How do Colombian software companies evaluate software product quality?

Wilder Perdomo[1 [0000−0003−1493−7307]], Julia Prior[1 [0000−0002−7182−0984]], and John Leaney[1 [0000−0001−5009−0401]]

University of Technology Sydney, 15 Broadway, Ultimo, Australia
Faculty of Engineering and Information Technology, School of Computer Science
Wilder.Perdomo@uts.edu.au
Julia.Prior@uts.edu.au
John.Leaney@uts.edu.au

Abstract. Software developers confuse product quality with process quality, leading them to think they are measuring product quality when they are not. This is the main finding of our study of software developers in young companies in Colombia. Software product quality (SPQ) reflects two perspectives: conformance to specifications, and satisfying expectations when it is used under specified conditions. Measuring product quality still remains a problem for software development companies in relation to factors such as cost, effort, time, and competitiveness. There are few studies that show the current state of SPQ in companies, how companies evaluate product quality, and which measures they use to develop and launch products that meet high-quality criteria.
This paper presents a study of SPQ in seven young software development companies in a developing country. We used a qualitative research approach to understand, through their experiences and knowledge, how 20 employees—developers, testers, and project managers—and their companies evaluate SPQ, and which measures they apply in their companies. Our results demonstrate that software process quality is better understood, and applied, by these software companies than SPQ. A greater difficulty is that most study participants 'overlaid' the idea of product quality with process quality, i.e. they talked about product quality as if it were process quality. These findings have implications for companies that wish to increase competitiveness and productivity, as they must develop a working knowledge of SPQ that is not confused with software process quality. It also has implications for educators, to ensure that the distinction between process and product quality is explicitly taught.

Keywords: Software product quality · software development · young companies · measures · open interviews · qualitative analysis.

1 Introduction

Software product quality (SPQ) reflects two perspectives: conformance to specifications and satisfying expectations when it is used under specified conditions [13, 17, 18, 37].

Software quality has gained major attention by software engineering researchers in the last three decades [41, 45], primarily focusing on the importance of the software process in the industry [20, 42]. Developing high-quality products requires a lot of effort, as developers have to also deal with challenges such as competitiveness, quality issues, and customer satisfaction during development [33]. Software product quality is thus becoming more of a concern [7, 20]; Perez *et al.* claim that "more companies and organisations insist not only on the quality of the processes that are followed in software development, but also on the quality of the products purchased or developed" [35, p.29]. Furthermore, SPQ is often not defined comprehensively, specifically or effectively because some approaches have focused separately on certain quality aspects such as software process quality characteristics, measures, definitions, and software stakeholders [9, 29].

Companies need to understand the importance of product quality and realise that the philosophy about "a quality process produces a quality product" [22] is no longer enough. *"... the software quality evaluations should be based on direct evidence about the product, not only on evidence about the process"* [24], since a high-quality process does not necessarily ensure a good quality product. However, measuring product quality by software development companies remains a problem related to factors such as cost, effort, time, and competitiveness [2, 44].

Several authors claim that companies lack industrial studies that show the current state of software product quality [23, 44], how best to evaluate the product quality, and which measures to use to deliver a good product to the market [1, 5, 8, 12, 34, 39, 40]. Furthermore, it may help developers and companies adopt SPQ measures to help them satisfy customer expectations, especially in countries where software development is in its youth [23, 26, 30–32]. Other studies state that the existence of inappropriate SPQ measurement approaches predominate, generating high costs in the remediation, defects correction, product development, user dissatisfaction and low competitiveness in global markets [10, 16, 38].

This paper presents a study carried out with seven young software development companies in a developing country that examines how they evaluate software product quality and which measures they apply in their companies. The problem of low competitiveness and SPQ evaluation affects the young software development companies that represent 19% of the 590 companies in the software and ICT industry in Colombia [11, 36]. Such companies must have between two and five years of operation to be considered a young company and have between one to 200 employees. We employed a qualitative research approach to under-

stand how 20 employees, including developers, testers, and project managers, and their companies, understand and assess SPQ. This data came from the experiences and knowledge of developers via semi-structured interviews.

This paper is organized as follows. In Section 2, we present a literature review. In Section 3, we provide an overview of the qualitative research approach that we used for this study. In Section 4, our data collection and analysis are explained in detail. In Section 5 we present the findings and discussion. Finally, in Section 6, the conclusions of our study are summarised and we describe possible future work.

