Developing a Quality Report for Software Maintainability Assessment: An Exploratory Survey

Pascal Giessler
and Michael Gebhart

iteratec GmbH
Stuttgart, Germany
Email: pascal.giessler@iteratec.de,
Email: michael.gebhart@iteratec.de

Manuel Gerster, Roland Steinegger
and Sebastian Abeck

Cooperation & Management
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany
Email: manuel.gerster@student.kit.edu,
Email: roland.steinegger@kit.edu,
Email: sebastian.abeck@kit.edu

***Abstract*—Maintainability can be a key factor concerning the success of a software product, since the majority of software life cycle costs is spent on maintenance. Therefore, there is a deep interest in analyzing and assessing the maintainability of a software product with the objective of identifying the need for action and subsequently minimizing maintenance expenses. Often software quality metrics are used to analyze the influencing factors of maintainability while an expert uses their results for the assessment. However, these metrics are distributed across several tools, dashboards, and literature. Moreover, there are further quality indicators for analyzing the maintainability that cannot be evaluated automatically by means of metrics. Hence, we aim to develop a quality report containing well-known quality metrics and further quality indicators that allows an expert to assess the system under review regarding its maintainability. For this reason, we conducted an exploratory survey in the area of research and industry to get the essential ingredients of such a quality report. In this paper, we present the survey and its outcomes. The survey shows potential ingredients of a quality report, i.e., metrics and quality indicators, which can be measured not only automatically but also manually.**

Keywords–maintainability assessment; software quality; quality report; quality analysis; quality indicators.

I. INTRODUCTION

Nowadays, software products can be seen as ubiquitous components in our daily life. Each software product is designed to satisfy one or more business or user needs. But, these needs may change over time due to several influencing factors, such as changing market conditions or customer behavior. As a result, software modifications are required to support these new needs [1]. From an economic point of view, these modifications should be performed fast with low costs, due to the fact that the majority of software life cycle costs (LCC) is spent on maintenance and not on development [1] [2] [3].

That is why it is important to develop and design software products with maintainability in mind. But, it is unclear how to analyze and assess the maintainability of a software product in order to derive actions for improvement. There is no uniform and agreed set of quality metrics or common quality indicators for maintainability, since they are distributed across several tools, dashboards, and literature. A quality indicator gives us a hint regarding the manifestation of a given quality aspect, such as the maintainability as part of the ISO/IEC 25010:2011 [4]. Therefore, we have to identify quality indicators and collect them in a so-called quality report. On the basis of this report, an expert should be able to assess the maintainability of a software product. Here, an expert is characterized by technical and domain knowledge of the software product.

For developing a quality report regarding software maintainability assessment, we conducted an exploratory survey across several institutions in research and industry. This survey should reveal essential ingredients of such a quality report. The results enabled us to design a quality report for a specific software product family in the area of the SmartCampus ecosystem [5]. The SmartCampus is a service-oriented system that provides functionality for students, guests, and members of a university to support their daily life. For instance, the CompetenceService as part of the SmartCampus system captures competences of students and offers a semantic search to get suitable candidates for projects in the area of information technology [6]. Besides the mentioned results, we have also gained insight into the importance of quality assessment and software maintainability for different software development project members. We think that our results can be used by experts in order to build their own quality reports for maintainability assessment, since we cannot give a universal quality report due to a missing uniform set of quality indicators.

The paper is structured as follows: In Section II, we lay the foundation for the upcoming sections by defining important terms in the area of software quality assessment regarding software maintainability. Afterwards, we present the related work in Section III to give an overview of the state of the art in this research field. By providing the methodology of this paper in Section IV, we illustrate the design of the study, as well as an overview of its goals and underlying research questions. The results of the conducted study are shown in Section V, followed by the threats to its validity in Section VI. By evaluating the results in Section VII, we answer our research questions and draw conclusions. Finally, in Section VIII we conclude with a summary and give an outlook on further work in this research field.

II. FOUNDATION

According to the standard ISO/IEC 25000:2005, software quality is the "capability of software product to satisfy stated and implied needs when used under specified conditions” [7]. Software quality assessment is the systematic examination of the extent to which a software product is capable of satisfying these needs [7]. As depicted in Figure 1, software quality can be decomposed into several quality characteristics, sub-characteristics, and attributes [4]. These attributes can again be broken down into quality indicators and measurable quality metrics [8], which are formalized quality indicators [9].

Based on the definition of the term process quality indicator in ISO/IEC 33001:2015 [10], a software quality indicator is an assessment indicator that supports the judgment of software quality characteristics. According to the standard IEEE 1061-1998, a quality metric is a “function whose inputs are software data and whose output is a single numerical value that can be interpreted as the degree to which software possesses a given attribute that affects its quality” [11].

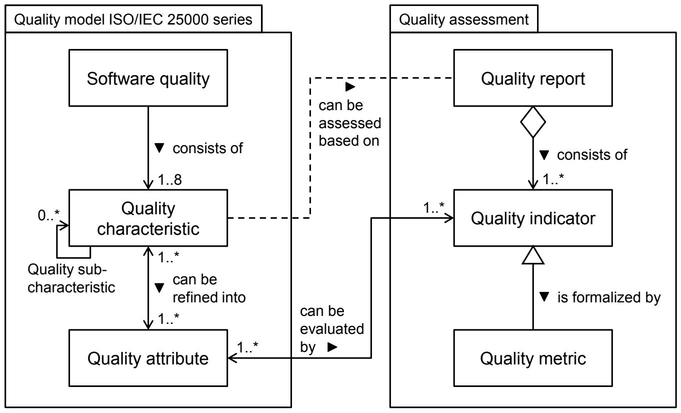

Figure 1. Meta model for software product quality [7] and quality assessment.

