An Empirical Investigation of Maintainability Metrics Adoption in Brazilian Software Companies

Micael Soares[1], Samuel Romeiro[1], Juliana Saraiva[2], Sérgio Soares[1]

[1]Center for Informatics (UFPE)
Recife, Brazil

[2]Department of Exact Sciences (UFPB)
Rio Tinto, Brazil

{msf3, scras, scbs}@cin.ufpe.br, julianajags@dcx.ufpb.br

Abstract. *Software Maintainability (SM) has been studied since it became globally accepted as part of the software quality model. Many researchers have been proposing a lot of metrics to be used as SM indicators. Nevertheless, the descriptions of these metrics are scattered in a larger number of studies, where many do not explain what these metrics should measure. Therefore, this paper presents a research about SM metrics' adoption in Brazilian software companies. We performed semi-structured interviews in a face-to-face fashion with 10 software companies resulting in 23 SM metrics listed, and 14 tools for supporting SM metrics collection. Our results showed evidence that most of the SM metrics proposed by researchers are not used by practitioners.*

1. Introduction

Software development includes a series of activities whose complexity is notorious. If a design flaw is not properly repaired, the cost of fixing it (after the software is delivery) can be 5-100 times higher than if it was fixed during design definition [Sahar R. Ragab 2010]. In this context, software quality can be defined as a set of at-tributes that must be achieved, so the final product meets the necessity of their users [Sommerville 2011, Meirelles 2008]. Many software attributes can be used as a quality indicator. Software maintainability (SM) is considered an attribute of software quality that plays an important role on its quality level. The higher the quality of the software, the lower tends to be the cost/effort on its maintenance cycle [Pressman 2011].

Software maintainability is defined through five characteristics [ISO/EIC 2011]: **analyzability** — software attributes that evidence the needed effort for identifying causes of failures or deficiencies in the system structure; **modifiability** — software attributes that evidence the effort required for correcting such failures (through adapting to environmental changes or removing defects), without introducing new defects or degrading existing product quality; **modularity** — the degree on which a system is composed of group of components in such way, that a change to one component has minimal impact on others components; **reusability** — the degree on which an asset can be used in more than one system or in building other assets; **testability** — software attributes that evidence the effort required to validate the modified software [ISO/EIC 2011].

With these definitions, it is possible to affirm that maintainability characteristics might present a significant challenge for software engineering [Whippany 1994].

Nevertheless, there is a way to facilitated controlling software maintainability steps and their attributes. Software Maintainability metrics can help identifying potential problems in the software structure [Bandi et al. 2003]. In addition, there are studies showing that these metrics can be used as qualitative and quantitative indicators for research in the Software Engineering (SE) [Mingguang et al. 2009, Beszedes et al. 2007, Saraiva et al. 2012].

Additionally, some tools have been developed aiming to facilitate analysis and collection of software metrics applied to different scenarios in software projects [Rudiger Lincke and Lowe 2008]. Many of them assisting cost/effort analysis in maintaining software products [Bitman 1999]. However, the wide variety of tools for supporting SM metrics selection and the lack of information to better evaluate their applicability and the context in which these metrics are adopted may hinder the usage and adoption of SM metrics and such tools.

Also, selecting and applying SM metrics able to reflect characteristics of a system is a real challenge for professionals [Leroy et al. 1994]. Over the years, SM metrics have been studied and several researchers have proposed a large set of metrics. These factors represent some problems for professionals and researchers worldwide. Firstly, the large number of identified metrics and their descriptions are scattered in a larger number of studies, where many of them do not explain clearly what these metrics should measure [Saraiva et al. 2015]. Second, there are metrics that have the same name, but measures different quality attributes [Saraiva et al. 2012].

Thus, this paper chose a methodological approach based on the Empirical Software Engineering for assessing SM metrics that have been adopted in the Brazilian industrial scenario. We also analyzed the adoption and applicability of tools to assist the metrics choice. We conducted a survey via semi-structured interviews to get the research data. With the results, we hope to contribute for understanding the software metrics and measurement tools used in the industry.

There are studies that aimed to accomplish this research proposal with the increased focus on academic and scientific contexts [Saraiva et. al. 2012, Saraiva et. al. 2015, Riaz et. al. 2009]. Nevertheless, the application of these metrics is not restricted to the academic scenario.

The remainder of the paper is organized as follow: Section 2 discusses the research method; The results and discussions of this work are depicted in Section 3; Section 4 presents the threats to validity; Section 5 exposes the related works; Finally, Section 6 presents the concluding remarks and Section 7 the acknowledgements.

2. Research Method and Methodology

This section exposes in detail the research method and methodology of our study. The subsections describe the definitions of the method that we had followed and discuss how all steps of the method was applied in our research. In addition, the data collection process and analysis are also depicted.

