UNIVERSITY
OF TECHNOLOGY
SYDNEY

How do Colombian software companies evaluate software product quality?

by Wilder Perdomo-Charry

Thesis submitted in fulfilment of the requirements for
the degree of

Doctor of Philosophy

under the supervision of Associate Professor Julia Prior and
Adjunct Professor John Leaney

University of Technology Sydney
Faculty of Engineering and Information Technology

February 2020

AUTHOR'S DECLARATION

I, *Wilder Perdomo-Charry* declare that this thesis, is submitted in fulfilment of the requirements for the award of Doctor of Philosophy, in the *School of Computer Science*, *Faculty of Engineering and Information Technology* at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This document has not been submitted for qualifications at any other academic institution.

This research is supported by Colfuturo Colombia and University of Technology Sydney.

This research is supported by the Australian Government Research Training Program.

Production Note:

SIGNATURE: Signature removed prior to publication.

[Wilder Perdomo-Charry]

DATE: 12th February, 2020

PLACE: Sydney, Australia

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 this study of software developers in young small to medium companies in Colombia.

Software product quality 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 software product quality in companies, how companies evaluate product quality, and which measures they use to develop and launch products that meet high-quality criteria.

This research presents a study of software product quality in seven young software development companies in a developing country. The candidate used a qualitative research approach to understand, through their experiences and knowledge, how 20 employees—developers, testers, and project managers—and their companies evaluate software product quality, and which measures they apply in their companies.

The results demonstrate that software process quality is better understood, and applied, by these software companies than software product quality. 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. This confusion leads them to think that they are measuring product quality when they are not.

These findings have implications for companies that wish to increase competitiveness and productivity as they must develop a working knowledge of software product quality 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.

Dedication

To God who is present in every place, moment and circumstance, and who always accompanies me on the path of life. To my mother, my father (rip), and Maria my wife, who are proof of love, persistence, and family union. Fundamental criteria to achieve a goal.

Wilder Perdomo…

Acknowledgments

Completion of this doctoral thesis would not have been possible without a number of people whom I would like to acknowledge and thank for their support, advice, and encouragement.

Firstly, an enormous thank you to the participant developers, testers, and project managers from the seven software development companies in Colombia who so generously allowed me to know their experiences in the companies. The CEO from each company gave exceptional help and was present during our contact with good enthusiasm and interest in this research, giving me access to whomever and whatever I deemed necessary to understand and carry out my fieldwork. I am extremely grateful to them all. I trust that the stories of their work experiences in this Phenomenological research and the findings of the research will make them proud to have played such a crucial part in the development of this project.

I also want to thank to Fedesoft (Federación Colombiana de la Industria de Software), Procolombia, and MinTIC (Ministerio de las Tecnologías de la Información y la Comunicación) in Colombia for allowing me to introduce my research and to be open to help me carry out my fieldwork with some Clusters and companies from different Colombian cities.

My UTS supervisor, Julia Prior who believed in me and recognised my research capabilities to develop this research and help me out to execute this journey. We know that this journey has had its up and down situations, but she was a good advisor in each step of my thesis. I would like to thank her for challenging me to think about epistemologies and research methodologies, interview techniques, reflection, software development in practice, software product quality, and a host of other disciplines. Her support and enthusiasm for my work never faltered. Most of all I would like to thank her for being able to readily understand my goal.

My mentor and UTS co-supervisor, John Leaney, was essential in this journey, as his experience and rigour in the research were vital for developing my project. I would like to thank him for challenging me to think critically, focusing on my topic and how my contributions can help Colombia and the young software development companies. John is the type of supervisor who is dedicated, responsible, enthusiastic, generous, and with enough knowledge to share in areas related to software development and software quality. I am glad to find people like him who always keep an eye on my research and on the clear cohesion of my problem with the research method, data analysis, and findings.

I have the honour to have been supervised in this research by two great people. Julia and John showed me alternative approaches to research in general, and software engineering practice in particular, two great aspects I had not previously experienced or had not realised were scholarly options. They were dedicated and engaged with me and my research topic with the best practice to advice PhD students from both professional and personal aspects. Thank you both for your commitment and belief in me.

I was privileged to share the joys and despairs of the PhD journey with my dear friends Muhammad Awais Bajwa and Avirup Dasgupta. We travelled this journey together over some years, although in different types of research. Muhammad's encouragement and advice helped me over many a hump in the road and Avirup and I debated software development topics and methodologies to apply in our studies. In particular, they helped me work through new ways of thinking about knowledge and our understanding of the world. I thank you all.

I also thank other members of UTS and friends for contributing to and being part of this research journey. I thank Marie Manidis for her role as my mentor during my hard days at UTS, her encouragement and advice helped me over many a hump in the road. MaryAnn McDonald from UTS:HELPS and Wendy Boynton from ABC News who believed in my English skills and helped me to improve those skills to increase my knowledge and understanding in this journey. Tom McBride for his role as my acting supervisor at the beginning of my candidature particularly, but also for his advice and determination. I would like to thank Jing Zhao from Graduate Research School (GRS) who listened to me when I was up and down and especially for her advice during my scholarship process. Thanks a lot to Carol Charman for your tremendous help during the reviewing of my thesis; she was incredible at checking my English and editing my draft and final thesis document. My dear friends, Nick, Asma, Tazin, Felipe, Luke, Wang, Federico, Kamal, Durga, Carlos, Suman, Gautum, Joanne, and James who helped me in different emotional, personal, and professional ways to focus my efforts on this PhD journey. Thank you all.

I want to thank to my principal sponsor, COLFUTURO Colombia (Fundación para el Futuro de Colombia) who award me a Doctoral Research Scholarship to do my PhD studies in Sydney and gave me the opportunity to live this fantastic experience in a developed and multicultural country. Furthermore, I would like to thank the University of Technology Sydney (UTS), especially to the Graduate Research Program for awarding me the UTS–International Research Scholarship to continue my PhD research.

I would like to thank and apologise to my family especially my beloved parents, Leo Charry and Serafin Perdomo (RIP) for my long absences —both physical and mental absences during these years when I could not drag my head out of the work. I am very grateful to them for inspiring me every moment of my life and showing me that education is an essential tool in life. Thank you all for your support and strength.

Finally, it is my privilege to thank my wife and hero, who gave me her unconditional support, experience, and knowledge to address this goal that we have always considered ‘ours‘. Thanks a lot. I have always thought that working as a team, with vision and persistence helps us move forward and achieve great goals.

LIST OF PUBLICATIONS

RELATED TO THE THESIS :

1. Perdomo, W., Prior, J., Leaney, J. (2019), 'Software product quality (SPQ) evaluation at young software companies from a developing country.' In: *School of Computer Science HDR Student Research Showcase 2019*.

2. Perdomo, W., Prior, J. Leaney, J. (2020), *How do Colombian software companies evaluate software product quality?*, in Proceedings of the 30th international Workshop on Software Measurement (IWSM) and the 15th international Conference on Software Process and Product Measurement (MENSURA), CEUR-WS, Mexico City, pp. 1-16.

3. Perdomo, W., Prior, J., Leaney, J. (2020), 'Evaluation and measurement of software product quality in the new Colombian software companies —A Systematic Literature Review', *Ingeniare X(XX), In review*.

TABLE OF CONTENTS

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

In a globalised market, where companies must constantly innovate and continuously improve to remain active in the market and be competitive, it is necessary to have access to high-quality technological tools that allow improving the business processes and products and that give a support to increase the productivity, reduce costs and time in the validation of process and products. Traditionally, enterprises have a local market and customer, but today companies can access global markets with different types of customers with their specific demands of products and services directly related to software development. The software industry has developed new and innovative applications to respond to the demand of customers globally and the environment needs. Software engineering has used different approaches to product development, which are constantly changing and adopting new and improved methods and models. One particular area which has transformed is software development and the software quality measurement process.

Software developers must compete in local and international markets on costs, response time, quality, reputation or specialized expertise (Liberatore & Pollack-Johnson 2013, Ralph & Kelly 2014). In addition, many software products present improvement needs related to product quality in aspects such as measurement, requirements fulfilment and user satisfaction, making efficient use of resources, time and costs. The above contributes to the development of products of high quality, which must evolve to adapt to the new market demands to be more competitive (Colomo et al. 2011, Garzas et al. 2013, Lampasona et al. 2012, Pino et al. 2008, Rodriguez et al. 2016). An important reason for low product acceptance in the market is due to low quality during the software development cycle, processes improvisation, time overestimation and resources (Pelaez et al. 2011).

Colombian software industry has low levels of international exports and high customer dissatisfaction due to the low software product quality (SPQ) (Fedesoft 2015). These problems are directly associated with the sector's low competitiveness and the lack of measures and evaluation methods clear and well structured (Carvalho et al. 2009, Metaute & Serna 2016, Putnam & Myers 2004). The candidate chooses Colombia as a developing country because of these issues which need to be improved to help software developers and their companies to have recognition inside and outside the country and the market. In addition, because the candidate knows the software development companies' situation in Colombia because of his experience and knowledge of the country. Therefore, this research will examine initially, how Colombian software developers evaluate software product quality.

1.1 What is the Problem?

The poor quality of much current software represents a significant financial burden for software development companies. In general, software products tend to need iterations in their development cycles due to process improvisation, time overestimation, and development resources (Pelaez et al. 2011). Also, the product return by the end user because of the lack of completeness of essential quality characteristics such as usability, reliability, security, efficiency and accuracy (Metaute & Serna 2016), thus generating investment in time, labour and high costs in the process review and adjustments to the product developed (Gasca et al. 2013). In addition, the poor product quality in young software companies is present because software product quality is not considered in the development processes, nor the use of objective and measurable criteria that can avoid customer dissatisfaction (Jacobson et al. 2013, Zelkowitz 2011, Perdomo 2019, Perdomo & Zapata 2021). This generates a bad image and low competitiveness in international markets.

The optimisation of the quality measures of existing software products or the adoption of new and more effective quality measures are important for the productivity and competitiveness of software development companies (Pelaez et al. 2011).

1.2 Why is this Problem Significant?

The economic and competitive benefits of companies depend largely on delivering good quality products to their customers. Companies cannot lose the goodwill of their customers because of an inadequate approach in the effective and efficient measurement of the quality of their products. Therefore, it is essential to carry out measurement and evaluation processes at each stage of the software development cycle, from the requirements stage to the implementation of the system and also to be aligned with the business objectives. The output of software development projects with high quality is directly linked to the company's positioning in the market and customer satisfaction.

The output of these projects is also related to variables such as cost, time, productivity and competitiveness. In order for software development companies to be internationally competitive, they must demonstrate good quality indicators in the measurement of software products, large-scale developments with measurement proposals on fundamental aspects such as functionality, maintainability, and security.

1.3 Why is it Critical to Solve this Problem?

The growth and speed of the software industry and the demand for good quality products has meant that many users accept that product costs can increase if there are improvements in software quality (Rodriguez et al. 2016). Achieving the quality level demanded by the market requires considering the adoption of models and standards recognised in the software industry.

For Colombian software companies this represents a high cost in terms of money, time, expertise in its implementation due to the fact that the Colombian software developers companies have a low competitiveness level on the international markets.

A major issue for Colombian companies with these models is that they were created to be implemented in large companies with high financial capability and solid experience. This limits the ability of young Colombian companies, because of their size and experience, to implement measurement models and practices of software product quality.

1.4 Where is this Problem Impacting?

This problem directly affects the ICT sector in Colombia, integrated by Ministry ICT, associations and organisations, productive sectors, and enterprises (MinTIC 2015). However, this research will focus on small and medium-sized software development enterprises (SMEs) and their quality and development teams that represent 19% of the 590 companies in the software and ICT industry in Colombia (Pino et al. 2010, Fedesoft 2015). These SMEs are further defined as companies which have been in operation for between two and five years and have between one and 200 employees. Further, they are located in Colombia. In the rest of this thesis they will be referred to as young companies.

1.5 Who are the Impacted Stakeholders?

The stakeholders directly involved in this research are the software development team, developers, project manager, testers, and users (acquirers). Some changes in the software development project will have a direct impact on these stakeholders; they will also be the direct beneficiaries of the recommended solution.

1.6 Scope of the Final Problem

The literature review described in Chapter 2 demonstrates that software product quality (SPQ) is important, because it helps by improving competitiveness, measurement, and product quality evaluation. Therefore, the next part of the introduction will be focused on SPQ evaluation and the fact that software product quality measurement still remains a problem for software development companies.

Software Quality is the *'capability of a software product to satisfy stated and implied needs when used under specified conditions'* (ISO24765 2010). Functional requirements specify the capabilities that the solution must provide to its customers and the final users. It defines the functionality the solution needs to work (Inayat et al. 2015). Meanwhile, a non-functional requirement is a software requirement that describes not what the software will do but how the software will do it, i.e. software performance requirements, software external interface requirements, software design constraints, and software quality attributes (ISO24765 2010). The term 'non-functional requirement' is used to delineate requirements focusing on "how well" software does something as opposed to the functional requirements, which focus on ‘what’ the software does (Chung & do Prado Leite 2009).

Software product quality (SPQ) reflects two perspectives: conformance to specifications and satisfying expectations when it is used under specified conditions (ISO24765 2010, Pressman 2005, ISO25000 2014).

Software quality has gained major attention by software engineering researchers in the last three decades (Sjoberg et al. 2007, Wirth 2008) focusing on the importance of software product in the industry (Solyman et al. 2015, Jinzenji et al. 2013). Developing high-quality products requires a lot of effort, as developers also have to deal with challenges such as competitiveness, quality issues, and customer satisfaction during development (Perdomo et al. 2020). Every day "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" (Ahmad et al. 2016, Curcio et al. 2016, Jinzenji et al. 2013). Furthermore, "SPQ is often not defined comprehensively, specifically or effectively because previous approaches have focused on certain quality aspects" (Fayad et al. 2000, p.118) such as software quality characteristics, specific measures, definitions, and software stakeholders (Fayad et al. 2000, Nakai et al. 2016, Tamai & Anzai 2018).

Companies need to understand the importance of product quality and prove that the philosophy about "a quality process produces a quality product" (Kitchenham & Pfleeger 1996) is no longer enough. *"...the software quality evaluations should be based on direct evidence about the product, not only on evidence about the process"* (Maibaum & Wassyng 2008), since a high-quality process does not necessarily ensure a good quality product. Measuring product quality still remains a problem for software development companies related to factors such as cost, effort, time, and competitiveness (Wagner 2013, Barney & Wohlin 2009).

Several authors claim that companies lack industrial studies that show the current state of software product quality (Wagner 2013, Lampasona et al. 2012), how they evaluate the product quality, and which measures they are using to deliver a good product to the market (Rodriguez et al. 2016, Escobar & Linares 2012, Sánchez-Gordón & O’Connor 2016, Fernandez et al. 2018, Carvalho et al. 2009, Baquero et al. 2018). Furthermore, it may help developers and companies adopt software product quality (SPQ) measures to help them satisfy customer expectations, especially in countries where software development is on the ascendance (Lampasona et al. 2012, Metaute & Serna 2016, Pelaez et al. 2011).

Other studies state that the existence of inappropriate measurement of the product quality predominate, generating high costs in the remediation, defects correction and product development, user dissatisfaction and low competitiveness in global markets (Wagner 2013, Febrero et al. 2016, Idri et al. 2016, Rodriguez, Pedreira & Fernandez 2015, Perdomo et al. 2020).

1.7 Aim and Objectives

The aim of this thesis is to identify how Colombian software development companies evaluate software product quality (SPQ). This research is focused on the following objectives:

1. Identify what does product quality mean to Colombian software developers and their companies.
2. Classify software developers' experiences when they evaluate SPQ.
3. Explore how organizations evaluate SPQ.
4. Identify which characteristics and measures software developers and their companies use to evaluate SPQ.

1.7.1 Thesis Conduct

This PhD thesis presents a study carried out with seven young software development companies in a developing country, Colombia, about how they evaluate software product quality and which measures they apply in their companies.

The thesis uses a qualitative research approach to understand 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.

The candidate did the work and the supervisor team progressively verified it and gave the candidate advice.

1.8 Structure of this Thesis

This thesis is organized as follows. Chapter 2 begins with a review of the existing literature. It defines the gap that this thesis intends to fill. In particular, section 2.7 discusses some studies on issues of software product quality. Chapter 3 outlines the research design proposed to address the gap identified in Chapter 2, including quantitative and qualitative research design, the epistemology, theoretical perspective, and methodology. In particular, section 3.6 introduces the use of the research method proposed and section 3.7 presents how the candidate knows the results will be valid. Chapter 4 presents the quantitative findings (Study 1), which includes data collection, findings, and the new research method proposed. Chapter 5 describes the qualitative analysis (Study 2) that includes the analysis approach and data collection process. Chapter 6 presents the results of this research. Chapter 7 develops a discussion of the results. Chapter 8 will conclude with future work and closing remarks.

CHAPTER 2

LITERATURE REVIEW

With the increasing importance of software products in industry as well as in our everyday life (Solyman et al. 2015), the software development process has gained major attention by software engineering researchers and practitioners in the last three decades (Sjoberg et al. 2007, Wirth 2008). However, software product quality has not had the same attention in industry (Ahmad et al. 2016, Curcio et al. 2016, Jinzenji et al. 2013), as previously stated on page 4, *"the software quality evaluations should be based on direct evidence about the product, not only on evidence about the process"* (Maibaum & Wassyng 2008, p.91), since a high-quality process does not necessarily ensure a good quality product.

Software product quality is reflected or measured by the degree to which a product conforms to its specifications (ISO24765 2010) or to its requirements (Pressman 2005, ISO25000 2014). It can be measured through time, cost, or productivity (Escobar & Linares 2012, p.2). Software product quality (SPQ) reflects two perspectives, conformance to specifications, and satisfying customer expectations when it is used under specified conditions (ISO24765 2010). Software quality attributes such as functionality, maintainability, security, and reliability are used to describe the characteristics of the product to ensure it meets the requirement needs (Al-Qutaish 2010, p.174). Therefore, 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.

Competitiveness is defined in terms of profit, market share, product value and buyer's expectations satisfaction (Berger 2008, p.94). A firm's product becomes part of its buyer's value chain. Buyer value is created when a firm lowers its buyer's cost or enhances its buyer's performance. This is the result of the ways a firm's product, as well as its other activities, affect the value chain of the buyer (Porter 2001, p.36). Therefore, the quality of a software product affects the buyer's costs since poorer quality will require that the buyer spend more on remediation, fixing defects and implementing enhancements (Metaute & Serna 2016, Rodriguez et al. 2016, Wagner 2013, Jones 2013). High-quality software therefore has a competitive advantage. On the other hand, when all software developers produce software at about the same quality level there is no advantage to any one developer. However, developers do not all produce software at the same quality levels.

There are many studies in the literature on issues of quality and its impact on the evaluation approaches used to measure the software product quality in software development companies. Their research presents fundamental problems such as insufficient measurement of quality which leads to low productivity and large schedule increases (Rahmani et al. 2016, Lampasona et al. 2012, Rodriguez & Piattini 2012*b*).

Some studies from different countries have explored different factors and criteria to evaluate software quality applying qualitative and quantitative methods to collect data (Lampasona & Kläs 2011, Phillips et al. 2012, Hering et al. 2015, Alsaqaf et al. 2017). They have applied open-ended questions in interviews to experts of software development companies as developers, testers and project managers, who were considered experts or practitioners. However, most of these studies used literature review data and surveys to collect information solving their hypothesis or research questions. Such studies focus on criteria related to software quality in general, but they do not describe any specific criteria related to software product quality. The authors describe which characteristics and sub-characteristics were applied in their studies and companies, but they do not show any important measure. Furthermore, they present some problems related to at least one quality characteristics. The essential characteristics described in the studies were maintainability, usability, and security. On the other hand, these studies explore new challenges that practitioners could be facing when they are designing requirements and developing products with poor quality. Nevertheless, they do not explore what is going on during all life-cycle with software development projects related to quality measurement. The candidate can not do a direct comparison among the above studies because they are not directly related to this study.

Recently, researchers have shown an interest in the study of software quality models and standards such as ISO/IEC 9126 and ISO/IEC 25000 by establishing direct relationships between characteristics and product quality measures. These models describe a general context about the application of methods and algorithms for the software product measurement with few established measures and without a correlation between measures and overall SPQ (Rodriguez, Pedreira & Fernandez 2015, Garzas et al. 2013, Perdomo & Zapata 2015). Other authors have been encouraged by different models, standards and methodologies to create their own models according to the companies' needs to measure the quality of their products (Ahmed & Capretz 2011, Rodriguez et al. 2016, Metaute & Serna 2016). These authors claim the quality measurement process used in these previous studies do not make appropriate good use of time and resources and not all companies implement the measures.

Some studies state that the existence of inappropriate measurement processes predominate, generating high costs in the remediation, defects correction and product development, user dissatisfaction and low competitiveness in global markets (Jacobson et al. 2012, Wagner 2013, Zelkowitz 2011, Hering et al. 2015). Other studies have confirmed that it is important to evaluate quality characteristics during and after the development of any type of software because this helps to identify defects and failures in the product (Febrero et al. 2016, Rodriguez, Sierra & Jaramillo 2015, Idri et al. 2016, Radjenović et al. 2013). In reference to these problems, software development companies need to guarantee coherence in the measures used in the SPQ evaluation.

This study intends to identify and characterize evaluation approaches used to measure the software product quality in young software development companies in Colombia. The study is relevant because such companies lack studies that show the current state of software quality, how they evaluate the product quality, and which measures they are using to develop and launch a good product to the market, which meets high-quality criteria. In addition, it may help developers and companies adopt SPQ measures to help them satisfy customer expectations, especially in countries like Colombia where software development is on the ascendance (Fedesoft 2015, MinTIC 2015). The findings of this literature review fit with the research aims which are focused on identifying and classifying software development companies' experiences when they evaluate product quality. Furthermore, it will identify characteristics and measures which software development companies are using and which they will consider using in the future to improve their software development projects.

Section 2.1 of this literature review surveys the field of software development and its positioning in the market. Section 2.2 shows customer requirements as a main objective of quality. Section 2.3 summarises software quality evaluated through different models and standards, which have been created to focus on the evaluation and measurement processes. Section 2.4 describes some success cases and needs related to the measurement of software product quality. Section 2.5 presents measures to improve the evaluation of software quality related to non-functional requirements (characteristics) and those most used by experts. Section 2.6 describes the Colombian software companies situation in terms of software quality and how they can meet their competition. Section 2.7 presents some papers gathered on issues of software product quality and how the software development companies measure their product quality. Together, these sections establish a foundation from which the thesis can explore what are the evaluation approaches used to measure the software quality in young software development companies and how do Colombian software development companies measure the software product quality.

2.1 Software Development and its Positioning in the Markets

The software industry is one of the largest, wealthiest and most important industries in the modern world. The software industry is also troubled by very poor quality and very high-cost structures due to the expenses of software development, maintenance, and endemic problems with poor quality control (Jones 2017, p.13). However, the current trends of the software industry have been the development of software products with an optimum use of resources, time and costs; in other words, the quest for the software development to be efficient (Escobar & Linares 2012).

In many parts of the world, software development companies consider that software process quality and software product quality (SPQ) level are always essentials to competitiveness in different markets (Darmawan et al. 2010, Hyatt & Rosenberg 1996, Rodriguez et al. 2016) to make sure that a company's products meet the required quality standards allowing software companies to compete, position their brand and constantly improve quality measurement terms. Developing countries and transition economies have played an important role in the software industry increasing from one-fourth to one-third their participation in the world software exports. Some countries such as

United States, China, Germany, France, and United Kingdom represented 36 percent of the global software trade in 2016. Meanwhile, Colombia was less dynamic, moving from a 3 percent share in 2000 to 10 percent share in 2016, but this figure is not enough to position itself internationally and compete with China, India, and the USA which have good software development companies (GARTNER 2015, Procolombia 2017, TradeMap 2015).

Colombian software developers must compete against international software developers for business within Colombia because software products do not have enough recognition inside and outside the country and the market is not sophisticated and is oriented only for customers in Latin America (TradeMap 2015). Software developers, as all product and service providers do, compete on price, productivity, specific knowledge and skills, and quality (Carvalho et al. 2009, Metaute & Serna 2016, Putnam & Myers 2004).

The most popular development approaches are those based on the famous capability maturity model integration (CMMI) developed by the Software Engineering Institute (SEI) and the newer "Agile" approach. When the companies adopt the Team Software Process (TSP) and Personal Software Process (PSP) methods, there is an additional boost in performance. However, these methods do not have enough examples to show that they are successful in improving quality, productivity and competitiveness at the same time (Jones 2017, p.21).

The CMMI approach has fundamentals problems. Firstly, the documentation is the second most expensive software activity in history. Secondly, without careful measurements continuous progress is unlikely and finally poor quality leads to low productivity and large schedules (Jones 2017, Rahmani et al. 2016).

Some methodologies such as Six Sigma, Lean and Scrum are expensive and demanding of time, effort and investment in the process. Furthermore, Scrum presents a high cost in training people due to the quality control that this methodology has in its process (Kitchenham & Pfleeger 1996, Maibaum & Wassyng 2008, Soley & Curtis 2013).

Over the last few years, there has been a change of philosophy, with a new focus on software product quality. As previously stated on page 9, there is greater and greater acceptance of the idea that *"software quality evaluations should be based on direct evidence about the product, not only on evidence about the process"* (Maibaum & Wassyng 2008, p.91), since a high quality process does not necessarily ensure a good quality product.

Even though existing different approaches, methodologies, and tools to improve and evaluate the software quality and to help the development companies be more productive exist, one possible way in which Colombian software developers can meet their competition is by improving the quality of their products and services (Lampasona et al. 2012, Pelaez et al. 2011, Pino et al. 2012, Rodriguez et al. 2016). Therefore, if the focus is on quality first, the organization's abilities are nurtured more that if the emphasis is on cost efficiency. In addition, if the quality effort achieves its goal, the system can be more dependable and the projects can be flexible to generate and implement ideas (Ferdows & De Meyer 1990, p.175). Finally, capabilities built in this way become formidable competitive weapons, and so the companies cannot be easily matched by competitors.

2.2 Customer Satisfaction is the Objective of Quality

Customers do know what they want but may not be proficient at describing their needs. It can be difficult for developers to understand customer needs. If they are not understood then the customer impact will be high. Customers have three expectations levels such as expected, normal and exciting (Xu et al. 2009, p.88).

Companies are concerned about their reputation and customer satisfaction; these criteria are critical to the continuity of an organisation in the market and financial stability over time. If a customer receives a product of lower quality, they can create a scenario of nonconformity with the community, leading to a loss of value in the market and less customer confidence in the company's products and services.

The concept of quality reflects two different perspectives, conformance to specifications and satisfying customer expectations. Of these the most common is that quality is reflected or measured by the degree to which a product conforms to its specifications (Radatz et al. 1990) or to its requirements (ISO24765 2010, ISO25000 2014, Pressman 2005). This view confines the role and responsibility of quality management and assurance to producing something after the requirements or specifications have been determined. The alternative view is that quality is reflected in the degree to which the product meets a customer's expectations (Deming 1982, p.43). This view places the role and responsibilities of quality management and assurance as determining what the customer's requirements are and then producing something to satisfy those requirements. This extends the role of quality management and assurance into product design and creation. While both perspectives are useful and have their champions, this research will consider only the first view, that quality management is a matter of production, since the problem under investigation is primarily concerned with producing software that meets the customer's requirements and is not concerned about discovering what those requirements might be.

2.3 Software Quality Evaluated through Different Models

In recent research, the frameworks, models, and standards have been created to focus the evaluation and measurement processes of the software quality correctly. However, according to Singh et al. (2012), many enterprises and researchers have proposed different models to improve and evaluate software quality, but these are not comprehensive and complete. In addition, the models still present some problems related to the absence of an association between quality models and the software development process, non-maintainable quality models, non-evolving quality models, subjectivity in quality evaluation and the absence of necessary documentation for a quality model.

To begin with, during 1977 and 2011, McCall, Boehm, Grady, and Dromey proposed models of software quality. Some models to evaluate the software quality try to decompose the quality into a category of simpler characteristics. The McCall model is based on the decomposition of quality concept in three important uses of a software product, from the viewpoint of the user: operation, review and transition of the product (McCall et al. 1977). This model was the first to propose the evaluation of software quality using automatic and quantitative methods. Meanwhile, Boehm et al.'s (1978) quality model, was introduced to quantitatively evaluate software quality. The model's attributes are partially integrated, and its measurement criteria are subjective. Grady's (1992) model was developed by necessity in an organization to prioritise and define quality attributes to measure. Afterwards, Dromey (1995) developed a quality evaluation framework and emphasized that all artefacts must fulfil the quality criteria. The ISO9126 (2001) standard, presents less issues related to model association with software development processes, risk-driven aspects and fairness in quality validation.

Since 2002, different researchers have produced new and improved quality models, focused on defining metrics and models for the components' evaluation and software and for the component's product quality (Bertoa & Vallecillo 2002, Georgiadou 2003, Ortega et al. 2003, Rawashdeh & Matalkah 2006). However, Khosravi & Gueheneuc (2004) and Trendowicz et al. (2003) implemented models to design patterns and integration of quantitative and qualitative approaches into product lines. Other studies were based on ISO 9126 to evaluate and validate different types of components and applications and reuse criteria (Andreou & Tziakouris 2007, Behkamal et al. 2009, Sharma et al. 2008, Sibisi & Van Waveren 2007). During 2010, models were created to evaluate and measure the software quality statistically (Jamwal & Jamwal 2009, Srivastava & Kumar 2009), a quality model of embedded software components (Carvalho et al. 2009), a quantitative evaluation of aspect-oriented software quality model (Kumar et al. 2009), and a novel method for quantitative assessment of software quality (Bawane & Srikrishna 2010). Alvaro et al. (2010) developed a quality framework to measure the quality process components and Kalaimagal & Srinivasan (2010) created another model to evaluate the quality components supported on ISO 25000. Lastly, in 2011 a framework was defined where the end-user can order and classify components to improve the software design and development (Upadhyay et al. 2011). In addition, Bassam et al. (2011) has a minor consideration with the software product and it only considers the user's opinion. In 2005 the ISO 25000 superseded ISO 9126, which specified characteristics and software quality measures (ISO9126 2001) and ISO 14598, which specified the evaluation of software products (ISO14598 2000, Marin & Bedoya 2015). In 2012 an investigation was carried out which identified 24 key software quality models (Singh et al. 2012).

It is interesting to note that for more than 20 years, many models were developed to give the solution to the process measurement paradigm that fixing processes would lead to high quality software products. Direct measurement of the product quality was of minor importance at the time. Currently, it there are some companies directly concerned about product quality as a way to be recognised in the market (Calero et al. 2008, Gall et al. 2008).

Despite the existence of many frameworks to evaluate and measure the software quality, this research will be guided by the most current model on software product measures, which is the ISO/IEC 25000 standard, known as SQuaRE (Software Quality Requirements and Evaluation) (ISO25023 2016, Marcos et al. 2008). This standard has an aim to create a common schema for evaluation software product quality, replacing previous ISO 9126 and ISO 14598 standards (Rodriguez, Pedreira & Fernandez 2015).

ISO 25000 has some attributes to measure the software quality and it contains eight characteristics. Such characteristics were reorganised by the candidate due to the need to give greater importance to features such as security and compatibility, which were previously considered sub-characteristics in the measurement processes (Table 2.1).

Table 2.1: Characteristics and sub-characteristics (ISO25023 2016)

Characteristic	**Sub-characteristic**
Functional suitability	Functional completeness, Functional correctness, Functional appropriateness
Performance efficiency	Time behaviour, Resource utilisation, Capacity
Compatibility	Co-existence, Interoperability
Usability	Appropriateness recognisability, Learnability, Operability, User error protection, User interface aesthetics, Accessibility
Reliability	Maturity, Availability, Fault tolerance, Recoverability
Security	Confidentiality, Integrity, Non-repudiation, Accountability, Authenticity
Maintainability	Modularity, Reusability, Analysability, Modificability, Testeability
Portability	Adaptability, Installability, Replaceability

Despite the existence of models and standards, projects are still being canceled, software quality is poor, productivity and competitiveness remains low, security flaws are alarming, and software literature will continue to offer data without being tested in industry.

2.4 Measurement of Software Product Quality and Success Cases

In order to improve software quality, it is necessary to measure it. Although there are some studies that relate the measurement of software product quality in different sectors and countries, it is also important to find out about existing research on evaluation and certification of software product quality around the world. Rodriguez & Piattini (2012*a*) carried out a detailed literature review of ten studies (Alvaro et al. 2007, Baggen et al. 2012, Burger & Reussner 2011, Carvalho et al. 2009, Hatcliff et al. 2009, Heck et al. 2010, Morris et al. 2001, Serebrenik et al. 2010, Yahaya et al. 2010, 2008, Nakai et al. 2016). As a result, it was concluded that most of the studies highlight the work that already exists in the certification of software development processes and the need to extend it to the characteristics of the product, to ensure that the result complies with requirements. Most of the research is based on certifying the quality characteristics extracted from ISO/IEC 9126 using evaluation methods as described in ISO/IEC 14598 and using the new model of ISO/IEC 25000 as a theoretical support. However, they found that all of these measures are expensive to implement.

Marcos et al. (2008) showed the implementation of the maintainability characteristic of ISO 25000, making use of free software tools. The KEMIS environment (Kybele Enviroment Mesaurement Information System) provides an infrastructure for measurement and runs in continuous integration environments, allowing a periodic and automatic collection of reports on product quality, obtaining code metrics and Micro-architecture within the framework of ISO 25000. The KEMIS restrictions are based on the maintainability that is included in the 2502n division, and that different public and private companies are using the environment for the evaluation of the products of software.

Mellado et al. (2010) present the MEDUSAS environment as an example of public-private cooperation. This framework allows companies and public entities to provide a set of software quality measures, to control services independently and is based on the ISO 25000. The framework allows for evaluating the quality and code safety (software), the quality and design safety. Most importantly, it allows for carrying out the software quality evaluation in the product deliverable during the phases of analysis, design and software development.

Other studies also show the software product evaluation and certification process based on ISO/IEC 25000 standard. The first case is the research on software product sustainability, which realized the evaluation, improvement and certification of the ’xCloud Bookstore’. ’Bookstore’ is a platform for the online distribution and sale of digital content developed by Enxenio (Rodriguez, Pedreira & Fernandez 2015). The second case is ’BITDOC version 2.0’, which carried out the same evaluation process by the Bitware Company. These studies show that the software product evaluation and certification generates benefits to the companies as the percentage reduction in the code generation, improvement and reduction in the project teams efficiency and fulfilment in the delivery times (Fernandez & Piattini 2012, Rodriguez, Pedreira & Fernandez 2015). Another case is the study of iPavement application certification for smart cities which presents an environment for assessing the quality of the services iPavement develops. The environment includes the evaluation process, a quality model and a set of evaluation tools based on the ISO 25000 standards, which are used to certify the quality services developed

(Oviedo et al. 2015). Finally, Febrero et al. (2016) identify and analyse existing works in modeling software product reliability based on international standards to achieve a reliability assessment proposal.

In summary, many studies focus on process certification models of software development and the need to extend these to the product characteristics measurement to ensure that they meet customer requirements, generation quantitative measures, and objective evaluation criteria. Other models evaluate software products by implementing features and quality measures drawn from some international standards. However, these models do not implement all the characteristics and measures defined in the standards, evidencing the need to investigate why they do not implement other measures and how they determine their implementation. 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 (Giraldo et al. 2014, Lampasona et al. 2012, Pino et al. 2012). 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. This limits the ability of young Colombian companies in a developing country, because of their size, extra resources, and experience, to implement measurement models and practices of software product quality.

On the other hand, the studies emphasize that the application of software product measures generates benefits such as the reduction in the code generation, improvement and efficiency in the products delivery, customer satisfaction and competitiveness. However, there are very few cases where it is evident that with the application of some measures, efficient compliance in the software product quality is achieved. Therefore, the following section shows the measures of software product quality that have been implemented in some researches.

2.5 Measures to Improve Software Product Quality Evaluation

There are different measures that some researchers have applied in their studies to measure the software quality. These measures are listed in Tables 2.2 and 2.3. In the tables, the first column lists the measures described in ISO/IEC 25000 standard and in the second column, we present some attributes described in the studies.