2 Literature Review

This section is a portion of a systematic literature review following the guidelines proposed by [21] and presented in [33]. This literature review focuses on issues of software product quality and how software development companies measure their SPQ, focused on Colombia. Note that, in the literature, software product quality is also referred to as non-functional quality. This empirical research cannot do a really direct comparison with other countries because they are not related to our study. Pelaez *et al.* [30] describe an exploration of recognized proposals focused on software quality, certifications, and organizations focused on improving software quality in Colombia. They state that Colombian companies have implemented ITMark approach that is a model based entirely on CMMI as a guide to systematic and organized steps that allows reaching optimal levels of capacity and maturity, and Light MECPDS approach that is an agile model of software process improvement based mainly on agile methodologies and principles for small and medium enterprises (SME). Those approaches are the most recommended practices for improving the software quality process in young business. In addition, these authors emphasize the Colombia SME must have high quality to be competitive in the worldwide market. They do not discuss SPQ.

Lampasona *et al.* [23] present experiences in developing custom-tailored quality models for Ecopetrol, a Colombian oil and gas company. They describe the creation of the quality model for the IT department to improve software quality, reduce the issues caused by unknown/probably poor software quality, and in turn contribute to the ability to develop and maintain software faster and support the company in becoming more agile. The custom-tailored quality model for Ecopetrol works, in that they defined and agreed upon a set of measurable quality goals that define the oil company's quality focus. The measures chosen were integrated into a comprehensive quality model, and the authors are working on visualizing the analysis results in an intuitive manner. Although the model is customized for the company's needs, they do not describe their model and how it works with the measures selected.

Escobar and Linares [8] show a model which could be used for measuring companies agility in four different levels i) project, ii) project management, iii) work-team, and iv) agile work-spaces coverage. They are focused more on management process rather than on product quality. This contribution helps to understand the current state of quality measurement, competitiveness, and productivity in Colombia.

Metaute and Serna [26] present an assessment of the ownership and use of metrics in medium enterprises in Medellin-Colombia, seeking to make recommendations that contribute to the strengthening of academia to support the industry. They found that some companies applied metrics during the years 2013-2015 in different stages of the software development life cycle. In addition, they use models such as CMMI, ISO/IEC 9000, ITIL, and customized methods to evaluate the software process quality. However, they did not ask companies about which product measures were implemented and how they are doing the product evaluation process. The authors combine different concepts and meanings from process quality and product quality, which leads to difficulty in understanding.

Febrero *et al.* [10] show the existing work on the modeling of software reliability based on ISO/IEC 25000 standard as the starting point for a reliability assessment proposal. They provide two main contributions: a systematic review of standard based software reliability modeling literature and an innovative method with which to model software reliability that integrates the stakeholders' needs. They describe that the standards are well constructed, but they do not appear to have had a great impact on academia and industry. They establish a reliability model layout and assessment schema, but they do not describe what measures they are using and how to evaluate the software product quality as a whole.

Nakai *et al.* [29] propose a SQuaRE-based software quality framework, which successfully made tangible many product metrics and qualities in use. These metrics were originally defined in the SQuaRE series [19]. Most of the work on quality in use is in human computer interaction (HCI). For instance, the authors validate that the framework is practically applicable to the software package or service product. However, they do not show how they selected 47 product metrics and 18 quality in use metrics, or how to evaluate SPQ in young software development companies. (This framework describes a procedure created to assess SPQ in large companies).

Fernandez *et al.* [12] propose a new model for assessment and selection of software products according to their quality. The model is validated with some non-functional requirements and measures. The process and the model could help companies to choose SPQ measures and how to evaluate them. However, they do not describe the SPQ evaluation processes and they lack justification as to how and why they choose the non-functional requirements and measures.

Baquero *et al.* [1] highlight the importance of usability and accessibility as quality factors of a secure web product. These aspects of quality of the products have reached the level of being a demand of the market and have become a differentiating factor for an increasingly demanding customer base. They state that in order to mitigate threats and vulnerabilities it is necessary to implement methodologies that guarantee a good software product. However, they do not describe a process or model to guarantee their claims or for describing usability as a non-functional requirement and some quality factors including related measures. In addition, they do not explain the SPQ evaluation process.

A major issue for Colombian companies with models or standards recognised in the software industry is that they were created to be implemented in large companies with high financial capability and solid experience. The high financial capability refers to the cost for extra resources such as time, human resources, and the payment for expensive certifications to achieve the goal of being recognised as a company with high-quality levels. It limits the ability of young Colombian companies in a developing country, because of their size, extra resources, and experience for implementing measurement models and practices of SPQ.

