test instruments existed. All participants were knowledgeable in English.

Even though we performed the evaluation according to the above design and procedures, we identified some deviations. First, the demographic questionnaire was carried out verbally to increase the speed of answering these questions instead of in paper form as planned. Still, one participant took 10 minutes longer to answer the questionnaires. Second, the evaluations carried out at Insiders Technologies lasted 15 minutes longer than planned. This decision was made because the participants were less experienced in software quality metrics and needed more time to comprehend the quality model. Third, despite previous reviews of the questionnaires with experts in empirical research, we found out during the first evaluation that the questionnaire regarding the system quality was still too long. Thus, six questions with low priority were removed in the remaining evaluations. These questions were either already captured by other items or just specific for the ProDebt project. For example, we removed the questions from [9] related to behavioral intention because they were captured in [13]. Tables 2 and 3 contain the final questions. Fourth, the questions about reliability and awareness were only included in the evaluations at Insiders Technologies because its representative considered it necessary to gain an in-depth understanding of their expectations on the ProDebt prototype. Finally, the contact person of QAware within the ProDebt project attended all evaluation sessions and (if required) provided the list of refactoring tasks in the last six months or explanations on the quality metrics. The contact person at QAware has a lot of experience in empirical studies and knew the evaluation guidelines. He is neither a direct superior nor a supervisor of the participants. At the beginning of each evaluation, we made clear that he was there just to provide support and will observe confidentiality in all matters concerning the evaluation. After analyzing the observational protocols and the raw data, we did not find any indication that his presence influenced the participants' behavior or answers.

4. Results

In this section we present only the aggregated results of the evaluation of the ProDebt prototype. During the within and cross-case analysis, we observed that the participants' perceptions were consistent among different roles and companies. We discussed the results about the information and system quality of the ProDebt prototype together with the identified needs for improvement, because this provides a better understanding of the quantitative findings.

4.1. Sample Description

In total, six developers and two managers of QAware and two developers and one manager of Insiders Technologies agreed to participate in the evaluation. Out of the eleven participants, the majority had more than four years of experience in software development at the time of this evaluation *(N = 11, Mdn = 4.5 , Min = 1, Max = 10)*. They consider (rating scale from 1: strongly disagree to 7: strongly agree): QM as important for the project (*N = 11, Mdn = 7, p < .001*) and tool support for managing software quality as important for their job (*N = 11, Mdn = 6, p < .001*). After the training, all participants felt confident and well prepared for using the ProDebt tool to analyze quality deficits.

4.2. Information Quality of the ProDebt Prototype

Table 4 shows the results regarding the information quality of the ProDebt prototype. The participants claimed the quality metrics included in the ProDebt prototype are understandable (*Mdn = 5, p = .007*), relevant (*Mdn = 6, p = .004*) and useful (*Mdn = 5, p = .03*). Most participants used metrics related to lines of codes (*N = 10*), branch coverage (*N = 7*), violations (*N = 7*), nesting (*N = 7*), and complexity (*N = 6*) during the analysis of quality deficits.

4.3. System Quality of the ProDebt Prototype

The participants tried all functionalities available in the ProDebt prototype. Table 5 shows the results regarding the system quality of the ProDebt prototype.

The participants claimed the ProDebt prototype is easy to use (*Mdn = 5, p = .01*). Almost all participants found it very positive that only a short training was sufficient to learn how to use the ProDebt prototype (*N = 10*). They also considered the visualizations of the ProDebt prototype to be average (e.g., Visual Clues: *Mdn = 4, p = .45*) and navigation options to be good (*Mdn = 6, p = .01*).

However, the participants perceived the ProDebt prototype as useless (*Mdn = 3, p = .05*) and do not intend to use

Authorized licensed use limited to: Idaho State University. Downloaded on July 29,2020 at 03:31:47 UTC from IEEE Xplore. Restrictions apply.

TABLE 4. Information Quality of the ProDebt Prototype

Information Quality	N	Mdn[a] (Min - Max)	Z	p
Understandability	11	5 (4 - 7)	2.683	0.007
Relevance	11	6 (4 - 6.5)	2.850	0.004
Usefulness	11	5 (2 - 6)	2.240	0.03

a. Rating scale from 1: strongly disagree to 7: strongly agree

it (*Mdn = 2, p = .03*). They found the time series analysis and the option to analyze quality metrics from the system to the file level very positive and useful (*N = 10*). Nevertheless, they argued there is a need for further functionalities to make the ProDebt prototype usable in practice.

The results regarding the ProDebt prototype's relevance (*Mdn = 2, p = .11*), efficiency (*Mdn = 5, p = .62*) and enjoyment are neutral (*Mdn = 0, p > .05*) and not statistically significant (cf.Table 5). The participants had contradictory opinions regarding these quality aspects. We were not able to identify any variable that could explain these results.