Table 2.2: Software Quality measures (1 of 2)

Measures (ISO25023 2016)	**Measures (studies)**
Functional suitability: Functional suitability Functional coverage, Functional correctness, Functional appropriateness of usage objective, Functional appropriateness of system.	Functional coverage, Functional correctness, Functional appropriateness of usage objective (Heck et al. 2010, Alvaro et al. 2007, Yahaya et al. 2008).
Performance efficiency: Mean response time, Response time adequacy, Mean turnaround time, Turnaround time adequacy, Mean throughput, Mean processor utilization, Mean memory utilization, Mean I/O devices utilization, Bandwidth utilization, Transaction processing capacity, User access capacity, User access increase adequacy.	Response time adequacy, Mean turnaround time (Burger & Reussner 2011), User access capacity (Yahaya et al. 2008, 2010).
Compatibility: Co-existence, Interoperability.	Co-existence with other products, Data formats exchangeability, Data exchange protocol sufficiency, External interface adequacy.
Reliability: Fault correction, Mean time between failures (MTBF), Failure rate, Test coverage, System availability, Mean down time, Failure avoidance, Redundancy of components, Mean fault notification time, Mean recovery time, Backup data completeness.	Fault correction, System availability, Redundancy of components (Febrero et al. 2016, Yahaya et al. 2008, 2010, Carvalho et al. 2009, Morris et al. 2001).
Security: Access controllability, Data encryption correctness, Strength of cryptographic algorithms, Data integrity, Internal data corruption prevention, Buffer overflow prevention, Digital signature usage, User audit trail completeness, System log retention, Authentication mechanism sufficiency, Authentication rules conformity.	Data integrity, Authentication mechanism sufficiency, Authentication rules conformity (Izzat 2013, Mellado et al. 2010, Yahaya et al. 2010).
Maintainability: Coupling of components, Cyclomatic complexity adequacy, Reusability of assets, Coding rules conformity, System log completeness, Diagnosis function effectiveness, Diagnosis function sufficiency, Modification efficiency, Modification correctness, Modification capability, Test function completeness, Autonomous testeability, Test restartability.	Coupling of components, Cyclomatic complexity adequacy, Reusability of assets, Diagnosis function sufficiency, Modification correctness, Test function completeness (Duque et al. 2011, Rodriguez, Pedreira & Fernandez 2015, Serebrenik et al. 2010, Baggen et al. 2012, Izzat 2013, Marcos et al. 2008, Carvalho et al. 2009, Yahaya et al. 2010).

Table 2.3: Software Quality measures (2 of 2)

Measures (ISO25023 2016)	**Measures (studies)**
Usability: Description completeness, Demonstration coverage, Entry point self-descriptiveness, User guidance completeness, Entry fields default, Error messages understandability, Self-explanatory user interface, Operational consistency, Message clarity, Functional customizability, User interface customizability, Monitoring capability, Undo capability, Understandable categorization of information, Appearance consistency, Input device support, Avoidance of user operation error, User entry error correction, User error recoverability, Appearance aesthetics of user interfaces, Accessibility for users with disabilities, Supported languages adequacy.	Description completeness, User guidance completeness, Operational consistency, Monitoring capability, Appearance aesthetics of user interfaces (Bevan et al. 2016, Heck et al. 2010, Izzat 2013).
Portability: Hardware environmental adaptability, System software environmental adaptability, Operational environment adaptability, Installation time efficiency, Ease of installation, Usage similarity, Product quality equivalence, Functional inclusiveness, Data reusability/import capability.	System software environmental adaptability (Yahaya et al. 2010).

Experts around the world have argued the importance of measuring different characteristics such as Portability (Po), Security (Se), Functional suitability (Fs), Performance efficiency (Pe), Usability (Us), Reliability (Re), and Maintainability (Ma) during and after software development (Alvaro et al. 2007, Arciniegas et al. 2010, Baggen et al. 2012, Burger & Reussner 2011, Carvalho et al. 2009, Febrero et al. 2016, Garzas et al. 2013, Heck et al. 2010, Marcos et al. 2008, Mellado et al. 2010, Morris et al. 2001, Rodriguez, Sierra & Jaramillo 2015, Serebrenik et al. 2010, Yahaya et al. 2008, 2010, Nakai et al. 2016) (see Table 2.4 for an analysis).

Some recent studies have shown that the measures of functional suitability characteristic are important to evaluate the degree to which a product provided functions that meet stated and implied needs when used under specified conditions. For example, Heck et al. (2010), argue that this type of measure helps organisations to obtain certainty about or confidence in software artefacts. Furthermore, it could help prevent poor quality of the requirements (incomplete and changing requirements), which is the main reason why so many projects continue to fail. The authors propose a model that evaluates two

Table 2.4: Characteristics most used by experts

Experts	**Po**	**Se**	**Fs**	**Pe**	**Us**	**Re**	**Ma**
Morris et al. (2001)	-	-	-	-	-	x	-
Alvaro et al. (2007)	-	-	x	-	x	-	x
Marcos et al. (2008)	-	-	-	-	-	-	x
Yahaya et al. (2008)	-	x	x	x	-	x	x
Carvalho et al. (2009)	-	-	-	-	-	x	-
Arciniegas et al. (2010)	-	-	-	-	x	-	-
Heck et al. (2010)	-	-	x	-	-	-	-
Mellado et al. (2010)	-	x	-	-	x	x	x
Serebrenik et al. (2010)	-	-	-	x	-	-	x
Yahaya et al. (2010)	x	-	-	-	x	-	x
Burger & Reussner (2011)	-	-	-	x	-	-	-
Baggen et al. (2012)	-	-	-	-	-	-	x
Garzas et al. (2013)	-	-	-	-	-	x	-
Rodriguez, Sierra & Jaramillo (2015)	-	-	-	-	-	-	x
Febrero et al. (2016)	-	-	-	-	-	x	-
Nakai et al. (2016)	x	x	x	x	x	x	x

attributes, the correctness and consistency of the functionality of software products.

In contrast, the work by Yahaya et al. (2008), addressed improving effort and functionality in software product quality. In addition, this study's survey indicated that functionality, efficiency, integrity, maintainability and reliability were the main characteristics with high consideration in assessing software product in Malaysian companies. In contrast, the study by Alvaro et al. (2007) highlighted the importance of reinvention, redesign, re-implementation and functionality when building a new software product. Alvaro's study argue that better products can be delivered in shorter times, reducing the maintenance costs and increasing the software product quality and functionality.

Other studies indicate the relevance of performance efficiency characteristic during the measurement process in the development stage, when the developer can check at any time if the requirements are still met and which attributes may be violated (Burger & Reussner 2011, Serebrenik et al. 2010, Yahaya et al. 2008).

Usability is another important characteristic to measure software quality due to the greatest impact on final user acceptance (Alvaro et al. 2007, Arciniegas et al. 2010, Mellado et al. 2010, Yahaya et al. 2010). These authors argue that the relationship between other characteristics and measures has not been completely characterised and implemented in the processes of software product evaluation. Otherwise, Mellado et al. (2010) mentioned the significant need to control and assess the software quality and security through the usability measures application with the help of customers, factories and software companies. In recent years, a key research area in the software engineering community has found reliability as a characteristic highly important in the evaluation of software processes and product. Also, to increase reliability in reusing software components is necessary to reduce development time and improved product quality attributes (Carvalho et al. 2009, Garzas et al. 2013). In 2008, Yahaya et al. (2008) published a paper in which they described that reliability, functionality, integrity, and security are the measures of greatest impact on software engineering. In addition, they argued that software engineering will need economical and reliability component resources.

Recent studies state that the increasing dependence of our community on software development has led software reliability to become a key factor in the research and industry (Febrero et al. 2016, Morris et al. 2001). According to Febrero et al. (2016), reliability is a key factor in software quality since it quantifies failures and misbehaviours. Furthermore, there are other problems with the reliability such as customer dissatisfaction, loss of prestige and reduction of total costs of the software product. Many authors argue that is not necessary to discuss the importance of software reliability in many industries (Littlewood & Strigini 2010, Pham & Pham 1991), due to software reliability being the crucial factor to estimate the software quality and cost (Farr 1996).

One of the measures less researched is that of security. Some studies cited by Yahaya et al. (2008) and Mellado et al. (2010), describe this characteristic and its measures as important to improve the software quality to the level of process and product. However, Mellado et al. (2010) indicated that security characteristic includes aspects such as controllability, data integrity, authentication mechanism, and data prevention, helping to identify mistakes in the software development and to give reliability to the customers with their data and information.

Some studies have demonstrated that the measures are important because they help to improve the product quality and service efficiency (Yahaya et al. 2008, 2010). However, maintainability continues to be a key factor during the software development life-cycle, it is constantly investigated, since it helps to correct errors over a long period, and it is closely related to the reliability and functionality of the system. Therefore, the errors generated directly affect the users and customers (Marcos et al. 2008). These errors are related with the technical quality of source code, which is a determinant for software maintainability. Thus, to implement some changes, avoid unexpected effects, and validate the changes performance it is useful to implement maintainability measures and to do monitoring (Serebrenik et al. 2010, Baggen et al. 2012). Serebrenik et al. (2010) claim quality control must be measured from a quantitative point of view, for which the code metrics provide the basis to improve and to give the best visibility in the product quality (Marcos et al. 2008). Furthermore, maintainability allows adapting the product to the customer needs to improve the commercialization (Rodriguez, Sierra & Jaramillo 2015).

2.6 Improving the Competitiveness of Colombian Software Companies

According to Procolombia (2017), the Colombian software industry has presented a dynamic and significant growth in the last 10 years but needs to improve some aspects related to the quality of its software products to achieve the products export to international markets. In addition, Colombia currently occupies the 31st position in the ranking of the world's largest economies and is the fourth in Latin America (IMF 2017). Procolombia is a government agency of the Executive Branch of the Colombian Government in charge of promoting Colombian non-traditional exports, international tourism, and foreign investment to Colombia by providing domestic companies with support and integral advisory services for their international trade activities, facilitating the design and execution of their internationalization strategies, and by providing foreign companies with trade, legal, and educational information about Colombia's market, products, services and companies (Procolombia 2017).

In the country, there are approximately 20 industry categories, for which many software development companies provide products and services such as government, finance, insurance, logistic and transportation, trading, agroindustry, health, tourism, construction, entertainment, real-estate, manufacturing, hydrocarbons-mining, education, energy, telecommunications, public services, software, and auto parts. According to the Ministry of ICT (MinTIC) on the 2015 Census and the Colombian Software Industry Federation (Fedesoft), the 10 top industries with the highest percentage of software products or services required are telecommunications (TC), software (SW), public services (PS), hydrocarbons-mining (HM), health (HE), energy (EN), tourism (TO), agroindustry (AG), manufacturing (MA), and education (ED) (Fedesoft 2015, MinTIC 2015).

Table 2.5 shows the relationship of the main industries with the most important quality characteristics (non-functional requirements) for their measurement in Colombia. Furthermore, it describes a numeric scale (1-low, 3-medium, 5-high) that represents the importance level of characteristics according to their application by different experts in some studies (Alba & Hurtado 2011, Arciniegas et al. 2010, Castellanos 2016, Delgado et al. 2016, Diaz et al. 2010, Farah et al. 2014, Giraldo et al. 2014, Guana & Correal 2013, Lampasona et al. 2012, Pino et al. 2012).

Table 2.5: Characteristics vs industries

Characteristic	**TC**	**SW**	**PS**	**HM**	**HE**	**EN**	**TO**	**AG**	**MA**	**ED**	**IMPORTANCE**
Suitability	1	3	1	1	3	1	3	1	1	3	Low-Medium
Performance	1	3	5	1	3	1	1	1	1	1	Low-Medium
Compatibility	3	1	1	3	1	3	3	1	3	5	Low-Medium
Portability	3	3	1	3	5	3	5	3	3	5	Low-Medium
Usability	3	3	5	3	5	3	5	3	3	5	Medium-High
Reliability	3	3	5	3	5	3	5	3	3	5	Medium-High
Security	3	3	5	3	5	3	5	3	3	3	Medium-High
Maintainability	3	5	3	3	5	3	5	3	5	5	Medium-High

Some research papers are focused more on non-functional requirements with medium-high importance such as usability, reliability, security, and maintainability. Furthermore, the industries with more incidents related to software quality are health, public services, tourism, and education. Such industries are concerned about the role of quality in their software products and how the software achieves their requirements. Although according to the findings of the attributes measured by software developers in Colombia (see Table 2.6), it is concluded that the most important characteristics that were taken primarily for the purposes of this study are security and performance efficiency. Since no measurements have been applied in the production of the software except for the security feature, performance efficiency has had low application in Colombian software development industries. Table 2.6 records the results of the relationship between the main industries and software quality measures (#m) used in some software development projects and companies (#c) in Colombia.

All the information in Table 2.6 is from the literature review. Some companies do implement some SPQs, but no statistical data is available from the government and software development industry in Colombia.

According to the Census conducted by Fedesoft (2015), 94 percent of entrepreneurs and managers argue as relevant the adoption of models and measures of quality in the development of their software products to strengthening their efficiency, effectiveness and competitiveness levels. In addition, 64 percent of the software companies do not apply quality models and measures, which represents a detriment to the competitive and positioning capacities in national and international markets.

On the other hand, 32% exports of software products and services from 2015 to 2017 were to countries such as the United States, Spain, Ecuador, Mexico, Chile, and Peru. Although these are only a few countries, it represents a significant growth of 13 percent in comparison with 2014, which obtained 19 percent of exports during the period.

Table 2.6: Characteristics and attributes measured by Colombian software companies

Reference	**Measures**	**#m**	**#c**
Castellanos (2016)	**Usability:** Operational consistency **Reliability:** Test coverage, System availability	3	1
Delgado et al. (2016)	**Compatibility:** Co-existence with other products, Data formats exchangeability	2	1
Giraldo et al. (2014)	**Maintainability:** Cyclomatic complexity adequacy	1	1
Farah et al. (2014)	**Usability:** Operational consistency; **Maintainability:** Coding rules conformity	2	1
Guana & Correal (2013)	**Maintainability:** Reusability of assets	1	1
Lampasona et al. (2012)	**Functional suitability:** Functional correctness, Functional appropriateness of usage objective; **Compatibility:** External interface adequacy; **Usability:** Appearance consistency; **Reliability:** Redundancy of components	5	1
Pino et al. (2012)	**Maintainability:** Coding rules conformity	1	1
Alba & Hurtado (2011)	**Maintainability:** Coupling of components	1	1
Arciniegas et al. (2010)	**Usability:** User guidance completeness; **Maintainability:** Cyclomatic complexity adequacy	2	1
Diaz et al. (2010)	**Usability:** Functional customizability; **Portability:** Operational environment adaptability; **Maintainability:** Reusability of assets	3	1

In summary, for software developers and industries in Colombia, all characteristics are important when evaluating and controlling software quality developed during and after the development life-cycle, but companies need to improve the software production process by emphasising high quality to be more competitive in different markets (Alba & Hurtado 2011, Arciniegas et al. 2010, Castellanos 2016, Delgado et al. 2016, Diaz et al.

2010, Farah et al. 2014, Giraldo et al. 2014, Guana & Correal 2013, Lampasona et al. 2012, Pino et al. 2012).

At the global level, experts have indicated that all features of software product quality are essential, but the studies identified have proven otherwise. Characteristics such as performance efficiency, usability, reliability, and maintainability are the most implemented in studies worldwide, but these do not evidence measures implemented and associated methods to evaluate and certify the software product quality. In the Colombian scenario, it has been identified that several industries make constant use of software production services and that the most relevant characteristics and measures that software developers implement to evaluate the product quality are maintainability, usability, reliability, and compatibility. However, characteristics and measures related to performance efficiency and security are less implemented in software development companies.

Colombian software industry has low levels of international exports (Fedesoft 2015) and high customer dissatisfaction (Metaute & Serna 2016, Sánchez-Gordón & O'Connor 2016) due to the low software product quality. These problems are directly associated with the sector's low competitiveness and the lack of measures and evaluation methods clear and good structured. Thus, it is evident there is a need to understand which measures the software development companies implement and how they evaluate SPQ. Furthermore, this research will propose or develop a measurement model in which the characteristics and elements are appropriate for evaluating software product quality in young software development companies (SMEs) and which of these should be measured by the Colombian software teams.

The literature review has shown the different models, frameworks and methodologies that different companies and researchers have developed/customized and implemented to measure the software product quality in software development companies. Moreover, it is evident that the characteristics and measures of different international standards have been implemented to evaluate and certify the product quality developed. However, the current focus of many companies is evaluating the software process quality, and neglecting product quality as a key factor for the companies' competitiveness international markets and end-user satisfaction.

2.7 Some Studies on Issues of Software Product Quality

This section is based on a systematic literature review following the guidelines proposed by Kitchenham (2004) and presented in Perdomo et al. (2020). This literature focuses on issues of software product quality (SPQ) and how young software development companies measure their SPQ in developing countries.

The papers studied were published between 2010 and 2019 and the search was extensive because many different papers were found related to the topic, but it needed to be filtered. Table 2.7 shows the search engine results which provided 3268 papers. Nevertheless, it should be pointed out that only 31 were accepted, which represents about 1% of the total articles, hardly even that. It is apparent that many articles were rejected. This is because if a more limited search had been carried out, there would have been with fewer results from the search engines, but at the same time, important papers would have been lost.

Table 2.7: Search Engines Used

Sources	**Search results**	**Reviewed**	**Accepted**
Google Scholar	1032	18	8
ProQuest	208	17	2
Web of Science	840	15	3
IEEE	921	23	10
ACM	267	21	8
Total	3268	94	31

Therefore, a less restrictive search was defined and as a result of this, many articles were obtained, of which very few were considered apt. In addition, the research attention focused on papers where keywords and titles included the research strings (see Table 2.8). These strings were also searched for in the whole document by some search engines.

Table 2.8: Keywords and strings

Strings
(a) TITLE-ABS-KEY (software product quality OR software quality) AND TITLE-ABS-KEY (measures OR metrics) AND TITLE ABS-KEY (software development) AND TITLE-ABS-KEY (Colombia) AND TITLE-ABS-KEY (young companies OR SMEs)
(b) TITLE-ABS-KEY(software product quality)AND TITLE ABS-KEY (software development) AND TITLE-ABS-KEY (Colombia) AND TITLE-ABS-KEY (small and medium enterprises OR SMEs)
(c) TITLE-ABS-KEY (software product quality) AND TITLE-ABS KEY (measures OR metrics) AND TITLE-ABS-KEY (soft-ware development companies) AND TITLE ABS-KEY (Colombia OR developing countries) AND TITLE-ABS-KEY (small companies OR young companies)
(d) TITLE-ABS-KEY (software product quality OR software quality) AND TITLE-ABS-KEY (measures OR metrics) AND TITLE ABS-KEY (software development) AND TITLE-ABS-KEY (South America OR Colombia) AND TITLE-ABS-KEY(young companies OR small organizations)
(e) TITLE-ABS-KEY (((((software quality) AND measures or metrics) AND software products) AND software companies) AND TITLE-ABS-KEY (small and medium enterprises OR SMEs)
(f) TITLE-ABS-KEY (+measures +OR +metrics +software +quality +software +developers +companies) AND TITLE-ABS-KEY (small companies OR young companies)

Below there are longer descriptions of some key activities related to software product quality found in the literature review. Some of the following studies are taken from a Colombian and South American context because this research is focused on Colombian software development companies that are essential for this study. Much of the literature had failings in various areas, which means no solution is effective but the candidate and this research will make suggestions to address this.

2.7.1 A New Model Based on Soft Computing for Evaluation and Selection of Software Products

Fernandez et al. (2018) propose a new model for the assessment and selection of software products according to their quality (see Figure 2.1). The model implements elements such as the manipulation of ambiguous and imprecise information from different resources. In addition, they validate this model 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.

Figure 2.1 shows the proposed model based on phases defined for the solution of a decision-making problem. It includes information modeling, determination of the essential criteria, the analysis/modeling interdependence between criteria, and the handling of heterogeneous data and with uncertainty. For this last point, it incorporates elements such as fuzzy logic and fuzzy linguistic modelling.

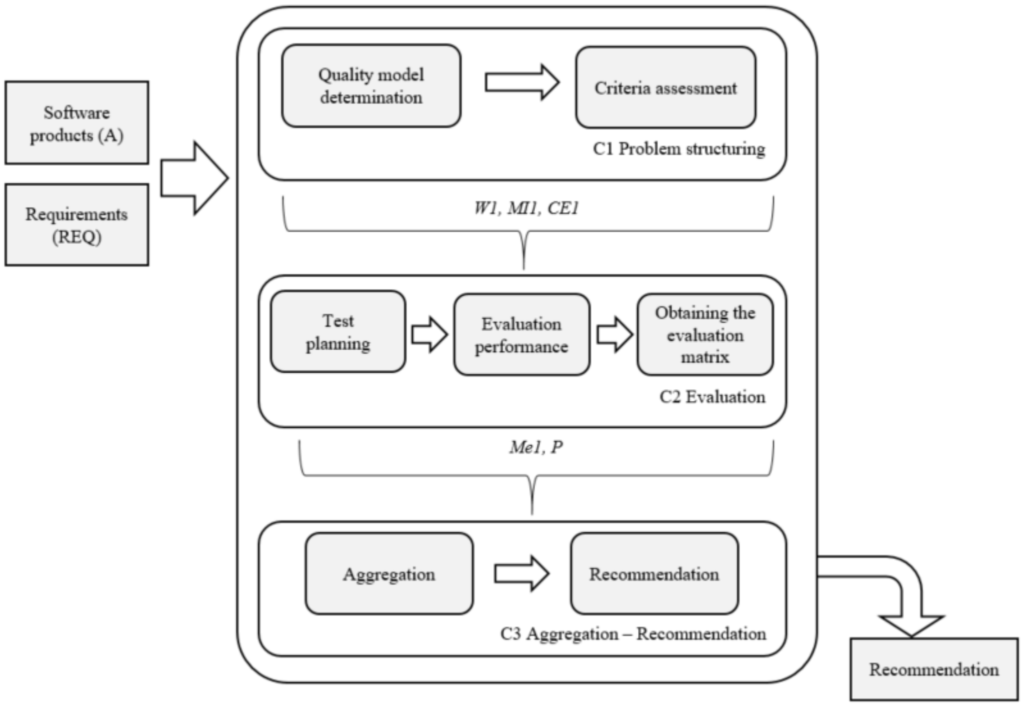


Figure 2.1: Model for assessment and selection of software products (Fernandez et al. 2018)

2.7.2 Usability and Accessibility as Quality Factors of a Secure Web Product

Baquero et al. (2018) 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.

The authors 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.

The framework presented in Figure 2.2 details the goals, decomposes the effectiveness, efficiency, and satisfaction measures, and lists all components of the context of use for measuring of usability attributes.

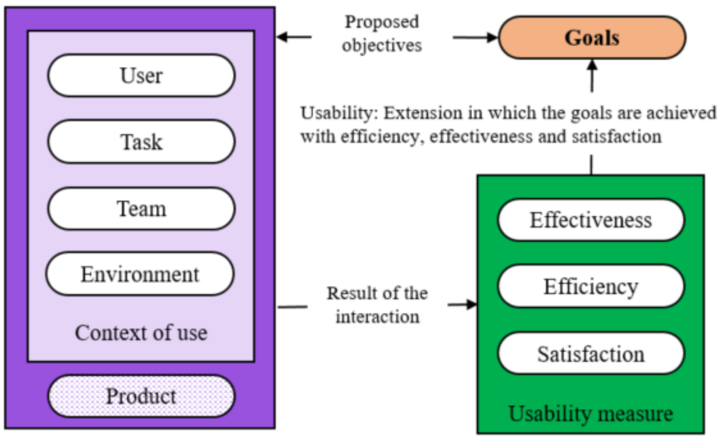

User
Task
Team
Environment
Context of use
Product
Proposed objectives
Goals
Usability: Extension in which the goals are achieved with efficiency, effectiveness and satisfaction
Result of the interaction
Effectiveness
Efficiency
Satisfaction
Usability measure

Figure 2.2: Usability definition framework according to ISO 9241-11 (Baquero et al. 2018)

2.7.3 A SQuaRE-Based Software Quality Evaluation Framework and Its Case Study

Nakai et al. (2016) propose a SQuaRE-based software quality framework, which successfully made tangible many product metrics and quality in use. These metrics were originally defined in the SQuaRE series ISO25023 (2016). 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 the 47 product metrics and the 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 (see Figure 2.3).

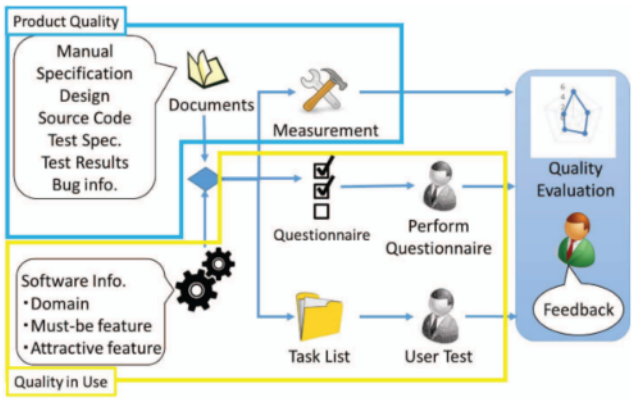


Figure 2.3: SQuaRE-based software quality evaluation framework (Nakai et al. 2016)

Figure 2.3 shows the overview of the procedure for employing the framework proposed. It requires manual specifications, bug information to measure the product quality, information collected via a questionnaire, and a user test using product and quality in use metrics. Then, the entire software quality is evaluated based on the findings to clarify which quality characteristics are sufficient or not.

2.7.4 Software Reliability Modeling Based on ISO/IEC SQuaRE

Febrero et al. (2016) 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 explain 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.

Figure 2.4 explains the decomposition of Reliability that can be mapped onto different stakeholders' needs, thus addressing the issue of capturing different viewpoints and needs. The low-level component allows obtaining a complete description, from the user's perception of the static measurable attribute and that mapping to the different user needs. This layout presents the complexity of the descriptions and the difficulties involved in applying them as a decision tool in control and management activities (Febrero et al. 2016).

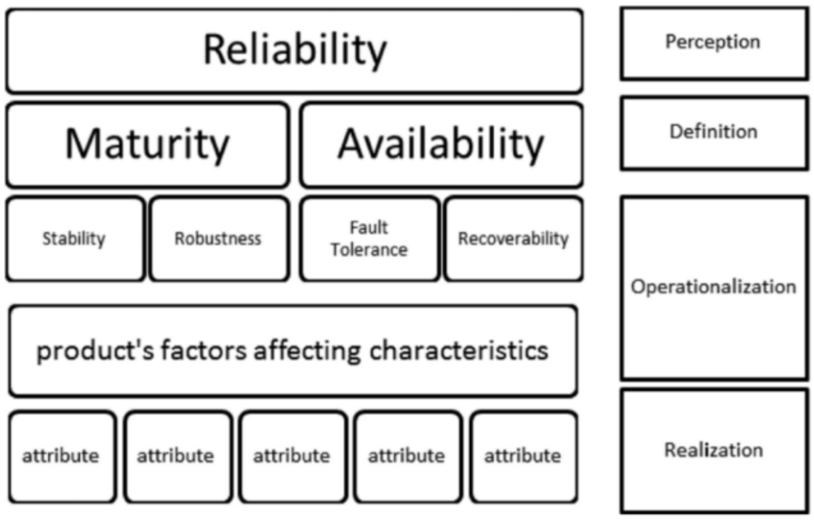


Figure 2.4: Reliability assessment schema (Febrero et al. 2016)

2.7.5 Diagnostic on the Appropriation of Metrics in Software Medium Enterprises of Medellin City

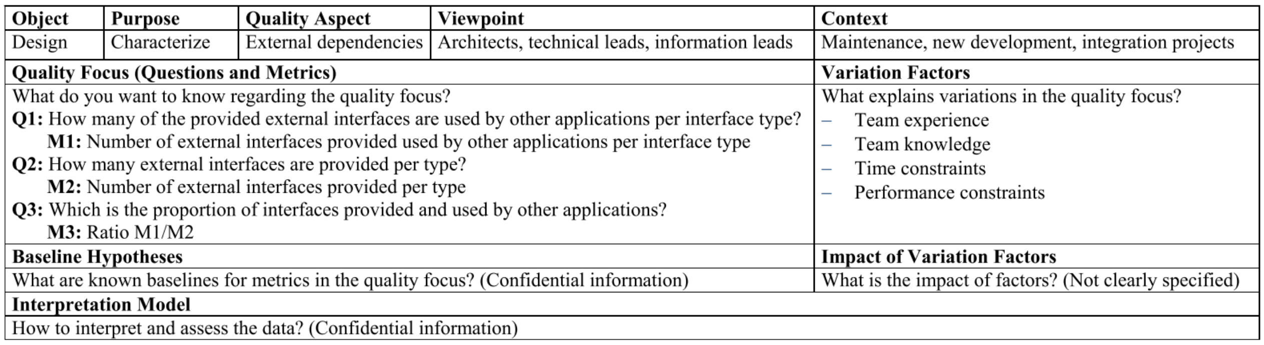
Metaute & Serna (2016) present an assessment of the ownership and use of metrics in medium enterprises in Medellin-Colombia, seeking to make recommendations that contribute to the strengthening of academia to support the industry. They found that some companies applied metrics during the years 2013-2015 in different stages of the software development life cycle. In addition, they use methods 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 did the product evaluation process. The authors combine different concepts and meanings from process quality and product quality, which leads to difficulty in understanding.

2.7.6 Software Quality Modeling Experiences at an Oil Company

Lampasona et al. (2012) 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 in their company to align the incorporation, acquisition, and development of IT solutions with the business goals. In addition, they developed this model 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.

Once the authors understood the major issues related to software development, they identified the quality needs and prioritize where measurement should be applied. They applied GQM paradigm for discovering each goal, metrics, and questions that were integrated into a comprehensive quality model for Ecopetrol.

Figure 2.5 describes the GQM abstraction sheet used in the initial quality model developed, which consisted of 10 measurement goals, 17 questions, 36 metrics, and 10 variation factors.



Object	**Purpose**	**Quality Aspect**	**Viewpoint**	**Context**
Design	Characterize	External dependencies	Architects, technical leads, information leads	Maintenance, new development, integration projects

Quality Focus (Questions and Metrics)	**Variation Factors**
What do you want to know regarding the quality focus? **Q1:** How many of the provided external interfaces are used by other applications per interface type? **M1:** Number of external interfaces provided used by other applications per interface type **Q2:** How many external interfaces are provided per type? **M2:** Number of external interfaces provided per type **Q3:** Which is the proportion of interfaces provided and used by other applications? **M3:** Ratio M1/M2	What explains variations in the quality focus? – Team experience – Team knowledge – Time constraints – Performance constraints
Baseline Hypotheses	**Impact of Variation Factors**
What are known baselines for metrics in the quality focus? (Confidential information)	What is the impact of factors? (Not clearly specified)
Interpretation Model	
How to interpret and assess the data? (Confidential information)	

Figure 2.5: GQM abstraction sheet for the design of external dependencies (Lampasona et al. 2012)

2.7.7 A Model for Measuring Agility in Small and Medium Software Development Enterprises

Escobar & Linares (2012) present 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. This contribution helps to understand the current state of the quality measurement, competitiveness, and productivity in Colombia.

Figure 2.6 shows the agility assessment model, which has four goals related to measuring the agility level of SMEs in project management, measuring the project agility or discipline in SMEs, measuring the work-team agility, and measuring the agile-workplace guidelines coverage. In addition, this model *"provides a general view of agility in the company and could allow companies to compare their agility levels and assess how successful are their transition-to-agile projects in terms of agile values, practices, and philosophy"* (Escobar & Linares 2012, p.9).

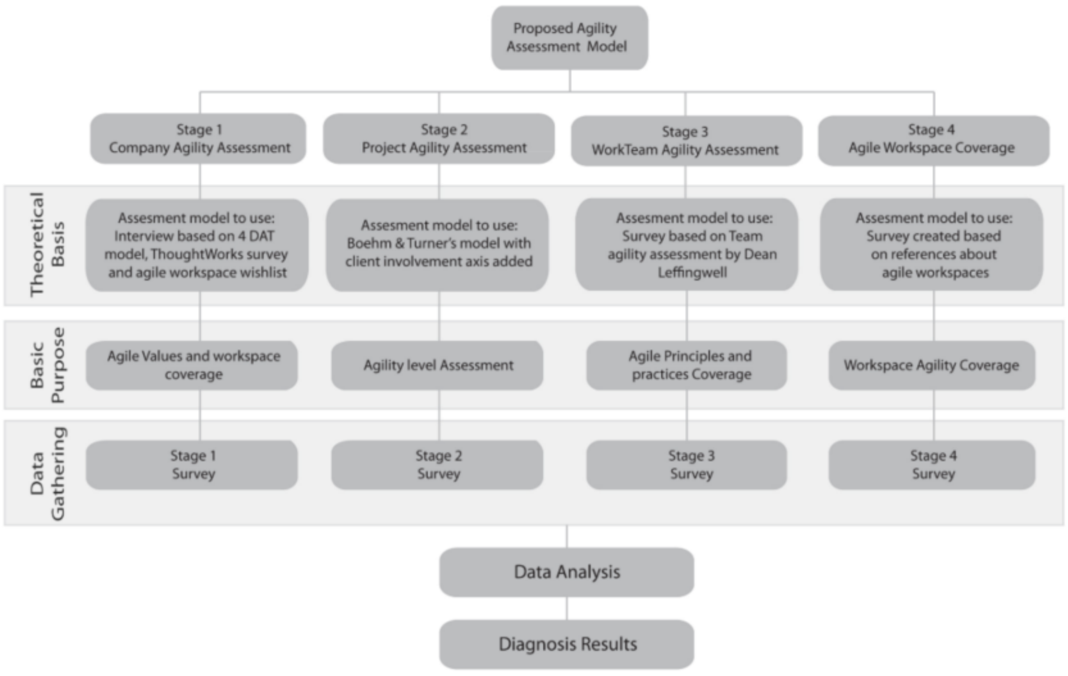

Proposed Agility Assessment Model
Stage 1 Company Agility Assessment
Stage 2 Project Agility Assessment
Stage 3 WorkTeam Agility Assessment
Stage 4 Agile Workspace Coverage
Theoretical Basis
Assesment model to use: Interview based on 4 DAT model, ThoughtWorks survey and agile workspace wishlist
Assesment model to use: Boehm & Turner's model with client involvement axis added
Assesment model to use: Survey based on Team agility assessment by Dean Leffingwell
Assesment model to use: Survey created based on references about agile workspaces
Basic Purpose
Agile Values and workspace coverage
Agility level Assessment
Agile Principles and practices Coverage
Workspace Agility Coverage
Data Gathering
Stage 1 Survey
Stage 2 Survey
Stage 3 Survey
Stage 4 Survey
Data Analysis
Diagnosis Results

Figure 2.6: Proposed agility assessment model (Escobar & Linares 2012)

2.7.8 A Framework for Evaluating the Software Product Quality of Pregnancy Monitoring Mobile Personal Health Records

Idri et al. (2016) present a set of requirements for mobile personal health records (mPHRs) for pregnancy monitoring and suggest the requirements that should be considered during the quality evaluation of these mPHRs. In addition, they calculate the impact of the requirements on software product quality using ISO/IEC 25000 standard.

The framework helps developers and evaluators to consider that the design of the checklist is the main key as regards deducing the impact of the requirements on software product quality. They lack focus on the other stages of software development (design, development, and testing) and only applied the process to the software requirements.

A checklist for discovering the requirements of an mPHR for pregnancy monitoring was created. The aim was comparing 30 external software product quality (SPQ) sub-characteristics to calculate the impact of the requirements on SPQ. Thus, analyzing the results, they discovered that certain sub-characteristics are more affected by the pregnancy monitoring mPHR requirements than others.

Figure 2.7 shows notably that characteristics such as functional suitability, reliability, performance efficiency, operability, and security, which are involved in the SPQ evaluation in health and also this framework could be adapted to other domains.

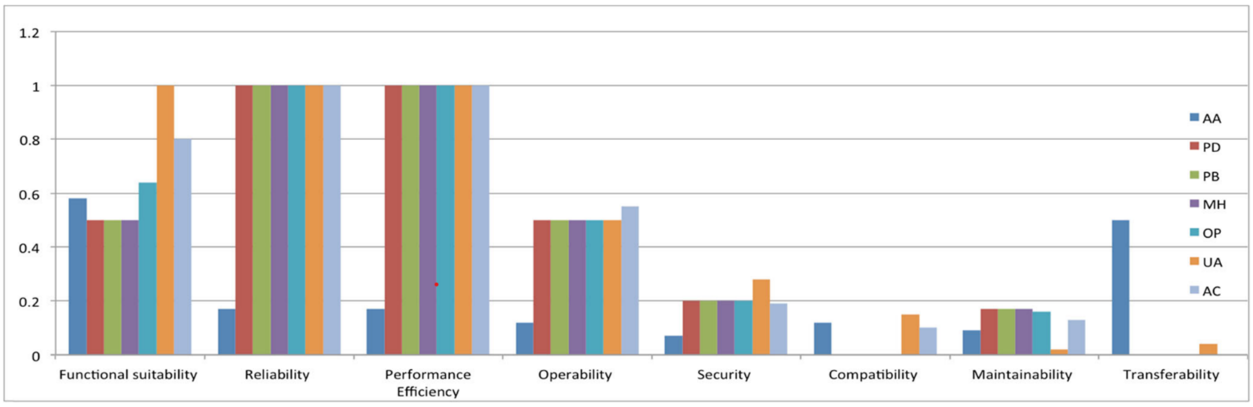


Figure 2.7: Impact of each block of requirements on the external sub-characteristics (Idri et al. 2016)

2.7.9 Evaluation of Software Product Functional Suitability: a Case Study

Rodriguez et al. (2016) describe a quality environment made up of a quality model, an evaluation process, and a set of tools for evaluating one of the characteristics proposed by the ISO/IEC 25000 standard: functional suitability. The evaluation results show a coherent level of compliance with the product requirements. The authors do not discuss the integration between characteristics and measures, which could help in assessing the overall quality process.