The aforementioned studies describe different customized models integrating some measures in their descriptions, but they do not explain: how those models are working in a real software project, which measures and relationships they have established, or help understand SPQ evaluation in young software development companies. The gap of understanding SPQ evaluation in young software development companies is the principal focus of this empirical research.

3 Research Approach

At the beginning of this study, we applied a quantitative research approach via a closed questionnaire, which had as the objective to find out which characteristics, measures, and methods are currently used by Colombian software developers to evaluate software product quality. The target participants were software development companies and their developers, testers, and project managers. We ran a pilot study on 11 people from four companies to test the questionnaire before proceeding to collect the final data. The pilot findings showed that the interviewees did not understand some questions, concepts, and elements defined in the survey, and the results were not useful.

As we were interested in what they did understand and why, a qualitative approach to explore software developers' experiences with and knowledge of SPQ was appropriate for further research. A phenomenological approach contributes to exploring and understanding the meaning that individuals or groups ascribe to a human or social problem [6]. So, we used a phenomenological approach, focused on software developers' lived experience, which would enable us to ex-

plore their understanding of SPQ. The steps followed in this approach were i) design the interview, ii) selection of interviewees, iii) discovering meanings, iv) data saturation, and v) validity of the analysis. This part of the research focused on the following objectives:

1. identify what does product quality mean to Colombian software companies;
2. classify software developers' experiences when they evaluate software product quality;
3. explore how organizations evaluate software product quality; and
4. identify which characteristics and measures software companies use to evaluate software product quality.

One technique used to get the necessary data is the semi-structured interview, with open-ended questions, which permits the understanding of the participants' experience and issues. Questions driving the interviews were grouped into four categories: i) classification or demography, ii) SPQ in the organisation, iii) SPQ personal perspective, and iv) SPQ evaluation at both the personal and organisational level. The participants answered research questions based on i) what do software developers/companies evaluate in product quality, and ii) how do software developers/companies evaluate software product quality. This study followed the steps described in [25] to design interviews and apply pilot tests.

3.1 Validity

We consider how to demonstrate the validity of our methods in this section.

Data Saturation. One of the markers of valid phenomenological research is data saturation. Data saturation is about the depth of the data [4]. The depth of the data is expressed in terms of quality, as the richness of the data [4]. Data saturation is reached when no new or relevant data seem to emerge regarding a pattern [15]. We planned to carry out 20 interviews initially. We may have done more if, after the initial analysis, data saturation was not reached. Guest *et al.* in an analysis of the development of ideas in a research study, found that saturation occurred within the first twelve interviews, although basic elements for patterns were present as early as six interviews [15].

Method Validity. To ensure that the process used is valid, we adopted a well-researched iterative method, further described in 4.1. This uses iterations between the collection, data analysis and conclusions drawn.

Validity of Analysis

Data Saturation in Practice. For all the patterns, we considered saturation had been reached when the authors agreed that no new or relevant data emerged regarding any patterns. This was the outcome of the iterative process described below.

Iterative Method Validation. The first author did the initial coding, and preliminary patterns. The coding and patterns were then reviewed and refined

by the second author with the first author. The coding and patterns were then reviewed and refined again by the third author with the first author. The three authors then reviewed the patterns and developed the overarching theme together.

4 Data Collection

In this section, we describe in more detail the data collection and analysis processes that we used. This study follows Miles *et al.* interactive model [28] to guide the thematic and interpretive analysis. We performed both manual analysis and semi-automatic analysis using NVivo.

4.1 Overview of the Approach

The approach has three concurrent flows of activity: data analysis, data display, and conclusion drawing and verification/validity [3, 27, 28]. Each of these three components continues during and after data collection. Data analysis involves analytic choices based on key phrases and nodes. Data display serves to organize and compress information, making it amenable to further analysis and interpretation, and meanings drawn from the data have to be tested against the data, with more being sought as necessary [3]. The continual validation is one of our strongest claim to validity. We iterated the approach cycle until we had reconciled all parts of each to the other. In other words, our termination of the iterations is defined by no discernible differences in patterns found by each of researchers.

4.2 Interviews

The aim of this research is to understand how software development companies and their employees evaluate and measure software product quality. The first and second authors designed a set of open-ended questions to use in interviews. Participants interacted directly with the first author through video-conference software for the interviews, sharing their views, experiences, and knowledge about software quality. The duration of each interview was 30 minutes and they were audio-recorded on the first author's laptop, as well as with the Voice Memos app on their phone.