In our work, we focus on the quality characteristic maintainability. ISO/IEC 25010:2011 defines maintainability as the “degree of effectiveness and efficiency with which a product or system can be modified by the intended maintainers” [4].

A so-called quality report comprises software quality metrics and quality indicators in a comprehensive artifact. On the basis of these quality metrics and indicators, an expert should be able to assess the maintainability of a software product under investigation.

III. Related Work

This section summarizes several papers in the context of software quality assessment and the usage of quality reports.

In Ogasawara et al. [12], an approach to software quality management is introduced and a quality report for software products is presented. This report is based on results which are measured regularly and automatically by a software quality evaluation tool, such as the number of loops or comments. However, the authors do not take a closer look at the quality report. Especially, they do not go into detail about the structure, properties, and ingredients of the quality report. Instead, they show a simple sample quality report for reviewing software product quality containing quality metrics, which can be automatically measured and easily understood.

Steidl et al. [13] present a quality control process that combines quality metrics and manual action: Metrics constitute the basis but software experts have to interpret the metric results within their context. This manual action is based on a quality report. Although we pursue a similar idea, the definition of the term quality report given by the authors differs from our definition. According to the authors, a quality report contains the interpretation of the current analysis results, as well as manual code reviews. As opposed to this, we define a quality report as an artifact comprising software quality metrics and quality indicators, which provides the basis for the interpretation. Moreover, the authors do not describe concrete properties and ingredients of the quality report.

Annex E of ISO/IEC 25040:2011 [14] gives guidance on the structure and contents of an evaluation report for software product quality. However, the structure and contents are only described on an abstract level. Especially, no concrete quality metrics are listed.

As opposed to this, Riaz et al. [15] summarize 15 studies regarding software maintainability prediction and metrics. Table 6 of their paper comprises 45 successful software maintainability metrics gathered at source code level. For example, it contains most of the metrics presented in the well-known Chidamber & Kemerer (CK) metrics suite [16]. Thus, the authors provide a set of potential ingredients of the quality report. However, they present only an unstructured set of metrics, which they extracted from several papers. This loose collection focuses on size, complexity, and coupling, ignoring metrics related to other aspects like testing, static bug detection, and compliance with conventions. Moreover, they do not consider metrics that have to be analyzed manually. Although this set is a good starting point for our work, it is unclear whether these metrics are adequate ingredients of a quality report for software maintainability assessment in research and industry.

Altogether, we identified three papers that show rough information about quality reports and do not go into detail, especially regarding the properties and ingredients of the reports. In addition, we identified one paper that presents an unstructured set of metrics and thus potential ingredients of the quality report, but does not point out whether these metrics are adequate ingredients of the report.

IV. Methodology

This section presents the methodology applied in the study. It outlines its goals and research questions, the design of the study and the study population.

A. Goals and Research Questions

We conducted this study in order to determine the importance of both quality assessment and maintainability in research and industry. Moreover, we wanted to identify properties and ingredients of a quality report for the purpose of software maintainability assessment. These research goals led to the following four research questions:

RQ1: *Is quality assessment considered to be important in research and industry?* There are miscellaneous techniques supporting the development of software products with high quality. In order to identify the status quo regarding software quality, a quality assessment can be conducted. The survey should show whether a quality assessment is actually considered to be important for software development project members in research and industry.

RQ2: *Is maintainability considered to be important in research and industry?* Several publications underline the importance of maintainability since a huge amount of the total software LCC can be spent on maintenance. The survey should reveal whether maintainability is actually considered to be important for software development project members in research and industry.

RQ3: *Which information should be part of a quality report for the purpose of software maintainability assessment?* The assessment of software maintainability is often based on software quality metrics. However, these metrics are distributed across several tools, dashboards, and literature. Moreover, there are further quality indicators for conducting an assessment. In order to develop a comprehensive quality report, which can be used to assess the maintainability of software, relevant quality metrics and indicators for maintainability assessment have to be identified.

RQ4: *How important are the given quality report properties?* Considering requirements engineering, software requirements should possess several properties, such as the traceability [17]. As with software requirements, a quality report should also possess miscellaneous properties. Since there are some properties, which we consider to be relevant to the quality report, the survey should reveal the importance of each property and identify further properties as appropriate.

B. Study Design

The design of the conducted study comprised three phases. In the initial phase, we planned and prepared the study. As part of this, we identified a free web survey tool, *Umfrage Online* [18], for conducting the survey. We settled on this tool, since it is free, easy-to-use and not subject to restrictions concerning the number of questions and answers.

In the second phase, we developed a checklist regarding the structure and content of a survey based on specialist literature (e.g., [19] [20] [21]). In accordance with this checklist, we created a rough sketch for the survey that contains 16 questions (12 open questions, 3 closed questions and 1 partially closed question). Thanks to feedback from researchers, we revised the survey especially by reducing the number of open questions, since open questions are prone to reduce the response rate [22]. Furthermore, we added some demographic questions. The pretest version of the survey consisted of 23 different questions (8 open questions, 12 closed questions and 3 partially closed questions).