2.1. Research Questions

The following questions were raised for this research:

- **RQ1**: What SM metrics are used in the industrial context?
- **RQ2**: How the SM metrics are collected and applied?
- **RQ3**: What tools are used to collect the SM metrics?
- **RQ4**: Is there a relationship between using tools to collect metrics and the decision-making process regarding the choice of SM metrics?

2.2. Research Design

The empirical method used was the cross-sectional survey [Shull et al. 2009]. This method was chosen because it is a research method used to collect information that can explain, describe, or compare knowledge, attitudes, and behavior of a certain population [Shull et al. 2009]. Survey method is composed of five well-defined activities [Shull et al. 2009]: (i) research goals definition; (ii) research design definition; (iii) development and validation of research instrument; (iv) instrument documentation; (v) re-search data collection and assessment [Kitchenham and Pfieeger 2001]. Figure 1 depicts an activity diagram that shows each phase of the survey used in our research.

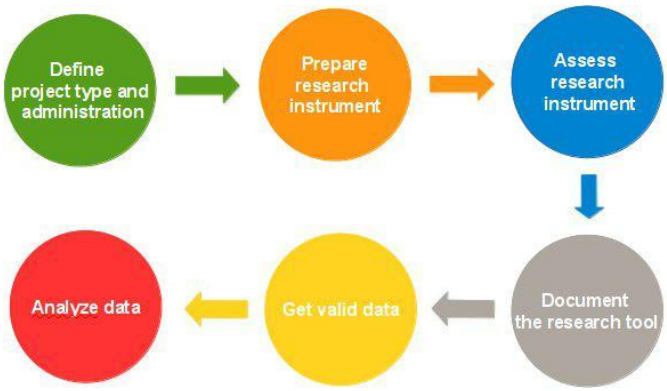

Figure 1. Survey Activity Diagram

Our study goals are: to list SM metrics and tools; to analyze their applicability and usage in industrial scenario. We focused on SM metrics collected during the process of software maintenance and evolution cycle; we collected information about tools that support the SM metrics' collection; we also identified correlations between the usage of these tools and maintainability metrics choice.

The data of our study was collected through semi-structured interviews. It was necessary to make face-to-face visits to gather information that allowed us a more comprehensive and in-depth research. It is important emphasizing that our research had an exploratory character, thus, we focus on the quality rather than quantity. The results will help us in future stages of the project, which we intend to compose an online questionnaire with similar questions of the interview and investigate the adoption scenario of SM metrics in companies around the globe.

The interviews were conducted between April to June and in November of 2015. The employees interviewed were project managers, quality engineers, programmers, and testers. All of them were directly related to the metrics collection process. Software companies located in northeastern and southeastern of Brazil were interviewed with an average of one hour per visit.

The interview questionnaire was composed of 16 questions divided into basically three parts: (i) the first explored the presence of maintenance cycle in the

company's systems; (ii) the second, collected information on the adoption of maintainability metrics during the development/evolution process; (iii) the third dealt with the tools for supporting collection of SM metrics. All the interview's questions are related to the RQs described in Section 2.1 as following:

- **Interview question(s) for RQ1**: (1) Does the company perform maintenance on their software products? (2) Does the company adopt some kind of software metrics? (It generally occurs during the development cycle or after — in the software maintenance cycle?) (3) Among those metrics, which ones are used during the maintenance cycle? (4) According to the company, what are the benefits and/or drawbacks in using SM metrics? (5) Have any of these metrics been used in order to measure and predict the effort to make changes, evolution, or correction of faults in the system?
- **Interview question(s) for RQ2**: (1) How did the company starting using metrics? What was the motivation? (2) Is there any process for collecting these metrics? Please, describe this process. (3) How does the company use these metrics? Was it during the development cycle (predictors) or after its implementation (maintenance cycle)? (4) Who does usually use or is responsible for collecting the metrics (project managers, developers, testers, others)? (5) According to the company, how useful are these metrics?
- **Interview question(s) for RQ3**: (1) Does the company use or know any tools for collecting these metrics? (2) What are these tools? (3) How long does the company use these tools? (4) Why have these tools been chosen? (5) What points are considered positive/negative on using tools for collecting SM metrics?
- **Interview question(s) for RQ4**: (1) Does the selection of the tools depend, in any way, on the projects or other circumstances? (2) Is there any process for collecting SM metrics? Please, describe this process.