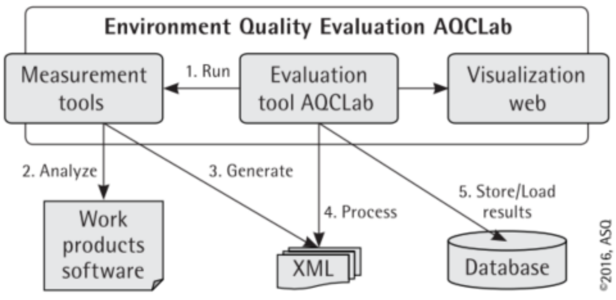


Figure 2.8: Environment quality evaluation (Rodriguez et al. 2016)

The technological environment showed in Figure 2.8 is made up of three levels, measurement tools that generates the results of the measures in an XML file, an evaluation tool which helps to obtain the values for the properties, characteristics, and sub-characteristics based on the measures results, and a visualization web that presents the information obtained in an understandable context. The authors validate this process in a case study applying this tool to the functional suitability characteristic.

2.7.10 Improving Software Product Line Configuration: a Quality Attribute-Driven Approach

Guana & Correal (2013) present a domain-specific modelling approach to determine whether the use of modelling tools can simplify and automate the definition process of a software product line (SPL), improving the selection process of reusable assets (including quality requirements i.e. security, performance, scalability).

The authors conclude that the approach significantly improves the software architect's decision making when they select the most suitable combinations of reusable components in the SPL context. They do not discuss SPQ and they neglect to explain how to evaluate software quality and which measures they use.

The Sensitivity Point Analysis Framework (SPAF) is a model-driven framework developed to support a product builder during the product derivation activities in an SPL. This framework has three key tasks; modeling the SPL core assets, modeling architectural sensitivity points and unifying these models into a conciliated model, and analyzing the conciliated model to select the most appropriate set of components for the SPL architecture implementation as is shown in Figure 2.9.

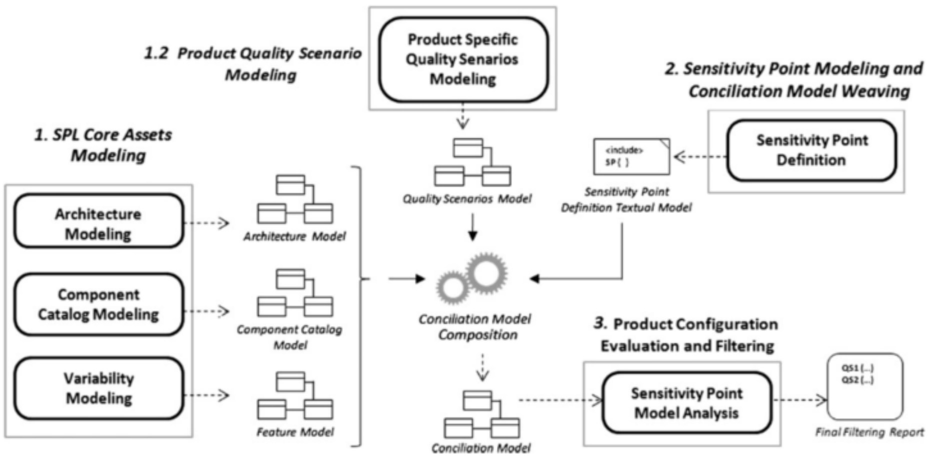

1.2 Product Quality Scenario Modeling
Product Specific Quality Senarios Modeling
2. Sensitivity Point Modeling and Conciliation Model Weaving
1. SPL Core Assets Modeling
Sensitivity Point Definition
Quality Scenarios Model
Sensitivity Point Definition Textual Model
Architecture Modeling
Architecture Model
Component Catalog Modeling
Component Catalog Model
Conciliation Model Composition
3. Product Configuration Evaluation and Filtering
Variability Modeling
Feature Model
Sensitivity Point Model Analysis
Conciliation Model
Final Filtering Report

Figure 2.9: SPAF execution process (Guana & Correal 2013)

2.7.11 Certification Process and Product Quality: Route Colombian SME Manufacturing Software

Pelaez et al. (2011) describe an exploration of recognized proposals focused on the software quality, certifications, and organizations focused on improving software quality in Colombia. They state that Colombian companies have implemented ITMark and Light MECPDS approaches as the most recommended practices for improving the software quality process. In addition, these authors emphasize the Colombia SME must have high quality to be competitive in the worldwide market. They do not discuss SPQ.

2.7.12 Assessment of Quality Factors in Enterprise Application Integration

Kumar et al. (2015) present a case study to identify detailed and operational level quality factors. They conducted a literature survey followed by interviews. They propose eight additional factors which impacted the quality of the software development projects. However, they identify the need to establish a framework that will provide a set of preventive and corrective actions for the quality factors which help in improving the system quality. The authors avoid selecting which evaluation approach they use for proposing the new factors and how they assess such quality factors.

2.8 Summary

The aforementioned studies describe different customized models integrating some measures in their descriptions, but they do not explain: how those models would work in a real software project, which measures and relationships they have established, or help to understand SPQ evaluation in young software development companies. Some models and standards recognised in the software industry were created to be implemented in large software Colombian companies with high financial capability and experience. These models are not necessarily appropriate because there is no empirical software engineering research available about the experience of young software development companies in a developing country. If this empirical research were available it would help these companies to evaluate SPQ. The gap of understanding SPQ evaluation in young companies is the principal focus of this research. Therefore, to address this gap the candidate presents in Chapter 3 an approach to find out how young software development companies do use and may use, SPQ.

Chapter 3

RESEARCH DESIGN

In this section, the candidate provides a description of the research approach used. At the beginning of this study, the candidate applied a quantitative research approach via a closed questionnaire, which had the objective to investigate which characteristics, measures, and methods are currently used by Colombian software developers to evaluate software product quality (SPQ). The target participants were software development companies and their developers, testers, and project managers.

The candidate ran a pilot study with 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 and called for a re-design of the approach.

The gap of understanding SPQ evaluation in young companies is the principal focus of this research. Therefore, this empirical research wants to explore how SPQ is evaluated in young software development companies in Colombia to put SPQ in practice and addressing this gap. The literature review from Chapter 2 helps with the research design for developing clear research questions and understand the method proposed.

3.1 Quantitative Research Design

3.1.1 Research Question and Hypothesis

Based on the literature review, the hypothesis developed for this research is: *Competitiveness of Colombian software development companies is positively related to software product quality measurement.*

The hypothesis is for understanding software quality attributes, measurement indicators, what software quality characteristics are important for Colombian companies and which of these characteristics the Colombian software teams must measure. In addition, how project deliverables are aligned with end-user requirements and market necessities. Assessing these elements will help to understand the clarity in the measurement of the software product quality (SPQ), and how well it is defined and communicated with all development life-cycles. The hypothesis is true if collected data from the literature review compares with data collected from experts' information and these indicate more software project success when SPQ measures are clearly defined and a common understanding of software quality is present in the software developers.

Software development projects' success is directly related to SPQ measurement. There are many studies in the literature on issues of quality and its impact on the evaluation approaches used to measure the software product quality in software development companies. They present fundamental problems such as insufficient measurement of quality which leads to low productivity and large schedule increases (Rahmani et al. 2016, Lampasona et al. 2012, Rodriguez & Piattini 2012*b*). In addition, as previously mentioned on chapter 2, there is greater and greater acceptance of the idea that *"software quality evaluations should be based on direct evidence about the product, not only on evidence about the process"* (Maibaum & Wassyng 2008) since a high-quality process does not necessarily ensure a good quality product. If the hypothesis is correct, most of the data collected during the research development will show which appropriate characteristics and measures software developers should implement in Colombian companies. If the hypothesis is false, the outcomes will show that it is not necessary to apply attributes to measure SPQ. A mixed result is when it is not possible to identify specifically a true or false result on which of the measurement criteria are suitable within the software organisations. Surveys were planned to collect data to confirm the hypothesis as well as if the result went against the hypothesis. The questionnaire had three layers; direction, confirmation, and solution. Data was collected to identify which SPQ measures and characteristics were important for software development companies in Colombia and which of those attributes the Colombian software developers must measure and how.

3.1.2 Quantitative Method

To find a solution to the identified problem, the hypothesis was developed. A quantitative research method approach (Creswell 2013) was selected to use for this research for collecting data, verifying and validating the hypothesis. Quantitative analysis was used because data collected from surveys were in numbers, using a Likert scale for the value determination (Jacoby & Matell 1971). Collected data analysis was a quantitative method to provide a clear indication of whether the hypothesis was validated.

This study was looking to develop a measurement model, in which the characteristics and elements were appropriate for evaluating SPQ in Colombian SME or young companies, which permit improvements to the positioning in the market, high-quality in developed product and user satisfaction. The research design was as follows:

- To identify the types of measures and methods existing and applied by software developers.
- To identify international standards to measure software quality and software product quality.
- To evaluate the current literature review on the subject critically. This included Colombian research studies as well as other international studies.
- To identify all companies and experts who are in contact with the software development industry and SPQ measurement processes.
- To contact the ICT Ministry, Fedesoft, Procolombia in Colombia to register their experiences and requested their support for the research development.
- To contact 12 cluster managers to filter software development companies by departments to be able to use their support and knowledge: 25 companies were required for the survey application to 3 employees/target (a developer, a tester, and a manager). The objective was to investigate the experts' experience and knowledge in those areas about software development cycle, measurement aspects, measures, methods, and standards applied in the software products developed.

- To conduct online surveys with approximately 75 employees from different software development companies in Colombia.

The research was conducted through the following stages:

- *Stage 1- Problem definition, Literature Review:* The objective was to identify the research problem and prepare a literature review about software product quality evaluation in the world markets. These included software development companies' main concern, SPQ measurement models and standards, SPQ evaluation success cases and needs, measures that contribute to improving SPQ evaluation, and ways to satisfy the competence.
- *Stage 2- Preparation:* Preparing a questionnaire based on the hypothesis and data collection criteria, complete ethics applications, data collection process, and data analysis. The questionnaire was to cover all software quality topics, dimensions, and criteria following a measurement structure. First, demographic data was collected in Colombia with a clear indication of organizational and technical levels. This section could confirm, or otherwise the hypothesis state with data evidence.
- *Stage 3- Sample set selection:* Drawing a sample set from software development experts for data collection, make initial contact with experts and companies, get preliminary consent to be part of the research and schedule survey application. The stage also covers the establishment of data collection and data analysis tools. The sample was divided into different categories based on the target experience.
- *Stage 4- Data collection:* The objective was to collect and preserve research data (using some tools) through online surveys, (through Google forms). The sample size for this research was 75 surveys according to the sample equation. It was important that the experts/participants had at least two years' experience and that the organisation has been legally constituted for two or more years. Data collection was from young software development companies in Colombia.
- *Stage 5- Data Analysis:* Analysing collected data using excel, dynamic tables, and ANOVA as the statistical analysis method.
- *Stage 6- Model:* The objective was to design the measurement model, describe outcomes, and organise a final research report.

There is more information and explanation about this quantitative research (Study 1) in Chapter 4.

3.2 Qualitative Research Design

The candidate now provided an overview of the qualitative research approach used for developing the research. As the research team was interested in what the software development companies 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 (Crotty 1998, p.7). So, the candidate used a phenomenological approach, focused on software developers' lived experience, which would enable them to explore their understanding of SPQ.

This research is focused on the following objectives:

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

The candidate needed to explore the understanding that the interviewees have on software product quality. To fulfil this objective of understanding, the candidate needed to ask open-ended questions, and use a constructionist research method, to understand what the interviewees understood by software product quality and how it was measured in the companies (Crotty 1998). Quantitative research (Study 1) will be described in Chapter 4 and qualitative research (Study 2) will be described in Chapter 5.

Figure 3.1 describes the research process that explains epistemology, the theoretical perspective, methodology and methods, which involves procedures including questions, data collection from the participants, data analysis, and researcher interpretations of data meanings.

According to Crotty (1998) and Creswell & Creswell (2017), *epistemology* is the theory of knowledge embedded in the theoretical perspective and thereby in the methodology itself. *Theoretical perspective* is the philosophical stance informing the methodology and thus providing a context for the process and grounding its logic and criteria. *Methodology*

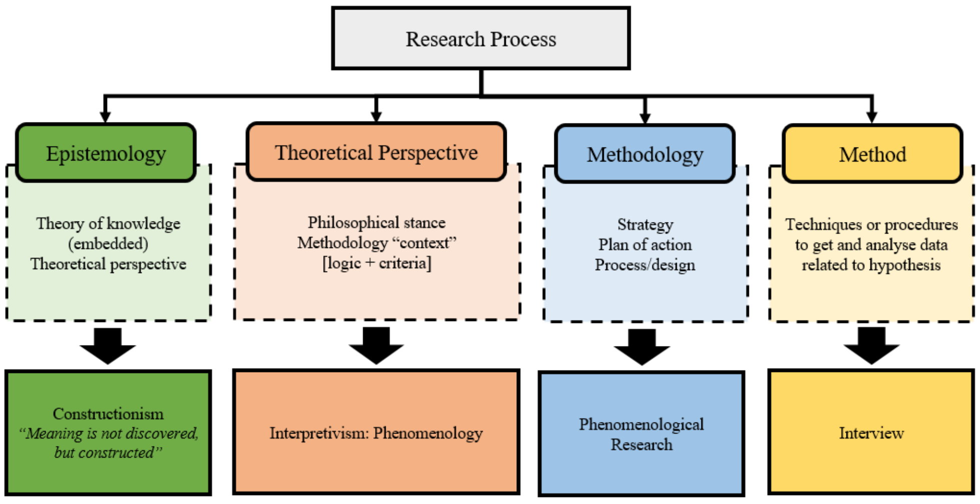

Research Process
Epistemology
Theory of knowledge (embedded) Theoretical perspective
Theoretical Perspective
Philosophical stance Methodology "context" [logic + criteria]
Methodology
Strategy Plan of action Process/design
Method
Techniques or procedures to get and analyse data related to hypothesis
Constructionism "Meaning is not discovered, but constructed"
Interpretivism: Phenomenology
Phenomenological Research
Interview

Figure 3.1: Research Process, adapted from Crotty (1998)

refers to the strategy, plan of action, process or design lying behind the choice and use of particular methods and linking the choice and use of methods to the desired outcomes. Finally, *Methods* are focused on the techniques or procedures used to gather and analysts' data related to some research question or hypothesis. More details about this research process will be explained on the next sections of this chapter.

In this work, with closed questionnaires not working when trialled, we needed to explore what people understood by software product quality (SPQ), which measures they used and how they evaluated SPQ. So we needed to discover the meaning and the understanding that people had.

Constructionism contributes to exploring and understanding the meaning that individuals or groups ascribe to a human or social problem (Creswell & Creswell 2017, p.15). One technique used to get data is the interview, which permits the understanding of the participants' experience and issues. It is important to consider that the participants are selected only if they have 'the experience of developing software' under study (Goulding 2005, p.302).

Questions driving the interviews were grouped into four categories:

- Classification or demography.
- SPQ in the organisation.

- SPQ personal perspective.
- SPQ evaluation at both the personal and organisational level.

The participants answered research questions based on i) what software developers/companies evaluated in product quality, and ii) how software developers/companies evaluated software product quality. This study followed the steps described in Marshall et al. (2015) to design interviews and apply pilot tests.

3.3 Epistemology Informs the Theoretical Perspective

Epistemology is the theory of knowledge embedded in the theoretical perspective and thereby in the methodology. There are different epistemology views such as objectivism, constructionism, and subjectivism. *Objectivism "holds that meaning, and therefore meaningful reality exists as such apart from the operation of any consciousness"* (Crotty 1998, p.8). *Constructionism* points out that *"people can construct meaning in different ways, even in relation to the same phenomenon"* (Crotty 1998, p.9). In *Subjectivism "the meaning does not come out of an interplay between subject and object but is imposed on the object by the subject. Here the object as such makes no contribution to the generation of meaning"* (Crotty 1998, p.9).

The research process (Figure 3.1) included a theory of knowledge embedded in a constructionism epistemology, which stated that there is no objective truth waiting for us to discover it, but the meaning is not discovered but constructed. Constructionism is *"the view that all knowledge, and therefore all meaningful reality as such, is contingent upon human practices, being constructed in and out of the interaction between human beings and their world and developed and transmitted within an essentially social context"* (Crotty 1998, p.8).

On the other hand, this epistemology permits that participants can construct the meanings of a situation, *"typically forged in discussions or interactions with other people"* (Creswell & Creswell 2017, p.39). According to the constructionist view, we do not create meaning, we construct meaning (Creswell & Creswell 2017, p.21).

"The theoretical perspective helps to describe the philosophical stance that lies behind our chosen methodology. Moreover, it explains how to provide a context for the process and grounds its logic and criteria with different assumptions" (Crotty 1998, p.7). This is because exposing and justifying such assumptions is necessary to know our view of the human world and social life within that world (Crotty 1998). Some recognised theoretical perspectives approaches are interpretivism (including phenomenology, symbolic interactionism, and hermeneutics), positivism (and post-positivism), critical inquiry, feminism, and postmodernism.

3.4 Phenomenological Research as a Methodology

The theoretical perspective is focused on interpretivism, which "was conceived in reaction to the effort to develop a natural science of the social. Its foil was largely logical empiricist methodology and the bid to apply that framework to the human inquiry" (Schwandt et al. 1994, p.226). Within this theoretical perspective is Phenomenology which "suggests that, if we lay aside, as best we can, the prevailing understandings of those phenomena and revisit our immediate experience of them, possibilities for new meaning emerge" (Crotty 1998, p.78). This was the chosen approach, based on the misunderstandings and interpretations from the quantitative questionnaire.

Crotty (1998) states that phenomenology could be seen as objective because it is an attempt to understand something new rather than being satisfied with another's description of that experience, we should therefore put our usual understanding into abeyance and see it afresh. Phenomenology helps us to question our culture and manner of seeing the world, perceiving the things as they present themselves to us as conscious human beings, what we experience directly, without being coloured or made opaque by prevailing understandings of some phenomena. In addition, phenomenology invites us to step away from the cultural meanings and understandings we inherit, and to engage in true meaning-making, to try to make sense of the phenomena as directly experienced by ourselves.

The discussion above suggests adopting a phenomenology research methodology as a suitable methodology for the research. The method used will be open ended interviews. The methodology and methods are discussed below.

This research approach will help to investigate in-depth the 'lived experiences' of software development companies the problems associated with software product quality, how they evaluate it, which measures and methods they are using to do so. Moreover, it helps to build meanings from their experiences and knowledge in software development.

In the research methodology, the candidate describes a strategy or plan of action that shapes his choice and use of particular methods and links them to the desired outcomes (Crotty 1998, p.3). Also, it provides a rationale for the choice of methods and the particular forms in which they are employed.

3.5 Methodology Governs Our Choice and Use of Methods

Moustakas states that *"phenomenological research is a strategy of inquiry in which the researcher identifies the essence of human experiences about a phenomenon as described by participants. Understanding the lived experiences marks phenomenology as a philosophy as well as a method, and the procedure involves studying a small number of subjects through extensive and prolonged engagement to develop patterns and relationships of meaning"* (Moustakas 1994, p. 61). For this research it was important to understand the participants' experiences and leave aside the candidate's own experiences.

The challenge in this research is focused on two aspects: how to help participants express their world as directly as possible and how to explain these dimensions through their experiences to reveal the lived world. In the case of this thesis, the lived world is a world the software developers developing software in a commercial environment in Colombia.

The methodology used is phenomenological research because, unlike ethnography (Goulding 2005, p. 295), it does not require direct and physical contact with the population to collect data. In using phenomenology, the research will achieve its aims if the phenomena present themselves to the candidate instead of the candidate imposing preconceived ideas on them.

Research methods help to describe the concrete procedures or techniques the candidate plan to use in the research design. There will be specific tasks to gather and analyse the data. These tasks are the research methods (Creswell & Creswell 2017, p.64).

3.6 Research Method Proposed

The basis for choosing the method will help the choice of methodology, given that interviews would be done remotely due to the candidate is being in Sydney, Australia. Such interviews need to be conducted with software development companies in Colombia. In addition, the method must allow the phenomena to reveal themselves, which suggests semi-structured interviews.

3.6.1 Selection of Interviewees

It was important to take account in this method that the participants were selected only if they have lived the experience under study. Sampling was therefore purposeful and prescribed from the start and the main instrument of data collection was the interview (Goulding 2005).

3.6.2 Discovering Meanings

The aim of this research process is to construct meanings as we interact with the context of the software development companies to know how they are interpreting the process of measuring the software product quality (SPQ) and what and how they are measuring the quality of their products. The candidate designed open-ended questions so that the participants could interact directly through mass media tools, such as Skype, Gtalk or video-conference with the interviewer and share their views about the research. Moreover, it was important to share information about their experiences and their own work during the product development process. After this step, this research sought to understand the context or setting of the participants through analysing this context and gathering information personally. The analysis process consisted of understanding the meanings collected from people, assisted by NVivo, which is a qualitative data analysis computer software package that aids managing the volume of data.

3.7 Will the Results be Valid?

3.7.1 Data Saturation

The candidate considered how to demonstrate the validity of the methods in this section. There was no one-size-fits-all method to reach data saturation. It was difficult to predict in advance the sample size, and in fact, a researcher probably should not try to do so in phenomenological research. Data saturation is a principle most frequently used in grounded theory that uses qualitative interviews for data collection. However, much qualitative research has an interpretivism theoretical perspective underpinning it, such as Phenomenology. In Phenomenological research data saturation is not about numbers per se, it is about the depth of the data (Burmeister & Aitken 2012, p.271). With this study, the research team would have data that is expressed in terms of quality (rich) and not quantity (thick) (Burmeister & Aitken 2012, p.272). In addition, the candidate must address data saturation using their specific research lens primarily (Fusch & Ness 2015, p.1410), (Mason 2010, p.5).

One of the markers of valid phenomenological research is data saturation. Data saturation is about the depth of the data (Burmeister & Aitken 2012, p.271). The depth of the data is expressed in terms of quality, as the richness of the data (Burmeister & Aitken 2012, p.271), (Fusch & Ness 2015, p.1409). Data saturation is reached when no new or relevant data seem to emerge regarding a pattern (Guest et al. 2006, p.60).

The candidate planned to carry out 20 interviews initially. This research 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 (Guest et al. 2006, p.78).

To ensure that the process used was valid, the candidate adopted a well-researched iterative method, further described in section 5.1. This uses iterations between the collection, data analysis and conclusions drawn.

3.7.2 Validity of Analysis

For the all patterns, the research team considered saturation had been reached when the candidate and supervisory team agreed that no new or relevant data emerged regarding any patterns. This was the outcome of the iterative process described below.

The candidate did the analysis, which was verified and clarified by the supervisory team. They participated in essential dialectic discussions several times per week during the fieldwork period, and beyond to clean the data, understand the data collected and suggest to the candidate new questions for improving the data. They also did some analysis and interpretations.

3.8 Ethical Issues

Ethical issues were considered, and risks mitigated as in the Appendix G. No ethical issues arose during the practical portion of this research. The language that the candidate conducted all interviews could be a potential problem during the translation Spanish to English, but it was managed by the supervisor who helped the candidate in the reviewing of each translation so as to avoid missing any important data.

Pre-interview data collected on the companies were confidential. Only the research team (the candidate and the supervisory team) had access to this data. Participant companies would be not identifiable when the research is published because the data and the companies were de-identified, and the results will only be published in summary form with regard to the companies. The participant's names have been changed using pseudonyms. No video recordings (re-identifiable data) were made, only audio recordings, and the data from these were de-identified and kept confidential.

The interviews were de-identified, non-personal, and non-intrusive. They were classified as Low Risk by the UTS Criteria (UTS 2008). This research required completion of the corresponding UTS Human Research Ethics Committee Approval Form and approval from the ethics committee, which was granted (ETH18-2553). No other ethical issues arose during the research.

3.9 Summary

This chapter has presented the research design used to find out which characteristics, measures, and methods are currently used by Colombian software developers and their companies to evaluate software product quality. Furthermore, the candidate presented the quantitative research design applied from the beginning of this research, followed by the qualitative research design focused on Constructionism as epistemology, its theoretical perspective leads by the phenomenology, the phenomenology research as the methodology, and the semi-structured interviews as the method for collecting data. The validity and data saturation of this research is addressed in this chapter as well as the ethical issues considered. The following Chapter 4 will discuss the findings of the quantitative study.

CHAPTER 4

QUANTITATIVE FINDINGS [STUDY I]

This chapter covers the first data collection in the quantitative research design including question design, analysis method, findings, and the new research method proposed. These decisions significantly impact and shape the chapters to come.

4.1 Question Design

The designed questions were defined in an online survey with closed questions and a Likert scale. The objective of the questions was to seek to know which characteristics, measures, and methods are used by Colombian software development companies for evaluating software product quality (SPQ).

Using some information from the literature review, the candidate decided to design questions based on non-functional requirements described on ISO 25000 series related to eight software quality characteristics, namely functional suitability, performance efficiency, compatibility, usability, reliability, security, maintainability, and portability (ISO25023 2016).

Appendix A shows the survey with four sections focused on i) measuring the software product quality, ii) measures and measurement methods to evaluate SPQ, iii) software tools to measure SPQ and certifications achieved, and iv) classification section. Appendix B presents the consent form, Appendix C shows the invitation letter for the participants, and Appendix D presents the participant information sheet (PIS) that describes all information about the project and the process for applying this fieldwork.

4.2 Analysis

The analysis process started with a quantitative research design, in which the candidate applied 11 surveys to young software development companies in Colombia to identify a set of measures appropriate for software products. However, after the candidate applied the survey to validate the information included, it was found that the participants did not understand or know the terminology, characteristics, and measures for evaluating SPQ that were used in the survey, and so they did not apply such measures. Tables 4.1, 4.2, and 4.3 show some measures described in such instrument, which were taken from ISO 25023 series (ISO25023 2016).

The candidate decided to do a pilot before the final fieldwork because the research needed to have validation and feedback from the software development teams' expertise. The next section describes the findings identified in those surveys.

4.3 Findings

The pilot fieldwork was done at the beginning of 2018 with 11 young software development companies which were considered by Fedesoft as Small and Medium Enterprises —SME— Fedesoft (2015). The candidate sent to the companies and employees who accepted participation in the pilot all the documentation including the consent form, participant information sheet, and the survey link to complete the closed questions. Once the participants had completed the instrument, the candidate received some feedback related to the misunderstandings of many questions, terminology matters, and imprecise knowledge of software product quality. Some of the participants revealed that they completed the survey based on their own experience but they did not apply it in real software development projects. Tables 4.1, 4.2, and 4.3 show the results collected from the data.

The 11 participants have roles related to project managers (M), developers (D), and testers (T). A Likert scale (Never | Rarely | Occasionally | Frequently | Very frequently) was used to respond to the closed questions. Frequently and Very Frequently were chosen as the best responses. Such responses were focused on measures and measurement methods to evaluate SPQ. The measures showed in Tables 4.1, 4.2, and 4.3 were imposed and chosen by the candidate based on ISO 25000 series (ISO25023 2016). In addition, all participants stated that they were not familiar with those measures and their meanings. Therefore, they researched the meaning and tried to connect it with their work in software projects and their experiences.

Table 4.1: Quantitative data results (1 of 3)

Charact.	**Measure**	**M1**	**M2**	**M3**	**M4**	**M5**	**D1**	**D2**	**D3**	**D4**	**T1**	**T2**
Functional suitability	Functional coverage	x	x		x	x		x	x	x		x
	Functional correctness	x				x	x	x				
	Functional appropriateness of usage objective	x	x			x	x	x		x		
	Functional appropriateness of system	x	x		x	x		x		x		
Performance efficiency	Mean response time	x				x	x					x
	Response time adequacy	x					x					x
	Mean turnaround time	x				x		x				
	Mean process utilization				x		x	x				
	Mean memory utilization				x		x	x				
	User access capacity	x										
	User access increase adequacy	x				x						
Compatibility	Co-existence with other products	x	x			x	x	x				
	Data formats exchangeability	x	x		x			x				
	Data exchange protocol sufficiency	x	x									x
	External interface adequacy	x	x		x			x				x
Maintainab.	Coupling of components	x	x									
	Cyclomatic complexity adequacy	x										x
	Reusability of assets	x				x						
	Coding rules conformity	x										
	System log completeness	x										
	Modification efficiency	x	x									
	Modification capability	x	x	x								
	Test function completeness	x										
	Test restartability	x										

Table 4.2: Quantitative data results (2 of 3)

Charact.	**Measure**	**M1**	**M2**	**M3**	**M4**	**M5**	**D1**	**D2**	**D3**	**D4**	**T1**	**T2**
Security	Access controllability	x	x		x	x		x	x		x	x
	Data encryption correctness	x				x		x	x		x	
	Data integrity	x	x	x	x	x	x	x	x		x	
	Internal data corruption prevention	x			x	x		x	x		x	
	Digital signature usage	x				x		x	x		x	
	User audit trail completeness	x				x		x	x		x	
	System log retention	x	x			x		x	x		x	x
	Authentication mechanism sufficiency	x	x		x	x		x	x		x	x
	Authentication rules conformity	x	x		x	x		x	x		x	x
Reliability	Fault correction	x			x	x	x	x	x			x
	Test coverage	x		x			x					x
	System availability	x		x	x	x		x	x			
	Redundancy of components	x		x				x	x			x
	Mean fault notification time	x				x		x	x			
	Mean recovery time	x			x	x		x	x			
	Backup data completeness	x	x	x	x	x		x	x			
Usability	Description completeness	x		x				x				
	Usage guidance completeness	x	x					x				
	Error messages understandability	x		x	x							
	Operational consistency	x	x	x	x		x					
	User interface customizability	x			x							
	Appearance consistency	x	x		x							
	User entry error correction	x	x									
	Appearance aesthetics of user interfaces	x	x		x		x					
	Accessibility for users with disabilities	x										

Table 4.3: Quantitative data results (3 of 3)

Charact.	**Measure**	**M1**	**M2**	**M3**	**M4**	**M5**	**D1**	**D2**	**D3**	**D4**	**T1**	**T2**
Portability	Hardware environmental adaptability	x					x					
	System software environmental	x	x		x	x	x					
	Operational environmental adaptability	x	x				x					
	Installation time efficiency	x			x	x		x		x		
	Ease of installation	x										
	Usage similarity	x										
	Product quality equivalence	x										
	Functional inclusiveness	x								x		
	Data reusability/import capability	x	x							x		

Some companies did not use non-functional requirements, measures, and measurement methods to evaluate SPQ. They evaluated SPQ as follows i) quantifying the hours dedicated to the re-processes of software development and the reopening of generated tickets ii) the quantity of product defects and time for correction or reprocessing iii) the value that the client perceives in the use and the number of incidents iv) the number of reported incidents vs. resolved incidents v) customized processes (using Excel) vi) the solution of incidents based on ISO 20000 vii) incidents that occurred viii) the intrinsic product quality.

For the question *"Does your company evaluate any software quality characteristics?"*, the participants highlighted security, functional suitability, and usability as the most implemented non-functional requirements. However, the questionnaire did not ask how they evaluated or implemented those characteristics and which measures they used to validate them.

Some participants evaluated SPQ using manual tools (six out of eleven) and five out of eleven participants implemented automatic software tools such as Cheekpoint, Sonar, Jenkins, NUnit, and JUnit. Some companies have achieved certifications related to software process quality rather than software product quality. Three companies have CMMI 3 and CMMI-Dev 5, one ITMark, and one ISO/IEC 291110, which is ongoing.

4.4 New Research Method

At the beginning of this study, the candidate started applying a quantitative research design through a questionnaire, which had the objective to know which characteristics, measures, and methods are currently used by Colombian software developers to evaluate software product quality (SPQ). Importantly, however, the candidate supposed that software development companies understood the SPQ meanings, also referred to as a non-functional quality. The candidate applied a pilot study to 11 people of four companies for testing the questionnaire before proceeding to collect the final data. The findings showed that the interviewees did not understand some questions (see Appendix A), concepts and elements defined in the questionnaire and so we found the initial results obtained were not useful.

The candidate therefore firstly needed to explore what the participants did understand by SPQ, based on the literature review described in Chapter 2, which described a process to explore the lived experiences of participants working in software development companies in Colombia. The candidate proposed a questionnaire for this software development audience. The audience were some developers, testers, and project managers. The candidate anticipated that interviewing 20 people from seven software development companies would be sufficient.

The concepts to be explored are related to non-functional characteristics and their meanings. The participants did not understand how to apply some of those concepts in real software projects. Table 4.4 and 4.5 show such concepts.

Table 4.4: Non-functional characteristic/requirement concepts (1 of 2) (ISO25023 2016)

Characteristic	**Concept**
Functional suitability	Used to assess the degree to which a product provides functions that meet stated and implied needs when used under specified conditions.
Performance efficiency	Used to assess the performance relative to the amount of resources used under stated conditions. Resources can include other software products, the software and hardware configuration of the system, and materials.
Compatibility	Used to assess the degree to which a product can exchange information with other products, systems or components, and/or perform its required functions, while sharing the same hardware or software environment.

Table 4.5: Non-functional characteristic/requirement concepts (2 of 2) (ISO25023 2016)

Characteristic	**Concept**
Usability	Used to assess the degree to which a product or system can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.
Reliability	Used to assess the degree to which a product performs specified functions under specified conditions for a specified time-period.
Security	Used to assess the degree to which a product or system protects information and data so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorization.
Maintainability	Used to assess the degree of effectiveness and efficiency with which a product or system can be modified by the intended maintainers.
Portability	Used to assess the degree of effectiveness and efficiency with which a product can be transferred from one hardware, software or other operational or usage environment to another.

On the other hand, elements are the attributes measured in each non-functional quality requirement/characteristic. For instance, Table 4.1, 4.2, and 4.3 present some measures chosen by the candidate for all characteristics. Some companies do not apply these elements or measures in their software development projects or, as they explained to the candidate, they use different protocols or customized ways to evaluate or track some attributes to the process level, not product level. In addition, they did not give a response to how SPQ is measured in their software projects. This could have been because the questions designed were closed. For example, the questionnaire asked about *'measurement methods'*, and *'measurement functions'*. Measurement method is "a logical sequence of operations used to quantify properties with respect to a specified scale. The results of applying a measurement method is called a quality measure element". A measurement function is "an algorithm used to combine quality measure elements. The result to applying a measurement function is called a quality measure" (ISO25023 2016).

These concepts were not explained before applying the questionnaire because the candidate imagined that the Colombian development teams already knew those concepts and elements types.

After submitting all questionnaires through Google forms, some participants contacted the candidate by email or a phone call and they gave their feedback about unclear questions, especially in the criteria and terminology related to some non-functional requirements and measures that they did not apply because they used their own indicators instead. Then, the findings showed that the interviewees did not understand some questions, concepts, and elements defined in the questionnaire and the results were not useful.

As a result, the candidate changed the research design and technique and he decided to not use the quantitative design. The candidate proposed a qualitative method that allowed meanings to be built from software development companies' experiences and their employees such as developers, testers, and project managers without imposing ideas, elements, and meanings from the standards and literature review. In addition, the candidate proposed to present the Study 1 findings in this Chapter for supporting the idea for changing the research design. This would be aimed at understanding the difficulties of young software development companies and the shortcoming of existing solutions in terms of software product quality. Such a research methodology would be critical to understanding how Colombian software development companies evaluate SPQ. This research turns now to discuss this methodology in the next chapter.

4.5 Summary

This chapter has presented the first data collection in the quantitative research design including question design, analysis method, findings, and the new research method. The candidate demonstrates through the analysis that to do a pilot before the final fieldwork is the best way to validate and get feedback from the software development team's expertise. In addition, the findings support the idea for exploring and applying a qualitative research method because the participants did not understand some questions, concepts, and elements defined in the questionnaire and the results were not useful. The decision significantly impacts and shapes the chapters to come.