In the third phase, we conducted a field pretest with chosen target subjects. Based on the outcomes of the pretest, there were only minor changes to the survey mainly affecting the wording of the questions and answers.

The final version of the survey consists of 23 optional questions (8 open questions, 12 closed questions and 3 partially closed questions) and is available at [23] in German.

C. Study Population

The population of the study comprises software development project members, such as software developers, software architects or research assistants, from various companies and research organizations.

The link to the online survey was sent to chosen members of several companies and research organizations, who distributed the link within their institution. These members and consequently the recipients of the link were selected using convenience sampling. Since we do not know how many people received the survey link, the initial sample and thus the response rate cannot be determined. In total, 113 people answered the survey including 32 respondents ($\approx 28\%$) who did not complete it. In average, the 81 respondents needed 14 minutes (adjusted, standard deviation: 13 minutes) respectively to complete the survey. For survey evaluation, both the respondents who did and did not complete the survey are taken into account. The survey was written in German and was sent to companies and research organizations in Germany; therefore the results may show a German attitude towards software maintainability assessment. Most of the 66 respondents who stated their occupation work as software developers ($\approx 23\%$), followed by software architects ($\approx 20\%$), research assistants ($\approx 18\%$), and project or group leaders ($\approx 12\%$). Approximately 6% of the survey respondents are students and about 21% are occupied otherwise.

As depicted in Figure 2, the respondents work at companies and research organizations with highly diverse size.

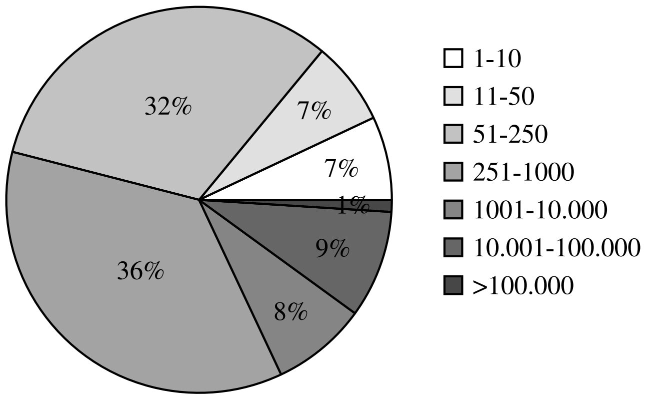

Figure 2. Size of companies and research organizations of survey respondents by the number of employees (n = 77).

Figure 3 shows that the survey covers software development project members working in the domain of software engineering for less than one year up to more than ten years, whereby the latter was stated by more than 30% of the respondents who answered the corresponding question.

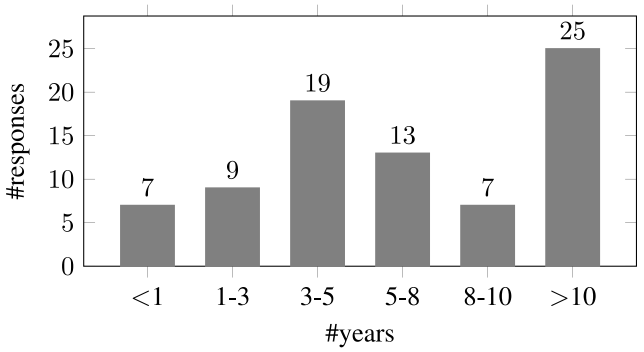

Figure 3. Number of years that respondents are working in the domain of software engineering (n = 80).

Furthermore, nearly 30% of the 81 respondents who answered the corresponding question stated that they have already participated in a software audit. According to the standard IEEE 1028-2008, an audit is an "independent examination of a software product [...] to assess compliance with specifications, standards, contractual agreements, or other criteria" [24]. About 30% of the 96 respondents who stated whether they have already read or created a quality report answered in the affirmative. If we separate the answers from research and industry, about 71% of the respondents from industry have already participated in a software audit and 63% have already read or created a quality report, compared with about 12% and 17%, respectively, in research. Due to the great amount of respondents from industry who are familiar with software audits and quality reports, and thus software quality assessment, we distinguish between answers from research and industry in the following sections.

V. Results

This section outlines the outcomes of the online survey. By reason of space limitations, we only present the survey questions which are most relevant for our research questions.

Question 1: How important is the quality assessment of already existing software for you? The first question should show the importance of quality assessment for our respondents. This question explicitly refers to the quality assessment of already existing software, e.g., in the context of a software audit. Since we aim to develop a quality report for software quality assessment, we should first determine whether there is a demand for assessing software quality. The participants of the survey could answer this question on a five-point ordinal scale: "very important", "important", "partly important", "less important", and "unimportant". Figure 4 shows the frequency distribution of all answers given by 108 respondents.

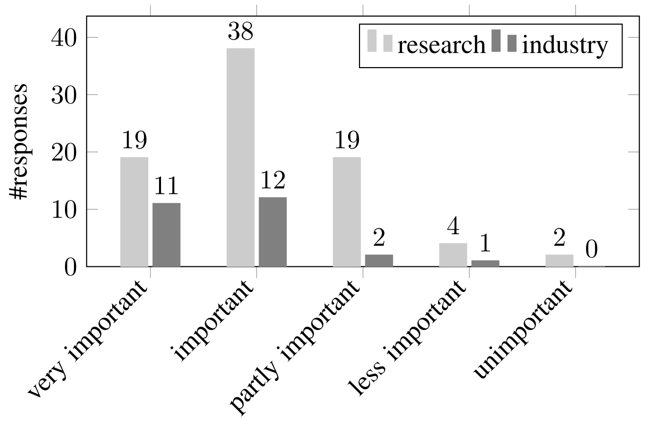

Figure 4. Importance of software quality assessment, absolute (n = 108).