We omit additional analysis regarding the reliability and awareness of the ProDebt prototype because the number of subject who answered the related questions is too small to form an anonymous group.

4.4. Suggestions for Improvement

Improvement of Information Quality. Regarding the quality model, all participants claimed that the quality metrics are useful. They stated the need for adding metrics regarding dead code, unreachable code, coupling, cohesion, and ignored tests (*N = 5*). Half of them also commented on the need to add an alarm system to the ProDebt prototype (*N = 5*) and further project- or person-specific configuration options (*N = 5*) (cf. Table 6). For example, to include functionalities to allow the selection of metrics as well as the definition and tailoring of threshold values for each metric. More than half of the participants preferred to have an alarm system over quality indicators (*N = 7*). They argued that quality indicators are often neither transparent nor comprehensive or easy to interpret.

Improvement of System Quality. The participants emphasized the need to link the measurement results to the source code and other information sources, such as user stories and commit data (*N = 11*). One participant said: *"It would be really cool if I could go to the source code"*. Another participant commented several times: *"I am itching to see the comments in the source code"*. The participants also commented the need for easily accessible information regarding the meanings of the quality metrics and guidelines for interpreting them (*N = 11*). Other needs for improvement mentioned by the participants include: (1) filtering of irrelevant files such as generated code and third-party code (*N = 10*), (2) search options for specific files or metrics (*N = 7*), (3) generation of hotspot lists (*N = 6*), and (4) refinement of the analysis up to the method level (*N = 6*). For instance, one participant emphasized: *"For me, a blacklist is missing, so that I could ignore generated code for example. This distorts the analysis"*. Finally, five out of eleven participants stated that files being restructured over time are not taken into account in the time series analysis. They believed that this issue falsified the analysis results. However, they argued this is a common issue in other QM tools as well.

4.5. Threats to Validity

Construct Validity. To avoid inappropriate measurement, we used reliable test instruments to measure information and system quality. To avoid mono-method bias, we compared the data collected using the quantitative and qualitative methods. That is, we used method-triangulation. There is a risk of mono-operation bias because we evaluated a single QM tool, i.e., the ProDebt prototype. Further comparisons of the ProDebt prototype and other QM tools are necessary to get more reliable and comparable results.

Internal Validity. To ensure treatment reliability, the responsible researcher performed all evaluations following the same procedures and using the same instruments and materials in a controlled environment. Another researcher acted as an observer to monitor and document any influence of the responsible researcher. A contact person of QAware also observed the evaluation sessions at QAware. He knew the evaluation procedures and strictly followed them. To avoid experimenter bias, all evaluation sessions were conducted by researchers who have not been involved in the development of the ProDebt prototype. Neither QAware nor Insiders Technologies have been involved in the development of the ProDebt prototype. To avoid diffusion or imitation of treatments, the participants committed to not sharing experiences in using the ProDebt prototype within their company until the end of the evaluation sessions. After analyzing the observation protocols and the raw data, we did not found any indication of experimenter or diffusion bias.

External Validity. To increase the transferability and representativeness of the results, we used extreme case sampling. All findings were discussed with representatives of the participants of both software companies involved in the evaluation. The use of extreme case sampling served to avoid bias because of the interaction of selection and treatment as well as setting and treatment. That is, the treatment (i.e., ProDebt prototype) only works with a particular type of participants in a specific setting. Because of the unbalanced sample in the extreme cases, there is still a risk that the results might be biased. Thus, the results are more likely to be representative of software practitioners with experience in using QM tools. Even though the participants analyzed the quality of a real software system, they did not perform QM under real project conditions. We conclude that further non-exact replications of this mixed-method study are needed to generalize the results regarding the ProDebt prototype.

Conclusion Validity. To increase trustworthiness in the analysis results, two researchers performed all analyses independent of each other. It is important to remark that the

Authorized licensed use limited to: Idaho State University. Downloaded on July 29,2020 at 03:31:47 UTC from IEEE Xplore. Restrictions apply.

TABLE 5. System Quality of the ProDebt Prototype

System Quality	Sub-attribute	N	Mdn [a] (Min - Max)	Z	p
Acceptance	Usefulness	11	3 (1.5 - 5.5)	-1.965	0.05
	Ease of use	11	5 (3 - 7)	2.511	0.01
	Behavioral intention	11	2 (1 - 6)	-2.118	0.03
Relevance		11	2 (2 - 6)	-1.609	0.11
Efficiency	Perception	11	5 (1 - 6.5)	0.492	0.62
	Speeds up work	11	3 (1 - 7)	-0.906	0.37
Visualization	Format	11	5 (3 - 7)	2.066	0.04
	Visual Clues	10	4 (2 - 6)	-0.750	0.45
	Diagrams and Tables	11	6 (3 - 7)	2.584	0.01
Navigation		11	6 (3 - 7)	2.461	0.014
Enjoyment[b]	dissatisfied - satisfied	11	0 (-2 - 2)	-0.250	0.80
	displeased - pleased	11	0 (-1 - 2)	1.000	0.34
	terrible - delighted	11	0 (-3 - 2)	0.687	0.49
	frustrated - contented	11	0 (-3 - 2)	-0.647	0.52