Chapter 5

QUALITATIVE ANALYSIS APPROACH [STUDY II]

This chapter describes the qualitative analysis approach and the data collection process that the candidate used after the pilot where problems with misunderstanding of terminology were encountered. This study follows Miles et al.'s interactive model (Miles et al. 2014) to guide the thematic and interpretive analysis. The candidate performed both a manual analysis and a semi-automatic analysis using NVivo.

5.1 Overview of the Approach

Thematic and interpretive analysis "are used by phenomenologists to make sense of their data" (Creswell 2013, Giorgi 1994, Smith 2015, Eatough & Smith 2008). Thematic analysis has been used "to refer to a number of different things, including, but not limited to, data analysis techniques in the social sciences" (Terry et al. 2017, p.17). There are three levels of codes recognised in this approach. However, most people begin with a very basic descriptive level of coding and work upwards in a systematic manner towards a more interpretative level. Interpretative phenomenological analysis (IPA) "allows to the researchers to try to understand the innermost deliberation of the lived experiences of research participants" (Alase 2017, p.9), (Pietkiewicz & Smith 2014, p.9).

In a qualitative research analysis, the interview transcript should be transcribed verbatim into a hard copy and then analysed by utilizing color-coding (or any other practical methods) and then categorization for analyses (i.e., to extract common patterns). Miles et al. (2014) stated that "credible and trustworthy analysis requires, and is driven by display that are focused enough to permit a viewing of a full data set in the same location and are arranged systematically to answer the research questions at hand" (Miles et al. 2014, p.90). Most importantly, in a phenomenological research study, Smith (2015) argues that IPA research approach has the ability to explore, in a deep way, the *'lived experiences'* of research participants and help to understand the phenomenological significance of this experience and how it impacts the participant.

This research study follows the Miles et al. (2014) interactive model to guide the thematic and interpretive analysis for the data collected using NVivo. The approach has three concurrent flows of activity: data analysis, data display, and conclusion drawing and verification/validity. Figure 5.1 shows the interactive, cyclical model.

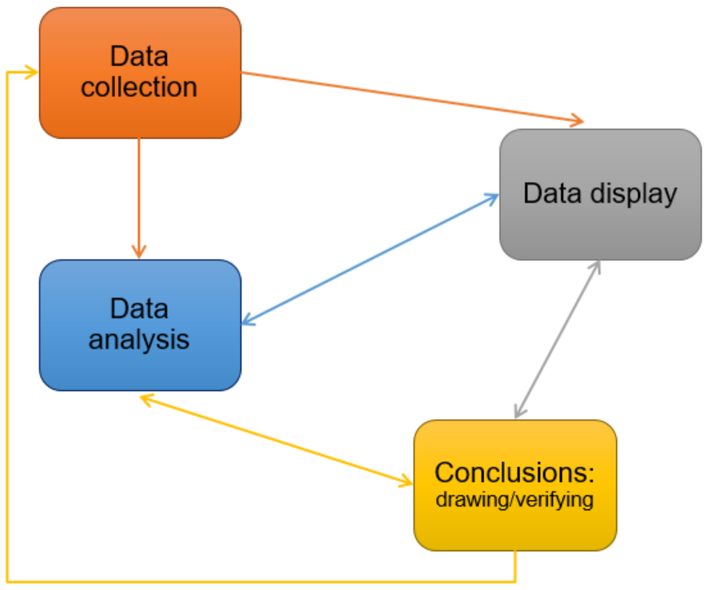


Figure 5.1: Components of data analysis. adapted from (Bazeley 2013, Miles et al. 2014)

Each of these three components continues during and after data collection. Data analysis involves analytic choices based on keyphrases 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 (Bazeley 2013, p.13). The continual validation (depicted in Figure 5.1) is one of our strongest claims to validity. The candidate cycled around the diagram until all parts of the diagram were reconciled, each to the other. The steps to qualitative analysis of interview data are described in the section data collection.

5.2 Data Analysis Process

5.2.1 Data Collection

The aim of this research is to understand how software development companies and their employees evaluate and measure software product quality.

The candidate and supervisory team designed a set of open-ended questions to use in interviews. Participants interacted directly with the candidate 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 candidate's laptop, as well as with the voice memos app on their phone.

5.2.2 Participant Selection

The criteria to select the participants were two-fold: i) they had to be employees of 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" (Goulding 2005, p.302).

Table 5.1 summarises the participants classification. 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. This research may not be generalisable, given that it focuses on young Colombian software development companies, but that the lessons learned could be verified in other developing countries.

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. A total of 20 interviews were conducted.

Table 5.1: Participants classification

Size	Role	Market	Industry
Small	Project manager	Local	Services, financial, agroindustry
		Local/National	Services, automotive, manufacturing, food
	Developer	National	Education, agroindustry, private sector, financial
		National	All types of industries
		Local	Manufacturing
		National	Government, agriculture, aeronautics, telecommunications
	Tester	National	Services and agroindustry
		National	Services and agroindustry
		National	Services, automotive, agroindustry
		Local	Services, automotive, manufacturing, food
		National	Government and private sector
		National	Government, agriculture, aeronautics, telecommunications
		National/Inter.	Energy
Medium	Project manager	Local/Nat./Inter.	Energy
		National/Inter.	Energy
		National/Inter.	All types of industries
	Developer	Local/Nat./Inter.	Energy
		National/Inter.	Energy
		National/Inter.	All types of industries
	Tester	National/Inter.	Energy

Interviews

Each interview included classification questions (company size, role, market, and industry), as described above, and four open-ended questions about software product quality. The open-ended questions were: i) what does software product quality mean for your organization?, ii) what about your everyday work is related to software product quality?, iii) how does your organization evaluate software product quality?, and iv) how do you evaluate software product quality?.

The candidate scanned all transcripts, made notes about his first impressions, and read the transcripts again in detail six times and very carefully, one by one and line by line, to understand the meanings. The language that the candidate conducted all interviews could be a potential problem during the translation Spanish to English, but it was managed by the research team who helped the candidate in the reviewing of each translation without missing any important data. In addition, the candidate created different descriptive tables to include key information such as ID or pseudonyms, answers for each participant, and keywords.

Once the candidate had started with the reading and understanding process of each transcript, he 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 (see Table 5.2).

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.

For this particular study, the candidate spent eleven-months doing the qualitative design, developing the interview, contacting companies, interviewing 20 participants from seven software development companies, collecting and analysing data, during 2018 and 2019.

Some participants replied to these extra questions, which were transcribed Spanish to English and included in the answers to the original questions respectively. The candidate sent two reminders to the other participants, but they did not reply to the emails. He made a decision to keep with their first response. In addition, the candidate discovered some difficulties related to time to meet and employees' availability. The candidate had to cancel some interviews and move some to another time because of company commitments and participants' personal issues. Also, the internet connection was another difficulty because it was not always working properly. However, mobile data helped to fix it.

5.2.3 Thematic Analysis

To begin the analysis, the candidate spent several weeks translating all interviews from Spanish to English, and then performed four iterations reading through each question and response transcripts to understand the data.

Table 5.2: Extra questions to the participants

Participant	**Question**
2	Which quality certifications have your company? | Please, could you give me more details?
4	What stage were they when started the quality evaluation in the organisation? | How old was the company? | What motivated the 'start'?
6	Which other security criteria do you need to measure your organisation?
7	You mentioned in our previous interview that you used cyclomatic complexity measure, could you give me more details? | How and Why did you used this particular measure in in your company?
8	You mentioned in our previous interview that you used 'minimum requirements', but could you give me more details about such minimum requirements? | What measures do you use to evaluate such requirements and security levels and Why?
9	How long have you had CMMI services certification? | How old is the company? | How many developers do you have? | Do you comply with their certification? (more details) | How does your company comply with this certification?
11	What specific measure or measures do you apply when you evaluate functionality, maintainability, and usability in your software products? Please, could you give more details?
12	How long have you had CMMI5 certification? | How old is the company? | How many developers do you have? | Do you comply with their certification? (more details) | How does your company comply with this certification?
17	Why does the organisation go to CMMI3? | How long have you had CMMI3 certification? | How old is the company? | How many developers do you have? | Do you comply with their certification? (more details) | How does your company comply with this certification?
20	How many employees do you have? | How many software developers do you have?

The supervisory team acted as an advisor, participated in essential dialectic discussions several times per week with the candidate, during the interview period and beyond, to clean and organize the data and to clarify the interviewees' responses. They acted as well as an adjudicator and evaluator of the interview data, analysis, and interpretations.

Coding

Coding/labels can be about actions, activities, concepts, differences, opinions, processes, or whatever the researcher thinks is relevant. For instance, the researcher "can decide that something is relevant because it is repeated in several places, it surprises you, the interview explicitly states that it is important, the researchers read about something similar in previously published reports (i.e. scientific papers). The important criteria are that as a researcher you are the interpreter and these phenomena are highlighted because you consider them important. In the coding try to be unbiased, stay close to the data (i.e., the transcripts), and do not hesitate to code plenty of phenomena" (Bazeley 2013, p.15).

The candidate 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 and also annotated the responses, to clarify meaning and their understanding, but they did not change the participant responses or their words in any way.

During this process, the candidate created different descriptive tables to include key information such as *ID* or *pseudonyms*, answers for each participant, and keywords and phrases. Table 5.3 shows some examples.

Table 5.3: Coding Examples

Participant answer	**Edited version**
The company is focusing on process quality. The process to control the logs or bugs is in implementation (because we do not have measures), the company has generated the indicators slowly.	*[We are]* focusing on process quality. The process to control the logs or bugs is in implementation (because we do not have measures), the company has generated the indicators slowly
We are more focused on product safety and maintainability. Quality area is more focused on product functionality and performance.	*[The development area]* is more focused on product safety and maintainability. *[The quality area]* is more focused on product functionality and performance.

Codes, Nodes, and Pattern Creation

For the manual analysis, codes are the keyphrases identified in the last step. Moreover, the candidate created patterns by colour, according to each question and common ideas;

integrating everything (meanings and responses) (see Table 5.4).

Table 5.5 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, the candidate grouped them together into nodes, based on similarity. He then developed patterns based on the nodes. Sometimes, the candidate changed the pattern, or read the transcript sentence again, to further understand it, as suggested in Figure 5.1.

Table 5.4: Pattern and common ideas by colour

Pattern	**Common ideas**
C	Perform product functionality tests until that application works correctly. [I apply] unit tests to ensure that what we are developing is well done. [We perform] smoke tests or manual review of the application. Interfaces must meet usability and user experience standards. We handle some defects categories: functionality, data, form, documentation, and environment.
I	[We consider and evaluate] software bugs/defects, customer perceptions [related to] user interface. [We have] under control, the coding rules [process]. [We use] unit tests for encoding it. [The company has the below metrics inventory]: Cyclomatic complexity | % Reverted changes | Deliveries without critical bugs | Efficiency of peer reviews | Rework cost.

Table 5.5: Definitions

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*” (Vaismoradi et al. 2016, p.104)
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

The candidate’s definition of a theme is derived from (Pietkiewicz & Smith 2014). In that paper, they talk about clustering themes, defined by "looking for connections between emerging themes, grouping them together according to conceptual similarities, and providing each cluster with a descriptive label" (Pietkiewicz & Smith 2014, p.12). In this thesis, the definitions in (Vaismoradi et al. 2016) are being used wherein the word pattern means the same as theme in (Pietkiewicz & Smith 2014). The idea of a cluster theme is referred to as a theme in this thesis.

Tables 5.6, 5.7, and 5.8 list the patterns established by the interview questions. Such questions are *Q5:* What does software product quality mean for your organization?, *Q6:* What about your everyday work is related to software product quality?, *Q7:* How does your organization evaluate software product quality?, and *Q8:* How do you evaluate software product quality?.

Table 5.6: Similarities and connections (1 of 3)

Qs	**Pattern**
5	A: the interviewees with roles such as developer, tester and project leader or manager have different viewpoints about the concept of software product quality. However, the most representative sentences and phrases from the data are related with customer satisfaction and fulfilment of the user requirements helping the companies for generating value to the customers, optimizing operations, and giving confidence to the final user.
	B: project managers agree with the criteria that quality is a key factor for success during a software development project. In addition, it must be integrated with the process for meeting standards, practices, and quality criteria such as LoC, code reuse, and performance.
6	C: the interviewees are developer, tester and project manager who are doing different activities related with their everyday work. Mainly, they are executing manual and automated tests to evaluate software product quality. Such tests are smoke, functional, regression, performance, user interface, and coding.
	D: the interviewees are implementing project management skills for planning, controlling, and organizing the team with good software practices and strategies. In addition, they have defined an engineering process for reviewing how the results came out, which defects are worth identifying, designing test plans, tracking test execution, management estimation, and approving the product with acceptance letters.
	E: software developers and testers interviewed are following processes customized by their companies, and standard guidelines such as CMMI in their everyday work. Such processes are focusing on the software development life cycle from the requirements, design, and development until quality.

Table 5.7: Similarities and connections (2 of 3)

Qs	**Pattern**
6	F: the interviewees pay attention to the customer requirements and support them during the quality verification and validation process. Such a process ensures that the entire customer needs from technical and business requirements are included in the final product.
7	G: the interviewees are developer, tester and project manager who are implementing customized protocols and methodologies for software quality verification and validation. Such customized protocols have some activities related to perform quality software tests (functional, unit, performance, and user interface, etc.), peer reviews evaluations, following checklists and excel formats with different indicators (errors complexity, tolerance percentage, acceptance criteria, vulnerabilities, user complaints/requests, etc.). In addition, some companies are certified in CMMI and they are using some tools such as SONAR for evaluating static code analysis, Jenkins for the integration tests, Buggy for following test plans, test cases and bugs reports, JavaDoc for documenting their code reuse, and Pronox used by project managers to see how their projects progress. Although, the companies are putting their effort on software quality, they are planning to work on DevOps (continuous integration) for coding inspection. H: software developers and testers interviewed are working on different software product quality (SPQ) characteristics such as maintainability, usability, functionality, performance, and portability. However, they consider such characteristics as non-functional requirements and they are evaluating some criteria related to variables such as complexity, user experience, effectiveness, efficiency, customer requirements, and vulnerabilities. Security characteristics are not very common in the SPQ process in the companies because they state that the application's security is a customer responsibility. I: the interviewees are evaluating SPQ in their small and medium enterprises (SME) in Colombia with approximately 50 different measures or indicators (see pattern description in section 6.1.5). J: the interviewees evaluate SPQ according to user requirements and perceptions. In addition, they listen to and receive feedback from the customers and they co-create all the usability criteria with their requirements.

Table 5.8: Similarities and connections (3 of 3)

Qs	Pattern
7	K: the companies are focusing on process quality rather than product quality. Although, some of them have CMMI process and customized methodologies or procedures, they are disorganized with their documentation; and they do not have a standard model or strategy to evaluate SPQ.
8	L: the developer and testers evaluate each product executing some manual and automated tests such as functional, smoke, load, performance, user interface, and unit tests. In addition, they do peer reviews during the development for validating if it is viable to start the product implementation.
	M: the interviewees would like to implement new measures/indicators and invest in training, new automation tools, and resources for evaluating software product quality. Such measures are response time, performance, technical debt, daily functionality, data protection, and database coding. Furthermore, they suggest that companies need developing and applying security policies for data protection, credentials, and other sensitive information.
	N: the interviewees implement existing procedures, indicators, and tools (SONAR, Jenkis) to evaluate SPQ. However, they are not correctly using the tools and they have a gap in automated tests.
	O: some developers, testers and project managers are focusing on customer expectations and they are working with them to guarantee information security, reduce the number of bugs, and increase their reliability.
	P: the companies need to have a coherent, standard, and clear measurement process for ensuring the quality from the beginning. Furthermore, they need to improve in the software documentation and functionalities related to the development cycle.

5.2.4 NVivo Data Analysis Process

The candidate decided to use NVivo for further analysis of participants' data as it allowed faster verification and validation of nodes and patterns. The steps developed during data analysis in NVivo followed the criteria suggested by Miles et al. (2014) and Bazeley (2013). NVivo can help to manage, explore, and find patterns in the data.

The steps developed follow the criteria showed in Figure 5.2. The aim of this cycle is to allow an interpretative analysis to understand the participants' lived experiences (Alase 2017, Pietkiewicz & Smith 2014).

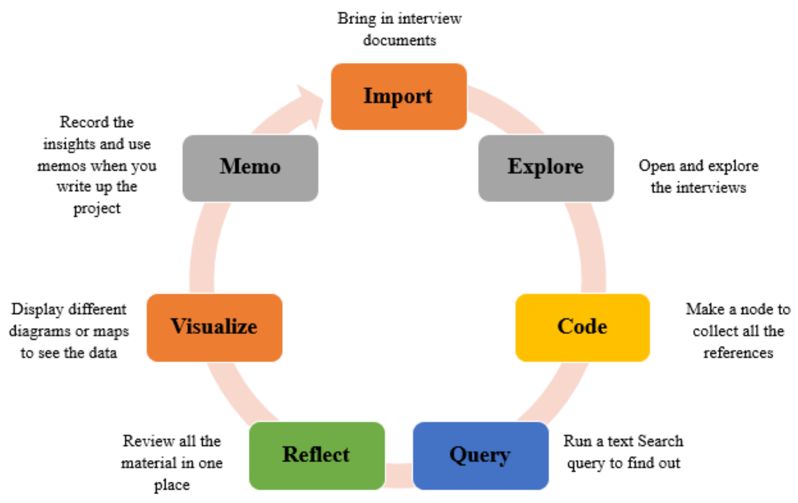

Bring in interview documents
Import
Explore
Open and explore the interviews
Code
Make a node to collect all the references
Query
Run a text Search query to find out
Reflect
Review all the material in one place
Visualize
Display different diagrams or maps to see the data
Memo
Record the insights and use memos when you write up the project

Figure 5.2: Path for exploring data in NVivo (Alase 2017, Pietkiewicz & Smith 2014)

Collect References

In this step, the candidate created cases according to the 20 interviews including a case classification named 'People', which has five attributes: role, company size, location, market, and industry. These come from the classification question of the interviews. These classifications help to understand the interviews by reference especially, to the interviewees' role, and discover and establish the relationships between data.

On the other hand, the candidate created nodes, reading all transcripts again (twice) and relate the description with the keyphrases defined (nodes). Some steps were not repeated in NVivo because those were completed in the manual analysis. The process followed for coding consisted of highlighting the keywords from each interview and dropping off those in each node created according to their meanings and relationships.

Visualizing Results

The candidate customized some diagrams and maps to visualize data and to investigate some associations or connections between patterns and nodes for establishing an overarching theme for this research. For example, Figure 5.3 shows the map *"customer satisfaction"*, which presents all relationships with the main nodes and their keyphrases such are acceptance criteria, generate value, and user satisfied.

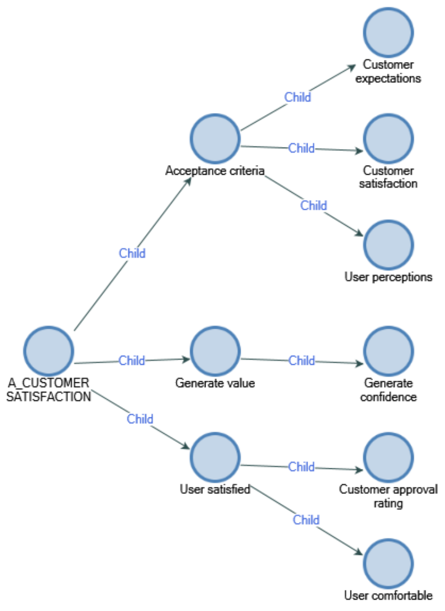


Figure 5.3: Map - Customer satisfaction

Figure 5.4 shows the pattern *"fulfilling customer requirements"* that is related to requirements validation and verification.

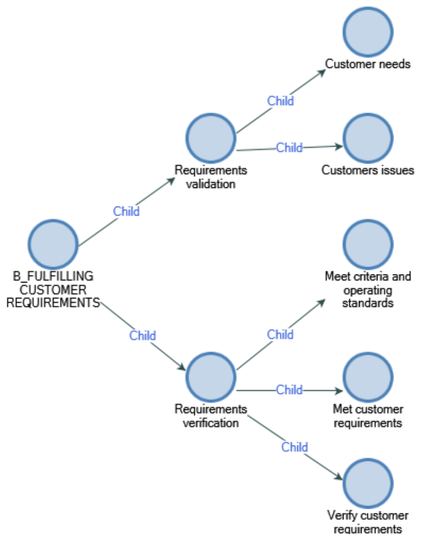

Customer needs
Child
Child
Requirements validation
Customers issues
Child
B_FULFILLING CUSTOMER REQUIREMENTS
Meet criteria and operating standards
Child
Child
Requirements verification
Child
Met customer requirements
Child
Verify customer requirements

Figure 5.4: Map - Fulfilling customer requirements

5.2.5 Resulting Patterns

According to LeCompte (2000) a pattern "is something like the middle stages of assembling a jigsaw puzzle, which involves items related to indicators of a variable/code" (LeCompte 2000, p.147).

After understanding the pattern meaning, the candidate read again all nodes, grouped them together and tried to find the right pattern. Sometimes, the candidate changed the pattern or read again the transcript sentence to understand it again (two or more times). Once he had finished the manual process, he identified that it was difficult to see and understand the connections between codes and patterns as a whole. Thus, the candidate decide to use NVivo software to combine the researchers' analysis and interpretation with an automatic and assisted process that could help to see the connections between all codes, patterns, and questions appropriately. The next section explains data analysis in NVivo.

The candidate initially identified 12 patterns, which were subsequently merged to form 10 patterns due to considerable overlap in shared nodes. Table 6.1 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.

In Chapter 6, the candidate will be describing the findings of this research, the complete patterns, and the relationships discovered.

5.3 Summary

This chapter has presented the second data collection using the qualitative research method including the approach overview and data analysis process performed both a manual analysis and a semi-automatic analysis using NVivo. The candidate identified 12 patterns, which were subsequently merged to from 10 patterns due to considerable overlap in shared nodes. The following Chapter 6 will describe the findings of this research.

Chapter 6

FINDINGS

This chapter presents the initial understanding gained from this research. This phase is focused on analyzing data and connecting everything from the software development companies' lived experiences. The results lead to the conclusion that software process quality is more important to software companies than software product quality. Furthermore, the candidate discusses the findings of the study and its significance to the evaluation of software product quality (SPQ).

6.1 Patterns

The candidate discovered 10 patterns from the codes and nodes marked up from the interviews. Table 6.1 shows these 10 patterns identified from the data collection and data analysis. Each pattern has been given a title, description, and, for validity, the number of references in the data. Patterns D and J are not included in the following list because the candidate integrated both patterns with pattern L and G respectively, according to their similarities.

The candidate summarized how the results of this study applied to seven young software development companies in Colombia and 20 participants in different roles such as developer, tester, and project manager for knowing how they evaluate SPQ and which measures they are applying. In the next sections, the candidate describes each pattern discovered and illustrates this with direct views expressed by research participants.

Table 6.1: List of patterns

Name	Description	Ref.
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
Fulfilling customer requirements	Customer requirements are fulfilled and supported during the requirements verification and validation process. Such a process ensures that the entire customer needs from technical and business requirements are included in the final product.	37
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
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
Measures and indicators evaluated	Colombian young companies are evaluating software quality with approximately 50 different measures and indicators.	36
Process quality over product quality	Companies are focused on process quality rather than product quality. Some companies have customized methodologies, procedures, and CMMI process. However, they are presenting product quality processes which are not well defined.	17
Existing procedures, best practices, guidelines, and policies	Best development practices, guidelines, and policies are implemented for evaluating software quality.	21
Project management skills	Project management skills are implemented for planning and controlling the team with good practices and strategies, defining a dynamic process for reviewing how the results came about, which defects are identified, tracking tests execution, and management estimation.	36
Software product quality characteristics	Software quality characteristics (such as functionality, maintainability, portability, security, and usability) are implemented for assessing product quality.	61
The quality environment is not well	The quality environment related to documentation, quality characteristics and measures, and automated tests are not working well in the young software development companies.	26

6.1.1 Pattern A - Customer Satisfaction

Customer Satisfaction Context

Software product quality (SPQ) is related to each stage of software development projects and needs to be assessed according to the requirements, customer perceptions and feedback for generating value, security, and confidence in each product developed for the final user.

The pattern is important for the customer and the end-user because it helps ensure compliance with the requirements defined from the early stages of product development. Moreover, some criteria that the company can take into account for generating value and providing customer satisfaction may be the confidentiality levels and portability of the application/system.

Customers do know what they want but may not be proficient at describing their needs. The customers have different expectations levels such as expected, normal and exciting (Xu et al. 2009, p.88). Where there is an understanding of these types of customer needs and how to meet them, it is possible to have a high impacts on the customer.

Companies are concerned about their reputation and customer satisfaction, these criteria are critical to the continuity of an organisation in the market and financial stability over time. If a customer receives a product of lower quality, he or she can create a scenario of nonconformity with the community, generating the enterprise loss of value in the market and less customer's reliability in its products and services.

Some studies argue the importance of customer satisfaction for evaluating software product quality (SPQ). Idri et al. state that *"stakeholders' needs and expectations are identified by means of software quality requirements, which have an impact on software product quality"* (Idri et al. 2016, p.1). Garzas et al. argue that *"software development firms are putting greater and greater emphasis on building their software products to a level of quality that allows them to meet the needs of their clients satisfactorily, as they strive to compete adequately in the local and international markets"* (Garzas et al. 2013, p.616).

On the other hand, Febrero et al. explain that *"there are other very important aspects such as customer dissatisfaction and loss of the manufacturer's prestige that can be traced to software product issues"* (Febrero et al. 2016, p.18).

Customer Satisfaction Pattern Description

Customer satisfaction is achieved for companies by fulfilling the acceptance criteria, generating value to the customers, and giving satisfaction to the final user. Figure 6.1 shows the relationships between nodes and keyphrases for discovering this pattern.

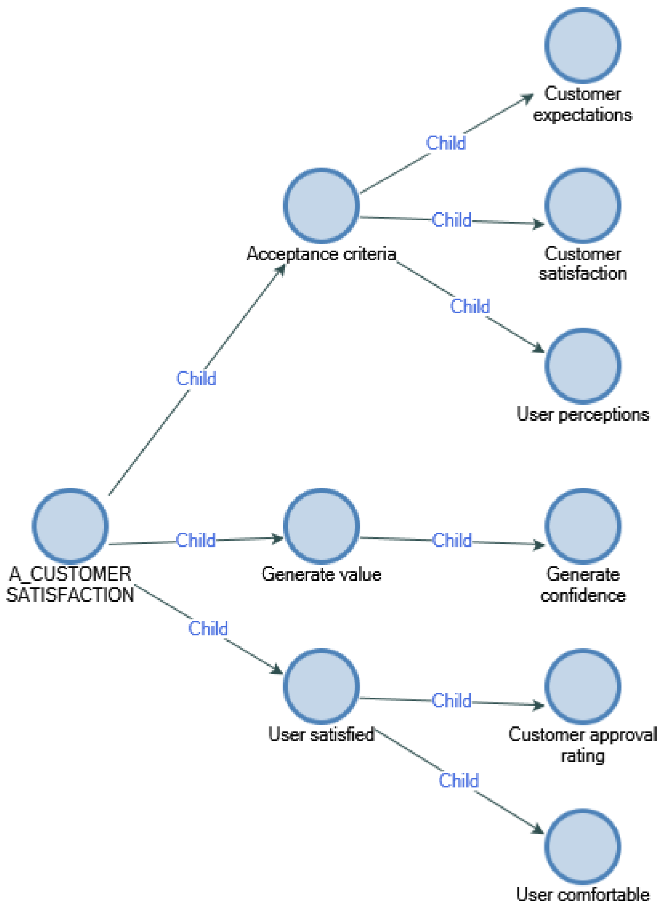


Figure 6.1: Customer satisfaction

For twelve out of the twenty participants, their work as software developers, testers or project managers related to software quality were focused mainly on customer satisfaction and perception. In the interviews, the data revealed that the twelve participants were all seeking ways for relating software product quality with usability, confidence, generating value, acceptance criteria, customer feedback, customer rating, and user comfort.

'Dan' suggested that software product quality *"means that the product that we release* *for a customer complies with the functional and non-functional requirements. In addition, it complies with the customer acceptance criteria, which were defined during the project start"*. However, some other participants are mostly focused on customer satisfaction:

- *"The product quality means that the customer is satisfied with the final product"* (Phillip)
- *"The product quality is related to compliance with the requirements and customer satisfaction"* (Josh)
- *"Oriented to the customer: that the need that arose is fulfilled at the functional level. For example, the market demands that we generate a report that adds A + B, so that we comply with this requirement in every sense (sum, volume, performance, response times and others)"* (Maria)
- *"The quality, from our concept, is that where our user feels satisfied or uses the product without any problem, this product meets customer requirements, regardless of whether it is beautiful, but it satisfies their requirements"* (James)

Moreover, some participants related SPQ with customer expectations, perceptions, and feedback as the following software developers, testers, and project managers described:

- *"If we have a quality product and we fulfil the client's expectations, we can be competitive"* (Kate)
- *"Now we are very interested in customer perceptions. For example, in Mexico, users use our products a lot and due to the usage level, we are very interested in measuring usability since it is fundamental, but we must migrate to the user experience issue"*. Furthermore, *"The issue of our quality goes much further, it is for generating value to the customer"* (Stiven)
- Although 'Kyra' describes that *"the customer is who gives the feedback on whether the product meets the agreed upon requirements"*, 'Nick' argues that *"the rating that the client gives us is a fundamental part of the quality"*
- A project manager argues that i) *"they evaluate the customer observations associated with the fact of the user interface perception to determine usability criteria before going to the field"* and ii) *"if the product meets customer expectations, then the product has the necessary quality to enter in the niche market identified"* (James)

'Andrew' and 'Kyra' argue that the customer needs to feel comfortable in quality terms:

- *"The company makes a very standardized quality evaluation process. The matter of having the customer service area also helps us identify errors and solve the customer's problems and needs. Also, customers feel comfortable with the products quality"* (Andrew)
- *"Quality is to fulfil the requirements that the customer needs, but also give to the client a benefit in usability terms, that the user feels comfortable in a tool/software"* (Kyra)

'Anna', a tester, and 'Julien', a developer, who believe that security and confidence are relevant quality aspects, illustrate another view:

- *"The quality goes beyond the functionality, it is to deliver something that continues generating confidence to the customer to continue with the company"* (Anna)
- *"Some added values are made to give security and confidence to the customer that the information that is being downloaded is real"* (Julien)

In summary, these extracts demonstrate that the participants, from diverse areas, are concerned about customer satisfaction throughout the software engineering life-cycle. They refer to SPQ attributes, even if they only talk about software quality, not software product quality.

The other eight interviewees made no comments on customer satisfaction.

6.1.2 Pattern B - Fulfilling Customers' Functional Requirements

Context for Fulfilling Customers' Functional Requirements

Some statistical studies that are published in the report of Standish Group Inc., identify causes of projects failure associated with: i) inadequate implementation of good practices in project management, ii) poor definition of requirements, iii) the lack of definition of techniques to guarantee the process quality and software product and; iv) lack of risk management (Gasca et al. 2013, p.2), (StandishGroup 2015).

Rodriguez et al. argue that there are greater demands placed on the software that is being produced to ensure that it meets the necessary requirements. The use of software in spheres that are very sensitive and where mistakes are especially serious, such as healthcare, banking, and security, has meant that there are more users willing to accept that costs may rise if that means there is a consequently higher level of software quality (Rodriguez et al. 2016, p.18). Meanwhile Heck et al. state that certification could help prevent poor quality of the requirements (incomplete and changing requirements), the primary reason why so many projects continue to fail. Moreover, it can help outsourcing partners and stakeholders, to convince the other party that deliverables are acceptable (Heck et al. 2010, p.38).

On the other hand, Yahaya et al. mention that companies and software houses are competing to produce software which is claimed to be good and fulfil user's expectation. Therefore, the quality aspect of a software product has seen as an important issue but companies that develop the software could not justify and guarantee the quality of their products, thus leaving users in uncertainties (Yahaya et al. 2008, p.1). Such fulfilling customer expectations or requirements are due to the end-users demands and the increase of a greater number of requirements in the industry. Thus, the quality needs to be immersed and implemented at the different development levels and it is necessary to count with new evaluation methods that allow knowing the answers given form the user opinions (Salcedo & Gil 2012, p.92).

Idri et al. (2016) present a framework that helps developers and evaluators to consider that the design of the checklist is the main key as regards deducing the impact of the requirements on software product quality. Therefore, they present a set of requirements for mobile personal health records (mPHRs) for pregnancy monitoring and suggest the requirements that should be considered during the quality evaluation of these mPHRs. In addition, they calculate the impact of the requirements on software product quality using ISO 25000 standard.

In Colombia, many solutions proposed by young software development companies do not meet the minimum customer requirements, much less allow them to compete at the level of software quality in international markets, generating various issues, which can lead not only to great investment costs but also to maintenance, administration, adaptation, and repairing implemented software solutions (Pelaez et al. 2011).

For mitigating such problems, Ecopetrol an oil and gas big company in Colombia has defined a quality model for supporting the definition of baselines for software quality helping to achieve the goal of standardization of quality requirements in their developed products. They developed this model to improve software quality, reduce the issues caused by unknown/probably poor software quality and customer requirements misunderstood, and in turn contribute to the ability to develop and maintain software faster and support the company in becoming more agile (Lampasona et al. 2012).

Pattern Description for Fulfilling Customers' Functional Requirements

Customer requirements are fulfilled and supported during the customer testing process. Such a process ensures that the entire customer needs from technical and business requirements are included in the final product (see Figure 6.2).

For fourteen of the twenty participants, a criterion for working in a software development environment is fulfilling customer functional and non-functional requirements in their roles as manager, developer or tester. In the interviews, the data showed that they were all implementing requirements verification and validation to ensure that the entire client needs from technical and business requirements are included in the final product with the highest quality.

Some participants were focused on requirements validation, which is related to customer needs and issues:

- *"To validate that the product which is delivered meets with what is defined and expected by the customer"* (Stiven)
- *"Quality must meet the customer needs, must satisfy each activity, each process that they require to finalize a product"* (Liz)

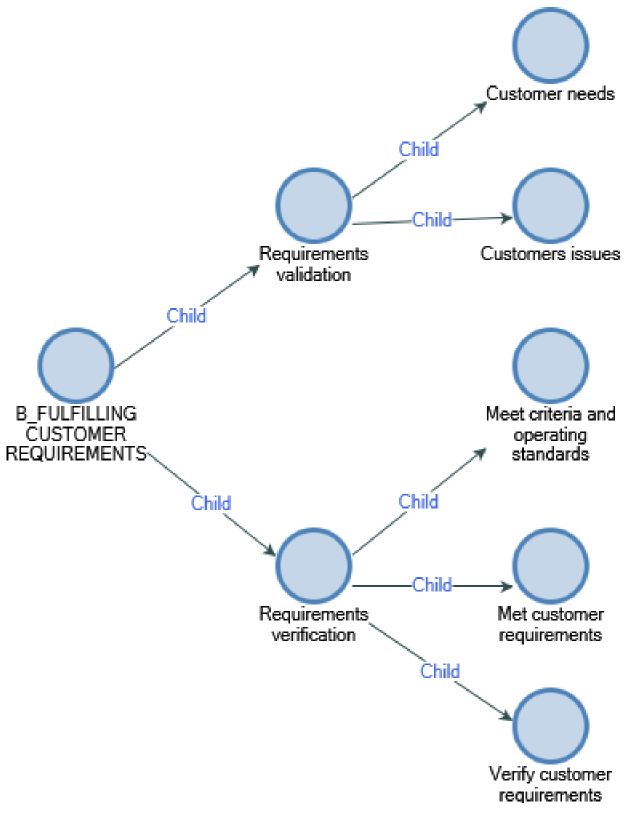

Customer needs
Child
Child
Customers issues
Requirements validation
Child
B_FULFILLING CUSTOMER REQUIREMENTS
Meet criteria and operating standards
Child
Child
Requirements verification
Child
Met customer requirements
Child
Verify customer requirements

Figure 6.2: Fulfilling customer functional requirements

- *"Product quality is part of the company's mission, which is providing services to the customer to guarantee good products. In addition, offering to the customers' products that meet their needs"* (Angel)
- *"Product quality is related to compliance with the requirements"* (Josh)
- *"Quality is to fulfil the requirements that the customer needs, but also give to the client a benefit in usability terms, that the user feels comfortable in a tool/software. The objective of our process is to ensure that the application we develop comply with the customer's needs"* (Kyra)
- *"We handle very high-quality standards. By orders of the company CEO, we have an area that depends directly on him, which is the customer service area. We listen to the customers, we listen to the inconveniences they have, and we respond to each one of the customer's needs or inconvenient"* (Andrew)
- *"We define user requirements from the beginning and define some aspects through*

co-creation for defining how the application will behave according to the customer's needs" (Lachlan)

Other interviewees were concerned about requirements verification, which is an understanding of the customer requirements and achieving technical and business requirements:

- *"Quality, from our concept, is that where our user feels satisfied or uses the product without generating any inconvenient, that product that meets the customer requirements, regardless of whether it is beautiful, but the product satisfies their requirements"* (James)
- *"Achieve customer needs and requirements. In addition, meeting the project scope considering the entire technical part"* and *"an understanding and verification that the requirements are actually those that the customer needs, from the inception until we deliver the product"* (Karol)
- *"Product quality is to guarantee that product work efficiently and effectively, that they meet criteria and operating standards. In addition, the products meet the requirements that the user wants"* (Sarah)
- *"From the requirements stage, we ensure that the requirements established with the customer are clear. We handle standard formats, accompanying experts in meeting requirements (to reduce the information duality and have greater requirements clarity)"* (Angel)
- *"Product quality is that the customer is satisfied with the final product. We have the testing area, which is responsible for reviewing the customers' requirements. Use cases/user stories are generated. The documents are signed by the customer and the company, so that they can be the input for the development area. With this same document, the testing area develops the test designs and validates that all customer requirements are met"* (Phillip)
- *"Product that we release for a customer complies with the functional and non-functional requirements. In addition, that it complies with the customer acceptance criteria, which were defined during the project start"* (Dan)
- *"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 requirements defined by the customer and the process defined by the quality area" (George)

- *"We may not meet 100% of all criteria in a good quality development, but we have selected some that should be met according to the application context, and the customer needs"* (Maria)

To summarise, the aforementioned data demonstrates that these interviewees are concerned about fulfilling customer requirements during all stages of the software engineering life-cycle and that the quality is directly related to this pattern. Such verification and validation process are correlated with the customer needs/issues and achieving client requirements. The other six interviewees made no comments on fulfilling customers' functional requirements.