The criteria to select the participants were two-fold: i) they had to be employees for a small (11-50 employees) or medium (51-200 employees) enterprise (SME), and, ii) they had to have at least two years of professional experience working in a software development area, i.e. "the experience of developing software" [14]. Table 1 summarises the participants classification. For this classification, Market is defined as a combination of Local (L), National (N), and International (I). These classifications describe the type of participant companies, the interviewees' roles in their companies. Most importantly, this information

provides some context for the study and its participants, as well as presenting some of its limitations with regard to scope. Further to the scope of the study, interviews were conducted within seven companies, with three employees from each young software development company (developer, tester, and project manager), except one which had two interview participants. So, we conducted a total of 20 interviews.

Each interview included classification questions (company size, role, market, and industry), as described above, and four open-ended questions about software product quality, which are specifically targeted for the participant vs company. The open-ended questions were:

Table 1. Participants classification

Size	**Role**	**Market**	**Industry**
Small (2 years of creation)	Project manager	L	Services, financial, agroindustry
		L/N	Services, automotive, manufacturing, food
	Developer	N	Education, agroindustry, private sector, financial
		N	All types of industries
		L	Manufacturing
		N	Government, agriculture, aeronautics, telecommunications
	Tester	N	Services and agroindustry
		N	Services and agroindustry
		N	Services, automotive, agroindustry
		L	Services, automotive, manufacturing, food
		N	Government and private sector
		N	Government, agriculture, aeronautics, telecommunications
Medium (3-6 years of creation)	Project manager	L/N/I	Energy
		N/I	Energy
		N/I	All types of industries
	Developer	L/N/I	Energy
		N/I	Energy
		N/I	All types of industries
	Tester	N/I	Energy
		N/I	Energy

Market: Local (L), National (N), and International (I).

1. What does software product quality mean for your organization?
2. What about your everyday work is related to software product quality?
3. How does your organization evaluate software product quality?
4. How do you evaluate software product quality?

Once the research team had started with the reading and understanding process of each transcript, we decided to ask more questions to some interviewees to clarify some details from their responses. These questions were relatively simple, and did not require the deep exploration of the open-ended questions. Participants replied to the extra questions in brief, follow-up interviews, which were transcribed from Spanish to English and included in their files respectively.

The interview instruments and procedures were piloted with three employees of a Colombian software development company. This experience confirmed our expectation that interviews would take approximately 30 minutes and also led to some changes in the delivery and sequencing of questions.

4.3 Thematic Analysis

To begin the analysis, the first author spent several weeks translating all the interviews from Spanish to English, and then performed four iterations reading through each question and response transcripts to understand the data. The second author acted as an advisor, participated in essential dialectic discussions several times per week with the first author, during the interview period and beyond, to clean and organize the data and to clarify the interviewees' responses. The third author acted as an adjudicator and evaluator of the interview data, analysis, and interpretations.

Coding. The research team read through all the responses question-by-question, eight times during four iterations, and highlighted key phrases (clauses, sentences, keywords, etc.), which included actions, activities, concepts, differences, opinions, or anything else intriguing. We also annotated the responses, to clarify meaning and our understanding, but we did not change the participant responses or their words in any way. During this process, we created different descriptive tables to include key information such as *ID* or *pseudonyms*, answers for each participant, and keywords and phrases. Participant names are anonymous, their names are unidentified.

Codes, Nodes and Pattern Creation. Table 2 describes the definitions of entities involved in the analysis and creation of codes, nodes and patterns. After establishing confidence in the meaning attributed to the codes, we grouped them together into nodes, based on similarity. We then developed patterns based on the nodes. Sometimes, we changed the pattern, or read the transcript sentence again, to further understand it, as suggested in Section 4.1.

Table 2. Definitions

Item	Description
Source	Research materials including documents, interviews, and audio
Code	Key phrase from the source data. Coding "*leads to breaking down data into incidents and examining their similarities and differences*" [43]
Node	Container that represents collection of codes which have a strong similarity
Pattern	Collection of nodes with common ideas
Theme	A dominant pattern, characterised by a large number of nodes and connections to other patterns

4.4 Resulting Patterns

We initially identified 12 patterns, which were subsequently merged to form 10 patterns due to considerable overlap in shared nodes. Table 3 shows these 10 patterns. Each pattern is presented with its name, description, and the number of occurrences of the codes forming that pattern in the interview data.

5 Findings and discussion

In this section, we discuss the findings of our study and their significance to the evaluation of software product quality.