a. Rating scale from 1: strongly disagree to 7: strongly agree

b. Rating scale from -3: very negative to 3: very positive

TABLE 6. Needs for Improvement regarding the ProDebt Tool

Quality Aspect	Needs for improvements	N
Information Quality	Missing metrics regarding dead code, unreachable code, coupling, cohesion, and ignored tests	5
	Need for an alarm system	5
	Need for further project- or person-specific configuration (e.g., metric selection)	5
System Quality	Need for further functionality:	
	– Link to source code and other information sources	11
	– Need for help regarding metric meaning, unit, interpretation, and granularity	11
	– Need for simultaneous display of time series of preselected metrics	7
	– Refinement up to the method level	6
	– Hotspot List	6
	Need for further interaction options:	
	– Filtering of irrelevant files (e.g., generated files, third parties, etc.)	10
	– Search for specific metric or file	7
	– Pre-selection of metrics	7
	– Zoom in on time series for preselected time periods	5
	Changes in the scales make interpretation difficult and lead to misunderstandings	6
	Too many clicks are needed to explore the software structure, even if the subsystems contain no measurements	8

results show low statistical power because of the small sample size. Thus, further non-exact replications of this mixed-method study are needed to achieve statistically significant results and to generalize them.

5. Lessons Learned

We performed a formative evaluation of a single QM tool, namely the ProDebt prototype, in a setting close to reality. We observed that the proposed evaluation design is feasible to be used with limited resources regarding the required sample, time, number of researchers, and materials. Therefore, the cost for industrial partners is kept low and hence the practicability for industrial settings increases. This turned out to be a decisive factor to motivate the participation of practitioners.

Designing an empirical study that demands few resources and still provides valid findings was possible because of several reasons. First, selecting the quality aspects of interest for the participating companies provided flexibility to scope the evaluation and shorten the duration of the evaluation sessions. Second, using real project data and daily tasks – that is, analyzing quality deficits and the impact of refactoring tasks in the own code – as part of the treatment allowed us to reduce the duration of the evaluation sessions even more. Third, using reliable test instruments to measure the quality of a QM tool and using triangulation of quantitative and qualitative methods increased the validity of our findings and helped us to get in-depth insights regarding the need for improvements. Finally, the consideration of two extreme cases allowed us to increase the findings' representativeness.

The proposed formative evaluation aimed at providing evidence to understand the quality of a single ProDebt prototype and identify needs for improvements. The developers of the ProDebt prototype considered the findings of this formative evaluation valuable for its further implementation. Although we only had access to a small sample, the triangulation of quantitative and qualitative analysis methods provided useful information to (1) evaluate the current quality of the ProDebt prototype, (2) collect precise feedback for improvements and derive necessary actions, and (3) better understand the findings and their implications.

The findings of this formative evaluation also provided support for dealing with conflicts of interest between the

Authorized licensed use limited to: Idaho State University. Downloaded on July 29,2020 at 03:31:47 UTC from IEEE Xplore. Restrictions apply.

developers and end-users in the ProDebt research project. After this evaluation, the developers of the ProDebt prototype prioritized the identified suggestions for improvement with representatives of QAware and Insiders Technologies and began with their implementation. The findings allow the developers of the ProDebt prototype to focus on the aspects that are expected to provide the most value to the end-users.

6. Conclusions and Future Work

This study contributes to the systematic evaluation of the quality of a single QM tool, including support for identifying suggestions for improvements. We identified operationalizations for information and system quality aspects to use reliable test instruments for such evaluations. Moreover, we reported a formative evaluation that can be used to evaluate the quality of a single QM tool including different aspects and perspectives such as the relevance of the provided information or the acceptance of the tool by end-users.

We evaluated the ProDebt prototype with eleven potential end-users. The participants rated the information of this prototype as understandable and relevant for analyzing quality deficits and the impact of refactoring tasks. Though the prototype is considered easy to use, there are needs for improvement regarding the system quality. These findings systematically provide input for the developers of the ProDebt prototype to effectively improve it. Further replications of this evaluation are needed to support the further development of the ProDebt prototype and allow comparison and further analyses of its development progress.

We found that our evaluation design was of practical value for evaluating the ProDebt prototype considering several limitations (simple design, small sample and low involvement by the companies). However, reducing the number of questions would speed up the execution time and may increase the willingness to participate in such evaluations. Getting access to additional participants would even increase the trustworthiness of the findings.