The absence of any mention on non-functional requirements suggests that the focus is on functional requirements only. For most companies, it reinforces the lack of focus on software product quality.

6.1.3 Pattern C - Customized Protocols and Methodologies

Customized Protocols and Methodologies Context

Small and medium enterprises/companies (SMEs) need efficient software engineering practices that are suitable for their particular context/environment (Pino et al. 2008). These practices would support the development of products of high quality which must evolve if they are to adapt to new demands and scenarios as they seek to make these companies become more competitive. Thus, companies could create practice communities to share their models, protocols, procedures, frameworks, and methodologies, which they have built or they are building for assessing software product quality (SPQ) with experts, academia, and industry. If the companies share their knowledge, SMEs probably could compete and be recognized worldwide.

Academia and industry could provide strategies or projects focusing on SPQ evaluation in a standard model/framework that allows the creation and measurement a software product in a common environment, but with essential and flexible characteristics, measures, protocols, and procedures (Metaute & Serna 2016, Rodriguez et al. 2016).

Some studies have developed models or methodologies according to the companies' needs and contexts for evaluating software product/process quality (Idri et al. 2016, Lampasona et al. 2012, Metaute & Serna 2016, Pelaez et al. 2011). According to Salcedo & Gil (2012) the end-users demands and the increase of a greater number of requirements compared to software products, it is necessary that the quality will be immersed and implemented at the different development process levels. Therefore, it is necessary to have new evaluation methods that allow for knowing the answers given from the users' opinions in needs and requirements terms.

Idri et al. (2016) present a framework for evaluating software product quality in a pregnancy monitoring mobile personal health records (mPHRs) for pregnancy monitoring and suggest the requirements that should be considered during the quality evaluation. In this specific case, they customize a framework to calculate the impact of the requirements on SPQ using ISO/IEC 25000 standard helping developers and testers to consider that the design of the checklist is the main key as regards deducing the impact of the requirements on SPQ.

Meanwhile, Pelaez et al. (2011) state that Colombian companies have implemented ITMark and Light MECPDS customized approaches as the most recommended practices for improving the software quality process.

In other study conducted by Metaute & Serna (2016), the authors 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. They found that some companies use methods as CMMI, ISO/IEC 9000, ITIL, and customized methods to evaluate the software product quality.

Lampasona et al. (2012) present experiences in developing custom-tailored quality models for Ecopetrol, a Colombian oil and gas company. They describe the creation of the quality model to align the incorporation, acquisition, and development of IT solutions with the business goals. The quality model supports the definition of baselines for software quality at Ecopetrol and it is part of its IT landscape. It helps Ecopetrol define policies to be followed by the organization in order to achieve the goal of standardization of quality requirements.

It is important to take into account that the software quality concept is often hard to capture for an organization. Quality models help to understand the concept in a more practical way by defining the product/process quality into sub-concepts down to the levels such as characteristics, measures, protocols, and procedures (Kumar et al. 2015, Lampasona et al. 2012, Rodriguez et al. 2016, Pelaez et al. 2011). In practice, *"it is difficult for an organization to come up with a reliable quality model because the quality depends on numerous organizational context factors, and the model, as well as the metrics and indicators, need to be tailored to the specifics of the organization"* (Lampasona et al. 2012, p.243).

Pattern Description for Customized Protocols and Methodologies

Protocols and methodologies are customized for software quality verification and validation. Such customized protocols have some activities related to performing engineering process, expert judgment/peer reviews, implementing checklists for reviewing, code strategies and business rules, continuous deployments, testing process, and indicators using some tools [such as Buggy, JavaDoc, Jenkins, Redmine, OTRS, and Sonar]. Figure 6.3 shows the nodes and keyphrases relationships.

For nineteen out of the twenty participants, customizing protocols and methodologies were fundamental for evaluating software quality in their companies. In the interviews, the data reveal that they were all applying diverse procedures related to checklists for reviewing, coding strategies and business rules, continuous deployments (DevOps), engineering processes, peer reviews, testing process and tools for tracking the quality. The other interviewee made no comments on customized protocols and methodologies.

Some participants are implementing checklists for reviewing usability criteria, design, architecture, coding, integration, acceptance, and technical issues (performance and completeness) according to their companies' guidelines and a standard process for ensuring and controlling the quality:

- *"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, in technical issues, criteria such as performance and the completeness of the entire requirements from all viewpoints. Checklists are made for criteria such as design, architecture, coding, unit tests, integration tests, and acceptance tests"*. Furthermore, *"we have seen that quality reprocessing becomes very expensive, so if we do not try to ensure quality from the beginning, we can lose control of quality"* (Karol)
- *"We handle standard formats, accompanying experts in meeting requirements (to reduce the information duality and have greater requirements clarity)"* (Angel)

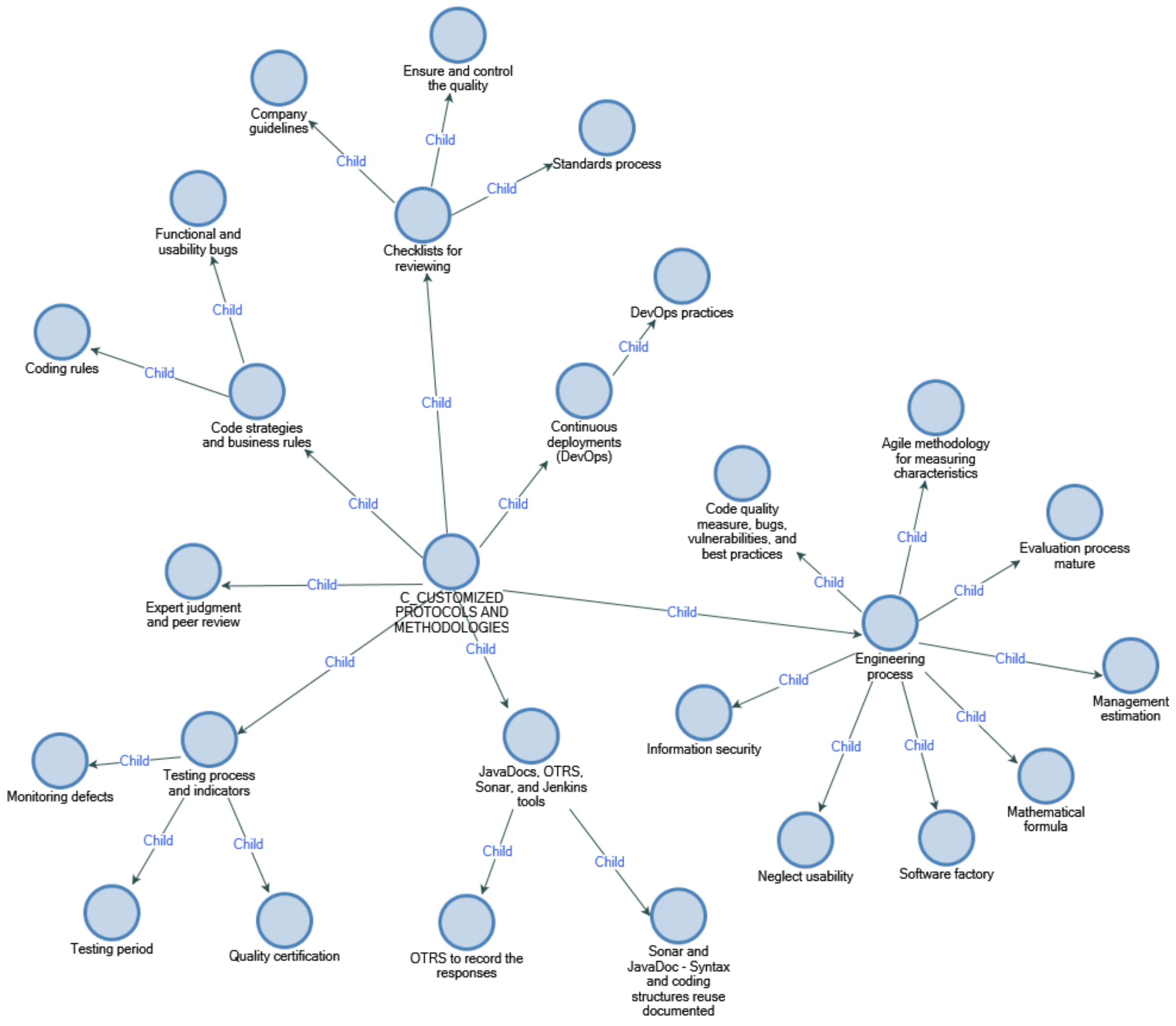

C_CUSTOMIZED PROTOCOLS AND METHODOLOGIES
Child
Checklists for reviewing
Company guidelines
Ensure and control the quality
Standards process
Code strategies and business rules
Functional and usability bugs
Coding rules
Continuous deployments (DevOps)
DevOps practices
Expert judgment and peer review
Testing process and indicators
Monitoring defects
Testing period
Quality certification
JavaDocs, OTRS, Sonar, and Jenkins tools
OTRS to record the responses
Sonar and JavaDoc - Syntax and coding structures reuse documented
Engineering process
Code quality measure, bugs, vulnerabilities, and best practices
Agile methodology for measuring characteristics
Evaluation process mature
Management estimation
Mathematical formula
Software factory
Neglect usability
Information security

Figure 6.3: Customized protocols and methodologies

- *"Regarding the customer data security, data and personal information of the company. In addition, in database terms, we handle SQL injection (for controlling that the data records in the database have not malicious information). In the registers we encrypt passwords, important data, the requests always have security criteria to avoid information theft"* (Liz)

- *"There are several levels of support (level 1, 2, and 3). Level 1: They are supports related to doubts and user interface. Here, we offer a guided support for the user. Level 2: Queries and reports generation that maybe the software does not offer because they were not contemplated in the requirements. This does not impact the code, they are queries in databases. Level 3: When the customer needs to modify or* *improve something at the code level. Once the adjustment is made, it goes back to the quality area for the respective validation and verification"* (Kyra)

- *"I evaluate the software quality in accordance with the guidelines established by company. Personally, I think that the process that has been established has generated good practices and results with customers"* (Phillip)
- *"I am currently developing a project to improve some functionalities and process. For instance, when the previous developers started working on the project, the customer requested a new functionality as 'form for three user types'. At that time, developers make a functionality copy of an existing registration form, it generates issues with the code duplication. When I started to review this, the maintenance was quite difficult. The strategy that I am implementing is to unify the forms with different business rules"* (Dan)
- *"What we do from our quality area is to evolve the product, we have certain functionalities that are established in the documents as a checklist and architecture guidelines"* (Maria)

Other interviewees are implementing coding strategies such as coding rules, and functional and usability bugs to unify the forms with different business rules:

- *"In the company, we are still organizing the code documentation. For example, there are certain standards to generate error alerts in the source code and we apply comments in the code"*. Moreover, *"we implement maintainability with a measure as static code analysis, using Sonar tool, integrated with Jenkins (integration test), so every time the project is reviewed"* (Nick)
- *"There is a quality area, we executed a project, this is reviewed by a quality analyst, through a 'Buggy' tool, here they report and solve the bugs"* (Angel)
- *"We have in our process, bugs records reported by the customer. For example, in the quality area, we found ten errors, but the customer identified only one or we found ten, but the client twenty. Then, something is wrong, and we must review the process for the bugs' identification. This is how we measure the quality of applications"* (Kyra)
- *"When I started to review this, the maintenance was quite difficult. The strategy that I am implementing is to unify the forms with different business rules"* (Dan)

- *"When we start with a project, we define a test plan with scope and types of tests that support the project. The types of tests are defined according to the project that is being developed. For instance, we develop communication protocols about the meters we support"* (George)
- *"As for coding rules they are totally under control. Exactly to ensure that the rules are met efficiently, it is that the technical review area arose (interdisciplinary area)"*, In addition, *"a lot of effort is being made in that the code, which is developed is also encoded in the unit tests, this being a relevant aspect for the software quality assurance"* (Julien)
- *"In quality we focus on quality of developed code. Here, we consider the best option to develop"* (Maria)

Some developers and project managers are doing continuous deployments (DevOps) as an internal strategy for creating an integration sequence, complementing the measurement process, and inspecting the code:

- *"We assembled some automatic tools (Development Operations-DevOps), which we have not finished implementing. The tool assesses 5 variables (maintainability, cyclomatic complexity, depth of inheritance, the coupling of classes, and LoC). DevOps tool complements the measurement process"*. In addition, *"DevOps create an integration sequence from the development to the deployments, making the integrations and adding different controls"* (Karol)
- *"Now, with the Team Services implementation, we have managed to bring a lot of continuous deployment. Here within the routines shows me since I make a 'commit' until the solution is deployed in production"* (Andrew)
- *"The continuous measurement during the project is done through the practices that are framed in DevOps (continuous integration), which refer to the continuous inspection of the code. For example, each time a developer uploads a piece of code, a sequence of automatic processes is executed, and the state of the code elements analysed. Here, we evaluated several indicators, including technical debt (which is the current state of the code based on a set of coding rules)"* (Lachlan)

Some participants recognised the importance of DevOps, but they are not happy at all:

- *"I consider that we do not use correctly the tools. For instance, continuous integration criteria, DevOps criteria, I think the company is very backward. When we neglect* *the process deliveries start to fail, reprocesses in the evaluation are presented, tests or quality, because manuals must be made and there is no agility in the process"* (Sarah)

- *"For a product to be of good quality, the process must be organized. I feel that even the company is disorganized in this regard. The idea is to work on the DevOps strategy internally. We still need to standardize processes and be strict in certain aspects of continuous integration and deployments"* (Nick)

Meanwhile, some project managers, developers, and testers argue that they carry out the defined engineering process by the company that includes criteria related to management estimation, evaluation process, code quality measures, bugs, and vulnerabilities, software factory criteria, and agile methodologies for measuring characteristics:

- *"The organization has defined some engineering processes; including the assurance processes. We have a software factory model where we have the service assurance (we have CMMI Services certification). The engineering processes incorporate quality assurance practices (quality tests). The responsibility of the project manager role is to carry out the defined engineering processes including the quality criteria"* (Stiven)
- *"A lot of attention is given to usability, but in things that the user gives not opinions, sometimes we tend to neglect it to finish very fast"* (Sarah)
- *"I offer support to the teams: solving doubts, validating that the deliverables that we make comply with the process standards that we have defined. In addition, we are making estimations and analysis of the test's times"*. In addition, *"in the application that we handle all the errors are counted and with the mathematical formulas, it also takes the implementation consumed times in the error's correction. All this is linked to the time record we use"* (Kyra)
- *"In the context that the development area manages, we perform an architecture to guarantee security in the information"* (Dan)
- *"We realise the estimation management of the execution time allocated, the documentation review, requirements, and that we have test environments. This is the initial planning"* (Naty)
- *"The company works as a software factory and we work with specific areas. I am in charge of the factory area that is responsible for software quality; we perform requirements, development, and quality"* (Maria)

- *"For maintainability and usability, we have an area responsible for these criteria. We carry out daily follow-ups and work with agile methodologies. On the other hand, for errors correction, we use tools as SONAR, which measures how the code quality is; from the innovation area are loaded some rules that are being updated and always the applications measurement is made through this application"* (Andrew)
- *"All measurement criteria are carried out manually and automatically depending on the criteria. Our evaluation process is mature enough compared to the industry. We have found cases where customers or other providers do not use this type of technique to assess the quality on software projects"* (Lachlan)

Other group of developers and testers state that they do expert judgment/peer review process, using checklists in different levels (specification and evidence):

- *"A list of what the system must do is given to the test analyst and from this, the analyst reviews the product functionality"* (Jen)
- *"Peer evaluations are handled by the quality team. We must evaluate the work developed by another developer"* (Dan)
- *"I do a peer review to these designs with the objective that they are completed and comply with what is really to be expected to prove in the system"*. Furthermore, the company *"at the specification level are doing peer reviews and involving the project team more in the project contextualization and at the level of evidence, we are trying to involve a person who does the peer review of the test cases, in such way that we contemplate all the possible scenarios and that we encompass what we want to evaluate"* (Naty)
- *"Recently as one of the quality assurance criteria is to do a peer review and verify that certain conditions of a review that is made in SONARQ are met"*. Moreover, the reported findings *"are done through manual tests, and we have a part oriented to expert judgment (peer review). In this peer review the checklist is delivered, it is verified that the development has been built based on an approved design, which has applied all the recommendations based on the architectural guidelines"* (Maria)
- *"We used the peer review (another developer) to identify errors in the application's encoding"* (Andrew)

In addition, some participants manage SONAR, OTRS, and Jenkins tool for measuring the solution integration degree recording the customer responses:

- *"We have the 'OTRS' tool, with this tool we can identify the incidents and propose strategies to mitigate the errors presented. As we analyse this information statistically, we can know what the errors that draw the user attention are"* (Kate)
- *"Through OTRS (where the requirements are recorded), the response of each measure is returned with the respective annotations of what was found during the review. An internal table is used to consider the entire measurement process"* (Jen)
- *"Regarding to the assets reuse (syntax, coding structures), we implement and execute them with the development tools (Java, C #). In addition, we documented the source code reuse through 'JavaDoc'. However, we leave comments in the code and generate a document that more eloquently reflects the comments and how we can reuse parts of the code"* (Dan)
- *"We have several tools to measure development quality as SONAR. In addition, we have a great partnership with Microsoft and have different ongoing training with them (we are Gold Partner)"* (Andrew)
- *"The tools are available so that the team can access to the project information and identify what are the errors in quality matters. SONAR is mainly for maintainability aspects, Jenkins to measure the integration degree of the solution, and then other tools for registration such as Mantis and Visual Studio Team Services"* (Lachlan)

Finally, they are following the development and testing process with some indicators for monitoring defects and achieving the quality certification:

- *"The activities are aimed at monitoring the observations made by customers through the technical support area and development, regarding the software operation. We are developing statistics about user behaviour and customer requests totals. The company is focused on evaluating the quality from the customer perceptions but to maintainability level. The company into the development process established some protocols, procedures and development stages during which we validate the product development once it is running"* (James)
- *"On a day-to-day basis, we evaluate how we do the process and, if it is necessary, make adjustments according to the experience and company's needs. In summary, my everyday activities are focused on organizing the team and making the quality process dynamic. We contemplated a testing period in the planning stages because we had many software development projects"*. It is developed because *"customers*

usually found errors that could easily have been detected during the testing periods by the Quality area". On the other hand, *"I like to run product pilot tests (with a partner company). In my experience in Colombia, this process is very complicated to make it understood. For instance, for applying tests are planned two weeks (just to a pilot test) and always the companies say that why so long if we can do quality reviews during the software development"* (Anna)

- *"Regarding the customer data security, data and personal information of the company. In addition, in database terms, we handle SQL injection (for controlling that the data records in the database have not malicious information)"* (Liz)
- *"We identify different errors, but it always happens when the developer has less experience. Therefore, we train developers and analysts to gain skills and each of them are responsible for reviewing and testing the software through the unit tests"* (Karol)
- *"In the quality stage (QA), the objective is that the delivery of any project is not made, without a product quality certification. This certification ensures that all errors founded are corrected, and that effectively it meets the customer's requirements"* (Angel)
- *"Within the organization, we have indicators. We follow the development and testing process"* (Kyra)
- *"We analyse and monitor defects. In 'Redmine' all project defects and the requirements description agreed with the customers are reported. Each requirement has its design task, development and testing. Furthermore, each defect is reported and automatically alerts are generated about the defects that were found, each responsible area adjusts and validates each defect"* (Phillip)
- *"We handle some defects categories: functional defects, design defects, documentation mistakes, if it is data defects, etc., they are of five types (functionality, data, form, documentation, and environment)"* (Naty)

To summarise, from the results embedded in the excerpts above:

- Companies implement checklists for review according to their companies' guidelines;
- Companies implement coding strategies such as coding rules;

- Companies are doing continuous deployment (DevOps) as an internal strategy;
- Companies carry out company defined engineering processes;
- Companies develop expert judgment/peer-review process, using checklists in specification and evidence levels;
- Companies manage some tools for measuring the solution integration degree recording the customer responses;
- Companies are following the development and testing process with some indicators for monitoring defects and achieving the quality certification

These results strongly validate that customizing protocols and methodologies is a way that software development companies have for evaluating the process or product quality in their projects. It is an essential practice and we need to provide some strategies or projects for focusing software product quality evaluation in a standard protocol, model or framework, which allows creating the same environment, but with flexible measures and procedures.

6.1.4 Pattern E - Manual and Automation Tests

Manual and Automation Tests Context

Even though the functionality field is receiving attention, companies need to pay attention to the quality and security of the software products (Mellado et al. 2010, p.4).

Companies need to pay attention to SPQ (notably security) of the software products. In addition, it is crucial that the industry invest more resources for solving the software product quality (SPQ) issues in the markets and developing automation testing to assess the quality in software development projects. Thereby, all companies can use their experiences and best practices to move from manual testing to automated testing. On the other hand, companies can implement software development practices and information-technology operations (DevOps) to shorten the system-development life cycle while delivering SPQ evaluation protocols with characteristics and measures that help to alignment with the business goals.

Currently, companies have different ways to evaluate software quality; applying manual and automated tests as an ideal testing option. However, some studies are focused more on the integration and unit test as Kumar et al. (2015) explain. The authors developed a case study that defined the top five factors that impact the quality of the enterprise. Such factors are Designing New Business Processes, Existing IT Infrastructure, and Skills of the team, Importance of Integration Testing, and Unit Test and Test Coverage. However, they state that is important to establish a framework that will provide practitioners with the set of preventive and corrective actions for these quality factors which help in improving the software quality (Kumar et al. 2015, p.5).

Metaute & Serna (2016) in their research results show information about metrics application time, the application of metrics in different software engineering stages, access to certifications in metrics, object-oriented programming application, type of methodologies used for software development, methods for the metrics application, automated tools for the metrics application, programming processes where metrics are applied, and contributions generated and types of tests where metrics are applied (Metaute & Serna 2016, p.12). Here, it is interesting to see how testing is important for applying metrics or measures in software development projects, but focusing more on automated tools.

In general, Mellado et al. (2010) state that there is little support that has been focused on the product quality and security of software products. Although the testing area is strongly working on functionality, they have still not developed techniques for evaluating in an effective way the quality and security of a software product (Mellado et al. 2010, p.1). In contrary to Mellado's conclusion, in the companies interviewed, there was good coverage of software process quality. Security is an SPQ, and some of the interviewed companies were concerned that security was not well covered.

With the aforementioned studies, the candidate can support that manual and automation tests can be a strategy to assess software product quality for the SMEs in the Colombian study.

Manual and Automation Tests Pattern Description

Amongst those interviewed, manual and automation tests are executed for evaluating software quality. Such tests are smoke, functional regression, performance, user interface (UI), and coding according to the context and company as is showed in Figure 6.4.

For sixteen of twenty participants, their work is focused on applying manual and au-

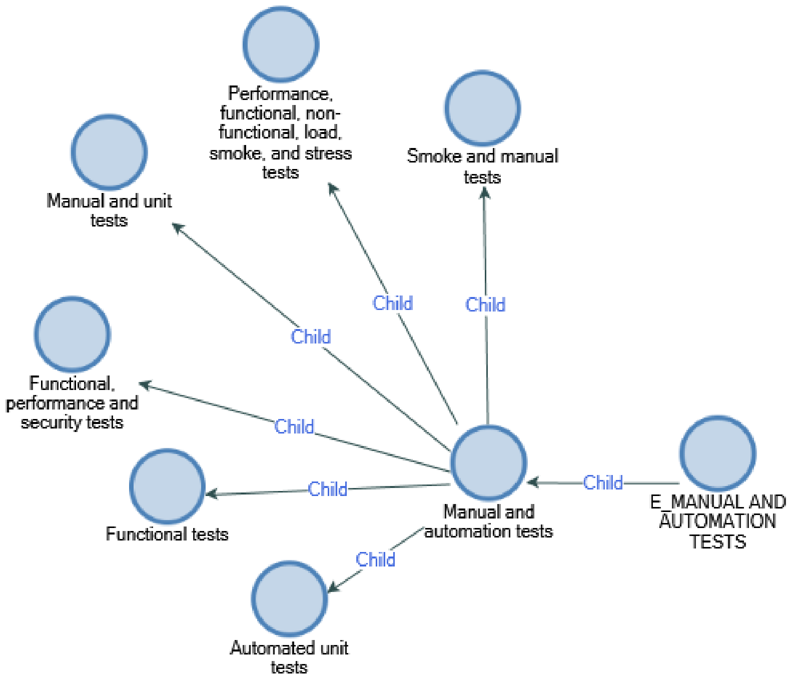

Performance, functional, non-functional, load, smoke, and stress tests
Smoke and manual tests
Manual and unit tests
Functional, performance and security tests
Functional tests
Manual and automation tests
E_MANUAL AND AUTOMATION TESTS
Automated unit tests
Child

Figure 6.4: Manual and automation tests

tomation tests during their quality evaluation process related to manual test, automated unit tests, functional tests, performance and security tests, smoke and stress tests, and non-functional tests. The other four interviewees made no comments on manual and automation tests.

Some participants are focusing on automated unit tests to reduce incidents that reach the development and quality area. Moreover, they run automatic deployments according to test cases established by the company. However, they state the following aspects:

- *"From the development area, there were some unit and performance tests were defined by technical leaders from the architecture area. They established formats and test cases according to the use cases and the analysts must comply with a part of unit tests. There are still quality analysts who perform more specific in detail tests"* (Sarah)
- *"I perform manual tests, previously I worked with different tools for automatic deployments, but it turns out that they were not so automatic, so we stopped working with this"* (Andrew)
- *"In Company Z history, they had a testing team. The first thing that they did were to*

develop all the requirements, design, coding and then test team entered and carried out the test cases and executed them. With this method, the test team was left with a lot of responsibility and it was a very late stage to return because it generated a lot of rework. Then, developers received training in test design and execution for strengthening this gap. Therefore, the schema was changed so that the functional analysts could carry out all the tests. This schema is the one that has been used up to now and has generated good results until the integration process" (Karol)

- *"At the development level, automated unit tests are being implemented to reduce incidents that reach the quality area (QA). Furthermore, they are conducting a peer-review process. At the level of evidence, we are trying to involve a person who does the peer review of the test cases, in such a way that we contemplate all the possible scenarios and that we encompass what we want to evaluate. In addition, we carry out a 'smoke test' to identify if it is viable to start the execution"* (Naty)

Most quality analysts are performing functional tests until applications work properly. They generate a baseline for running the test cases and verify the customer requirements against these:

- *"The company gives me the application working and I perform product functionality tests. If an error happens, a report is delivered, development area fixes the issue, I test it again and the entire process runs again until the application works correctly"*. Furthermore, during the customer requirements validation *"we carried out different functional test cases, which are documented in the 'OTRS' tool and the control and traceability of the inconsistencies and errors identified are taken"* (Jen)

- *"In the verification perspective to ensure that the product is being developed under the quality criteria defined by the organization at the engineering level. For instance, test cases design, development stage verification, testing stage and obviously the functional verification"* (Stiven)

- *"In 'Redmine' all project defects and the requirements description agreed with the customers are reported. Each requirement has its design task, development and testing. Furthermore, each defect is reported and automatically alerts are generated about the defects that were found, each responsible area adjusts and validates each defect. Then, a new development baseline is released for the test area to rerun the test cases, perform a regression, and verify that the customer's requirements are effectively met"* (Phillip)

- *"For the development process release, we generate a baseline, which is reported by e-mail to subsequently start the test execution cycles. We currently have three test cycles: Cycle 1: complete execution, Cycle 2: defects review, and Cycle 3: complete regression of the entire system"* (Naty)
- *"We applied functional tests, integration, user interface, security, installation to the protocols. In addition, we performed user documentation tests"* (George)

Meanwhile, some software developers and quality analysts combine different tests during a software development project. They execute some functional, unit, performance, and security tests for guaranteeing the application quality:

- *"For some requirements, functional or performance tests are required, but everything is linked to the customer's needs"* (Kyra)
- *"Since the company has improved its processes, to guarantee quality, regression tests, smoke tests, acceptance, and functional tests are carried out to guarantee compliance with the customer's needs"* (Dan)
- *"We applied functional tests, integration, user interface, security, installation to the protocols. Also, we performed user documentation tests"* (George)
- *"We performed tests on desktop computers and then we performed functional tests to the application with the API (mobile)"* (Jen)
- *"We handle performance tests, here we make constant measurements to the applications to know what the desired ones are, what we lack if we have a script or something that is failing (bugs, time). In general, we have the part of security, quality in code and performance up to what I have worked"* (Andrew)

In addition, other participants are running manual and unit tests to ensure that the development process is well done:

- *"From the development area, there were some unit and performance tests were defined by technical leaders from the architecture area. They established formats and test cases according to the use cases and the analysts must comply with a part of unit tests"* (Sarah)
- *"The application of unit tests, which ensure that what we are developing is well done. They helps us to measure that the process quality is good"*. Moreover, *"in the*

development area, we apply in .Net unit tests with Visual Studio (through a specific framework that the tool has). These unit tests are stored in a repository, in the same space where the project source code is stored" (Nick)

- *"In the future, I would like to involve more indicators and automate the tests. We work with manual tests. We are reviewing automation tools and training processes"* (Naty)
- *"The reported findings are done through manual tests, and we have a part oriented to expert judgment (peer review)"* (Maria)
- *"I perform manual tests, previously I worked with different tools for automatic deployments, but it turns out that they were not so automatic, so we stopped working with this"* (Andrew)

On the other hand, 'James', as a project manager and 'Anna', as a quality analyst argue that they have diverse type of tests such as smoke and quality tests, functional and non-functional tests (load, performance, and stress). However, they only do functional and smoke tests which are executed by the development area:

- *"We did (the quality is focused on tests) the corresponding tests for validating the specific user operation and further back was the interface design and so on in terms of image design"*. Nevertheless, *"all application starts from the technical requirement established with the customers for then evaluating their functionalities and applying tests such as load, performance, interface (identity through wire-frame and mock-up the product perceptions), and field tests"* (James)
- *"The development area realises smoke and quality tests; performs functional and non-functional tests (load, performance, stress), but we have not a history of the tests and measures implemented to evaluate the software product quality"* (Anna)

Finally, four participants state that they only perform the smoke test (checking the application viability) or manual review of the application (through test cases):

- *"Quality is evaluated in terms of usability (user interface) and maintainability. Smoke tests are performed, load tests (customer response times). Before the final product delivery, the company performs quality software tests in the cloud (company server) prior to the product launch"* (Kate)

- *"We perform smoke and functional tests according to the component's functionalities (we request help to a software development partner). From the development area, the tests are not documented, they are performed shortly"*. Moreover, *"from the development area, we do not contribute more to quality (only smoke tests and security measures in terms of servers)"* (Josh)

- *"Test area performs the 'smoke test', the main idea is to check that there are no high priority errors, for saying that it is a blocking error, that does not let us run our already designed test case"*. Furthermore, *"we do unit tests through Visual Studio. The 'smoke test' is a manual review of the application. Regarding the design execution of test cases, a document was developed in which are all the steps that must be done functionally to validate the test case"* (Phillip)

- *"At the level of evidence, we are trying to involve a person who does the peer review of the test cases, in such a way that we contemplate all the possible scenarios and that we encompass what we want to evaluate. In addition, we carry out a 'smoke test' to identify if it is viable to start the execution"* (Naty)

In summary,

- These results describe the importance of manual and automation tests during software development projects and how such tests are implemented in different young software companies

- Some companies are still using manual tests for validating software usability and functionality and they are less focused on security and performing matters

- Companies mix some tests and peer review strategies to verify customer requirements and product functionalities

- The question needs to be asked about why the companies are not passing all their experiences to an automated protocol using technologies or, as they said, implementing continuous integration under the umbrella of DevOps. DevOps is a software development practices that combine software development (Dev) and information-technology operations (Ops) to shorten the systems development life cycle while delivering features, fixes, and updates frequently in close alignment with business objectives

6.1.5 Pattern F - Measures and Indicators Evaluated

Context for Measures and Indicators Evaluated

The software quality concept is often hard to capture for any organization. According to Lampasona et al., quality models aim at making the concept more operational by refining the 'quality' of software development products and processes into sub-concepts down to the level of concrete metrics and indicators. They highlight that in practice, *"it is difficult for an organization to come up with a reliable quality model because the quality depends on numerous organizational context factors, and the model, as well as the metrics and indicators, need to be tailored to the specifics of the organization"* (Lampasona et al. 2012, p.243).

Many companies need to establish clear and coherent measures for assessing software product quality (SPQ) and also to create an economic model that contributes to measuring the product quality for SMEs from the early stages of the life cycle. However, the software product family that defines the quality characteristics and sub-characteristics that can be measured (ISO 25000 standard) does not determine a specific set of indicators, metrics, and thresholds that can be taken by organizations as a reference when evaluating and certifying the quality of their products (Rodriguez, Pedreira & Fernandez 2015, p.129).

Academia and industry could work together for creating and adopting new and better measures with the aim to increase the productivity and competitiveness of young software development companies. They are still concerned that the product is being returned from the customer with some issue and that it does not have quality criteria evaluated through measures and indicators (Castellanos 2016, Delgado et al. 2016, Farah et al. 2014, Giraldo et al. 2014).

On the other hand, it is crucial that the industry and software development companies establish baseline measures for SMEs and keep updating those according to the project scope, context, and customer requirements. Thereby, all companies can be integrated through a general goal related to software quality.

Currently, companies are customizing their own measures or indicators according to their necessities. Lampasona et al. (2012) show a custom-tailored quality model for Ecopetrol, an oil company in Colombia, which has 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.

Meanwhile, Metaute & Serna (2016) present an assessment of metrics ownership and use in medium enterprises in Medellin-Colombia. They discovered that some companies applied metrics in distinct software development life cycle stages and customized methods for evaluating SPQ. However, they missed asking the companies about which specific measures they were implemented. Furthermore, the authors point out that:

- *"There are still companies dedicated to developing software but they are not interested in applying metrics in their processes because they see it as something complicated and without tangible benefits,*
- *Metrics for testing process depend on the software type, measurement needs, and even the baseline proposed by the developer or product quality certifier,*
- *Metrics are not defined from project planning stages, which can maximize risks, costs, reprocesses, and times. In addition, metrics do not apply to all processes in the software life cycle,*
- *There is no appropriation of up-to-date standards related to software quality, especially in metrics, such as SQUARE,*
- *There is a shortage of qualified staff in software metrics, especially in the interpretation of existing models, methodologies, and the creation of new metrics"*

On the other hand, some studies describe that "the standards are doing good work, but they do not appear to have had a great impact on academia and industry". They establish different models, but they do not show the measures implemented and the integration among non-functional requirements (SPQ characteristics) and measures/indicators (Febrero et al. 2016, Rodriguez et al. 2016).