The interviews were initially applied in software development companies in Porto Digital, a nationwide recognized technological park with over $375 million in annual revenue located in Recife (Pernambuco state) [PortoDigital]. Others interviews were also applied in companies located in Joao Pessoa (Paraíba state). This city has a group of software industries that have brought technological growth for the region and discussion about the creation of a new technology park [Prefeitura Municipal de Joao Pessoa 2015]. We also interviewed companies from Tecnopuc, the technological and scientific park of the Pontifical Catholic University of Rio Grande do Sul (PUCRS), located in Porto Alegre (Rio Grande do Sul state). Tecnopuc houses 120 organizations, totaling more than 6,300 workstations [Tecnopuc 2016].

Interviews were recorded and then transcribed for helping us to carefully analyze all gathered data. It is important to clarify that we explained to the interviewees the meaning of "metrics": every indicator, attribute, and measure used to provide quantitative values regarding any software product. Another relevant concern taken into account for the survey application was the participants' motivation. According to [Shull et al. 2009], people are more motivated to provide complete and concise answers about any topic, if they know for sure the purpose of the study [Shull et al. 2009]. Consequently, before each interview, a brief presentation was performed to expose an overview of our study, and consequently, the benefits that it will provide to the companies. This presentation lasted 15 minutes.

3. Results and Discussions

In this section, the results and discussion of our study are depicted. The discussion is organized according to the research questions presented in Section 2.1.

3.1. Respondents' Demographics

To classify the companies that participated in our study, we followed the Support Service for Micro and Small Enterprises (SEBRAE) recommendations [SEBRAE 2006]. Where businesses with up to 19 employees are considered micro-enterprises; those with between 20 and 99 employees are considered small ones; 100 to 499 employees are identified as medium-sized; and finally, more than 500 employees are considered large companies [SEBRAE 2006].

We performed 13 interviews in 10 different software companies with 15 interviewees. In two of the 13 interviews, there were two interviewees, and in the other 11 only one person. In one company we conducted four interviews with four different people, making this 13 interviews in total. We interviewed software managers, quality analysts, developers, testers and company's director. Most of the companies (four) are headquartered in the Digital Port of Recife-PE. Others three are set at Joao˜ Pessoa-PB. The remaining (three) are located on Tecnopuc (Porto Alegre city). Two companies are considered as micro enterprises, four are classified as small businesses, three are considered medium ones and finally, one is classified as large businesses.

As for quality certifications (CMMi, MPS.BR, ISO 9001, Oracle Platinum Level Partner, SOA Specialized Partner), half of the companies reported having at least one of these certifications. The other half claimed they don't have any kind of these. However, two of them confirmed that they had had at least one of the quality certifications mentioned. But there was not an interest in renewing it. All companies interviewed were in the Commercial Software Industry area and/or the Software Factory area. In other words, they have worked in the software development area and also software factory (target population of our research).

3.2. RQ1: MS Metrics

All companies answered "yes" for the interview's first question: "Does the company perform maintenance on Their products software?". When we asked if the company uses software maintainability metrics, nine answered "yes". Only one reported not using such metrics. This only company affirmed that the reason for not using SM metrics is the relationship between time versus human resources to manage its data. In other words, the company used to use SM metrics. However, developers/managers/testers had lost a lot of time updating these metrics' data and often, such data were not enough to reflect the real status of projects or simply had not been used.

Table 1 presents the maintainability metrics adopted by the Brazilian software companies interviewed, collected through the answered of the nine companies that reported using SM metrics. The first column represents the metric's name and the second column presents the number of companies using it.

According to the Table 1, we identified 23 software maintainability metrics. For supporting our discussions, we decided to divide these metrics into three groups based on the number of companies using it. The first group refers to the metrics used by six or

more companies. This is the case of Customer Satisfaction, Effort, Complexity, Time and Number of Bugs. Time and Effort are among the most mentioned metrics in the interviews. The metric "Time" refers to the total time for developing/maintaining software systems. "Effort" indicates the total cost for developing systems (covering the financial costs, how many programmers were allocated, hours worked, and so on). If we analyze frequency mentions of words that refer to the collected metrics, "Time" was the most mentioned one (with more than 100 mentions through all the interviews). Effort metric also had many mentions (also mentioned more than 100 times). One of the reasons is because both metrics are considered a risk factor for software companies.

Table 1. Software Maintainability Metrics

Maintainability Metrics' Name	Number of Companies Using It
Customer Satisfaction	8
Effort	7
Complexity	6
Number of Bugs	6
Time	6
Backlog Validation	4
Number of Tasks	4
Size	4
Costs	3
Lines of Code	3
Point/Hour	3
Speed	3
Adherence to Process	2
Code Coverage Level	2
Coupling	2
Completeness	2
Instability	2
Number of Code Inspections	2
Cohesion	1
Efficiency	1
Function Point	1
Schedule Control	1
Technical Debt	1

Time is also used as a parameter to control the productivity of their own programmers and to calculate other indicators such as Point/Hour (number of points —— tasks performed per hour) and Effort, for instance. The main difference between Effort and Time is that the first one is used as an indicator by engineers for designing internal control of the projects while the second one is used for both internal controls and for customers' feedback.