The answers are coded with a scale ranging from 1 to 5, where 1 codes "very important" and 5 codes "unimportant". As depicted in Figure 4, the large majority of the respondents answered with "very important" or "important", namely almost 75%. Calculating some statistical numbers based on the coding lead to the following results: arithmetic mean = 2.06, median = 2, standard deviation = 0.91. As the median indicates, most of the respondents selected the answers "very important" or "important".

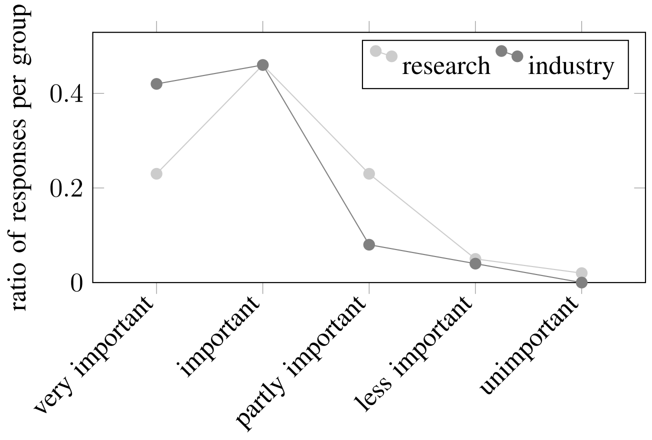

Figure 5. Importance of software quality assessment, normalized (n = 108).

Figure 5 shows that there is a difference between respondents from research and industry. More than 42% of the respondents from industry answered with "very important" compared to about 23% of the respondents from research. This result suggests a more distinct quality awareness in the industry. Nevertheless, the responses to this question show that software quality assessment is important for respondents from both research and industry. Hence, this question reinforces our decision to develop a quality report that aims to support and simplify software quality assessment.

Question 2: How important is software maintainability for you? The second question should reveal the importance of maintainability for our respondents. Due to the fact that the quality report should focus on maintainability initially, the importance of maintainability is crucial to us. The scale of this question and the coding correspond to those in the first question. Figure 6 shows the frequency distribution of all answers given by 91 respondents.

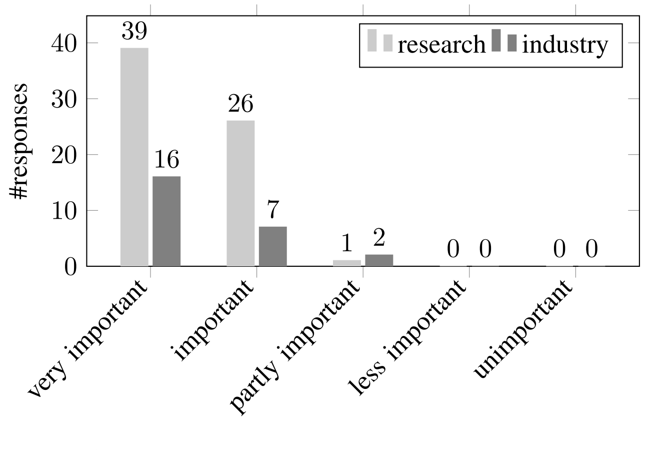

Figure 6. Importance of software maintainability, absolute (n = 91).

As depicted in Figure 6, not a single respondent answered with "less important" or "unimportant". Instead, approximately 97% of the respondents selected the answers "very important" or "important". Quantifying the answers based on the coding lead to the following results: arithmetic mean = 1.43, median = 1, standard deviation = 0.56.

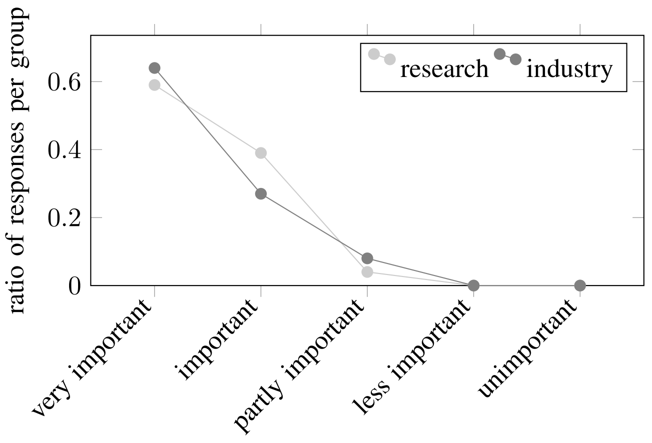

Figure 7. Importance of software maintainability, normalized (n = 91).

In contrast to the first question, Figure 7 shows that there is no significant difference between the answers given by the respondents from research and industry. The responses to this question reveal that maintainability is very important for both groups. Therefore, this question reinforces our decision that the quality report should focus upon maintainability initially.

Question 3: Which information should a quality report provide for you in order to assess the maintainability of a software product? By means of the third question, essential ingredients of the quality report for software maintainability assessment should be determined. Therefore, we conducted a brainstorming session in order to identify relevant answers to this closed question. The identified answers are not only quality metrics, but also more general quality indicators, such as the compliance with conventions. These answers are shown in Figure 8. Beyond this closed question, we created a subsequent open question asking for further information, which should be part of the quality report. Since the determination of ingredients is an essential task for developing a quality report, these two questions can be understood as key issues. Figure 8 depicts the frequency distribution of all answers given by 83 respondents.