Evaluating other QM tools according to the presented evaluation design would allow comparing them, e.g., by performing subsequent quantitative syntheses. This would support companies in selecting a QM tool. Further future work would be to consider additional operationalizations of quality aspects or to validate the selection of those we used ones through new findings.

Acknowledgments

We would like to thank the members of QAware and Insiders Technologies who participated in this empirical evaluation. Parts of this work were funded by the German Ministry of Education and Research under Research grant N 01IS15008A.

References

[1] A. Hunt and D. Thomas, "Zero-tolerance construction," *IEEE Software*, vol. 19, pp. 100–102, 2002.

[2] I. Inayat, S. S. Salim, S. Marczak, M. Daneva, and S. Shamshirband, "A systematic literature review on agile requirements engineering practices and challenges," *Computers in Human Behavior*, vol. 51, Part B, pp. 915 – 929, 2015.

[3] L. Guzman, M. Oriol, P. Rodriguez, X. Franch, A. Jedlitschka, and M. Oivo, "How can quality-awareness support rapid software development? a vision paper," in *Proceedings of the 23rd Working Conference on Requirements Engineering Foundation for Software Quality*, ser. REFSQ 2017. Berlin, Heidelberg: Springer-Verlag, 2017, pp. 1–7.

[4] S. Wagner, *Software product quality control*. Springer Verlag, 2013.

[5] S. Wagner, A. Goeb, L. Heinemann, M. Kls, C. Lampasona, K. Lochmann, A. Mayr, R. Plsch, A. Seidl, J. Streit, and A. Trendowicz, "Operationalised product quality models and assessment: The quamoco approach," *Information and Software Technology*, vol. 62, pp. 101 – 123, 2015.

[6] 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, 2015.

[7] A. M. Vollmer, "Empirical evaluation of the prodebt approach," Master's thesis, Technical University of Kaiserslautern, 2016.

[8] B. Kitchenham, "Guidelines for performing systematic literature reviews in software engineering," Keele University and University of Durham, Tech. Rep., 2007, ver. 2.3 EBSE-2007-01.

[9] V. McKinney, K. Yoon, and F. M. Zahedi, "The measurement of web-customer satisfaction: An expectation and disconfirmation approach," *Information Systems Research*, vol. 13, no. 3, pp. 296–315, 2002.

[10] R. R. Nelson, P. A. Todd, and B. H. Wixom, "Antecedents of information and system quality: An empirical examination within the context of data warehousing," *Journal of Management Information Systems*, vol. 21, no. 4, pp. 199–235, 2005.

[11] Y. W. Lee and D. M. Strong, "Knowing why about data processes and data quality," *Journal of Management Information Systems*, vol. 20, no. 3, pp. 13–39, 2003.

[12] D. L. Goodhue and R. L. Thompson, "Task technology fit and individual performance," *MIS Quarterly*, vol. 19, no. 2, pp. 213–236, 1995.

[13] V. Venkatesh and H. Bala, "Technology acceptance model 3 and a research agenda on interventions," *Decision Sciences*, vol. 39, no. 2, pp. 273–315, 2008.

[14] B. Vandenbosch and M. J. Ginzberg, "Lotus notes and collaboration: Plus a change..." *Journal of Management Information Systems*, vol. 13, no. 3, pp. 65–81, 1996.

[15] P. Xu and B. Ramesh, "Impact of knowledge support on the performance of software process tailoring," *Journal of Management Information Systems*, vol. 25, no. 3, pp. 277–314, 2008.

[16] D. H. Mcknight, M. Carter, J. B. Thatcher, and P. F. Clay, "Trust in a specific technology: An investigation of its components and measures," *ACM Trans. Manage. Inf. Syst.*, vol. 2, no. 2, pp. 12:1–12:25, Jul. 2011.

[17] L. Goel, N. A. Johnson, I. Junglas, and B. Ives, "From space to place: Predicting users' intentions to return to virtual worlds," *MIS quarterly*, vol. 35, no. 3, pp. 749–772, 2011.

[18] J. Cresswell and V. L. P. Clark, *Designing and conducting mixed-methods research*, 2nd ed. London: SAGE Publications, 2011.

[19] J. Daniel, *Sampling essential. Practical guidelines for making sampling choices*. SAGE Publications, 2012.

[20] M. Miles and M. Huberman, *Qualitative data analysis*, 2nd ed. London: Sage Publications, 1994.

[21] R. Woolson, *Wilcoxon Signed-rank test*. John Wiley and Sons, Inc, 2008, pp. 1–3.

[22] V. Braun and V. Clark, "Using thematic analysis in psychology," *Qualitative research in psychology*, vol. 3, pp. 77–101, 2016.

Authorized licensed use limited to: Idaho State University. Downloaded on July 29,2020 at 03:31:47 UTC from IEEE Xplore. Restrictions apply.