Customer Satisfaction is the most used metric. 8 companies said they are using attributes to measure customer satisfaction level. This fact can represent an evidence for the constant concern of software industry related to customers' satisfaction. Months, or even years after launching a particular system, companies have sectors dedicated to

maintaining contact with the customers. The main idea is to measure some indicators, such as client satisfaction with their products. This reality, many times, justifies the maintenance of some systems because the client needs new features or because it was detected problems. The following statements help to consolidate the above discussions.

> "(...) Then, the first deal that we have made here was the following: 'Guys, we have eight hours, but I want an Effort of seven hours. I don't want eight' (...)"

> "(...) This client satisfaction survey where quality is one of its points, it runs once or twice a year. I have always tried to reconcile with important deliveries because the information is more recent (...)"

The second group refers to the metrics used by three, four or five companies. We can highlight two metrics in this group, Size and Cost. Although some metrics of this group, such as Lines of Code and Point/Hour, are also important. Unsurprisingly, companies estimate the costs of their projects during the development cycle. The same is true for the maintenance cycle. Three companies reported using Cost as a metric.

As for the Size metric (a number that represents how big or small is the system), four companies reported using mechanisms for estimating the size of tasks. According to the research data, this estimation can be done in two ways: (i) through the mechanism for scoring systems, commonly used in agile methodologies, such as Scrum; (ii) through the project manager expertise. Based on the experience of the software manager, a task may be more or less complex compared to another. Here are some speeches of two project managers interviewed.

> "(...) So, I have a set and topics that I can check. After that, come the part of registration, which is focused on me, as I know the business rule and software structure, I can measure how complex it is and how much effort will be required to be able to do this task. Most of the time I don't miss, but that's what we do, based on experience (...)"

Finally the third and last group, composed of metrics used by one or two companies. We can emphasize Cohesion, Completeness, Coupling, and Function Point. These metrics are well known in the Software Engineering area. Nevertheless, they were rarely mentioned. In some interviews, engineers have reported that the amount of formalism for collecting and using those metrics do not help its acceptation in the software industry. On the other hand, two companies claim that they had stopped using classical software metrics, like those mentioned before, because of the amount of information required for a complete analysis of this indicator. This fact helps to explain why software companies are using metrics where the measurement is simpler than those commonly used in the academia, which presents a complex way to collect and interpret its data. Our results, discussed so far, have shown evidence of that.

> "(...) So I see a very large advent of people focused on standardization in the matter of CMMI, I think 2008. But soon the company itself had no more interest in renewing it, you know?! It's much more important to be effective than to be efficient (...)"

There is another Metrics that draws attention in this discussion group: Technical debt. This SM metric measures the flaws in the development/maintenance software process (covering from the product itself to the process in question). It is measured using various indexes, such as Cyclomatic Complexity, Effort, Number of Tasks, Costs, among others. A single company said using this measure. However, this attribute is

configured as a risk factor for the company, since it is used for decision making related to any product in the development stage, involving, many times, other projects.

The definitions of these metrics were taken from the responses itself. That is, our intention is not to expose "formal definitions", we are just reporting the descriptions of metrics considering the companies' point of view. Generally, companies consider the using of metrics a very positive experience. The topics below list negative and positive points on using SM metrics.

- Drawbacks in using SM metrics - bureaucracy to gather information; high cost implantation (programmers' time demand); in the case of those companies that does not have well-defined collection processes, time for collecting such kind of metrics; company dependence on metrics that are involved in the project and the experience they had in past projects.
- Advantages in using SM metrics - project control progress; transparency about the activities of each developer; feedback for the client; commitment; timely product delivery; increased quality of produced systems.

3.3. RQ2: Metrics Application

The word "Process" was repeated about 100 times in the interviews. This can be an evidence that the practitioners are worrying about the collection process of these metrics. Nevertheless, it is important to highlight that our results show no standardization for the collection process. Each company has a process that best suit their own demands. The interviews' responses allowed us to separate the companies into two groups based on the collection process. This section will focus on the nine companies that reported using SM metrics.

The first group is composed of companies with a well-defined metrics collection method. There is no pattern, though, only common characteristics between the companies in this group. The first one is the fact that all these companies hold meetings to discuss key values for the adopted metrics. The meetings are the basis for the next development steps. The development does not begin until all metrics have been estimated and formalized. These meetings are concerned in transform the past estimation into information for developers. For instance, development time can be transformed into Effort measures, Number of Tasks, Point/Hour, and so on.

Another characteristic of this group is that the metrics' analyses have been centered in one person: the software manager. He/She is responsible for controlling and briefing these numbers to the company's board. He/She is also responsible for charging developers to give reports to evaluate the performance of each employee.