Figure 8 shows the number of votes for each software quality metric or quality indicator we identified in our brainstorming session. The most popular answers are “quality of comments” and “understandability of documentation”, the most unpopular one is “cyclomatic complexity”. Figure 8 depicts the answers in order of declining popularity from the most popular answer on the left to the most unpopular answer to the right. However, there is no obvious point which separates the answers in two parts: ingredients and non-ingredients of the quality report. Instead, the most popular answers were selected by about 80% of the respondents and the most unpopular one by about 20%. Therefore, we cannot identify any quality metric or quality indicator which is indispensable or dispensable for the quality report. Moreover, there are no respondents who selected exactly the same answers. These facts lead us to the assumption that there is no uniform set of quality metrics and indicators for the quality report for software maintainability assessment.

Even the 18 responses to the related open question strengthen our assumption. Out of 12 proposed quality metrics and indicators in total, there are nine metrics and indicators which are only mentioned by one respondent. A possible explanation for our assumption is that the ingredients of a quality report depend on different aspects. For example, if we want to assess the maintainability of a web service based on SOAP, the quality report does not have to contain information about the compliance with RESTful best practices. Moreover, the content of the quality report seems to depend on the objective of the assessment. For instance, if we want to assess the readability of a software product, we usually do not need information about the test coverage. Due to the fact that there are no respondents who selected exactly the same answers, the ingredients of the quality report also appear to depend on the software expert who conducts the assessment. Hence, the assessment of maintainability seems to be a subjective task, e.g., based on the experience of the software expert.

Thanks to the related open question, we identified several additional quality metrics and indicators, which can be part of the quality report. These potential ingredients are listed below in descending order of popularity: results of static code analysis, compliance with software development principles, description of fundamental design decisions, application of best practices, maturity level of used technologies, design of interfaces, results of architectural code reviews, description of the basic architecture, age of the examined software product, usage of dependency injection, comprehensibility of the source code and development practices. Due to the fact that the most popular response to this question is stated only by five respondents, we do not elaborate on it.

Separating the respondents from research (n = 60) and industry (n = 23), Figure 8 reveals some differences between both groups. The most important quality indicator for the respondents from industry provides information about the technologies which were used during development. Moreover, the quality indicator concerning the functional naming of classes and methods is more relevant for respondents from industry than for respondents from research. As opposed to this, the completeness of the documentation is more important for respondents from research. Determining the number of responses for each answer relative to the number of respondents from research and industry respectively, the mean deviation between both groups amounts to about eleven percentage points. This indicates that the responses from both groups are quite similar. Merely, the three mentioned quality indicators above and the cyclomatic complexity are considerable outliers (deviation >20 percentage points).

Furthermore, the answers to this question point out that the quality report may contain not only software quality metrics and indicators which can be measured automatically. Instead, it may also contain quality metrics and indicators which have to be determined manually, such as the quality of comments or the understandability of documentation.

Since well-known quality metrics regarding software maintainability are mainly located on the right-hand side of Figure 8, such as the number of packages, comment ratio or cyclomatic complexity, these metrics seem to be less important for software maintainability assessment.

Question 4: How important are the given quality report properties for you? The fourth question should reveal the importance of several quality report properties for our