Pattern Description for Measures and Indicators Evaluated

Colombian young companies are evaluating software quality with approximately 50 different measures and indicators related to coding rules, code reuse, user interface coherence, technical debt, rework indicator, quality process with indicators base, and satisfaction measures (see Figure 6.5).

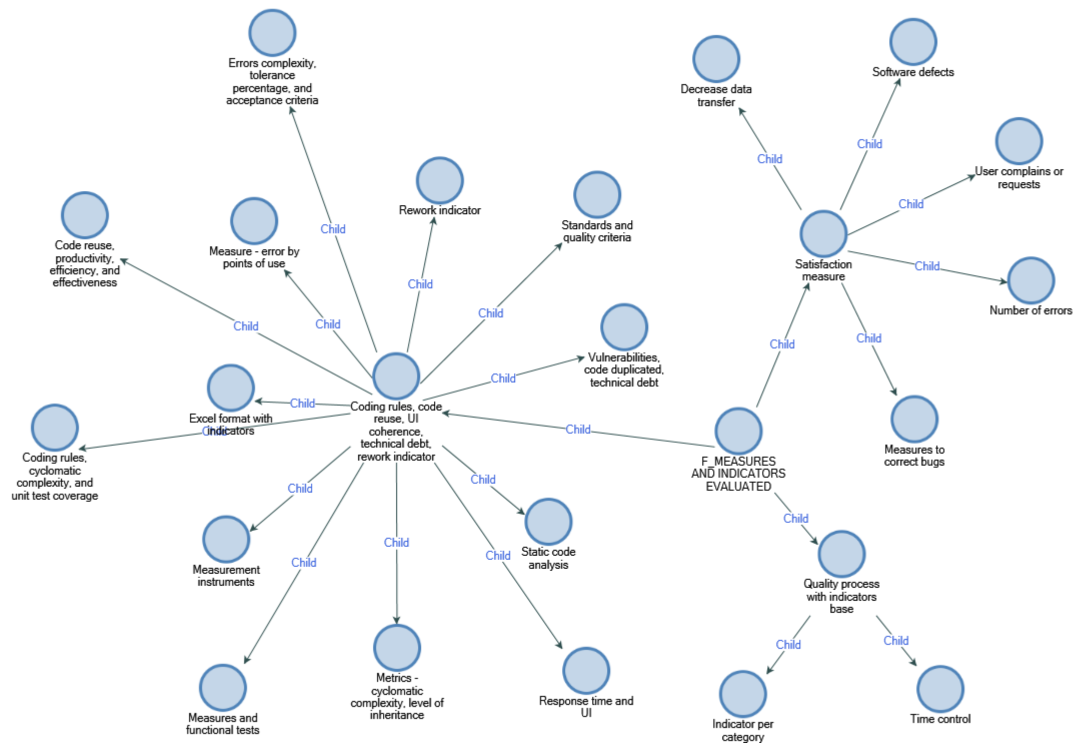

Errors complexity, tolerance percentage, and acceptance criteria
Rework indicator
Standards and quality criteria
Code reuse, productivity, efficiency, and effectiveness
Measure - error by points of use
Vulnerabilities, code duplicated, technical debt
Excel format with indicators
Coding rules, code reuse, UI coherence, technical debt, rework indicator
Coding rules, cyclomatic complexity, and unit test coverage
Measurement instruments
Static code analysis
Measures and functional tests
Metrics - cyclomatic complexity, level of inheritance
Response time and UI
Decrease data transfer
Software defects
User complains or requests
Satisfaction measure
Number of errors
Measures to correct bugs
F_MEASURES AND INDICATORS EVALUATED
Quality process with indicators base
Indicator per category
Time control
Child

Figure 6.5: Measures and indicators evaluated

For fifteen of the twenty participants, their work is focused on evaluating software code quality with distinct measures and indicators. Of those fifteen, eight make measures related to some software product quality characteristics such as maintainability, functionality, usability, and security. They create their own indicators or measures for tracking the quality based on their experience and knowledge. In the interviews, the data revealed that they are all applying measures and indicators for evaluating the time, software defect, cyclomatic complexity, rework indicator, code duplicated, vulnerabilities, user interface (UI), effectiveness, performance, and critical bugs. The other five interviewees made no comments on measures and indicators evaluated by their companies.

As can be seen from the following quotes, most developers, testers, and project leaders are focusing on measures related to code quality and process quality such as coding rules, code reuse, UI coherence, technical debt, and rework indicator according to their following statements. 'Jen' as a tester is an exception as she is more focused on product quality.

- *"The quality issue is associated with the code reuse, software life-cycle productivity, measures that allow you to see where there are wastes with large magnitudes in the development process, requirements engineering, design, etc. For instance, efficiency and effectiveness in the correction and modification of errors"* (Stiven)
- *"The measurement is made through the technical debt indicator for maintainability, which allows evaluating through some coding rules as it is founded in the application code, that is, if the code is clean. It implies nomination, validation in cyclomatic complexity criteria, and alignment on the unit test coverage"* (Lachlan)
- *"We have the following indicators or measures: number of errors that a component can have (high, medium, low complexity) and we have a tolerance percentage. If a component has low complexity, errors should not exceed a certain percentage"*. Moreover, *"another general measure is to take all the total requirements (format with the acceptance criteria for a subsequent delivery)"* and *"we are working on an Excel format and we have the respective indicators, but it is still in test to validate if we continue using it"* (Anna)
- *"We have return indicators, so in all phases we count how many returns we have, and we apply a measure. For custom development, we have errors by points of use cases, and we have statistically controlled it (currently, we are not doing it)"* (Karol)
- *"We have quality certification, different measurement instruments (compliance with code lines, bugs). In terms of support, measurement in response time, attention time"* (Kate)
- *"We performed tests on desktop computers and then we performed functional tests to the application with the API (mobile). Such measures are program response time, response reliability, performance, and daily functionality"* (Jen)
- *"In the development good practices, we have some metrics to know if the code is appropriate: cyclomatic complexity, coupling index between classes, level of inheritance, it is verified that the chains are defined correctly, that the variables and properties are declared in a correct way, that the classes meet a unique and particular objective and if it is not in this way, we try that the classes are built in the best way"* (Julien)
- *"We register with a network client for the Web Service part, and we observe the response (time: minus one second which is normal), if the response takes longer (five, seven or eight seconds), then, I as software developer leader check what is going on*

(a request as simple as accessing a form cannot take so long). Furthermore, in the front-end part, we checked that visually it looks good, that there are no bad fields, that the answer goes well, and that we do not have security issues" (Josh)

- *"At the development level, we use different standards that help us measure the code quality (lines of code, code reuse, security, databases, etc.)"*. In addition, *"we implement maintainability with a measure as static code analysis, using Sonar tool, integrated with Jenkins (integration test), so every time the project is reviewed, we perform a code analysis to identify the vulnerabilities, and how much code is duplicated, technical debt (this is important for the company)"*. Finally, the participant *"perform a code analysis to identify the vulnerabilities, and how much code is duplicated, technical debt (this is important for the company). Technical debt is measured in time"* (Nick)

Meanwhile, 'Angel' and 'Kyra' points out about rework indicator with the below quotes respectively:

- *"We handle the rework indicator, which measures the time spent by a developer uses to correct errors found by the customer or internally"*

- *"We have the rework indicator, oriented to the error's correction: how many times a product must be corrected or to say something, if it is delivered in time for tests, but finally not delivered with the appropriate quality"*

Some specific measures and indicators are listed by 'Maria', as a quality analyst (from 1 to 26) and 'Lachlan', as a Project manager (from 27 to 33) (see Table 6.2). The candidate has classified each of them as a type of measure (code, process, and product quality).

On the other hand, 'Nick', as a developer and 'Phillip', as a tester, argue respectively that *"we control the time of adjustments/changes of each project, this being a way to measure the product"* and *"when we report a defect, we are obliged to register what is the category, if it is shape, functionality, security, etc. For each category an indicator is generated, according to this indicator, the quality area reports what is the quality level of the product developed"*.

Finally, some participants describe in the interviews that they are implementing some satisfaction measures focused on decrease data transfer, bugs correction, number of errors, software defects, and user complaints/requests:

Table 6.2: List of measures and indicators

No	Measure and indicator	Type
1	% Changes without commitment guidance	Process
2	% Reverted changes	Process
3	Deliveries without critical bugs	Product
4	Efficiency of peer reviews	Process
5	Quality of installers	Process
6	Rework cost	Process
7	Volatility of requirements	Process
8	Quality of requirements	Process
9	Effectiveness of planning	Process
10	Effectiveness of the development test	Process
11	Effectiveness of defect elimination	Product
12	Actual percentage of progress	Process
13	Prioritization of critical bugs	Process
14	% Changes without responsibility	Process
15	% Changes without tracking	Process
16	% Effort deviation in protocol development	Process
17	Periodic behaviour of maintenance cases	Code quality
18	% Protocol delivery deviation	Product
19	Cost of reworking protocols	Process
20	Effectiveness of the QAP test	Process
21	Development quality	Process
22	Stability of requirements	Process
23	% Bugs reported in regression	Process
24	Effectiveness of the test	Process
25	% Bugs rejected	Process
26	Critical Bugs	Process
27	Technical debt	Process
28	Indicator of duplicate lines of code	Code quality
29	Indicator of unit tests	Code quality
30	Indicator of the source code complexity	Code quality
31	Response time for the performance case in different scenarios: peak loads or basic usage load of the application	Product
32	Percentage of application airtime in a specified time period	Product
33	Findings related to security (number of problems found in the application)	Product

- *"We contribute to decrease the data transfer speed of the product"*. Furthermore, *"It is important to reduce the number of errors as much as possible to generate reliability and for customers to access our products again"* (Josh)

- *"I need a greater effort to correct the bugs. We need to establish these measures and validate if the product is delivered with quality"* (Kyra)
- *"A requests number or detected bugs. As is in quality terms, we consider the software errors that are presented during its basic performance. Furthermore, in second place, we evaluate the user observations associated with the fact of the user interface perception"* (James)
- *"There are some indicators that are generated from the support area (failures reported by the user and the modifications made)"* (Sarah)

To summarise, from the results embedded in the excerpts above, i) companies implement more measures related to process quality and code quality rather than product quality, ii) companies point out about rework indicator, and iii) companies are implemented satisfaction measures.

These results show that the participants create their new measures and indicators according to the companies' needs. However, they are interested in evaluating the software quality without any attention to product quality because most of them are focused on code quality and process quality. It is evident that in some way they are evaluating the quality of their products, but with less conceptual and technical knowledge about existing measures/attributes for their planning and implementation.

6.1.6 Pattern G - Process Quality over Product Quality

Context for Process Quality over Product Quality

Real-world problems are more non-functionally oriented than they are functionally oriented, e.g., poor productivity, low quality, slow processing, high cost, and unhappy customers (Chung & do Prado Leite 2009, Haigh 2010).

The findings below are lacking focus on software product quality, but they are relevant to academia, industry, and government because they need to understand the importance of product quality and prove that the philosophy about *"a quality process produces a quality product"* (Kitchenham & Pfleeger 1996, p.13) is no longer enough, because we can reinforce the perspective stated on chapter 2 on *"the software quality evaluations should be based on direct evidence about the product, not only on evidence about the process"*

(Maibaum & Wassyng 2008, p.91), since a high-quality process does not necessarily ensure a good quality product.

Some authors state that quality is an important criteria and a main concern for software development companies (Idri et al. 2016, Rodriguez et al. 2016, Yahaya et al. 2008). Yahaya et al. (2008) argue that companies and software houses are competing to produce software which is claimed to be good and fulfil user's expectation. Therefore, the quality aspect of a software product is seen as an important issue but companies that develop the software could not justify and guarantee the quality of their products, thus leaving users in uncertainty. Meanwhile, Rodriguez et al. (2016) explain that software quality is a hot topic at the moment, due mainly to the fact that software has spread its influence into a large part of industry and existing disciplines.

Many companies need engineering and software quality practices suitable for their particular characteristics. These practices contribute to high-quality development (Pino et al. 2008, p.238), which must evolve and improve to adapt to new demands and scenarios in order to be more competitive (Colomo et al. 2011, p.395).

Adopting new and better quality practices could increase the number of customers with greater reliability in the design and development of software products and reduce the investment, in time and effort dedicated to software maintenance.

Companies could create communities to share the meanings and criteria to consider the quality as a key factor for software development projects and contribute to the SPQ evaluation and building new meanings with other companies' experiences and knowledge.

On the other hand, in accordance with the importance of software in different contexts, it is important that Universities restructure their Software Engineering courses, with the aim of training professionals to contribute significantly to the industry, because it requires each day a higher qualification in this regard. Villavicencio & Abran recommend the inclusion in the curriculum of topics related to quality, based on the curricular guidelines for University programs in Software Engineering such as Swebook and the body of knowledge of software measurements (Villavicencio & Abran 2015, p.126).

The poor quality of current software represents a significant financial burden for software developers companies. In general, software products tend to present reprocessing in their development cycles due to processes improvisation, time overestimation, and development resources. Furthermore, this particular study has expressed a *"special interest in improving their software processes in order to increase the quality and productivity of their software products developed. Nevertheless, product quality is always complex to evaluate"* (Pelaez et al. 2011, p.42).

An important concern of software industry companies has been the development of software products with optimum use of resources, time and costs; in other words, the quest is to be efficient (Ahmed & Capretz 2011, Garzas et al. 2013). Therefore, for young Colombian companies, it will help to reduce their workload, time, and resources if the international standards focused on software product quality building by new frameworks with characteristics and attributes for evaluating SPQ or developing SPQ measures which could be economical to produce.

In addition, the lower quality in software in young companies is present because it is not considered in the development processes. There are not objective and measurable criteria to avoid customer dissatisfaction (Febrero et al. 2016, Jacobson et al. 2013, Zelkowitz 2011). This generates a bad image and low competitiveness in international markets. Developing effort to support quality means less effort for the development cycle. Therefore, low sales level in software development companies will represent less profit to companies and they could go out of business (Pino et al. 2008, p.250).

According to Kumar et al. (2015), the software development industry has been growing recently but the most important issue that is causing worries to the software industry is quality. There are still more than 50% of software projects that overrun their budget. A lot has been discussed in the literature about the factors that impact the quality of these systems but most of these factors are high-level factors. The case study which also depicts that the top 5 factors that are impacting the quality of the enterprise are designing new business processes, existing IT infrastructure, skills of the team, importance of integration testing and unit test, and test coverage.

For the aforementioned reasons, software products must be developed in less time, with quality, and with a good understanding of the customer requirements. Furthermore, software development companies are concerned to have high-quality indicators to be recognized in the market (Calero et al. 2008, Gall et al. 2008).

This pattern is essential because this research found that most of the participants consider quality as a key factor for software development projects, but not software product quality. In Colombian scenarios, software development companies face several quality problems with the software they develop. This fact causes fast developments to be made with limited resources, but despite the fact they make user-request-oriented products, the time dedicated to their quality is not enough (Moreno et al. 2010, p.23). Moreover, software product quality is one of the main concerns of software development companies (Colomo et al. 2011, Pino et al. 2008, Rodriguez et al. 2016).

According to Lampasona et al. (2012) the concept of 'software quality' is often hard to capture for an organization. Quality models aim at making the concept more operational by refining the 'quality' of software development into sub-concepts down to the level of concrete metrics and indicators. In practice, it is difficult for an organization to come up with a reliable quality model because the quality depends on numerous organizational context factors, and the model, as well as the metrics and indicators, needs to be tailored to the specifics of the organization. The authors state that *"the major goal related to developing the quality model was 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 Ecopetrol in becoming more agile"* (Lampasona et al. 2012, p.243).

Finally, it is still necessary to implement quality in the development process and achieve a clear and coherent evaluation process. This assertion is supported by Salcedo & Gil, who state that *"given the demands of the end-users and the increase of a greater amount of requirements compared to software products, it is necessary that the quality is immersed and implemented at different development process levels. Therefore, it is necessary to count with new evaluation methods that allow knowing the answers given from the opinions of the users"* (Salcedo & Gil 2012, p.92).

Pattern Description for Process Quality over Product Quality

The strongest and most significant pattern to emerge from the analysis, was what the candidate 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 CMMI (see Figure 6.6), 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 (SPQ), and they do not have well-defined SPQ assurance procedures. Just over half of

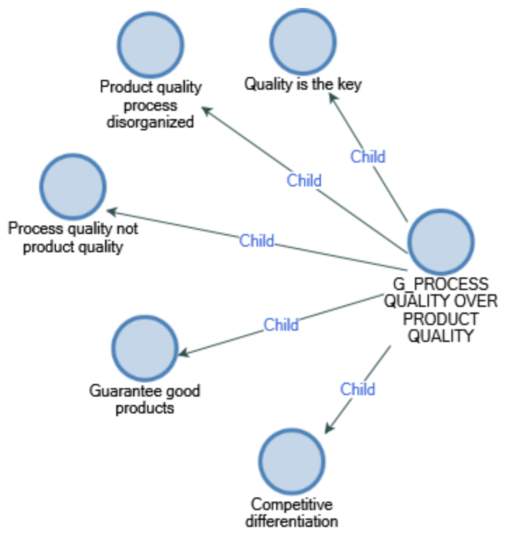

Product quality process disorganized
Quality is the key
Child
Child
Process quality not product quality
Child
G_PROCESS QUALITY OVER PRODUCT QUALITY
Child
Guarantee good products
Child
Competitive differentiation

Figure 6.6: Process quality over product quality

the 20 participants confused 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 were process quality.

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 to control the logs or bugs is in implementation (because we do not have measures), the company has generated the indicators slowly"* (Anna)
- *"With the CMMI process, the documentary shortcomings are closing, but internally it is necessary to improve and strengthen the software documentation. The focus is on process rather than product quality"* (Julien)
- *"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. Therefore, we have in our process, bugs records reported by the customer"* (Kyra)
- *"The development area is more focused on product safety and maintainability. The quality area is more focused on product functionality and performance"* (Josh)

- *"All the characteristics associated with quality are immersed in the quality processes defined by the company"*. In addition, *"I think that the quality issue is associated with the code reuse, software life-cycle productivity, measures that allow you to see where there are wastes with large magnitudes in the development process, requirements engineering, design, etc. For instance, efficiency and effectiveness in the correction and modification of errors"* (Stiven)

- *"When we evaluating the computer systems security, it is necessary to determine and build standards for the platforms and architectures designs that are viable internally at the company (software legal level) and externally complying with international standards for information security (COBIT, ITIL, OSSTMM, etc.), and the laws that apply to different platforms"* (Liz)

- *"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, in technical issues, criteria such as performance and the completeness of the entire requirements from all points of view"* (Karol)

- *"The measurement traceability and indicators were lost; it was done up to a period. It is because the work group was reduced, developers were external. Traceability is done, in the support area by analysing which are the requests that arrive if they are related to previous requests, response time, time invested, but it is a personal exercise"* (Sarah)

- *"There are normal or known stages in the project execution. From the requirements stage, we ensure that the requirements established with the customer are clear. We handle standard formats, accompanying experts in meeting requirements (to reduce the information duality and have greater requirements clarity)"* (Angel)

- *"We do not implement security, portability, and reliability measures because it always starts from the customer's requirements and the project dimension. We have had projects, which due to time and resources, it is not necessary to start a specific measurement process"* (Dan)

- *"When we start with a project, we define a test plan with scope and types of tests that support the project. The types of tests are defined according to the project that is being developed"*. Therefore, *"We have defined in the quality area, certification letters, which consider the defined scope, what was fulfilled, software version, testing* *environment used, project observations, documents which were considered to carry out the project, those responsible, and if the project was conditioned (time, cost, incidents)"* (George)

- *"Currently, I do not count an indicator such as, i.e. cyclomatic complexity. Until recently as one of the quality assurance criteria is to do a peer review and verify that certain conditions of a review that is made of the SONARQ are met. The only indicator we have is how many findings they reported in the peer review"* (Maria)

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). Maybe if we differentiate the product focus, likewise we have the product evaluation perception"* (James)
- *"Product quality is part of the company's mission, which is providing services to the customer to guarantee good products... I believe that quality is ultimately something tangible, but it is accompanied by other criteria such as time, accompaniment and guarantees."* (Angel)
- *"What we seek is that the functionality corresponds to what should be implemented, which is determined by the use cases and 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, in technical issues, criteria such as performance and the completeness of the entire requirements from all points of view."* (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 (SPQ). Furthermore, the latter group confused process quality with product quality. As a consequence of this confusion, they do not recognize that they are not, in fact, measuring product quality.

6.1.7 Pattern H - Existing Procedures, Best Practices, Guidelines, and Policies

Context for Existing Procedures, Best Practices, Guidelines, and Policies

According to Pino et al. (2008) and Colomo et al. (2011), many companies need engineering practices and efficient software quality processes suitable for their particular development and product needs. These practices contribute to high-quality products development, which must evolve and improve to adapt to new demands and scenarios in order to be more competitive.

Software product quality (SPQ) is demonstrably important, but the processes of the Colombian companies focus mostly on software quality (SQ) as a process.

Some studies have implemented mixed models, frameworks, and best practices for evaluating software product/process quality according to the companies' context and needs (Lampasona et al. 2012, Metaute & Serna 2016, Pelaez et al. 2011).

Pelaez et al. (2011) state that Colombian companies have implemented ITMark and Light MECPDS customized approaches as the most recommended practices for improving the software quality process.

Meanwhile, a study conducted by Metaute & Serna (2016) 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. They found that some companies use methods as CMMI, ISO 9000, ITIL, and customized methods to evaluate the software product quality.

Nevertheless, Gasca et al. (2013) analyse the current state of the implementation of best practices in software development in Colombian small and medium organizations. To achieve this goal, a survey was designed which was applied to a representative region of Colombia (Medellin). The authors present the analysis of the collected statistical results, using a pilot study given the target population (Latin America). In addition, they argue that statistical studies that are published in the report of Standish Group Inc. identify causes of projects failure associated with: a) inadequate implementation of good practices in project management, b) the poor definition of requirements, c) the lack of definition of techniques to guarantee the process quality and software product and; d) the lack of risk management (StandishGroup 2015).

On the other hand, Lampasona et al. (2012) present experiences in developing custom-tailored quality models for Ecopetrol, a Colombian oil and gas company. They describe the creation of the quality model to align the incorporation, acquisition, and development of IT solutions with the business goals. The quality model supports the definition of baselines for software quality at Ecopetrol and it is part of its IT landscape. It helps Ecopetrol define policies to be followed by the organization in order to achieve the goal of standardization of quality requirements.

Febrero et al. (2016) show the existing work on the modelling of software reliability based on ISO 25000 standard as the starting point for a reliability assessment proposal establishing a reliability model layout and assessment schema. Furthermore, they describe that the standards are doing good work, but they do not appear to have had a great impact on academia and industry.

Quality models help to understand the concept of software quality more practically by defining the product/process quality into sub-concepts down to the levels such as characteristics, measures, protocols, and procedures (Kumar et al. 2015, Lampasona et al. 2012, Rodriguez et al. 2016, Pelaez et al. 2011). In addition, Lampasona et al. argue that *"it is difficult for an organization to come up with a reliable quality model because the quality depends on numerous organizational context factors, and the model, as well as the metrics and indicators, need to be tailored to the specifics of the organization"* (Lampasona et al. 2012, p.243).

Pattern Description for Existing Procedures, Best Practices, Guidelines, and Policies

Participants indicated that they believe they are using the best development practices, guidelines, and policies are implemented for evaluating software quality by Colombian companies. Figure 6.7 shows the relationships between pattern, nodes and their keyphrases.

For eleven of twenty participants, their work as software developers, quality analysts, and project managers are focused on the best development practices and following guidelines and policies provided by the company for evaluating software quality. The other nine interviewees made no comments on existing procedures, best practices, guidelines, and policies used.

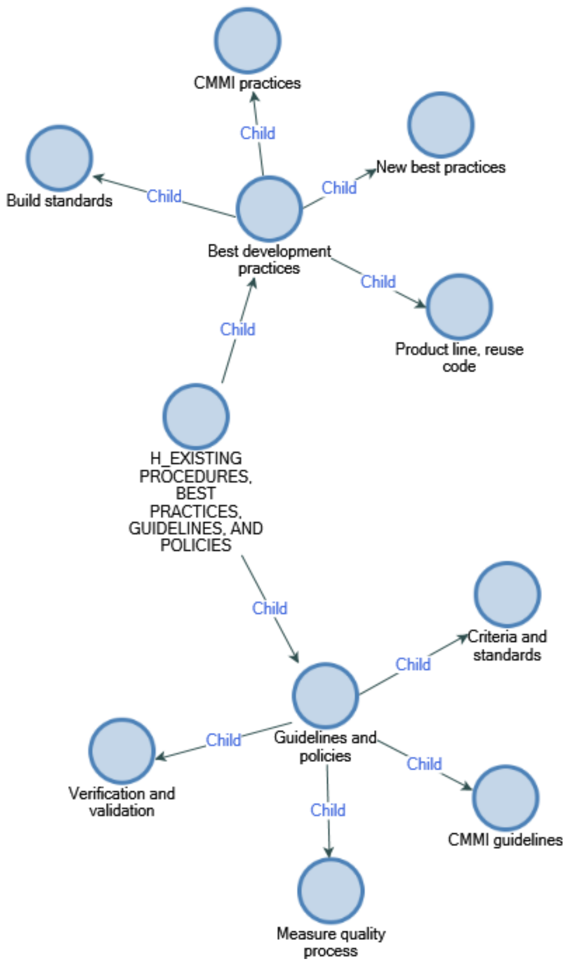


Figure 6.7: Existing procedures, best practices, guidelines, and policies

The views of 'Stiven' show elements of SPQ with a focus on product line architecture and security and usability. So do those of 'Liz'.

- *"We generate product line (base-line of architecture on technology), to all software that we develop with .Net. It is done under the baseline; many criteria are predefined and helps to reuse code. This helps mitigate many errors and risks in development and quality issues"*. In addition, *"we are very interested in measuring usability since it is fundamental, but we must migrate to the user experience issue"*. Meanwhile, *"in security matters we are working the minimum. United States and Mexico companies ask us for advanced security practices. The security criteria in software development are key"* (Stiven)

- *"When we evaluating the computer systems security at an organization, it is neces-*

sary to determine and build standards for the platforms and architectures designs that are viable internally at the company (software legal level) and externally complying with international standards for information security (COBIT, ITIL, OSSTMM, etc.), and the laws that apply to different platforms". Furthermore, *"developing and applying security policies for data protection, database coding, credentials, and other sensitive information; determining the adequate resources for the proper functioning of the organization and its assets in order to meet the integrity, availability, and confidentiality criteria; and implementation of a risk control and prevention system"* (Liz)

Some participants claim to be following good development practices to make the code integrations:

- *"The senior developer has advanced knowledge and experience in the software development area. What I do is looking for new implementations, new cutting-edge solutions, always implementing the best practices, so that our software is optimal, efficient, and generates lower costs"* (Andrew)
- *"We must follow good development practices to make the code integrations, here is where the product quality is assured, in which the processes are optimal"* (Julien)
- *"Within the organization, we have indicators. We follow the development and testing process"*. However, they carry out software traceability with a tool that *"was developed in the company, it is a custom development (we developed it to achieve the CMMI 3 certification). It was developed in the IBM suite. We are certified in CMMI 5, and every time we make improvements to the applications. For instance, in the market for the design of the tests, tools such as Test Link and Mantis are generally used, but the company does not use them. We use a developed tool called 'Buggy'. Here the test plan, test cases, bugs report are recorded. The records are counted, and this information feeds another application called 'Pronox" and with this, the project management is carried out"* (Kyra)

Meanwhile, most developers and quality analysts have guideline documents and policies in their companies for developing applications. They state as follows:

- *"When we started the quality theme in the organization, the application basic process of software tests was carried out to comply with the customer's requirements. However, currently, we are looking to organize a formal process"*. In addition, *"we* *need to have coherent and clear measurement processes because the quality at the end, it is the company image and confidence, so if we deliver product quality, we generate trust in the market, customers, and obviously, it gives us a plus to compete"* (Anna)

- *"We do not have a coding standard (the process of retaking the person's code is difficult, we lose time, our learning curve is much slower). I believe that our measurement process is failing"* (Josh)
- *"Developing and applying security policies for data protection, database coding, credentials, and other sensitive information"* (Liz)
- *"Being a company that bases its capacity of engineering in processes, processes endorsed with practices of global recognition as CMMI. Therefore, we look at the quality from two perspectives: i) verification, to ensure that the product is being developed under the quality criteria defined in the organization at the engineering level. For instance, test cases design, development stage verification, testing stage and obviously the functional verification, and ii) validation, to validate that the product which is delivered meets with what is defined and expected by the customer"* (Stiven)
- *"When we receive the requirements in the form of user stories, use cases, for achieving the quality, we must comply with a series of standards, implicit or explicit within each project. We must meet certain standards as scripts for databases to work. These scripts must comply with a structure and design. Furthermore, the interfaces must meet usability and user experience standards"* (Dan)
- *"The objective as quality analysts is to ensure that the quality process of software manufacturing meets the requirements defined by the customer and the process defined by the quality area"* (George)
- *"We follow architectural guidelines (solid principle). Furthermore, we have the CMMI level 3 certification, we have support documents, documents generated in the development area and checklists"*. On the other hand, *"we have been weak in the documentation. For instance, if we want to check which functionality supports a meter developed three years ago, this information does not exist. Therefore, this information should be raised from the code that was developed. However, with CMMI process, the documentary shortcomings are closing, but internally it is necessary to improve and strengthen the software documentation"* (Julien)

- *"We have a guidelines document and policies for the development of solutions. Here we have the SONARQ review and that it does not exceed certain levels, that it has complied with all Microsoft standards, so it is a checklist aimed at ensuring that the code that is being developed is of good quality"* (Maria)

- *"I address the leaders of the different disciplines (requirements, agility, architecture, development, etc.) to achieve the quality guidelines"* (Lachlan)

To summarise, these results show that companies have guidelines, policies, and their own best practices for evaluating software quality according to the project scope, context, and user requirements/needs. However, they need to reinforce the idea to create a good framework or standard to follow steps/stages according to the development process and quality criteria. It is possible to create it if they think as a software community, and they share their experiences and ideas to build an economical and flexible model.

6.1.8 Pattern I - Project Management Skills

Project Management Skills Context

Some statistical studies identify causes of failure associated with the inadequate implementation of good practices in project management and lack of risk management (StandishGroup 2015).

An important concern of software industry companies has been the development of software products with optimum use of resources, time and costs; in other words, the quest to be efficient (Ahmed & Capretz 2011, Garzas et al. 2013). Therefore, for young companies, investing time and resources in training for their software development teams could help to reduce their workload, time, and product return from customers, if all team understand how to manage a project, which best practices (characteristics, protocols, and measures) they need to use for evaluating SPQ, and which techniques are best for software development.

Some studies show that small and medium enterprises (SMEs) in Colombia present various obstacles including project management skills in software development teams (Escobar & Linares 2012, Gasca et al. 2013, Kumar et al. 2015).

According to Escobar & Linares (2012) an obstacle for SMEs development in Colombia includes lack of managers with management skills and strategic thinking. In addition, the authors 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. The study contribution helps to understand the current state in Colombia in terms of quality measurement, competitiveness, and productivity.

On the other hand, StandishGroup (2015) report identifying causes of failure associated with: i) inadequate implementation of good practices in project management, ii) the poor definition of requirements, iii) the lack of definition of techniques to guarantee the process quality and software product and; iv) the lack of risk management.

Finally, Kumar et al. (2015) present a case study to identify detailed and operational level quality factors. It depicts that the top 5 factors that are impacting the quality of the enterprise integration projects are i) designing new business processes, ii) existing IT infrastructure, iii) skills of the team, iv) the importance of integration testing and unit test, and v) test coverage. The authors, identify the needs to establish a framework that will provide a set of preventive and corrective actions for the quality factors which help in improving the system quality.

Project Management Skills Pattern Description

Project management skills are implemented for planning and controlling the team with good practices and strategies. In addition, companies have defined a dynamic process for reviewing how the results came out, which defects are identified, tracking test execution, and management estimation. The relationships established are presented in Figure 6.8.

For ten of twenty participants, in their work, they are implementing project management skills by organizing the team and making the quality process dynamic. In addition, they are defining and designing test plans with scope and strategies during the development and evaluation of their software projects. The other ten interviewees made no comments on project management skills.

Some participants prefer to work on a dynamic process that allows supporting the back-end, database management, and the team who are doing traceability on tests execution. In addition, they want to have efficient solutions working directly with the customers:

- *"My everyday activities are focused on organizing the team and making the quality*

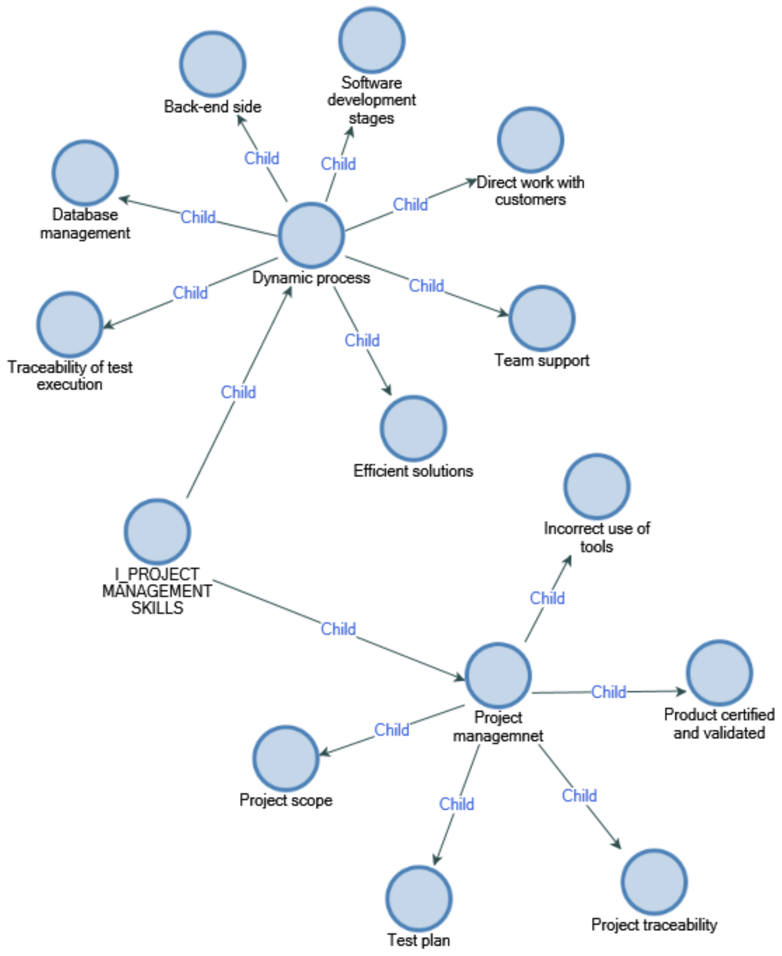


Figure 6.8: Project management skills

process dynamic" (Anna)

- *"We work hand in hand with the client. They require information care and data reliability. We guarantee to the user that the information is protect and always available at the time it is required"* (Kate)
- *"I am a developer working on the back-end side (functionality and quality when interacting with customer information). Also, I work with the internal part of the application, and database management"* (Liz)
- *"For improving the performance and functionality of the product. The idea is not to do reprocessing, if I dedicate three hours to work on the same thing, then I like to look for efficient solutions for giving maintenance and good performance to the application"* (Sarah)

- *"I offer support to the teams: solving doubts, validating that the deliverables that we make comply with the process standards that we have defined"* (Kyra)
- *"We have traceability and support of the tests execution, obviously there are going to be failures and errors"* (Phillips)
- *"We execute the test case, we give the approval that the product complies with the customer's requirements, and we generate an internal certification that guarantees that the product complies with the established requirements. We have defined in the quality area, certification letters, which consider the defined scope, what was fulfilled, software version, testing environment used, project observations, documents which were considered to carry out the project, those responsible, and if the project was conditioned (time, cost, incidents)"* (George)
- *"The company works as a software factory and we work with specific areas. I am in charge of the factory area that is responsible for software quality; we perform requirements, development, and quality"* (Maria)

Participants claim that software development teams need to define and design test plans with clear project scopes and well-defined strategies for executing such tests, doing traceability, and avoiding incorrect use of tools:

- *"We have the 'Redmine' tool for classifying and perform versioning of each project in development (the project is controlled from its beginning to the end). With this tool, the organization can know the development time invested to each requirement, the project status, generate a schedule and we can keep track of project changes"* (Kate)
- *"I consider that we do not use correctly the tools. For instance, continuous integration criteria, DevOps criteria, I think the company is very backward. When we neglect the process deliveries start to fail, reprocesses in the evaluation are presented, tests or quality, because manuals must be made and there is no agility in the process"* (Sarah)
- *"As a company policy, none product is delivered to the customer if the defined quality processes have not been met. It is considered a serious fault in which a product is delivered without having carried out the quality process defined in the development and testing phase, unit tests, functional tests, integration tests, validation with the client, etc. With the criteria established by the company, the product quality has been certified and evaluated before being delivered to the customer"* (Stiven)

- *"The process begins with the project manager request (resources) to execute the tests, we review the people available and then we assigned them. We carry out a project contextualization process. From here, we generate proceedings, which are the support of how to contextualize, which was delivered, with what inputs (resources) we have, and we sign the document to record the scope of tests application. The above is done through a project baseline"*. Moreover, *"I review again how the test results came out, how many defects were identified, we generate the acceptance letter, and we prepared a closing report with the support of the test analysts. Some report generates indicators, which are presented in a monthly management meeting with the entire project managers' team to take corrective actions and review strategies to improve these indicators"* (Naty)

- *"We define and design a test plan with scope, strategies, and tests to be carried out on a specific product. Then, we design scenarios and test cases where we ensure that all cases developed by the development area are considered. Finally, we execute the test case, we give the approval that the product complies with the customer's requirements, and we generate an internal certification that guarantees that the product complies with the established requirements"* (George)

To summarise, from the results embedded in the excerpts above participants claim that i) companies need to work on a dynamic process to have efficient solutions for the customers and ii) companies need to define clear test plans with well-defined scopes and strategies.