The second group consists of companies that do not have formal methods for collecting maintainability metrics. Every project manager assesses the reality of the project and verifies the need for seeking metrics, based on their necessities. These metrics are used as indicators for managing projects and also as measures for code analyzing. Unlike the first group, these companies do not have the collection process focused on the project manager. This process is more dynamic and developers actively participate in this application. Consequently, the process is not a critical agent for maintainability metrics adoption. Each company has a reality, and it adapts to the process that brings best answers to their problems.

We also have noticed that maintainability metrics are used during the project development as effort predictor. Out of the nine companies surveyed (and uses SM metrics), six affirmed they have used these metrics during development cycle for providing maintenance measures (predictors). Another important information is regarding the utility of these metrics. When the companies were asked about how useful they are to the company, all nine respondents said that using SM metrics is "important" or "very important". They also acknowledge the importance of these metrics for software maintenance activities.

3.4. RQ3: Tools

Figure 2 presents the tools for supporting SM metrics collection, used by the companies interviewed. We listed 14 different ones. These tools have features to assist the collection process of maintainability metrics and manage information about them. Figure 2 is a graphical representation of the frequency mention of each tool. The higher the frequency, the more evident is this tool.

According to the Figure 2, five tools can be highlighted as most mentioned: home-made tools, Sonar, Jira, Redmine, and CRM tools. Jira has one of the highest incidences among companies. Four of the nine companies that have used SM metrics, mentioned they are aware of this tool. Three said they have already used it in their internal processes. The Sonar is also well known. Three companies have used this tool.

It is noteworthy that all businesses have used more than one tool in their processes. Two companies, for instance, reported using four different tools for managing their SM metrics. Many companies seek for adapting the use of these tools for customers accessing and project monitoring. This is the motivation of why the incidence of CRM tool was so high. It was mentioned as a system capable of doing this interface with the customer, and it simplifies the contact between the company's employee and the customers. In fact, the customer can report problems for justifying maintenance task. Another tool that provides this interface is Mantis. It is used in three companies surveyed. Below, we present some statements confirming our discussions.

> "(..) First of all, we are using Jira, which is recommended, but not for all projects. Sonar is also used in several projects. Jenkins is used for supporting the matter of continuous integration with Sonar and others stuff. I would say that these three tools are basics. We may use others, but it can vary according to the project's context (...)"

> "We have used Excel and Mantis (...) So, all the bugs are reported to them, both intern bugs and client bugs (...) Then, we use these same tools to make metrics collection. (...)"

An interesting fact to point out is the high incidence of tools that were developed by the company. With 40 mentions, the expression "Home-made tools" presents a prominent position in the graph of Figure 2. Four companies reported having systems that have been developed in order to assist the processes of metrics collection. The justification of it: the lacking of features in existing tools, such as having more personal contact with the customer.

We have also investigated the acceptance level of these tools. The topics below list negative and positive points of using tools to support collection of SM metrics:

- Drawbacks in tools' using - bureaucracy to collect metrics for tasks and projects considered small by the companies (require small effort); better integration between different tools; low automation process (many manual activities).
- Advantages in tools' using - deployment cost; easy customization of the systems; keeps updated all the information about the metrics; constantly evolving (the systems are always aligned with the customer's needs); data Consistency presented on the tool.

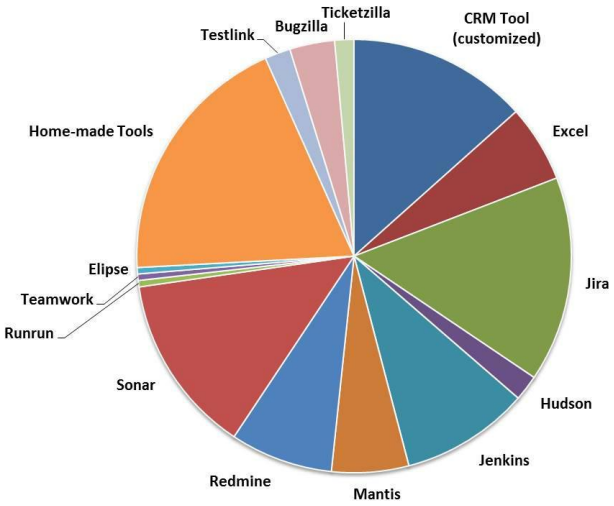

Figure 2. Tools Mentions Frequency

3.5. RQ4: Tools vs Metrics

In this section, the relationship between metrics collection and the use of tools for supporting metrics collection is discussed. Based on the interviewees' answers, it is possible to highlight the following points:

- **The adoption of tools came from the need for managing information from maintainability metrics** — Companies often introduce some tools in their processes because of the need for organizing and tracking these indicators, or even the need to forward this information to their customers. The following statements support this idea.