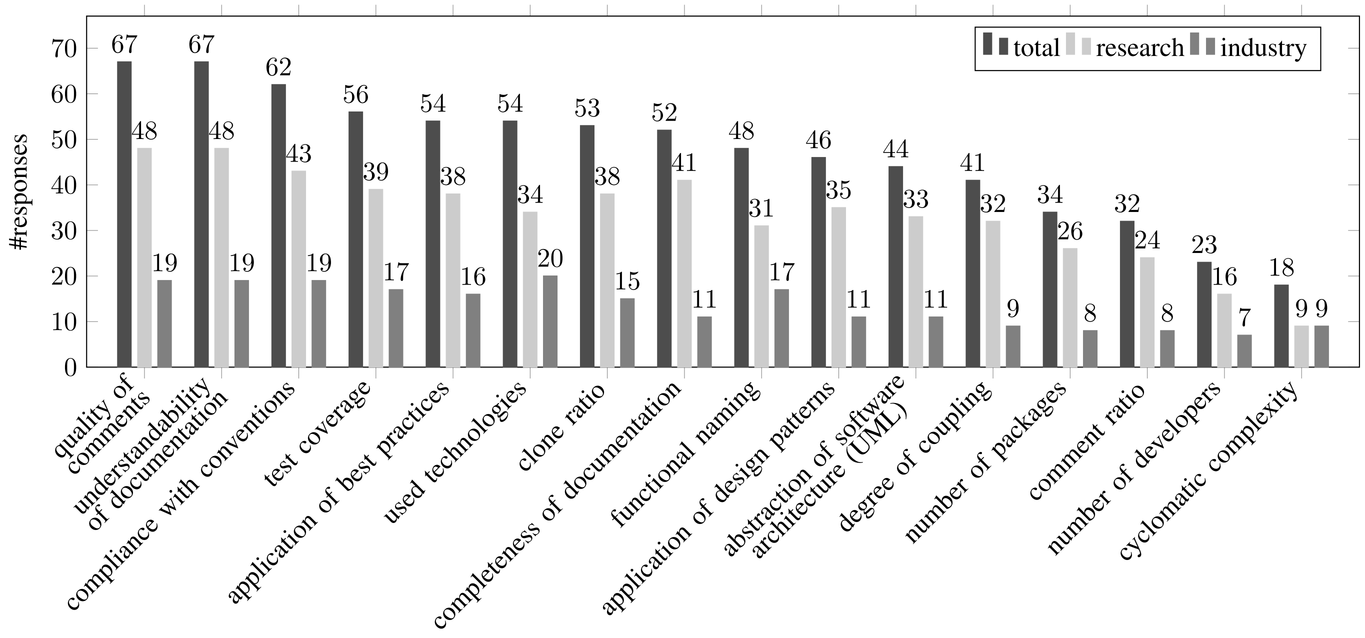

Figure 8. Ingredients of the quality report for software maintainability assessment (n = 83).

respondents. For this purpose, we derived eight properties from different standards and papers published in the software quality field (e.g., [25] [26] [27]). These properties are shown in Figure 9. Beyond this closed question, we created a subsequent open question asking for further quality report properties. Since the quality report has to possess certain properties in order to be used effectively and efficiently, these two questions are crucial to us. The participants of the survey could rate each property on a five-point ordinal scale: "very important", "important", "partly important", "less important", and "unimportant". Alternatively, the respondents could answer with "not judgeable". The number of answers ranges from 80 to 81 per property including 2 to 7 responses with "not judgeable" respectively. As these responses were counted as missing answers, Figure 9 shows the importance of the given properties based on 74 to 79 responses per property.

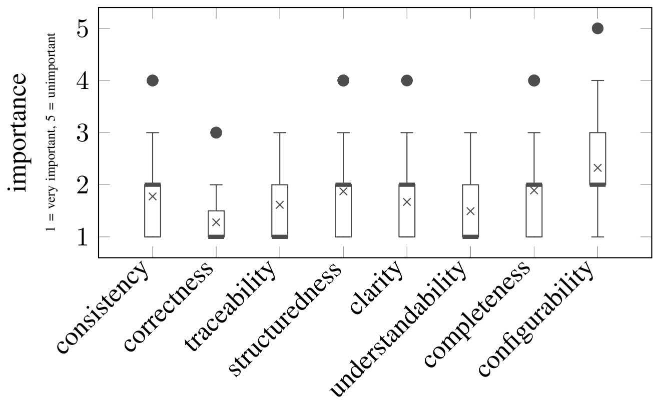

Figure 9. Importance of several quality report properties (n = 74 to 79, median = thick line, arithmetic mean = cross, outlier = dot).

The answers are coded with a scale ranging from 1 to 5, where 1 codes "very important" and 5 codes "unimportant". Figure 9 contains a boxplot, which comprises a box for each quality report property. The bottom and the top of each box are the first and third quartiles, whereas the thick line inside the box depicts the median (second quartile). The lower and the upper whiskers (horizontal lines outside a box) denote the minimum and maximum, respectively, of the coded answers with maximum 1.5 interquartile range. Furthermore, the crosses visualize the arithmetic means and the dots illustrate outliers.

The analysis of the responses shows that all of the given quality report properties are considered as "very important" or "important", due to the fact that the median ranges from 1 to 2 and the arithmetic mean ranges from 1.3 to 2.3.

Upon closer examination, the three most important properties are correctness, traceability, and understandability with a median of 1 and the lowest values for the arithmetic mean. The least important property, but still important with a median of 2 and an arithmetic mean of 2.3, is the configurability of the quality report. This result is a bit surprising considering the outcome of question 3. Question 3 suggests that the ingredients of the quality report depend on different aspects, e.g., the used technologies or the objective of the assessment. In order to equip the quality report with the required ingredients, it has to be configurable. While answering the survey, the respondents possibly think of several concrete quality reports instead of a general report. Therefore, the configurability is circumstantial. However, in our work we aim to develop a general quality report from which we deduce these concrete reports by adapting the general one. Hence, the configurability is a property which we deem very important.

Separating the respondents from research (n = 54 to 57) and industry (n = 19 to 22) shows that the importance of the given quality report properties is quite similar for both groups. There is only one difference in the median concerning the traceability (research: 1.5, industry: 1). Moreover, the average difference between the arithmetic means of each property for respondents from research and industry is 0.14, which supports our statement.

The responses to the related open question yield merely two usable properties, simplicity and robustness. Both properties were proposed by one respondent respectively. Therefore, we do not elaborate on it.

Question 5: Do you use tools for static code analysis? If so, which tools? By means of the fifth question, the usage of tools for static code analysis should be determined and relevant tools should be identified. Primarily, the respondents were asked whether they use tools for static code analysis. If so, they were asked which tools they use. Therefore, we conducted a brainstorming session in order to identify several tools for static code analysis. In addition to these given answers, the respondents could state further tools. Since software quality metrics and indicators provided by static code analysis tools are potential ingredients of the quality report, this question matters to us. Figure 10 shows the number of votes for the different tools. Due to space limitations, we only present the six most popular tools out of 27 in total.