6.1.9 Pattern K - Software Quality Characteristics

Software Quality Characteristics Context

Although the engineering community has classified requirements as either functional or non-functional, most existing requirements models and requirements specification languages lack a proper treatment of product quality characteristics (Chung & do Prado Leite 2009, Haigh 2010). Therefore, the government, academia, and industry need to have a better understanding of software quality requirements. They need to understand the business need, what aspects of software product quality are important to them and to users so that they can ensure that developers of the system implement the features with the highest priority.

According to Pino et al. (2008) and Colomo et al. (2011), small companies need efficient Software Engineering practices that are suitable for their particular characteristics. These practices would support the development of products of high quality which must evolve if they are to adapt to new demands and scenarios as they seek to make these companies become more competitive. Thus, companies could create communities to share their concepts and views about non-functional requirements or software quality attributes for building meanings with other experts and companies' experiences.

SME can select their main non-functional characteristics and define which one they need according to project scope, context, requirements, and customer needs. In addition, they can build a knowledge bank with the best practices (characteristics and measures related and how it could be operated).

Some studies describe the software product quality (SPQ) characteristics such as reliability, efficiency, security, maintainability, performance, scalability, and suitability as essential quality requirements for evaluating both development process and the final product (Febrero et al. 2016, Guana & Correal 2013, Heck et al. 2010).

Heck et al. argue that *"more and more applications depend on the reliability, availability, and integrity of software systems due to the increase of complexity at the hardware, software, and communication level, creating quality systems has become both a major scientific and engineering challenge"* (Heck et al. 2010, p.38). Furthermore, they state that implementing other characteristics such as completeness, uniformity, conformance, correctness, and consistency could help prevent poor quality of the requirements (changing and incomplete requirements), which are the primary reason why projects are failing. Guana & Correal state that *"during the software product definition it is necessary to choose the components that appropriately fulfil a product's intended functionalities, including its quality requirements (i.e., security, performance, and scalability)"* (Guana & Correal 2013, p.541). Thus, the selection of the appropriate set of assets from many possible combinations is usually done manually, turning this process into a complex, time-consuming, and error-prone task.

Nevertheless, Febrero et al. (2016) consider that software reliability based on ISO 25000 is the crucial factor as regards estimating both software quality and software cost. The authors describe that *"the standards are doing good work, but they do not appear to have had a great impact on academia and industry"*. To which Metaute & Serna (2016) affirm that there is no appropriation of up-to-date standards related to software quality, especially in metrics, such as SQUARE.

Meanwhile, and according to the systematic review carried out in Rodriguez & Piattini (2012*b*), functional suitability is one of the most relevant characteristics, and it is among those that has generated the greatest interest by companies. This study was focused on ISO 25000 standard.

Hofman (2010) considers that quality means a different set of characteristics for different perspectives, and for different contexts of use (the same product may be perfect for one and useless for another context of use).

On the other hand, Mellado et al. (2010) state that there are few contributions about software product quality and security and although testing area (focus on functionality) is a field that has well worked, it has still not developed the necessary techniques for evaluating in an effective way the quality and security of a software product.

Software Quality Characteristics Pattern Description

Four participants (Anna, George, Josh, and Kyra) defined software product quality (SPQ) characteristics and measures. Sixteen focused only on software process quality and functional characteristics. Figure 6.9 shows the relationships found.

In the interviews, the data revealed that the four participants were applying SPQ characteristics [called as well non-functional requirements (Haigh 2010, p.362)] considered functionality, maintainability, portability, security, and usability of the products. They also combined those to get some results to focus on measurement, product context, and clients requirements.

'Anna' and 'George' as quality analysts are focused on aspects related to functionality and maintainability. 'Anna' argues that from her experience *"in many software factories, only the product functionality is measured and the quality evaluation goes beyond the functionality"*. However, 'George' states that *"they are more concerned with aspects such as functionality and maintainability. However, they are starting to improve a bit the usability criteria"*.

Although 'Josh' is not implementing Reliability measures, he states that *"the development area is more focused on product safety and maintainability. The quality area is more focused on product functionality and performance"*.

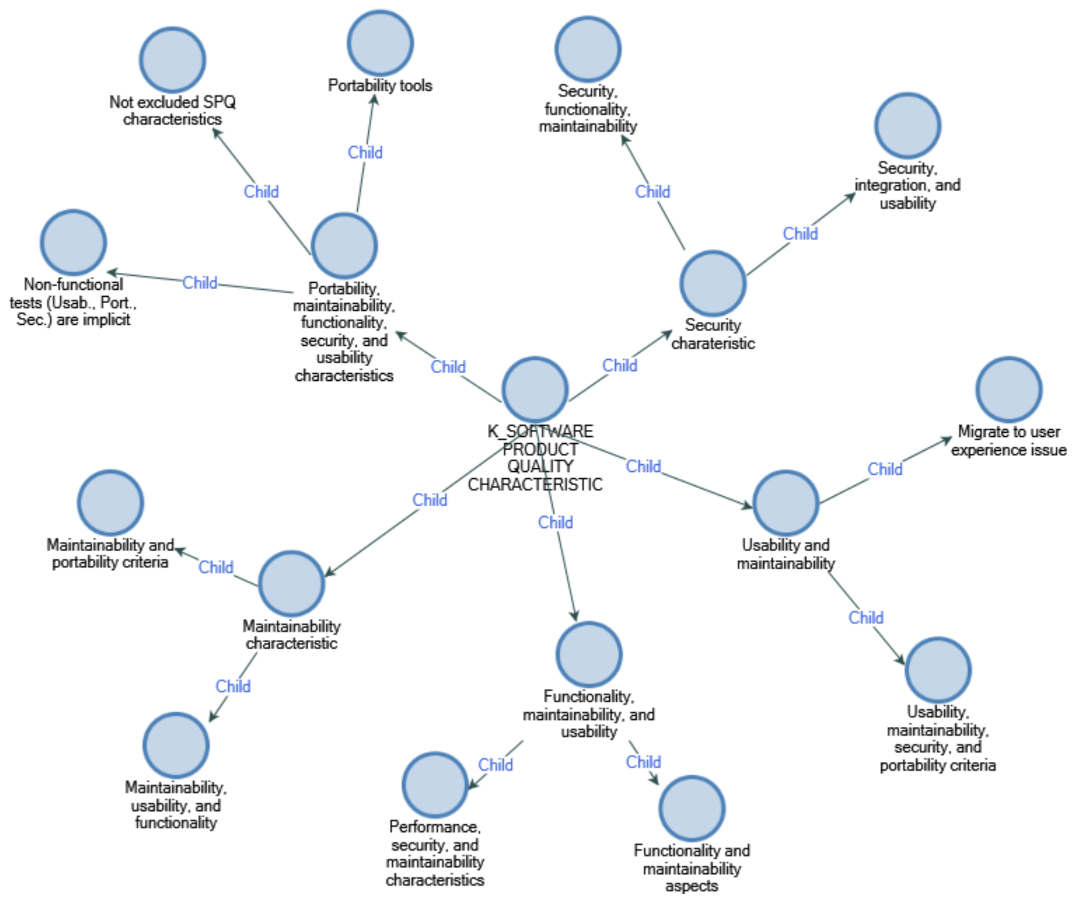

Not excluded SPQ characteristics
Portability tools
Security, functionality, maintainability
Security, integration, and usability
Non-functional tests (Usab., Port., Sec.) are implicit
Portability, maintainability, functionality, security, and usability characteristics
Security charateristic
Migrate to user experience issue
K_SOFTWARE PRODUCT QUALITY CHARACTERISTIC
Maintainability and portability criteria
Usability and maintainability
Maintainability characteristic
Functionality, maintainability, and usability
Usability, maintainability, security, and portability criteria
Maintainability, usability, and functionality
Performance, security, and maintainability characteristics
Functionality and maintainability aspects
Child

Figure 6.9: Software quality characteristics

The following participants have a focus on maintenance from different perspectives, but mainly on process quality:

- *"We assembled some automatic tools (Development Operations-DevOps), which we have not finished implementing. Maintainability and coupling measures are being made"*. Moreover, *"based on cyclomatic complexity, we identified how the code will be maintainable, it is then sought to improve the value by coding simplifying. It is an indirect measure and it is focused on module level. It does not guarantee all the modules together. Moreover, cyclomatic complexity only measures a module"*. (Karol)
- *"From my experience, maintenance has been the most important because it is what I have suffered too much. In many cases, no attention is given to the tools that are suitable to really evaluate the product, it is something that is neglected"* (Sarah)
- *"We have a generic product, therefore, for us, it is something new the entrepreneurial*

capacity in the development of new products. We have focused more on the maintenance of large applications. There are other types of measurements that are of greater interest" (Stiven)

- *"We apply maintainability criteria. However, we are working on mobile developments related to portability, but in security, we must pay more attention to this. In other companies, security is not very important, it is mentioned, but not enough time is dedicated to evaluating this criterion"* (Nick)
- *"The idea is to guarantee that when I request a modification, then I go to the generic form, make the changes and they apply to everyone, in this way I apply some maintainability and code reusability criteria"* (Dan)
- *"The organisation has two procedures: mature products which are in the market (maintainability), and new products (usability and functionality). Maybe if we differentiate the product focus, likewise we have the product evaluation perception"* (James)

'Kyra' understands the software quality characteristics and she states that *"they define characteristics such as maintainability, functionality, portability, security, and usability. This means that the non-functional requirements are identified"*

'Angel', 'Phillip', and 'Naty' do not understand software quality characteristics and they are implementing different characteristics according to their context and needs:

- *"Usability, portability, security, functionality, and maintainability criteria and measures are important, and I believe we should not be excluding. One thing does not exclude the other"* (Angel)
- *"In usability, maintainability, security and portability terms, we evaluate some criteria which are reported in the 'Redmine' tool"* (Phillip)
- *"As for non-functional tests (usability, portability, security, etc.), let's say that these aspects are implicit in the product development, that is to say for all the equipment it is clear that the system that we are going to deliver must comply with minimum requirements, i.e., adaptable, usable, etc"* (Naty)

'Kate' is focusing on portability and she says that *"they have tools to measure portability, but it is not 100% covered. However, they have products which can be installed on a computer and deploy applications in the cloud"*.

Even though 'Julien' argues that *"in security criteria, he knows that the software has been including cybersecurity aspects. However, there are functionalities of the applications that are not covered by security criteria"*. 'Maria' says that *"they are not meeting 100% of all quality criteria, but it is more oriented to the customer's need. For instance, there are customers who prefer to sacrifice performance, but they want to be sure that applications will not have certain vulnerabilities"*.

In addition, 'Josh' and 'Stiven' are focused on the applications' security and in certain development parameters respectively as follows:

- *"I am working in software architecture, coding, and products development. We handle quality from the security of applications. Sessions and SQL requests, which are secure, and access to Web Services (through encryption)"* (Josh)
- *"We have incorporated software product line concepts. We development in .Net, but we do that with certain parameters, then the security, integration, access to database and usability aspects are made based on these concepts"* (Stiven)

Finally, some participants evaluate usability:

- *"Quality is evaluated in terms of usability (user interface) and maintainability"* (Kate)
- *"We are very interested in measuring usability since it is fundamental, but we must migrate to the user experience issue"* (Stiven)
- *"We have addressed usability and maintainability criteria, which have been more focused on aspects as interface and user experience"* (Julien)

In summary, some participants have focused on maintenance from different perspectives, but mainly on process quality. The results demonstrate that the participants are trying to measure different non-functional requirements or software quality characteristics according to the project scope, context, and user requirements/needs. Pattern K reveals that software development companies need to pay attention to defining which key SPQ characteristics need to be focused on.

6.1.10 Pattern L - The Quality Environment is Not Well

Context for the Quality Environment is Not Well

Companies are concerned about their reputation, quality, and customer satisfaction and these criteria are critical to the continuity of an organization in the market and financial stability over time (Febrero et al. 2016, Rodriguez et al. 2016). If a customer receives a product of lower quality, he or she can create a scenario of nonconformity with the community, generating the enterprise loss of value in the market and less customer's reliability in its products and services. Therefore, companies need to establish new strategies and show their strengths for improving such problems related to documentation, design, and implementation of quality criteria that include characteristics, measures, and indicators (Carvalho et al. 2009, Gasca et al. 2013, Metaute & Serna 2016, Pelaez et al. 2011).

In addition, end-users need to generate a guideline with the quality criteria that they would like to see in their products. It helps the software development teams and companies to know which best practices and criteria they need to pay attention to during the development (Gasca et al. 2013).

It is essential that companies pay attention to key aspects such as measures, characteristics, indicators, documentation, and automation tests rather than manual tests, which have been a concern to software development teams for improving the development process and product quality (Colomo et al. 2011, Garzas et al. 2013, Rodriguez et al. 2016). Furthermore, it is crucial that software development teams be involved in diverse SPQ training to stay up to date on the new technologies, models/frameworks, and learn ways for improving software quality in their products. Finally, this pattern is important to let the population know the current problems and propose some mitigation strategies.

Some studies in the Colombian scenarios argue that software development companies face some quality problems based on limited resources and the time to dedicate to their quality is not enough (Pelaez et al. 2011, p.53). Furthermore, currently, software quality is one of the main concerns of software development companies (Colomo et al. 2011, Pino et al. 2008, Rodriguez et al. 2016).

Meanwhile, more than 50% of software projects still overrun budget because of some factors based on designing new business processes, existing IT infrastructure, and skills of the teams, that include project management, software development, and testing skills

(Kumar et al. 2015, p.20).

In fact, quality is not going well in the software development companies because it is not considered in the development process with objective and measurable criteria that manage to avoid customer dissatisfaction (Febrero et al. 2016, Jacobson et al. 2013, Zelkowitz 2011).

In addition, some authors argue the importance of criteria such as process improvisation, time underestimation, customer satisfaction, development resources, and high-quality levels which have an impact on software product quality, financial burden, and competitiveness in the markets (Ahmed & Capretz 2011, Garzas et al. 2013, Idri et al. 2016, Pelaez et al. 2011).

On the other hand, Febrero et al. explain that *"there are other very important aspects such as customer dissatisfaction and loss of the manufacturer's prestige that can be traced to software product issues"* (Febrero et al. 2016, p.18).

Pattern Description for the Quality Environment is Not Well

The quality environment related to software product quality (SPQ), documentation, quality characteristics and measures, and automatic tests are not working well in young software development companies (see Figure 6.10).

For thirteen of twenty participants, the things that are not doing well in their companies are related to software product quality (SPQ) evaluation, documentation, quality characteristics and measures, and automatic tests. The other seven interviewees made no comments on this pattern. Each participant has different views and statements focused on such aspects.

Some participants are concerned about characteristics, measures, and indicators which are not considered in their software quality plans/processes and they need to be implemented in different stages/areas:

- *"We are not considering other characteristics and measures such as functionality, usability, and security because we have own products and we work in an incremental process way"* (James)
- *"From my experience, it is necessary to implement new measures and indicators that help for identifying errors in the applications quality"* (Kate)

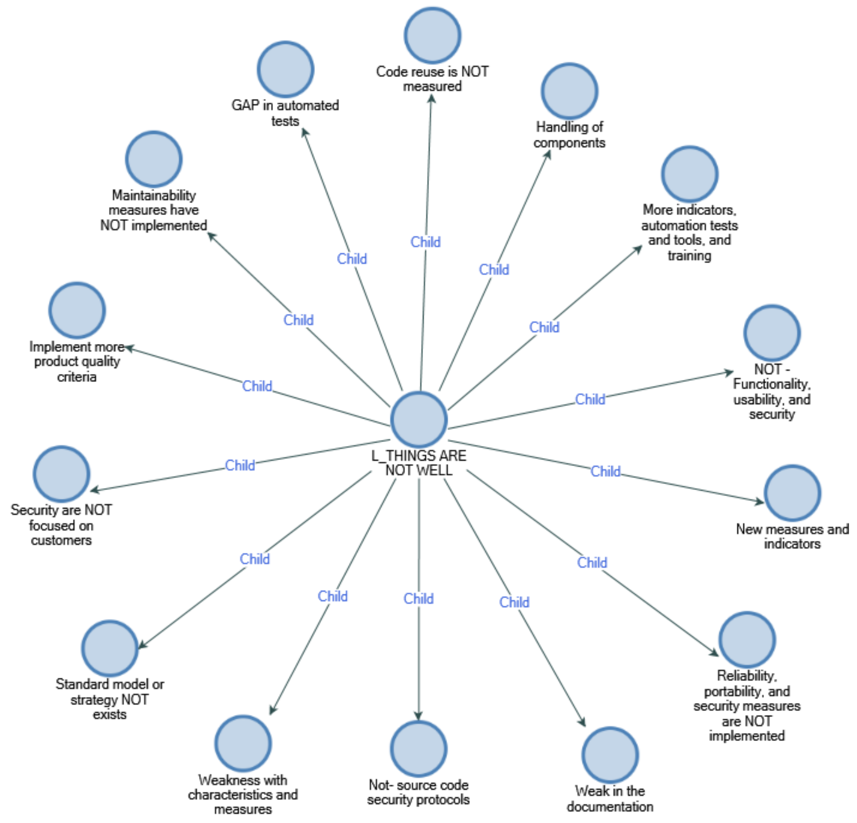

L_THINGS ARE NOT WELL
Code reuse is NOT measured
GAP in automated tests
Handling of components
Maintainability measures have NOT implemented
More indicators, automation tests and tools, and training
Implement more product quality criteria
NOT - Functionality, usability, and security
Security are NOT focused on customers
New measures and indicators
Standard model or strategy NOT exists
Reliability, portability, and security measures are NOT implemented
Weakness with characteristics and measures
Not- source code security protocols
Weak in the documentation
Child

Figure 6.10: The quality environment is not well

- *"The company needs to implement different measures in quality terms (the company still needs to improve and implement many more aspects). We need to measure the handling of components. Let me explain since we have different developers, sometimes the code is a bit personalized (depending on the development methods of each person), we do not have a coding standard (the process of retaking the person's code is difficult, we lose time, our learning curve is much slower). I believe that our measurement process is failing"* (Josh)

- *"The maintainability quality measures (complexity, effectiveness and efficiency), have not been implemented yet. However, everything depends on the project and who is developing, so there is no development standard for both the data model and code documentation (it includes new projects and new developers)". In addition, "I have not much time in the company (only 8 months), but they need to implement more criteria to measure the product quality. The security levels we have are very basic. It requires more strength to improve quality. Furthermore, we must have other criteria* *in mind, invest in training and new tools to measure quality"* (Liz)

- *"Product quality evaluation is important to keep thinking about the product quality, but undoubtedly, we must continue investing in resources such as time, cost and human talent with the aim for improving the quality of our software developments"* (Stiven)
- *"We have not a standard model or strategy that validates the software product quality. Everything is in accordance with customer needs"* (Kyra)
- *"We do not implement security, portability, and reliability measures because it always starts from the customer's requirements and the project dimension. We have had projects, which due to time and resources, it is not necessary to start a specific measurement process"* (Dan)
- *"Reuse measure is not considered. In the company, we have many customers with different approaches and the reuse criterion is complex to implement"*. Moreover, *"the security of applications is not a very important criterion. In general, it falls more on the customer. We deploy the application in the customer's facilities, and the customer is responsible for protecting the application's information (infrastructure). In conclusion, we have not source code security protocols"* (Nick)

In addition, they are working on testing, but focusing on manual tests rather than automated tests:

- *"In the future, I would like to involve more indicators and automate tests. We work with manual tests. We are reviewing automation tools and training processes"* (Naty)
- *"The company has been oriented only to a couple of lines and we have identified that we are weak in others that are equally important. From my viewpoint, all the criteria must be implemented in an integral way"* (Angel)
- *"We have a gap in aspects related to automated tests for the purpose of unit tests. What we do from our quality area is to evolve the product, we have certain functionalities that are established in the documents as a checklist and architecture guidelines"*. In addition, *"throughout this process for improving the way we evaluate quality, we have identified that there are certain factors that require investment in specialized tools and training. For instance, in testing area we have people who have been working with the company for many years, but they are outdated in*

quality aspects, and which tools they can implement to perform automated tests, since measuring quality does not only consist of applying functional tests. They must have more technical skills" (Maria)

- *"The security criteria are not focused on the customer, but on law aspects or aspects that certain industries or customers have"* (Lachlan)

On the other hand, 'Julien' as a developer considers that they are weak in the documentation. He says *"for instance, if we want to check which functionality supports a meter developed three years ago, this information does not exist. Therefore, this information should be raised from the code that was developed. However, with the CMMI process, the shortcomings are closed, but internally it is necessary to improve and strengthen the software documentation."*

To summarise, from the results embedded in the excerpts above, the young software development companies are facing problems associated with software product quality, testing, and documentation. In addition, they all had different views and also would need new measures or indicators to evaluate software product quality (SPQ) and investing in resources and training. This pattern is important for the industry because it helps to know the software quality context in some young companies and give them some advice about what they are not doing well and also so they can compare with each other and check within their companies what happens in software quality aspects.

6.2 An Overall Understanding of the Patterns

The candidate identified that Pattern G *"process quality over product quality"* presents a high number of references (codes from the interview data). It means that 19 out of 20 participants point out that software quality/assurance is dominated by software process quality and not by software product quality (SPQ) in their companies and they do not have well-defined SPQ processes.

Therefore, the candidate decided to define pattern G as a theme and reference all other patterns (A, B, C, E, F, H, I, K, L) to it. A theme or also called 'meaning unit' should be able to take the significant statements from the data and then group them into larger units of information (Alase 2017, Creswell 2013).

This theme is further discussed and developed in Chapter 7.

6.3 Summary

This chapter has presented the patterns and an overall understanding of them. The candidate describes the data analysed and connected everything from the software development companies' lived experiences. The extracts and statements from the participants demonstrate that they are concerned about customer satisfaction throughout the software engineering life-cycle; fulfilling customer requirements during all stages focusing more on functional requirements rather than non-functional requirements reinforcing the lack of focus on software product quality. Furthermore, customizing protocols and methodologies strongly validate that this pattern is a way that software development companies have for evaluating the process or product quality in their projects. Manual and automation tests demonstrate the importance of using them and the need to be asked about why the companies are not passing all their experiences to an automated protocol using technologies or implementing continuous integration (DevOps). Measures and indicators demonstrate that the participants create their new measures and indicators according to their needs and their attention is focused on code and process quality rather than product quality. Process quality over product quality shows that 80% of the interviewees understand that they need to attend to quality in their companies, but are focused only on process quality. Existing Procedures, Best Practices, Guidelines and Policies, and Software quality characteristics show that the companies have their own best practices and have focused on different non-functional requirements for evaluating SPQ according to the project scope, context, and user needs. Through the project management skills pattern this research demonstrates that companies need to work on a dynamic process to have efficient solutions for the customer. The pattern 'the quality environment is not well' shows that the young software development companies are facing problems associated with SPQ, testing, and documentation. Finally, everything above leads to the conclusion that software process quality is more important to software companies than software product quality. Therefore, the candidate decided to define the pattern '*process quality over product quality*' as a theme and reference all other patterns to it. The theme will be developed in Chapter 7.

Chapter 7

Discussion

As demonstrated in the literature review, software product quality must be an essential criterion for evaluating software quality from the early stages of the software engineering life cycle and it needs to consider some attributes/measures related to documentation and best practices. However, it has been found that some companies who participated in this research do not understand the difference between software process quality and software product quality (SPQ) because they cannot identify a clear concept nor how they can apply SPQ in their companies. Moreover, this problem needs to be fixed, by creating new training programs to strengthen the companies' capacities, providing an inexpensive method to evaluate SPQ, and creating communities of practice to share their knowledge and experiences.

7.1 Pattern Discussion

Pattern G - Process quality over product quality, section 6.1.6 was found to be a very strong pattern, and strongly related to all other patterns [customer satisfaction (A); fulfilling customers' functional requirements (B); customized protocols and methodologies (C); manual and automation tests (E); measures and indicators evaluated (F); existing procedures, best practices and policies (H); project management skills (I); software quality characteristics (K); and the quality environment is not well (L)].

Other patterns are connected to Pattern G because they reinforce the confusion between process quality and product quality. In addition, such patterns are connected to the main theme because the data and key descriptions, for example, demonstrate that they need each other to evaluate SPQ in order to achieve business requirements and customer satisfaction.

7.1.1 Relating Patterns to Software Product Quality

All patterns discovered during this research are crucial to evaluate software product quality (SPQ) in any young software development company in Colombia and worldwide (Nakai et al. 2016). Also, those patterns are associated with an essential theme *'process quality over product quality'*. The main criteria to take into account is the clear concept and differences between process quality and product quality. Companies need to generate a culture about this terminology in their software development teams. Customer satisfaction and fulfilling customer requirements are essential from the beginning of the software engineering life-cycle to understand all end-user requirements and needs. Customized protocols and methodologies help to figure out how the companies are evaluating SPQ, using some procedures or models to do so. Here, it is important to know which guidelines and policies are used during the SPQ evaluation process for identifying their best practices and procedures. In addition, manual and automation tests are necessary to verify and validate the product state during the life-cycle. SPQ characteristics and measures/indicators are crucial to measuring SPQ for different non-functional requirements associated with the product. Project management skills are fundamentals in all software projects for tracking their scopes, objectives, work-plan, and resources in terms to achieve the evaluation process and a good final product for the customers. Finally, all companies in this research do not have the right quality environment and they are presenting different issues related to software product quality, which must be fixed before launching the product to the end-user.

Customer satisfaction needs to be included as another criterion to evaluate SPQ and needs to be checked iteratively during the development stages. It helps to address customer satisfaction early on from each functional and non-functional requirement. Quality is directly related to fulfilling customer requirements. The verification and validation process is correlated with customer needs/issues and achieving client requirements. The absence of any mention of non-functional requirements from study participants suggests that their focus is on functional requirements only and for most companies, it reinforces the need to focus on software product quality (*called non-functional requirements*).

Customized protocols and methodologies strongly validate that customizing protocols and methodologies is a way that software development companies have for evaluating the process or product quality in their projects. It is an essential pattern and this research needs to provide some strategies or projects for focusing SPQ evaluation in a standard protocol, model or framework, which allows creating the same environment, but with flexible measures and procedures. Furthermore, manual and automation tests describe the importance of such tests during software development projects, and how they are implemented in different young software companies.

This research found that companies still have guidelines, policies, and their own best practices for evaluating software quality according to the project scope, context, and user requirements/needs. However, they need to reinforce the idea to create a good framework or standard to follow steps/stages according to the development process and quality criteria. It could be useful to investigate whether they follow different strategies and workflows to implement project management skills in their employees and software projects. These patterns are important to complement the SPQ evaluation process from the early stages of the software development life cycle in young companies. In addition, software quality characteristics are a clear pattern that software development companies need to pay attention to by defining which key characteristics need to be focused on and related those measures defined. Measures and indicators are created according to the company's needs. However, companies are interested in evaluating software quality without, it would appear, any attention to product quality because most of them are focused on code quality and process quality.

The pattern L "the quality environment is not well" shows that young software development companies in Colombia appear to be facing problems associated with software product quality, testing, and documentation. Moreover, they need new measures or indicators to evaluate SPQ and investing in resources and training.

A pattern emerged from the interview data; process quality over product quality that is achieved for companies customized methodologies, procedures, and CMMI process. However, companies do not have well-defined product quality processes.

In summary, interviewees understand that they need quality in their companies, but they are focusing on process quality. The candidate argues that product quality must be an essential criterion for evaluating from the early stages of the software engineering life cycle and it needs to consider some attributes/measures related to documentation and best practices. To achieve this, companies could standardize processes and be strict in some criteria of continuous integration and deployments.

7.2 The Theme "Process Quality over Product Quality"

As defined in the literature review (see Table 5.5), a theme is a dominant pattern, characterised by a large number of nodes in itself and with connections to other patterns.

The dominant theme from all the patterns found in this research, is that companies are focused on process quality rather than product quality. Some companies have customized methodologies, procedures, and CMMI processes. However, their product quality processes as not well defined. In addition, it was found that some software development teams did not understand the difference between software process quality and software product quality (SPQ).

This is a clear conclusion from examining Pattern G (Process Quality over Product Quality). Nineteen out of twenty participant companies focused on process rather than product quality. Furthermore, 12 out of 20 participants are confusing software product quality (SPQ) with software quality (SQ). For instance, 'Karol' argues that they *"try to ensure the quality from the requirements stage. Checklists are made to review from this stage some features to meet the usability criteria, in technical issues, criteria such as performance and the completeness of the entire requirements from all points of view"* and 'Angel' states that *"there are normal or known stages in the project execution. From the requirements stage, we ensure that the requirements established with the customer are clear. We handle standard formats, accompanying experts in meeting requirements (to reduce the information duality and have greater requirements clarity)"*. Those are clear statements about confusion between SPQ and SQ.

When the candidate re-examined the other patterns, he found that many of them substantially reinforced Pattern G "process quality over product quality". The candidate 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. This is visualised in Figure 7.1.

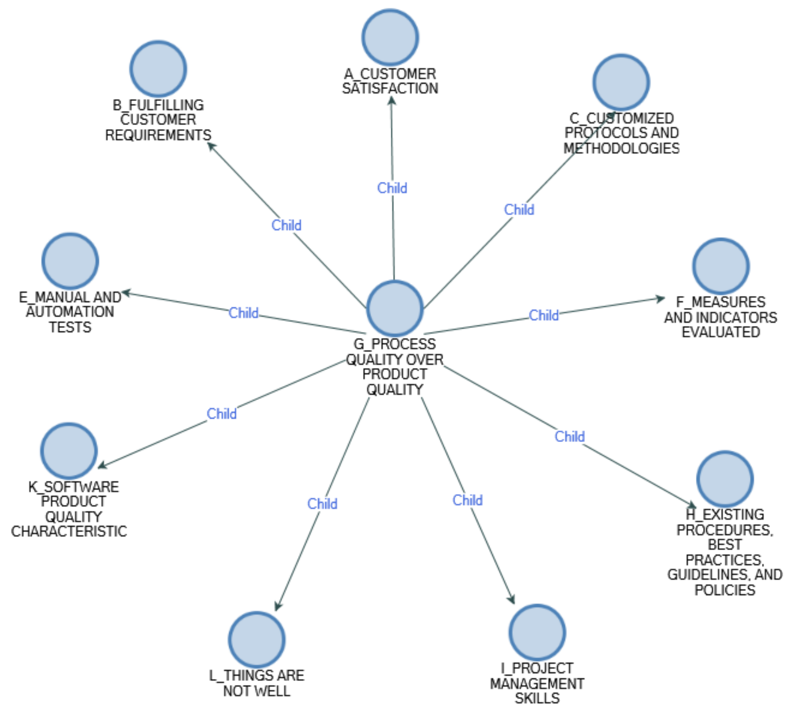

A_CUSTOMER SATISFACTION
B_FULFILLING CUSTOMER REQUIREMENTS
C_CUSTOMIZED PROTOCOLS AND METHODOLOGIES
E_MANUAL AND AUTOMATION TESTS
F_MEASURES AND INDICATORS EVALUATED
G_PROCESS QUALITY OVER PRODUCT QUALITY
K_SOFTWARE PRODUCT QUALITY CHARACTERISTIC
H_EXISTING PROCEDURES, BEST PRACTICES, GUIDELINES, AND POLICIES
L_THINGS ARE NOT WELL
I_PROJECT MANAGEMENT SKILLS
Child

Figure 7.1: Theme as Related to Patterns

The following patterns reinforce that participants are focusing on SQ over SPQ. The pattern names and their relevance to process quality over product quality are given below. Table 6.1 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 for customer satisfaction, such as usability, confidence, and acceptance criteria. Customer satisfaction is a product quality issue, not a process quality concern. The fact that they did not mention SPQ measures for customer satisfaction highlights the claim that they are focusing on process quality.

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

Pattern L (The quality environment is 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.

Thus, the evidence from this research is clearly very strong that the companies studied are focused on software process quality (SQ) over software product quality (SPQ). In addition, the evidence of confusing SQ with SPQ is that companies talk about SQ when asked about SPQ. For instance, already discussed in pattern G, when the candidate asked about software product quality, some participants' responses were:

"We are focusing on process quality. The process to control the logs or bugs is in implementation (because we do not have measures). The company has generated indicators slowly".

"With the CMMI process, the documentary shortcomings are closing, but internally

it is necessary to improve and strengthen the software documentation. The focus is on process rather than product quality".

"There are normal or known stages in the project execution. From the requirements stage, we ensure that the requirements established with the customer are clear. We handle standard formats, accompanying experts in meeting requirements (to reduce the information duality and have greater requirements clarity)".

7.3 Consequences and Benefits

The most important consequence of this research is that Colombian companies are not able to take advantage of the advances in software product quality because they do not understand that software product quality is different from software process quality.

The benefits of this research are that the candidate has gained an understanding of this consequence and can deliver guidelines, education and community groups that can be set up to improve this situation.

Below additional consequences and benefits for young software development companies in Colombia related to software product quality (SPQ) evaluation are stated. All of these consequences have the benefit that the insight gained from each consequence will help to develop corrective action to aid Colombian software companies. These key ideas come from each pattern defined and described in Chapter 6:

- Developers, testers, and project managers are concerned about customer satisfaction during all software engineering phases, but they do not have proper measures for evaluating customer satisfaction based on software product quality (SPQ). Companies need to focus on SPQ measures to ensure customer satisfaction.

- Companies are concerned about fulfilling customer requirements during all stages of software engineering, but this concern is focused on functional requirements instead of product quality or non-functional requirements. Developers need to establish a reviewing process iteratively for each SPQ criteria during the development stages. This process will help to address and understand business requirements and customer needs early reducing time, cost, and resources invested.

- Customized protocols and methodologies for evaluating the process or product quality are used by companies in their projects. It is a best practice for those companies, but they are still confusing technical concepts such as measures, indicators, and characteristics, which leads to misunderstandings about SPQ evaluation. Therefore, a standard and economical model is required for having an environment with flexible measures and procedures and avoiding such confusion.

- People continue to focus on process quality and not product quality. It highlights the necessity of a training education program for specific companies to understand the differences between software product quality (SPQ) and software process quality (SQ).
- Project management skills are adopted by companies for planning and organizing the resources during the software development process. A dynamic process needs to be defined for having efficient solutions for the customers and establishing clear test plans with well-defined scopes and strategies.
- Companies have focused on maintenance from different perspectives based on process quality. They are trying to measure some non-functional requirements or software quality characteristics according to the project scope, context, and user requirements/needs. They need to define which key software product quality (SPQ) characteristics need to be focused on.

7.4 Summary

This chapter has discussed my thesis in terms of how young software development companies evaluate SPQ. The perspectives from pattern discussion and the theme that were chosen state the importance of product quality and how companies are still working and planning based on process quality. All the findings were discovered from developers' lived experiences based on phenomenological research. The candidate has described consequences and benefits for the stakeholders involved in this research with the aim to improve SPQ evaluation and increase productivity and competitiveness in the young software development companies. Together these findings present a fitting conclusion to my phenomenological research cycles. For a conclusion to my thesis as a whole, we turn now to the final chapter.

Chapter 8

CONCLUSIONS AND FUTURE WORK

In this thesis, the candidate presents a study aimed at understanding how young software development companies in Colombia 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 & Wassyng (2008): *"... the software quality evaluations should be based on direct evidence about the product, not only on evidence about the process"*.

The candidate used a phenomenological research methodology to investigate participants' experiences in software quality with the aim of improving SPQ evaluation and increasing the productivity and competitiveness in the young software development companies.