  > "(...) The need is emerging. So let's say, over the last 5 years, there were many times we have to create tools to measure (...)"
  >
  > "(...) We needed a tool that was collaborative, could be available on the web, all that together. Then I had conducted a research for a long time and decided I would make use of the Jira tool to my projects (...)"

- **The projects use tools for managing information about SM metrics** — There is no "condition for using tools". The companies reported they have made use of at least, one tool for gathering information regarding their maintainability metrics. We have confirmed that through the following statements.

  > "(...) Most of the projects use, at least, Jira and Jenkins tools. (...). I cannot remember at least one project that doesn't use these tools. (...)"
  >
  > "(...) We have always used it. Without the tool, it is impossible to measure!"

Assessing the previous statements, it is possible to observe a clear dependence between metrics choice and tools selection. While the first one was a prerequisite for implementing the second one, after its adoption, SM metrics are not adopted without a tool for collecting them. The Company always has its indicators under control by using tools for collecting them.

Another fact to point out is that the tools to support maintainability activities are also a software, therefore, it is necessary to maintain and include new definitions for collecting SM metrics. This fact can explain why there are few tools to do it and why some companies have chosen to develop their own tools.

4. Threats to Validity

This section discusses the threats to validity suggested by [Wohlin et al. 2012].

4.1. Internal Validity

This validity is concerned with matters related to participants of the study. Irrelevant interviewee or answers, which can lead to systematic errors in the research without the researcher's knowledge, for instance [Wohlin et al. 2012, Usman et al. 2015]. To mitigate this threat we adopted three strategies: (i) on the interview schedule period, we made sure that people who would attend the meeting had some knowledge about metrics collection — An invitation letter was sent to all participants describing the study and its goals. Consequently, the companies previously allocated a specialized employee on the subject to receive us; (ii) face-to-face interviews — although this led to a low number of participants, we have made sure that all questions were answered in a proper way. If some response was considered irrelevant, the question was redone, directing the respondent to give a more acceptable one. (iii) We ensure the confidentiality of all data from participants, allowing more comfort for them in answering all the questions.

Even adopting those strategies, there is another threat relating to the internal validity of this work: the extension of metrics' definitions. For instance, how do we know that the respondents are talking about the same metric when say they are using "complexity" or "cohesion" metric? One strategy that we have used trying to mitigate this threat was the instigation of some metrics definition. We have conducted a survey via semi-structured interviews, having that in mind, we were allowed to make some extra questions about the metrics and/or tools. Throughout these extra questions, we encouraged the interviewee for defining with some more details the metrics used on his company.

4.2. External Validity

Also known as generalizability, this validity refers to the extension in which the results of a particular study is valid [Wohlin et al. 2012]. Our study is an exploratory research, and as there are not many works such as ours in the literature, generalize the data is not our goal. We intend only to show the state of practice of software maintainability metrics and tools for supporting SM collecting in some Brazilian software industries. The idea is to replicate this study with new sets of companies obtaining more data. The work discussions may be valid for companies with similar characteristics to participants of the study.

4.3. Construct Validity

This validity refers to problems that arise due to improper design of the survey instrument. Such issues can cause errors in the data collection process. In other words, what we are willing to evaluate, would not be properly measured [Wohlin et al. 2012, Usman et al. 2015]. We tried to mitigate this threat through questionnaire (interview questions) validation by software engineering researchers and through two pilot interviews at the beginning stages of the research execution. With the results collected during this validation period, we have found that the questionnaire returned information able to respond our main objectives.

4.4. Conclusion Validity

Conclusion validity is related to erroneous conclusions retrieved from the results [Wohlin et al. 2012]. This is caused through bad data evaluation, either by applying misguided statistical methods or poor qualitative analysis of the results, the latter been identified as a threat in our research. As a strategy to mitigate this threat, we used the QSR NVivo software (version 10) [International 2015]. This system is a platform for unstructured data analysis. It means that the tool has an interface that allows a better qualitative assessment of large sets of data. We also followed the guidelines suggested by Dyba (2005) to assess data through Evidence-Based Software Engineering [Dyba et al. 2005].

5. Related Works

There are some studies in the literature that aimed to seek and analyze SM metrics. This section depicted the main works related to our research. The first related work was conducted by [Saraiva et al. 2012]. In this study, the authors performed a systematic mapping study, searching for Aspect Oriented (AO) SM metrics, and Object Oriented (OO) SM metrics. More than 570 metrics was listed [Saraiva et al. 2012]. The results of this research showed that there are many SM metrics proposed by experts worldwide. However, there is no information about which metrics are actually being used by software companies.