5.1 The Pattern "process quality over product quality"

The strongest and most significant pattern to emerge from the analysis, we named "process quality over product quality". Although the participant developers and their companies tend to have well-developed approaches to process quality, including customized methodologies, procedures, and following CMMI guidelines, they do not have well-defined approaches to product quality.

All except one participant indicated that software quality assurance in their companies is dominated by software process quality and not by software product quality, and they do not have well-defined SPQ assurance procedures. Just over half of the 20 participants confuse product quality with process quality. More than that, these participants 'overlaid' the idea of product quality with process quality, i.e. they talked about product quality as if it was process quality.

In examining their answers across the four questions, 16 participants explained that software quality is key. However, all their answers are about software process quality, even though the questions are specifically about software product quality. This is exemplified in the following statements:

Table 3. List of Patterns

Name	Description	#Codes
A.Customer satisfaction	Customer satisfaction is achieved for companies by fulfilling the acceptance criteria, generating value to the customers, and giving satisfaction to the final user	29
B.Fulfilling customer requirements	Customer requirements are fulfilled and supported during requirements verification and validation. This ensures that customer needs as a whole, from the technical and business requirements, are included in the final product	37
C.Customized protocols and methodologies	Protocols and methodologies are customized for software quality verification and validation. Such customized protocols have activities related to performing engineering process, peer reviews and implementing checklists for reviewing	77
E.Manual and automation tests	Manual and automation tests are executed for evaluating software product quality. Such tests are smoke, functional regression, performance, user interface, and coding according to the context and company	24
F.Measures and indicators evaluated	Colombian young companies are evaluating software quality with approximately 50 different measures and indicators	36
G.Process quality over product quality	Companies are focused on process quality rather than product quality. Some companies have customized methodologies, procedures, and alleged CMMI compliant process. However, they present product quality processes that are not well defined	17
H.Existing procedures, best practices, guidelines, and policies	Best development practices, guidelines, and policies are implemented for evaluating software quality	21
I.Project management skills	Project management skills are implemented for planning and controlling the team with good practices and strategies, defining a dynamic method for reviewing how the results came out, which defects are identified, tracking tests execution, and management estimation	36
K.Software product quality characteristics	Software quality characteristics (such as functionality, maintainability, portability, security, and usability) are implemented for assessing product quality	61
L.Things are not well	Things related to documentation, quality characteristics and measures, and automated tests are not working well in the young software development companies	26

– *"Software quality is important because we know that for customers it is essential that the product is useful and functional. The objective as quality analysts is to ensure that the quality process of software manufacturing meets the re-*

quirements defined by the customer and the process defined by the quality area." (George)
- *"Quality is the key to the company's success. In addition, some added values are made to give security and confidence to the customer that the information that is being downloaded is real."* (Julien)
- *"That the product meets standards at the level of coding and performance."* (Nick).

Despite having a clear idea of quality and its importance, most participants are focused only on process quality. In response to the questions on product quality, they responded with the following quotes, which demonstrate the tendency to overlay, or conflate, product quality with process quality.

- *"We are focusing on process quality. The process for controlling the logs/bugs is on implementation (because we do not have measures); the company has generated the indicators slowly."* (Anna)
- *"Within the organization, we have indicators. We follow the development and testing process" and "this process is important during the validation because the customer is who gives feedback on whether the product meets the agreed-upon requirements."* (Kyra)

Only four participants demonstrated that they understood software product quality, as distinct from process quality:

- *"The organisation has two procedures: mature products which are in the market (maintainability), and new products (usability and functionality)."* (James)
- *"Product quality is part of the company's mission, which is providing services to the customer to guarantee good products."* (Angel)
- *"What we seek is that the functionality corresponds to what should be implemented, which is determined by the use cases, user stories, and quality attributes that are fulfilled with the expected maintenance, performance, availability, and other quality attributes that are defined for a particular project".* (Lachlan)
- *"We try to ensure the quality from the requirements stage. Checklists are made to review from this stage some features to meet the usability criteria."* (Karol)

In summary, 80% of the interviewees understand that they need to attend to quality in their companies, but are focused only on process quality. Only a few of the interviewees appear to have a good understanding of SPQ. Just over half of the interviewees appear to not understand the difference between software process quality and software product quality. Furthermore, the latter group confuse process quality with product quality. As a consequence of this confusion, they do not recognize that they are not, in fact, measuring product quality.