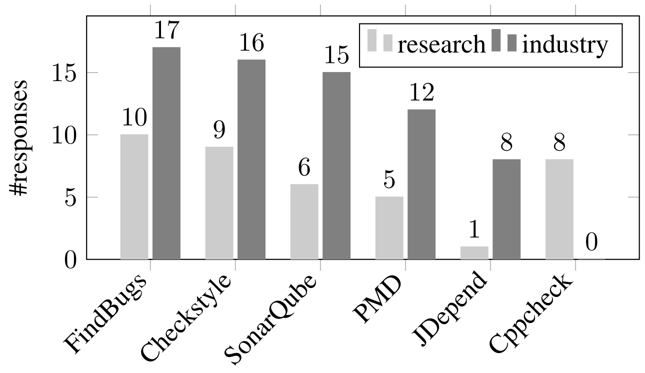

Figure 10. Most popular tools for static code analysis (n = 47).

Altogether, 49 out of 83 respondents who answered the corresponding question stated that they use tools for static code analysis. Separating the respondents from research (n = 59) and industry (n = 24), 88% of the respondents from industry use these tools in contrast to 47% from research. Although the usage of static code analysis tools seems to be more common in industry than in research, these answers underline the importance of quality assessment for both groups.

Figure 10 contains the six most popular tools, which are stated by 47 respondents. All these tools are well-known and provide a huge set of software quality metrics and quality indicators, which can be used for software maintainability assessment.

Considering the three tools for Java code, FindBugs, Checkstyle, and JDepend, we make an interesting discovery. Altogether, 22 out of the 32 respondents who selected at least one of these tools use two or all of them. Moreover, these tools support the same programming language. These two facts support the statement that currently quality metrics and indicators are distributed across several tools and dashboard. Hence, a comprehensive quality report comprising relevant software quality metrics and indicators provided by these tools and dashboards seems to be useful.

VI. Threats to validity

This section comprises an analysis of potential aspects threatening the validity of the survey and its outcomes.

The population of the study comprises software development project members from companies and research organizations of different size (see Figure 2). Moreover, we got responses from participants with a great diversity in experience regarding the number of years they are working in the domain of software engineering (see Figure 3). Therefore, the study population does not pose a threat to the validity.

In total, 113 people answered the survey, 86 from research and 27 from industry. Here, the relatively small number of respondents from industry may threat the validity of the survey and its outcomes.

The survey was written in German and was sent to companies and research organizations in Germany. This definitely is a threat to the validity of the survey and its outcomes, since different cultures and customs may imply different attitudes towards software maintainability assessment.

VII. Answers to research questions

In this section, we summarize the answers to the four research questions mentioned in Subsection IV-A.

RQ1: *Importance of quality assessment in research and industry:* Due to the answers to question 1 and question 5, we come to the conclusion that quality assessment is considered to be an important task, e.g., in order to identify areas of improvement. This is true for respondents from both research and industry, although the answers suggest more distinct quality awareness in industry.

RQ2: *Importance of software maintainability in research and industry:* The answers to question 2 lead us to the conclusion that software maintainability is a very important quality characteristic for respondents from both research and industry.

RQ3: *Ingredients of the quality report for software maintainability assessment:* Due to the answers to question 3, we conclude that there is no uniform set of quality metrics and indicators for the quality report for software maintainability assessment. Instead, the ingredients of the quality report seem to depend on different aspects, e.g., the used paradigms. Therefore, we cannot identify any quality metric or indicator which is indispensable or dispensable for the quality report. In addition, the answers to this question point out that the quality report may contain not only quality metrics and indicators which can be measured automatically. It may also contain quality metrics and indicators which have to be determined manually, e.g., due to the fact that domain knowledge is required.

RQ4: *Importance of quality report properties:* The answers to question 4 outline that all given quality report properties are considered as very important or important for respondents from both research and industry. Correctness, traceability, and understandability are the properties considered to be most important.

VIII. Conclusion

Today, the majority of software life cycle costs is spent on maintenance. The systematic or even automatic evaluation of software regarding its maintainability by means of a quality report might help to reduce these costs. However, there is no uniform understanding about the metrics and quality indicators required to assess the maintainability of software. For that reason, we showed the structure and the results of an exploratory survey for developing an appropriate quality report.

With our survey, we reached 113 respondents, partially software developers, software architects, research assistants, and project or group leaders. The survey presented us the following three key finding: 1) Our initial assumptions about the importance of quality assessment and maintainability of software were confirmed by the first questions of the survey. There was not one respondent who considers that software maintainability is less or unimportant. 2) The answers to question 3 showed that there is no uniform set of quality metrics and indicators for the quality report for software maintainability assessment. Instead, the software expert who conducts the assessment has to determine the ingredients of the quality report depending on different aspects. 3) The answers to question 3 point out that the quality report may contain not only quality metrics and indicators which can be measured automatically. Instead, it may also contain metrics and indicators which have to be determined manually.

In the future, we will further work on the development of a quality report for assessing software maintainability. On the one hand, as we could not identify concrete metrics or indicators being indispensable or dispensable for the quality report, we will categorize the identified metrics and indicators initially. Subsequently, we will try to identify additional metrics and indicators for each category based on literature research and an examination of several tools for static code analysis. These metrics and indicators should be understood as potential ingredients, which can be added to or removed from the quality report. On the other hand, we will develop a hybrid approach, which combines automatic and manual analyses in order to generate a quality report tool-based and with minimal effort. The basic idea for such an approach already exists [9] and will be seized by us.

Acknowledgment

We would like to thank all survey participants for their participation. Furthermore, we would like to thank the members of the research group Cooperation & Management (C&M), Karlsruhe Institute of Technology (KIT), and the engineers of iteratec GmbH for their valuable comments and suggestions regarding the development of the survey.