The study proposed in [Saraiva et al. 2012] served as the basis for other research presented in [Saraiva et al. 2015]. In this study, the authors proposed a catalog of metrics based on categories proposed by them. However, this catalog was focused on metrics often used in academic context, leaving out many metrics used in the industry setting.

[Riaz et al. 2009] conducted a systematic review of SM metrics used in the development process (predictors). The results showed that out of 700 studies initially selected, only 15 reached the main goal of the systematic review. In these 15 studies, 12 proposed models for SM prediction, and from them, only 6 used accuracy measures. These data only emphasize how much this area demands scientific research for supporting software quality [Riaz et al. 2009].

Another research that addressed the same issue was carried out in Brazil. Arruda and Filho conducted a study that shows the impacts of software metrics usage in software companies on the Digital Port of Recife. Although the focus of the research is metrics in general, this study adopted as the empirical method, the survey method (very

similar to our proposal) [Arruda and Filho 2014]. The authors interviewed 20 companies of the Digital Port and revealed that about 90% of them have used software metrics in their development processes. Some of those metrics was also identified in our study (Size, Cost, Complexity, for instance). However, the majority of metrics analyzed in [Arruda and Filho 2014] are also well-known metrics in academia (NOC, LCOM, for instance). This fact doesn't seem to be a reality when it comes to SM metrics. Our results showed evidence of that [Arruda and Filho 2014].

In addition, other studies [Lincke et al. 2008], [Ragab and Ammar 2010] [Budimac et al. 2012] deal with tools that support software metrics. However, there is still a lack of studies that addressed tools for supporting software maintainability metrics collection and their use in industrial scenario.

6. Concluding Remarks

This paper presented an exploratory study on the adoption of maintainability metrics in some Brazilian software companies. This research was conducted through a survey via semi-structured interviews resulting in 23 metrics listed as software maintainability indicators and 14 tools to support these metrics application.

Most of the interviewed companies seek maintainability metrics that provide a simple way to interpret their measures. The majority of them have been adopting the following metrics: "Time", "Effort", "Number of Bugs", "Customer Satisfaction", "Point/Hour", and so on. In addition, Companies have shown themselves divided when it comes to documented processes for collecting SM metrics. There are those companies documenting this process and those that was not concerned with this formalization. Our results showed no standardization process for the collection, though. Also, Home-made tools and Jira were the most prominent tools adopted by the interviewed companies. Based on the interviewees' speech, the tools for collecting SM metrics have to be able to easily adapt to the users' needs.

Our results showed evidence that may be happening a technology transfer problem between academia and industry. The research presented in [Saraiva et al. 2012, Saraiva et al. 2015], depicted a list and categorization of more than 500 SM metrics widely used in academia. However, our results showed only 23 SM metrics adopted by the industry. If we were looking only at the metrics proposed and used by academics, out of 23 metrics listed in our study, less than 10 are SM metrics also listed in [Saraiva et al. 2012]. There must be a reflection on how these metrics are being proposed and investigate the reasons for these discrepant data.

Also, there is another fact that is important to emphasize. The SM metrics listed in this paper can be misused, misunderstood or poorly analyzed by any company. In fact, we have reasons to believe that this really happens. However, our study has an exploratory and initial character. In this context, we do not have enough data to discuss in a deeper way, the applicability of these metrics, which will be our future work. Our goal with this study was to give an overview of what are those metrics, how they are being used and what tools assist in this process.

Finally, our research was able to answer the research questions presented in Section 2.1, helping researchers and practitioners to initially understand the context of metrics and tools adoption in the Brazilian software industry. These initial findings will provide the basis for further and in-depth studies in order to better characterize and

discuss the SM metrics' adoption scenario. We hope to provide better vision regarding tools and SM metrics already used in software industry.

7. Acknowledgements

This work was partially supported by the National Institute of Science and Technology for Software Engineering (INES)[1], funded by CNPq grant 573964/2008-4. Sérgio Soares is partially supported by CNPq grants 309234/2007-7 and 471381/2012-8. Samuel Romeiro was supported by CAPES grants.

References

Arruda, D. F. and Filho, J. G. A. T. (2014). Software metrics: A survey conducted with brazilian it companies. In CONTECSI: Proceedings of the eleventh International Conference on Information Systems and Technology Management, pages 1801–1817.

Bandi, R. K., Vaishnavi, V. K., and Turk, D. E. (2003). Predicting maintenance performance using object-oriented design complexity metrics. IEEE Transactions on Soft-ware Engineering, 21(1).

Beszedes, A., Gergely, T., Farago, S., Gyimothy, T., and Fisher, F. (2007). The dynamic function coupling metric and its use in software evolution. In CSMR'07: Proceedings of 11th European Conference on Software Maintenance and Reengineering.