5.2 The Theme "Process Quality over Product Quality"

When we re-examined the other patterns, we found that many of them substantially reinforced Pattern G "process quality over product quality". We thus developed this dominant pattern into an overarching theme, and relate these other patterns (A, B, C, E, F, H, I, K, L) to Pattern G because they provide further evidence for the finding that developers confuse process quality and product quality. Customer satisfaction (usability) and fulfilling customer requirements (functional suitability) are clearly product quality characteristics, but the participants talk about them as if process quality is needed to assure it [1].

The following patterns reinforce that participants are focusing on software process quality (SQ) over SPQ. The pattern names and their relevance to process quality over product quality are given below. Table 3 has more complete definitions of the patterns.

Pattern A (Customer satisfaction)—twelve participants claim to be focused on customer satisfaction and perception, but most did not talk about measures of customer satisfaction. Only three of these twelve mentioned SPQ measures such as usability, process, and product measures for customer satisfaction.

Pattern B (Fulfilling customer requirements) reinforces the theme by the lack of SPQ attributes as requirements. Only four participants talked about fulfilling well-defined SPQ criteria as important.

Pattern C (Customized protocols and methodologies)—19 participants customize protocols and methodologies for verifying and validating software process quality, rather than SPQ. This supports the idea that software development companies are focused more on software process quality than SPQ.

Pattern E (Manual and automation tests) reinforces the theme by the lack of SPQ evaluation criteria. Sixteen participants focused on applying manual and automatic code tests (but not product tests) during their quality evaluation process.

Pattern F (Measures and indicators evaluated) strengthens this theme by the lack of SPQ attributes as measures and indicators. Fifteen participants evaluate software process quality with approximately 50 process measures and indicators.

Pattern H (Existing procedures, best practices, guidelines, and policies) support this theme by the lack of SPQ evaluation practices or guidelines. 11 participants focused on the best development practices and following guidelines and policies provided by the company for evaluating software process quality alone.

Pattern I (Project management skills) reinforces the idea by the lack of SPQ evaluation skills as project management indicators. Half of the participants are implementing project management skills to organize the team and make the quality process dynamic, but they are not focused on managing product quality.

Pattern K (Software product quality characteristics) reinforces this theme by the lack of SPQ attributes as characteristics and measures. Most participants focused on software process quality. As mentioned previously, only four partic-

ipants focused on SPQ characteristics including functionality, maintainability, portability, security, and usability.

Pattern L (Things are not well) reinforces this theme by the lack of SPQ evaluation attributes. Thirteen participants argue that things need to improve in relation to documentation, characteristics, measures, and automated tests.

6 CONCLUSIONS

In this paper, we present our study aimed at understanding how young software development companies evaluate software product quality and which measures they apply in their companies to do so. This research was driven by the assertion that software product quality assurance is essential to producing high quality products, as argued by Maibaum and Wassyng [24]: *" . . . the software quality evaluations should be based on direct evidence about the product, not only on evidence about the process"*.

We used phenomenological research to investigate participants' experiences in software quality. Our findings demonstrate that young software companies in Colombia focus more on process quality than software product quality. Moreover, software developers in these companies confuse product quality with process quality, and even conflate the former with the latter. As a consequence, they mistakenly believe that they are evaluating software product quality, when, in fact, they are not. This finding has implications for quality assurance of these companies' software products, and the companies' competitiveness and success [10, 38].

The companies could distinguish software product quality from software process quality, being focused in the product rather the process for getting the software product. In addition, we suggest that the companies at the beginning of their projects could follow some steps to evaluate SPQ. First, identifying which particular product quality characteristics (so-called non-functional requirements) you are implemented in your project or you are going to implement (i.e. usability, security, maintainability, so on). Second, classifying the measures that you are going to assess for identifying some issues in the software product (according to some models/standards, i.e. ISO/IEC 25000). Third, applying the measures selected to the software product according to the criterion given by the models/standards. Finally, analyzing the results generated and propose some strategies to mitigate or fix the issues before launching the final product.

Although this study presents an empirical understanding of software product quality, of course, it is not a complete picture of how SPQ is evaluated in practice in general. The following limitations should be considered in assessing and using this research: i) the phenomenological research design, ii) the seven participants companies are only young Colombian software development companies (2-5 years old), iii) 20 interviews were carried out with individual participants from these

companies, iv) participants are all software developers, testers, and project managers, each with a minimum of 2 years' experience in a software engineering role, v) the interviews were semi-structured, consisting of four open-ended questions, vi) the interviews were conducted in Spanish via video-conferencing tools, and after those were translated to English, vii) thematic analysis was performed to understand how these participants and their companies evaluate software product quality, and viii) the author's background as both an experienced professional software engineer and a lecturer in software engineering for many years.