References

[1] NASA, "Software Design for Maintainability," URL: https://oce.jpl.nasa.gov/practices/dfe6.pdf [accessed: 2015-08-08].

[2] T. Pearse and P. Oman, "Maintainability Measurements on Industrial Source Code Maintenance Activities," Proceedings of International Conference on Software Maintenance, 1995, pp. 295–303, ISSN: 1063-6773.

[3] J. Choudhari and U. Suman, "An Empirical Evaluation of Iterative Maintenance Life Cycle Using XP," ACM SIGSOFT Software Engineering Notes, vol. 40, no. 2, 2015, pp. 1–14, ISSN: 0163-5948.

[4] ISO/IEC, "Std 25010:2011: Software engineering – Software product Quality Requirements and Evaluation (SQuaRE) – System and software quality models," 2011.

[5] M. Gebhart, P. Giessler, P. Burkhardt, and S. Abeck, "Quality-Oriented Requirements Engineering of RESTful Web Service for Systemic Consenting," International Journal on Advances in Software, vol. 8, no. 1&2, 2015, pp. 156–166, ISSN: 1942-2628.

[6] P. Giessler, M. Gebhart, D. Sarancin, and S. Abeck, "Best Practices for the Design of RESTful Web Services," Tenth International Conference on Software Engineering Advances (ICSEA 2015), 2015, ISSN: 2308-4235, to be appear.

[7] ISO/IEC, "Std 25000:2005: Software engineering – Software product Quality Requirements and Evaluation (SQuaRE) – Guide to SQuaRE," 2005.

[8] M. Gebhart and S. Sejdovic, "Quality-Oriented Design of Software Services in Geographical Information Systems," International Journal on Advances in Software, vol. 5, no. 3&4, 2012, pp. 293–307, ISSN: 1942-2628.

[9] M. Gebhart, "Query-Based Static Analysis of Web Services in Service-Oriented Architectures," International Journal on Advances in Internet Technology, vol. 7, no. 1&2, 2014, pp. 136–147, ISSN: 1942-2652.

[10] ISO/IEC, "Std 33001:2015: Information technology – Process assessment – Concepts and terminology," 2015.

[11] IEEE, "Std 1061-1998: IEEE Standard for a Software Quality Metrics Methodology," 1998.

[12] H. Ogasawara, A. Yamada, and M. Kojo, "Experiences of Software Quality Management Using Metrics through the Life-Cycle," Proceedings of the 18th International Conference on Software Engineering, 1996, pp. 179–188, ISSN: 0270-5257.

[13] D. Steidl, F. Deissenboeck, M. Poehlmann, R. Heinke, and B. Uhink-Mergenthaler, "Continuous Software Quality Control in Practice," 2014 IEEE International Conference on Software Maintenance and Evolution (ICSME), 2014, pp. 561–564, ISSN: 1063-6773.

[14] ISO/IEC, "Std 25040:2011: Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – Evaluation process," 2011.

[15] M. Riaz, E. Mendes, and E. Tempero, "A Systematic Review of Software Maintainability Prediction and Metrics," 3rd International Symposium on Empirical Software Engineering and Measurement (ESEM), 2009, pp. 367–377, ISSN: 1938-6451.

[16] S. Chidamber and C. Kemerer, "A Metrics Suite for Object Oriented Design," IEEE Transactions on Software Engineering, vol. 20, no. 6, 1994, pp. 476–493, ISSN: 0098-5589.

[17] ISO/IEC/IEEE, "Std 29148:2011: Systems and software engineering – Life cycle processes – Requirements engineering," 2011.

[18] enuvo GmbH, "Umfrage Online," URL: https://www.umfrageonline.com/ [accessed: 2015-07-27].

[19] D. A. Dillman, Ed., Mail and Internet Surveys – The Tailored Design Method. John Wiley & Sons Inc., Hoboken, New Jersey, Jun. 2007, ISBN: 978-04-70-03-85-6.

[20] A. Bryman, Ed., Social Research Methods. Oxford University Press Inc., New York, Mar. 2008, ISBN: 978-01-99-20-29-5.

[21] L. Bickman and D. J. Rog, Eds., The SAGE Handbook of Applied Social Research Methods. SAGE Publications Inc., Thousand Oaks, California, 2009, ISBN: 978-14-12-95-03-1.

[22] D. Robbins, Ed., Understanding Research Methods – A Guide for the Public and Nonprofit Manager. Taylor & Francis Group LLC, Abingdon, 2009.

[23] M. Gerster and P. Giessler, "Entwicklung eines Qualitätsberichts," 2015, URL: https://www.umfrageonline.com/s/7e3cb3d [accessed: 2015-08-07].

[24] IEEE, "Std 1028-2008: IEEE Standard for Software Reviews and Audits," 2008.

[25] ISO/IEC, "Std 25012:2008: Software engineering – Software product Quality Requirements and Evaluation (SQuaRE) – Data quality model," 2008.

[26] R. Plösch, A. Dautovic, and M. Saft, "The Value of Software Documentation Quality," 14th International Conference on Quality Software (QSIC), 2014, pp. 333–342, ISSN: 1550-6002.

[27] R. E. Al-Qutaish, "Quality Models in Software Engineering Literature: An Analytical and Comparative Study," Journal of American Science, vol. 6, no. 3, 2010, pp. 166–175, ISSN: 1545-1003.