Bitman, W. R. (1999). A metrics-based decision support tool for software module interfacing technique selection to lower maintenance cost. In Software Metrics Symposium. Proceedings. Sixth International.

Budimac, Z., Rakic, G., Hericko, M., and Gerlec, C. (2012). Towards the better software metrics tool. Software Maintenance and Reengineering (CSMR), 2012 16th European Conference on.

Dyba, T., Kitchenham, B. A., and Jorgensen, M. (2005). Evidence-based software engineering for practitoners. IEEE Computer Society.

International, Q. (2015). Qrs web site. http://www.qsrinternational.com.

ISO/EIC, B. S. P. (2011). Systems and Software Quality Requirements and Evaluation (SQuaRE) Models. [S.l.]: ISO/IEC.

Kitchenham, B. and Pfieeger, S. L. (2001). Principles of survey research part1: Turning lemons into lemonade. Software Engineering Notes.

Leroy, A., C., V., A.T.&T., Bell Laboratories, and Whippany, C. (1994). Integrated maintainability analysis: A practical case study. In Annual Reliability and Maintainability Symposium.

Lincke, R., Lundberg, J., and Lowe, W. (2008). Comparing software metrics tools. In In ISSTA'08: Proc. of the 2008 int. symp. on Software testing and analysis. ACM.

Meirelles, P. R. M. (2008). Searching software metrics for free software projects evaluation in portuguese levantamento de métricas de avaliação de projetos de software livre. http://ccsl.ime.usp.br/files/relatorioPauloMeirelles_final.pdf.

[1] http://www.ines.org.br

Mingguang, Z., Haqhua, Z., Weiyi, Q., Shijun, M., and Chuanyin, A. W. (2009). The mea-surement and evaluation for large-scale object-oriented software system. In HIS.'09: Proceedings of 9th International Conference on Hybrid Intelligent Systems.

PortoDigital. Digital port in portuguese porto digital.

Prefeitura Municipal de Joao Pessoa (2015). Forumtec shows the ways to build a techno-logical park in the capital in portuguese forumtec indica caminhos para construção de parque tecnológico na capital. http://www.joaopessoa.pb.gov.br/.

Pressman, R. S. (2011). Software Engineering: A Practitioner's Approach in portuguese Engenharia de Software uma abordagem profissional. Number 7. AMGH.

Ragab, S. and Ammar, H. (2010). Object oriented design metrics and tools a survey.Informatics and Systems (INFOS), 2010 The 7th International Conference on.

Riaz, M., Mendes, E., and Tempero, E. (2009). A systematic review of software main-tainability prediction and metrics. Proceedings of the Third International Symposium on Empirical Software Engineering and Measurement, pages 367–377.

Rudiger Lincke, J. L. and Lowe, W. (2008). Comparing software metrics tools. In ISSTA '08 Proceedings of the international symposium on Software testing and analysis.

Sahar R. Ragab, H. H. A. (2010). Object oriented design metrics and tools a survey.Informatics and Systems (INFOS), The 7th International Conference on.

Saraiva, J., Barreiros, E., A., A., Lima, F., Alencar, A., Lima, G., Soares, S., and Cas-tor, F. (2012). Aspect-oriented software maintenance metrics: A systematic mapping study. In EASE'12: Proceedings of 16th International Conference on Evaluation and Assessment in Software Engineering.

Saraiva, J. A. G., Franc¸a, M. S., Soares, S. C. B., Filho, F. J. C. L., and Souza, R. M. C. R. (2015). Classifying metrics for assessing object-oriented software maintainability: A family of metrics' catalogs. The Journal of Systems and Software, pages 85–101.

SEBRAE (2006). Companies classification criteria in portuguese critérios de classificação de empresas. http://www.sebrae-sc.com.br/leis/default.asp?vcdtexto=4154.

Shull, F., Singer, J., and Sjoberg, D. I. K. (2009). Guide to Advanced Empirical Software Engineering. Springer.

Sommerville, I. (2011). Software Engineering in portuguese Engenharia de Software. Number 9. Pearson Prentice Hall.

Tecnopuc (2016). Tecnopuc. http://www3.pucrs.br/portal/page/portal/inovapucrs/Capa/Tecnopuc.

Usman, M., Mendes, E., and Borstler, J. (2015). Effort estimation in agile software de-velopment: A survey on the state of the practice. Proceedings of the International Conference on Evaluation and Assessment in Software Engineering.

Whippany, L. A. V. C. A. B. L. C. (1994). Integrated maintainability analysis: A practical case study. In Reliability and Maintainability Symposium.

Wohlin, C., Runeson, P., Host, M., Ohlsson, M. C., Bjorn, R., and Wesslen, A. (2012).Experimentation in Software Engineering. Springer.