We plan to propose an economical and flexible model for evaluating software product quality in young software development companies. The aims of the model are to help reduce workload, time, and resources in order to make SPQ assurance more economical.

We also plan to build communities of practice to share the meanings and criteria to consider SPQ as a key factor for software development projects. They could contribute to SPQ evaluation and building new understanding with other companies' experiences and knowledge. These groups could also generate a thematic guideline by industry (i.e. finance, services, health, information technology, manufacturing, education, etc.) for recording best practices to evaluate their software products according to non-functional requirements.

References

1. Baquero, L., Gil, C., Hernández, M.: The usability and accessibility as quality factors of a secure web product. International Journal of Applied Engineering Research **13**(23), 16288–16294 (2018)
2. Barney, S., Wohlin, C.: Software product quality: Ensuring a common goal. In: International Conference on Software Process. pp. 256–267. Springer (2009)
3. Bazeley, P.: Qualitative data analysis: Practical strategies. Sage (2013)
4. Burmeister, E., Aitken, L.M.: Sample size: How many is enough? Australian Critical Care **25**(4), 271–274 (2012)
5. Carvalho, F., Meira, R.L., Freitas, B., Eulino, J.: Embedded software component quality and certification. In: 35th Euromicro Conference on Software Engineering and Advanced Applications. pp. 420–427. IEEE (2009). https://doi.org/10.1109/SEAA.2009.90
6. Crotty, M.: The foundations of social research: Meaning and perspective in the research process. Sage (1998)
7. Curcio, K., Malucelli, A., Reinehr, S., Paludo, M.A.: An analysis of the factors determining software product quality: A comparative study. Computer Standards & Interfaces **48**, 10–18 (2016)
8. Escobar, V., Linares, M.: A model for measuring agility in small and medium software development enterprises. In: XXXVIII Conferencia Latinoamericana en Informatica (CLEI). pp. 1–10. IEEE (2012)
9. Fayad, M.E., Laitinen, M., Ward, R.P.: Thinking objectively: software engineering in the small. Communications of the ACM **43**(3), 115–118 (2000)
10. Febrero, F., Calero, C., Moraga, M.A.: Software reliability modeling based on ISO/IEC SQuaRE. Information and Software Technology **70**(2), 18–29 (2016). https://doi.org/10.1016/j.infsof.2015.09.006

11. Fedesoft: Caracterizacion del sector teleinformatica, software y ti en colombia. Report (2015)
12. Fernandez, Y., Cruz, C., Verdegay, J.L.: A new model based on soft computing for evaluation and selection of software products. IEEE Latin America Transactions **16**(4), 1186–1192 (2018)
13. Garvin, D.A.: What does "product quality" really mean? Sloan management review **25** (1984)
14. Goulding, C.: Grounded theory, ethnography and phenomenology: A comparative analysis of three qualitative strategies for marketing research. European journal of Marketing **39**(3/4), 294–308 (2005)
15. Guest, G., Bunce, A., Johnson, L.: How many interviews are enough? an experiment with data saturation and variability. Field methods **18**(1), 59–82 (2006)
16. Idri, A., Bachiri, M., Fernández-Alemán, J.L.: A framework for evaluating the software product quality of pregnancy monitoring mobile personal health records. Journal of medical systems **40**(3), 50 (2016)
17. ISO24765: Systems and software engineering - vocabulary (ISO/IEC:24765) (2010)
18. ISO25000: Systems and software engineering - systems and software quality requirements and evaluation (SQuaRE) - guide to SQuaRE (ISO/IEC:25000) (2014)
19. ISO25023: Systems and software engineering - systems and software quality requirements and evaluation (SQuaRE) - measurement of systems and software product quality (ISO/IEC:25023) (2016)
20. Jinzenji, K., Hoshino, T., Williams, L., Takahashi, K.: An experience report for software quality evaluation in highly iterative development methodology using traditional metrics. In: 2013 IEEE 24th International Symposium on Software Reliability Engineering (ISSRE). pp. 310–319. IEEE (2013)
21. Kitchenham, B.: Procedures for performing systematic reviews. Keele, UK, Keele University **33**(2004), 1–26 (2004)
22. Kitchenham, B., Pfleeger, S.L.: Software quality: the elusive target. IEEE software **13**(1), 12–21 (1996)
23. Lampasona, C., Heidrich, J., Basili, V.R., Ocampo, A.: Software quality modeling experiences at an oil company. In: Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement. pp. 243–246. ACM, 2372296 (2012). https://doi.org/10.1145/2372251.2372296
24. Maibaum, T., Wassyng, A.: A product-focused approach to software certification. Software Technologies **41**(2), 91–93 (2008). https://doi.org/10.1109/MC.2008.37
25. Marshall, C., Brereton, P., Kitchenham, B.: Tools to support systematic reviews in software engineering: a cross-domain survey using semi-structured interviews. In: Proceedings of the 19th international conference on evaluation and assessment in software engineering. p. 26. ACM (2015)
26. Metaute, P., Serna, A.: Diagnostic on the appropriation of metrics in software medium enterprises of medellin city. Revista Antioquena de las Ciencias Computacionales y la Ingenieria de Software **6**(1), 6–14 (2016)
27. Miles, M.B., Huberman, A.M., Huberman, M.A., Huberman, M.: Qualitative data analysis: An expanded sourcebook. sage (1994)
28. Miles, M.B., Huberman, A.M., Saldana, J.: Qualitative data analysis: A Method Sourcebook. Sage (2014)
29. Nakai, H., Tsuda, N., Honda, K., Washizaki, H., Fukazawa, Y.: A square-based software quality evaluation framework and its case study. In: 2016 IEEE Region 10 Conference (TENCON). pp. 3704–3707. IEEE (2016)

30. Pelaez, L.E., Hurtado, R.A., Franco, J.A.: Certificacion de la calidad del proceso y producto: ruta para pymes colombianas que fabrican software. Ventana Informatica **25**(2), 41–61 (2011)
31. Perdomo, W., Zapata, C.M.: Identificacion de criterios para relacionar la usabilidad con el alfa sistema de software del nucleo de semat. In: Proceedings of the Latin American Software Engineering Symposium (LASES). pp. 17–22 (2015)
32. Perdomo, W., Zapata, C.M.: Software quality measures and their relationship with the states of the software system alpha. Ingeniare. Revista chilena de ingeniería **29**(2), 1–23 (2021)
33. Perdomo-Charry, W.: How do Colombian software companies evaluate software product quality? Ph.D. thesis, University of Technology Sydney (2020)
34. Perdomo-Charry, W., Zapata-Jaramillo, C.M.: Modelo para la medición del progreso del alfa sistema de software del núcleo de Semat con base en las normas ISO/IEC 2502n e ISO/IEC 2504n. Ph.D. thesis, Universidad Nacional de Colombia-Sede Medellín (2019)
35. Perez, M., Velez, Y., Ayaz, M.: Evaluation of the non-functional requirements of usability: A systematic study. International Journal of Advanced Research in Computer Science **7**(3) (2016)
36. Pino, F.J., Ruiz, F., Garcia, F., Piattini, M.: A software maintenance methodology for small organizations: Agile-mantema. Journal of Software: Evolution and Process **24**(8), 851–876 (2012). https://doi.org/10.1002/smr.541
37. Pressman, R.S.: Software engineering: a practitioner's approach. Palgrave Macmillan, United States (2005)
38. Rodriguez, J.I., Sierra, E., Jaramillo, L.K.: Model for measuring usability of survey mobile apps, by analysis of usability evaluation methods and attributes. In: 10th Iberian Conference on Information Systems and Technologies (CISTI). pp. 1–6 (2015). https://doi.org/10.1109/CISTI.2015.7170420
39. Rodriguez, M., Piattini, M.: Systematic review of software product certification. In: 7th Iberian Conference on Information Systems and Technologies (CISTI),. pp. 1–6. IEEE (2012)
40. Sánchez-Gordón, M.L., O'Connor, R.V.: Understanding the gap between software process practices and actual practice in very small companies. Software Quality Journal **24**(3), 549–570 (2016)
41. Sjoberg, D.I., Dyba, T., Jorgensen, M.: The future of empirical methods in software engineering research. In: 2007 Future of Software Engineering. pp. 358–378. IEEE Computer Society (2007)
42. Solyman, A.M., Ibrahim, O.A., Elhag, A.A.M.: Project management and software quality control method for small and medium enterprise. In: 2015 International Conference on Computing, Control, Networking, Electronics and Embedded Systems Engineering (ICCNEEE). pp. 123–128. IEEE (2015)
43. Vaismoradi, M., Jones, J., Turunen, H., Snelgrove, S.: Theme development in qualitative content analysis and thematic analysis (2016)
44. Wagner, S.: Software product quality control. Springer, Germany (2013)
45. Wirth, N.: A brief history of software engineering. IEEE Annals of the History of Computing **30**(3), 32–39 (2008)