8.1 Contributions to Field

This thesis has made the following contributions to the field.

1. The most important contribution is that young software companies in Colombia currently focus more on process quality than on software product quality (SPQ). Moreover, software developers in these companies currently 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 related to the company revenue, engineering life-cycle, HR productivity, time to market, quality assurance of these companies' software products, and their competitiveness and success.

2. Second, five SPQ characteristics are used by the participant Colombian companies. Those companies are using functionality, usability, maintainability, security, and portability. As mentioned previously, SPQs are often difficult and expensive to measure. However, four companies have managed to implement SPQ measures, despite the obstacles. These four companies provide a starting point to assist other companies to implement SPQ measures.

3. Third, this thesis demonstrates a deep empirical understanding of software product quality (SPQ), how SPQ is evaluated in a young software development company, and which measures are applicable and suitable according to the lived experiences of software practitioners. It is important to industry and government because they lack a research/report where they can present clear and useful information about how young companies are evaluating SPQ for contributing to their competitiveness and productivity.

8.2 Limitations

Although this study presents an empirical understanding of software product quality (SPQ), 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:

- The phenomenological research design
- The seven participants companies are only young Colombian software development companies (2-5 years old)
- 20 interviews were carried out with individual participants from these companies
- Participants are all software developers, testers, and project managers, each with a minimum of 2 years' experience in a software engineering role
- The interviews were semi-structured, consisting of four open-ended questions

- The interviews were conducted in Spanish via video-conferencing tools, and after those were translated to English
- Thematic analysis was performed to understand how these participants and their companies evaluate software product quality
- The candidate's background as both an experienced professional software engineer and a teacher in software engineering for many years

8.3 Industry Adoption

The objective throughout this thesis has been a focus on how young software development companies evaluate software product quality (SPQ) and which measures they are using to do so that should be applicable to and adopted by industry (see section 3.2). It has successfully achieved this goal. This is evidenced by the qualitative research design (see section 3.6) and research findings (see chapter 6).

This research could be adopted by the industry for understanding how they are evaluating software product quality, which non-functional requirements/quality attributes are the most important criteria in the young companies and which measures they need to implement for improving the quality of their products.

This research could be replicated with other significant samples of young software development companies to do comparative studies on software quality evaluated and implemented in a similar developing country with a low rate of exportation.

The candidate was intrigued by the idea that the industry could develop with all participants an economical and flexible model to evaluate SPQ with the essential criteria to measure the product quality and improve the competitiveness for young software development companies.

8.4 Challenges

8.4.1 Lack of Standardisation

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. This limits the ability of young Colombian companies, because of their size and experience, to implement measurement models and practices of software product quality.

Industry consensus takes time. At the time of writing, there is not enough consensus to justify proposing a standard as ISO for young companies. Standardization requires broad agreement and, ideally, multiple mature and competing implementations. The teams behind such implementations can then come together, identify their commonalities, and work to generate them. It is premature to pursue standardization before such a broad consensus has been achieved. However, with corporations such as ISO promoting standardization, the candidate believes it is only a matter of time.

8.4.2 Lack of Education in SPQs

The misunderstandings of software process quality and software product quality will require education. The form of the education is a real challenge for developers, as they need to change the way they see things, not just learn new techniques.

However, the suggestions in this thesis that companies can form communities to share SPQ experiences, in the light of current standards, may be an immediate, practical way forward, as discussed in the first recommendation below.

8.5 Recommendations

In order to respond to the two challenges above, the following recommendations are made.

- Companies should develop an economical and flexible model for evaluating software product quality for young companies. It needs to include non-functional requirements, attributes, measures, and clear procedures for evaluating and measuring product quality. In general, this type of model strives to integrate all lived experiences and needs from software development companies about software quality. The candidate recommends developing a consensus between different stakeholders (industry, government, users) to define and establish the key criteria for evaluating SPQ and finding a way to be competitive and increase the software exportation level by young companies.
- Companies could create communities of practice (Wenger 1998) to share their approaches or frameworks (i.e. models, protocols, procedures, and methodologies),

which they have built or they are building for assessing SPQ with help from experts, academia, and industry. If the companies share their knowledge, SMEs probably could compete and be recognized worldwide.

- Companies need to relate SPQ to each stage of software development projects and it needs to be assessed according to the requirements and customer perceptions for generating value, security, and confidence in the product developed.
- Companies need to pay attention to SPQ (notably security) of the software products.
- Developers need to review iteratively each SPQ criteria during the development stages. It helps to address business requirements and customer needs early on from each functional and non-functional requirement.
- End-users could generate a thematic guideline by industry (i.e. finance, services, health, technology, manufacturing, education, etc.) for recording best practices about how they would like their software products evaluated according to non-functional requirements.
- Companies can implement software development practices and information-technology operations (DevOps) to shorten the system-development life cycle while delivering SPQ evaluation protocols with characteristics and measures that help to alignment with the business goals.
- Software development companies need to establish baseline measures for young companies and keep updating those according to the project scope, context, and customer requirements. Thereby, all companies can integrate through a general goal related to software quality.
- Academia and industry could work together the purpose of creating and adopting new and better measures to increase the productivity and competitiveness of young software development companies in Colombia.
- The government should invest more resources creating new training programs to strengthen the companies' capacities and generate more interaction among academia and industry.
- Industry needs to invest more resources for solving SPQ issues in the market and developing automatic testing to assess the quality of software development projects.

- SMEs need to invest time and resources in training for their software development teams. It helps to reduce their workload, time, and product returning from customers, so all teams understand how to manage a project, which best practices (protocols, characteristics, and measures) they need to use for evaluating SPQ, and which techniques implementing to develop better quality software.
- Standards institutions could provide an inexpensive method to evaluate SPQ.

8.6 Future Work

In future work, the candidate plans to:

- Propose an economical and flexible model for evaluating SPQ in young software development companies. The aims of the model are to help reduce their workload, time, and resources in order to make SPQ assurance more economical.
- 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 build new understandings 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.
- Work with end-users to generate a thematic guideline by industry (i.e. finance, services, health, information technology, manufacturing, education, etc.) for recording best practices on how they should like their software products to be evaluated according to non-functional requirements.
- Create a company for validating and verifying software product quality based on the economical and flexible model to be developed.

To summarize, the main finding of the thesis is that young software development companies in Colombia focus more on process quality than software product quality (SPQ) or non-functional requirements.

APPENDIX A

SURVEY - QUANTITATIVE DESIGN (QTD)

IDENTIFICATION OF A SET OF QUALITY MEASURES APPROPRIATE FOR SOFTWARE PRODUCTS IN COLOMBIA
FACULTY OF ENGINEERING AND INFORMATION TECHNOLOGY
SCHOOL OF SOFTWARE
SYDNEY, AUSTRALIA

I consent to participate in this research.
( ) Yes (go to the next question)
( ) No (go at the end of the survey)

SECTION I: MEASURING OF THE SOFTWARE PRODUCT QUALITY

Software development is becoming an international business and Colombian software developers soon must compete against international developers who want to enter the Colombian market. In order to compete, the quality of Colombian software must be of a similar or better quality, and to achieve that it is necessary to know which quality characteristics matter and how to measure them. This research will investigate which quality characteristics are important to Colombian industry and determine some suitable methods to measure those characteristics.

The software quality is reflected or measured by the degree to which a product conforms to its specifications or requirements. Therefore, for this research is important to know which characteristics, measures and measurement methods software development companies in Colombia use to evaluate the product quality.

1. Does your company measure quality by counting the number of defects and their severity, or by measuring some specific quality characteristics? Defects only/Specific quality characteristics.
   ( ) Yes
   ( ) No
   If defects only, go to question 12.

2. If you do not use characteristics, measures and measurement methods, tell us how you evaluate the software product quality and which method(s) do you use?
   Describe here

3. Does your company evaluate any of the following software quality characteristics?
   (Frequency: Never, Rarely, Occasionally, Frequently, Very Frequently)

No	Characteristic	Frequency
1	Functional suitability	Choose an item
2	Performance efficiency	Choose an item
3	Compatibility	Choose an item
4	Usability	Choose an item
5	Reliability	Choose an item
6	Security	Choose an item
7	Maintainability	Choose an item
8	Portability	Choose an item
9	Other(s): *(Please answer Question 12)*	

SECTION II: MEASURES AND MEASUREMENT METHOD TO EVALUATE THE SOFTWARE PRODUCT QUALITY

For this research, a measurement function means a mathematical formula, which shows how the quality measurement elements are combined to produce the quality measure (e.g. X=A/B). The measurement methods, are operations mapping an attribute to a scale (e.g. count number of lines and count number of defects). Thus, it is essential to know which measures your company uses to evaluate different characteristics and how they are measured.

1. If you evaluate Functional suitability, which measures do you use?
   (Frequency: Never, Rarely, Occasionally, Frequently, Very Frequently)
   Functional suitability: used to assess the degree to which a product provides functions that meet stated and implied needs when used under specified conditions.

Measure	**Frequency**	**Measure method**	**Measure function**	**Other *(which?)***
Functional coverage	Choose an item			
Functional correctness	Choose an item			
Functional appropriateness of usage objective	Choose an item			
Functional appropriateness of system	Choose an item			
Other(s):	Choose an item			

2. If you evaluate performance efficiency, which measures do you use?
   Performance efficiency: used to assess the performance relative to the amount of resources used under stated conditions. Resources can include other software products, the software and hardware configuration of the system, and materials.

Measure	**Frequency**	**Measure method**	**Measure function**	**Other *(which?)***
Mean response time	Choose an item			
Response time adequacy	Choose an item			
Mean turnaround time	Choose an item			
Mean processor utilization	Choose an item			
Mean memory utilization	Choose an item			
User access capacity	Choose an item			
User access increase adequacy	Choose an item			
Other(s):	Choose an item			

3. If you evaluate compatibility, which measures do you use?
   Compatibility: used to assess the degree to which a product can exchange infor-

mation with other products, systems or components, and/or perform its required functions, while sharing the same hardware or software environment.

Measure	**Frequency**	**Measure method**	**Measure function**	**Other *(which?)***
Co-existence with other products	Choose an item			
Data formats exchangeability	Choose an item			
Data exchange protocol sufficiency	Choose an item			
External interface adequacy	Choose an item			
Other(s):	Choose an item			

4. If you evaluate usability, which measures do you use?
   Usability: used to assess the degree to which a product or system can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.

Measure	**Frequency**	**Measure method**	**Measure function**	**Other *(which?)***
Description completeness	Choose an item			
User guidance completeness	Choose an item			
Error messages understandability	Choose an item			
Operational consistency	Choose an item			
User interface customizability	Choose an item			
Appearance consistency	Choose an item			
User entry error correction	Choose an item			
Appearance aesthetics of user interfaces	Choose an item			
Accessibility for users with disabilities	Choose an item			
Other(s):	Choose an item			

5. If you evaluate reliability, which measures do you use?
   Reliability: used to assess the degree to which a product performs specified functions under specified conditions for a specified time-period.

Measure	**Frequency**	**Measure method**	**Measure function**	**Other *(which?)***
Fault correction	Choose an item			
Test coverage	Choose an item			
System availability	Choose an item			
Redundancy of components	Choose an item			
Mean fault notification time	Choose an item			
Mean recovery time	Choose an item			
Backup data completeness	Choose an item			
Other(s):	Choose an item			

6. If you evaluate security, which measures do you use?
   Security: used to assess the degree to which a product or system protects information and data so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorization.

Measure	**Frequency**	**Measure method**	**Measure function**	**Other *(which?)***
Access controllability	Choose an item			
Data encryption correctness	Choose an item			
Data integrity	Choose an item			
Internal data corruption prevention	Choose an item			
Digital signature usage	Choose an item			
User audit trail completeness	Choose an item			
System log retention	Choose an item			
Authentication mechanism sufficiency	Choose an item			
Authentication rules conformity	Choose an item			
Other(s):	Choose an item			

7. If you evaluate maintainability, which measures do you use?
   Maintainability: used to assess the degree of effectiveness and efficiency with which a product or system can be modified by the intended maintainers.

Measure	**Frequency**	**Measure method**	**Measure function**	**Other *(which?)***
Coupling of components	Choose an item			
Cyclomatic complexity adequacy	Choose an item			
Reusability of assets	Choose an item			
Coding rules conformity	Choose an item			
System log completeness	Choose an item			
Modification efficiency	Choose an item			
Modification capability	Choose an item			
Test function completeness	Choose an item			
Test restartability	Choose an item			
Other(s):	Choose an item			

8. If you evaluate portability, which measures do you use?
   Portability: used to assess the degree of effectiveness and efficiency with which a product can be transferred from one hardware, software or other operational or usage environment to another.

Measure	**Frequency**	**Measure method**	**Measure function**	**Other *(which?)***
Hardware environmental adaptability	Choose an item			
System software environmental	Choose an item			
Operational environmental adaptability	Choose an item			
Installation time efficiency	Choose an item			
Ease of installation	Choose an item			
Usage similarity	Choose an item			
Other(s):	Choose an item			

9. If you answered "Other" to question 3, please answer this question:

Measure	**Frequency**	**Measure method**	**Measure function**	**Other *(which?)***
	Choose an item			
	Choose an item			
	Choose an item			
	Choose an item			

SECTION III: SOFTWARE TOOLS TO MEASURE THE SOFTWARE PRODUCT QUALITY AND CERTIFICATIONS ACHIEVED

1. Gathering quality information can be time consuming and expensive if it is done manually. However, automated measurement requires tools that must be configured to provide information about software quality. We would like to know whether you gather quality information automatically or manually.

   Do you evaluate the software product quality by applying manual or automatic software tools?

   ( ) Manual *(go to Question 2)*
   ( ) Automatic *(go to Question 3)*

2. If you do apply manual tools, please list these in the following table and include the use frequency:

Tool	**Frequency**
	Choose an item
	Choose an item
	Choose an item
	Choose an item

3. Below are listed some automatic tools which could be used by software development companies to measure the software product quality. If you do apply automatic tools, please choose the use frequency and include more if you have implemented:

Tool	**Frequency**
AdaQuest	Choose an item
Cheekpoint	Choose an item
Inspector	Choose an item
Q/Audito	Choose an item
TestGen	Choose an item
SQAmanager	Choose an item
XRunner	Choose an item
JUnit	Choose an item
FindBugs	Choose an item
NUnit	Choose an item
Sonar	Choose an item
Jenkins	Choose an item
pbUnit	Choose an item
jBehave	Choose an item
Other(s):	Choose an item

4. Finally, it is important to know if your company achieved any of the following certifications after the measurement of software products quality (more than one box may be checked). If yes, please indicate below:
   ( ) ISO/IEC 14598
   ( ) ISO/IEC 9126
   ( ) ISO/IEC 25000
   ( ) None
   ( ) Other (which?):

CLASSIFICATION SECTION

This section has a series of questions concerning the participant's and companies' demographic or socio-economic characteristics such as organisation size, type of organisation, market, organisation age, and participant's experience in IT or their particular field. For this research is important to know this information because we need to identify different response categories and the relevance of this topic to the Colombian software companies.

Criteria	**Response**
Level *(Manager, developer, tester)*:	
Company name:	
Company size *(Small, medium, large)*:	
Year established:	
Market *(Local, national, international)*:	
City/Country:	

END OF SURVEY

APPENDIX B

CONSENT FORM (QTD)

IDENTIFICATION OF A SET OF QUALITY MEASURES APPROPRIATE FOR SOFTWARE PRODUCTS IN COLOMBIA (ETH17-1576)

I *[participant's name]* agree to participate in the research project to identification of a set of quality measures appropriate for software products in Colombia and ETH17-1576 being conducted by *Wilder Perdomo*, Wilder.Perdomo@uts.edu.au, phone: (+61) [redacted]. I understand that funding for this research has been provided by COLFUTURO Colombia and University of Technology Sydney.

I have read the Participant Information Sheet or someone has read it to me in a language that I understand.

I understand the purposes, procedures and risks of the research as described in the Participant Information Sheet.

I have had an opportunity to ask questions and I am satisfied with the answers I have received.

I freely agree to participate in this research project as described and understand that I am free to withdraw at any time without affecting my relationship with the researchers or the University of Technology Sydney.

I understand that I will be given a signed copy of this document to keep.

I agree to be: ( ) Audio recorded | ( ) Video recorded

I agree to keep confidential all information including all conversations and discussions, materials and methods provided to me by the UTS research team.

I agree that the researcher could get my permission if any published paper or thesis wants to quote something I have said. In addition, whether there are any additional use of data in future research projects: ( ) Yes | ( ) No

I am aware that I can contact Wilder Perdomo if I have any concerns about the research.

Name and Signature [participant]	**Date (dd/mm/yy)**

Name and Signature [researcher or delegate]	**Date (dd/mm/yy)**

NOTE: *This study has been approved by the University of Technology Sydney Human Research Ethics Committee [UTS HREC]. If you have any concerns or complaints about any aspect of the conduct of this research, please contact the Ethics Secretariat on ph.: +61 2 9514 2478 or email: Research.Ethics@uts.edu.au, and quote the UTS HREC reference number. Any matter raised will be treated confidentially, investigated and you will be informed of the outcome.*

APPENDIX C

Invitation Letter (QtD)

IDENTIFICATION OF A SET OF QUALITY MEASURES APPROPRIATE FOR SOFTWARE PRODUCTS IN COLOMBIA

Dear *[participant's name]*

My name is *Wilder Perdomo Charry* and I am a student at the University of Technology, Sydney.

I am conducting a research to identify a set of quality measures and methods appropriate for software products in Colombia and would welcome your participation. The research will involve responding to a series of semi-structured questions and should not take more than 20 minutes of your time. I have asked you to participate because I know of your experience and knowledge in the software industry, as a manager, developer or evaluator of software products.

Software development is becoming an international business and Colombian software developers must compete against international developers who want to enter the Colombian market. In order to compete the quality of Colombian software must be of similar or better quality and to achieve that it is necessary to know which quality characteristics matter and how to measure them. This research will investigate which quality characteristics are important to Colombian industry and determine some suitable methods to measure those characteristics.

This research has been funded by COLFUTURO Colombia and the University of Technology Sydney for my research studies in Computing Sciences.

If you are interested in participating, I would be glad if you would contact me Wilder.Perdomo@uts.edu.au / (+61) [redacted].

You are under no obligation to participate in this research.

Yours sincerely,

Wilder Perdomo Charry
15 Broadway Ultimo NSW 2007
Phone: +61 295142000
wilder.perdomo@uts.edu.au

NOTE: *This study has been approved by the University of Technology, Sydney Human Research Ethics Committee. If you have any complaints or reservations about any aspect of your participation in this research which you cannot resolve with the researcher, you may contact the Ethics Committee through the Research Ethics Officer (ph: +61 2 9514 2478 Research.Ethics@uts.edu.au), and quote the UTS HREC reference number. Any complaint you make will be treated in confidence and investigated fully and you will be informed of the outcome.*

APPENDIX D

PARTICIPANT INFORMATION SHEET (QTD)

IDENTIFICATION OF A SET OF QUALITY MEASURES APPROPRIATE FOR - SOFTWARE PRODUCTS IN COLOMBIA (ETH17-1576)

WHO IS DOING THE RESEARCH?

My name is *Wilder Perdomo Charry* and I am a student at UTS.

WHAT IS THIS RESEARCH ABOUT?

This research is to find out about which software quality characteristics, measures and methods are important for Colombian organisations, and which of these characteristics the Colombian software developer must measure.

FUNDING

Funding for this project has been received from COLFUTURO Colombia and the University of Technology Sydney.

WHY HAVE I BEEN ASKED?

You have been invited to participate in this study because your experience and knowledge in the software industry, as a director, developer or evaluator of software products. Your contact details were obtained from ICT Ministry and Colombian Software Federation (Fedesoft).

IF I SAY YES, WHAT WILL IT INVOLVE?

If you decide to participate, I will invite you to answer the questionnaire that will take approximately 20 minutes to complete.

ARE THERE ANY RISKS/INCONVENIENCE?

Yes, there are some risks/inconvenience. They are:

- You may be asked sensitive questions for your company
- Concern for your privacy or confidentiality. Please note that in most cases, participants cannot be anonymous unless they are completely unidentifiable to the research at the point of data collection. Once data has been de-identified, data is classed as confidential
- Inconvenience for participating (you do not time, etc.)
- Burden or commitment i time

DO I HAVE TO SAY YES?

Participation in this study is voluntary. It is completely up to you whether or not you decide to take part.

WHAT WILL HAPPEN IF I SAY NO?

If you decide not to participate, it will not affect your relationship with the researchers or the University of Technology Sydney. If you wish to withdraw from the study once it has started, you can do so at any time without having to give a reason, by contacting Wilder.Perdomo@uts.edu.au.

If you decide to leave the research project, we will not collect additional personal information from you, although personal information already collected will be retained to ensure that the results of the research project can be measured properly and to comply with law. You should be aware that data collected up to the time you withdraw will form part of the research project results.

CONFIDENTIALITY

By signing the consent form, you consent to the research team collecting and using personal information about you for the research project. All this information will be treated confidentially. Only the research team will know the participants' information, which will be kept confidential in the Survey Monkey tool, online repository (Dropbox) and personal computer provided by the University. Your information will only be used for the purpose of this research project.

We plan to discuss and publish the results. The research outcomes will be released to COLFUTURO and University of Technology Sydney and will be published in some recognised Journal in the area. In any publication, information will be provided in such a way that you cannot be identified.

WHAT IF I HAVE CONCERNS OR A COMPLAINT?

If you have concerns about the research that you think my supervisor or I can help you with, please feel free to contact me (us) on:

Wilder Perdomo Charry
wilder.perdomo@uts.edu.au
(+61) [redacted]

NOTE: *This study has been approved by the University of Technology Sydney Human Research Ethics Committee [UTS HREC]. If you have any concerns or complaints about any aspect of the conduct of this research, please contact the Ethics Secretariat on ph.: +61 2 9514 2478 or email: Research.Ethics@uts.edu.au], and quote the UTS HREC reference number. Any matter raised will be treated confidentially, investigated and you will be informed of the outcome.*

Appendix E

Interview - Qualitative Design (QLD)

HOW DO COLOMBIAN SOFTWARE DEVELOPERS EVALUATE SOFTWARE PRODUCT QUALITY? (ETH18-2553)
FACULTY OF ENGINEERING AND INFORMATION TECHNOLOGY
SCHOOL OF SOFTWARE
SYDNEY, AUSTRALIA

INTRODUCTION

In this section, I will introduce the research project as well as definitions of various terms we will use in the interview. *(Helping notes for the researcher are in italic and within parentheses)*

Software development is becoming an international business and Colombian software developers soon must compete against international developers who want to enter the Colombian market. In order to compete, the quality of Colombian software must be of similar or better quality, and to achieve that it is necessary to know which quality measures matter and how to evaluate the software product quality. This research will investigate which quality measures are important to Colombian industry and determine some suitable criteria to evaluate the product quality with those measures.

Quality: The degree to which a user perceives that software meets his or her composite expectations.

Product: artefact that is produced, is quantified, and can be either an end item in itself or a component item.

Product Quality: The collection of characteristics of a product that contribute to its ability to meet given requirements.

Software quality: degree to which the software product meets established and implied needs when used under specific conditions.

Measure: variable to which a value is assigned as, a result of a measurement.

Measurement: Set of operations performed to determine the value of a measure.

Quality assessment: a systematic review of the extent to which an entity is able to meet specified requirements.

The purpose of this interview is to understand and find how the Colombian software developers evaluate the software product quality and which measures they use to do it. In addition, the impact of this research is related how the software developers assess their products during the development cycle.

CONSENT

Are you willing to participate in this research voluntarily, and do I have answered your questions? ( ) Yes | ( ) No

CLASSIFICATION

This section has a series of questions concerning the participant's and companies' demographic or socio-economic characteristics such as organisation size, participant's experience in IT, market, organisation age, and the industry for which you develop your products. For this research is important to know this information because we need to identify different response categories and the relevance of this topic to the Colombian software companies.

Tool	**Frequency**
Company size *(What is the company size?)* *Small (11-50 employees) | Medium (51-200 employees) | Large (more than 200 employees)*	Choose an item
Enterprise Level Role *(Project manager, developer, tester)*	Choose an item
Market *(Are you working for a local, national or international market?)*	Choose an item
Industry *(Are you developing product for some special industry?)*	Choose an item

SOFTWARE PRODUCT QUALITY

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?
    - What kinds of characteristics/measures are used to evaluate software product quality?
    - How is that characteristic/measure evaluated? *(Can you tell me more details?)*
4. How do you evaluate software product quality?
    - What kinds of characteristics/measures are you using to evaluate software product quality?
    - How do you evaluate that characteristic/measure? *(Can you tell me more details?)*

END OF INTERVIEW

Thank you very much for your time and for participating in this research.

APPENDIX F

AGREEMENT LETTER (QLD)

HOW DO COLOMBIAN SOFTWARE DEVELOPERS EVALUATE SOFTWARE PRODUCT QUALITY? (ETH18-2553)

I *[authorised company representative's name]*, *[company role]* from *[company name]* agree that the employees to participate in the research project to identify how Colombian software developers evaluate the product quality, and which measures they use during the development cycle (ETH18-2553), being conducted by Wilder Perdomo, Wilder.Perdomo@uts.edu.au, phone: (+61) [redacted], and supervised by Julia Prior, Julia.Prior@uts.edu.au, phone: (+61) 295144480. I understand that funding for this research has been provided by COLFUTURO Colombia and University of Technology Sydney.

I understand the purposes, procedures and risks of the research. Furthermore, I have had the opportunity to ask questions and I am satisfied with the answers I have received.

I freely agree that the employees to participate in this research project as described and understand that they are free to withdraw at any time without affecting their relationship with the researchers or the University of Technology Sydney.

I agree to keep confidential all information including all conversations and discussions, materials and methods provided to me and my employees by the UTS research team.

I am aware that I can contact Wilder Perdomo if I have any concerns about the research.

Name and Signature [participant]	**Date (dd/mm/yy)**
Name and Signature [researcher or delegate]	**Date (dd/mm/yy)**

NOTE: *This study has been approved by the University of Technology Sydney Human Research Ethics Committee [UTS HREC]. If you have any concerns or complaints about any aspect of the conduct of this research, please contact the Ethics Secretariat on ph.: +61 2 9514 2478 or email: Research.Ethics@uts.edu.au, and quote the UTS HREC ETH18-2553. Any matter raised will be treated confidentially, investigated and you will be informed of the outcome.*

APPENDIX G

PARTICIPANT INFORMATION SHEET FOR COMPANY REPRESENTATIVE (QLD)

HOW DO COLOMBIAN SOFTWARE DEVELOPERS EVALUATE SOFTWARE PRODUCT QUALITY? (ETH18-2553)

WHO IS DOING THE RESEARCH?

My name is Wilder Perdomo and I am a student at University of Technology Sydney (UTS). My supervisor is Dr. Julia Prior, email: Julia.Prior@uts.edu.au, phone: +61 2 9514 4480.

WHAT IS THIS RESEARCH ABOUT?

This research is to find out about how Colombian software developers evaluate the product quality, and which measures they use during the development cycle.

FUNDING

Funding for this project has been received from COLFUTURO, Colombia and the University of Technology Sydney, Australia.

WHY HAVE I BEEN ASKED?

You have been invited to participate in this study because your experience and knowledge in the software industry, as a company representative and you could put us in contact with three possible participants from your company. Your contact details were obtained from ICT Ministry or Colombian Software Federation (Fedesoft).

Managers will only be providing information about the study to potential participants; managers will not be seeking consent from individual participants —the researcher Wilder Perdomo will manage this process directly with the possible participants. Further, the manager will not see any interview questions or, most importantly, any participants' responses. This information will be confidential, only available to the research team (researcher and supervisors). Also, when the research is published it will be not possible to identify any individual participants from the results because it will be de-identified.

IF I SAY YES, WHAT WILL IT INVOLVE?

If you decide to be involve, you can put us in contact with the participants.

ARE THERE ANY RISKS/INCONVENIENCE?

Yes, there are some risks/inconvenience. They are:

- While we have made every effort to ensure data about employees and companies are not disclosed or identifiable in any way, in the very unlikely event that it is disclosed to their employer company that a participant, or their company, lacks significant knowledge or has inadequate quality evaluation processes, the company's response to this disclosure is unpredictable and beyond the research team,Äôs control.

DO I HAVE TO SAY YES?

Participation in this study is voluntary. It is completely up to the company whether or not the company decide to take part in this research.

WHAT WILL HAPPEN IF I SAY NO?

If your company decide not to participate, it will not affect your relationship with the researchers or the University of Technology Sydney. If you wish to withdraw company consent from the study once it has started, your company can do so at any time, without having to give a reason, by contacting Wilder.Perdomo@uts.edu.au.

If you company decides to withdraw from this research project, the interview audio-recordings and transcripts from your employees will be destroyed. However, it may not be possible to withdraw your company's data from the study results, if these have already had identifying details removed.

CONFIDENTIALITY

By signing the agreement, your providing consent for your employees to participate and to the research team collecting company information for the research project. Only the research team (researcher and supervisors) will have access to the company and participants' information, which will be kept confidential in the online repository and personal computer provided by the University. Company's information will only be used for the purpose of this research project.

The research outcomes will be released to COLFUTURO and University of Technology Sydney. It will also be published in recognised peer-reviewed journals in the field. In any publication or report, information will be provided in such a way that none of the companies or individuals participants can be identified.

WHAT IF I HAVE CONCERNS OR A COMPLAINT?

If you have concerns about the research that you think my supervisor, or I can help you with, please feel free to contact me (us) on:

Julia Prior (Supervisor)	Wilder Perdomo (Student)	Maria Lut Siza (Sponsor)
julia.prior@uts.edu.au	wilder.perdomo@uts.edu.au	marialut.siza@colfuturo.org
(+61) 2 9514 4428	(+61) [redacted]	+57 (1) 3405394

NOTE: *This study has been approved by the University of Technology Sydney Human Research Ethics Committee [UTS HREC]. If you have any concerns or complaints about any aspect of the conduct of this research, please contact the Ethics Secretariat on ph.: +61 2 9514 2478 or email: Research.Ethics@uts.edu.au, and quote the UTS HREC ETH18-2553. Any matter raised will be treated confidentially, investigated and you will be informed of the outcome.*

APPENDIX H

CONSENT FORM (QLD)

HOW DO COLOMBIAN SOFTWARE DEVELOPERS EVALUATE SOFTWARE PRODUCT QUALITY? (ETH18-2553)

I *[participant's name]* agree to participate in the research project to identify how Colombian software developers evaluate the product quality, and which measures they use during the development cycle (ETH18-2553), being conducted by *Wilder Perdomo*, Wilder.Perdomo@uts.edu.au, phone: (+61) [redacted]. I understand that funding for this research has been provided by COLFUTURO Colombia and University of Technology Sydney.

I have read the Participant Information Sheet or someone has read it to me in a language that I understand.

I understand the purposes, procedures and risks of the research as described in the Participant Information Sheet.

I have had an opportunity to ask questions and I am satisfied with the answers I have received.

I freely agree to participate in this research project as described and understand that I am free to withdraw at any time without affecting my relationship with the researchers or the University of Technology Sydney.

I understand that I will be given a signed copy of this document to keep.

I agree to be: ( ) Audio recorded

I agree to keep confidential all information including all conversations and discussions, materials and methods provided to me by the UTS research team.

I agree that the researcher could get my permission if any published paper or thesis wants to quote something I have said. In addition, whether there are any additional use of data in future research projects: ( ) Yes | ( ) No

I am aware that I can contact Wilder Perdomo if I have any concerns about the research.

Name and Signature [participant]	**Date (dd/mm/yy)**

Name and Signature [researcher or delegate]	**Date (dd/mm/yy)**

NOTE: *This study has been approved by the University of Technology Sydney Human Research Ethics Committee [UTS HREC]. If you have any concerns or complaints about any aspect of the conduct of this research, please contact the Ethics Secretariat on ph.: +61 2 9514 2478 or email: Research.Ethics@uts.edu.au, and quote the UTS HREC reference number. Any matter raised will be treated confidentially, investigated and you will be informed of the outcome.*

Appendix I

Participant information sheet (QLD)

HOW DO COLOMBIAN SOFTWARE DEVELOPERS EVALUATE SOFTWARE PRODUCT QUALITY? (ETH18-2553)

WHO IS DOING THE RESEARCH?

My name is Wilder Perdomo and I am a student at University of Technology Sydney (UTS). My supervisor is Dr. Julia Prior, email: Julia.Prior@uts.edu.au, phone: +61 2 9514 4480.

WHAT IS THIS RESEARCH ABOUT?

This research is to find out about how Colombian software developers evaluate the product quality, and which measures they use during the development cycle.

FUNDING

Funding for this project has been received from COLFUTURO, Colombia and the University of Technology Sydney, Australia.

WHY HAVE I BEEN ASKED?

You have been invited to participate in this study because your experience and knowledge in the software industry, as a quality manager, developer or tester of software products. Your contact details were obtained from ICT Ministry or Colombian Software Federation (Fedesoft).

IF I SAY YES, WHAT WILL IT INVOLVE?

If you decide to participate, I will invite you to answer the interview that will take approximately 30 minutes to complete.

ARE THERE ANY RISKS/INCONVENIENCE?

Yes, there are some risks/inconvenience. They are:

- The participant may be asked sensitive questions about their company
- Participants may be concerned about their privacy and/or confidentiality. Participants will not be anonymous because of the nature of data collection, i.e. interviews
- Participants may not have time, extra commitments, etc.
- While we have made every effort to ensure data about employees and companies are not disclosed or identifiable in any way, in the very unlikely event that it is disclosed to their employer company that a participant, or their company, lacks significant knowledge or has inadequate quality evaluation processes, the company's response to this disclosure is unpredictable and beyond the research team's control

These risks will be managed in the following ways:

- The participants will be not identifiable in any published results
- The company will be not identifiable in any published results
- The data only be available to the researcher and their supervisors
- Interview data will be de-identified in the transcripts to make it confidential
- The participant's interview answers will not be available in any form to their employer/company
- Participants do not have to answer any questions that they are uncomfortable answering
- Participation in the research is entirely voluntary; see below
- Information such as company classification and/or company size will not be in published reports if it is possible to identify the company from the summary data

DO I HAVE TO SAY YES?

Participation in this study is voluntary. It is completely up to you whether or not you decide to take part.

WHAT WILL HAPPEN IF I SAY NO?

If you decide not to participate, it will not affect your relationship with the researchers or the University of Technology Sydney. If you wish to withdraw from the study once it has started, you can do so at any time, without having to give a reason, by contacting Wilder.Perdomo@uts.edu.au.

If you decide to withdraw from this research project, the interview audio-recording and transcript will be destroyed. However, it may not be possible to withdraw your data from the study results, if these have already had your identifying details removed.

CONFIDENTIALITY

By signing the consent form, you consent to the research team collecting and using personal information about you for the research project. Only the research team (researcher and supervisors) will have access to the participants' information, which will be kept confidential in the online repository and personal computer provided by the University. Your information will only be used for this research project.

The interview will be conducted by Wilder Perdomo for software development companies in Colombia. The data will be audio-recorded through Skype calls and will be stored in password protected files on the researcher's University computer, and on backup facilities managed by the University. The data from the interview will be de-identified before it is transcribed. The participants will have the opportunity to review their interview transcripts before the results are published.

The research outcomes will be released to COLFUTURO and University of Technology Sydney. It will also be published in recognised peer-reviewed journals in the field. In any publication or report, information will be provided in such a way that none of the companies or individuals participants can be identified.

WHAT IF I HAVE CONCERNS OR A COMPLAINT?

If you have concerns about the research that you think my supervisor, or I can help you with, please feel free to contact me (us) on:

Julia Prior (Supervisor)	Wilder Perdomo (Student)	Maria Lut Siza (Sponsor)
julia.prior@uts.edu.au	wilder.perdomo@uts.edu.au	marialut.siza@colfuturo.org
(+61) 2 9514 4428	(+61) [redacted]	+57 (1) 3405394

NOTE: *This study has been approved by the University of Technology Sydney Human Research Ethics Committee [UTS HREC]. If you have any concerns or complaints about any aspect of the conduct of this research, please contact the Ethics Secretariat on ph.: +61 2 9514 2478 or email: Research.Ethics@uts.edu.au, and quote the UTS HREC ETH18-2553. Any matter raised will be treated confidentially, investigated and you will be informed of the outcome.*

References

Ahmad, M. A., García, I. V. P. & Rodríguez, Y. (2016), 'Evaluation of the non-functional requirements of usability: A systematic study', *International Journal of Advanced Research in Computer Science* **7**(3).

Ahmed, F. & Capretz, L. F. (2011), 'A business maturity model of software product line engineering', *Information Systems Frontiers* **13**(4), 543–560.

Al-Qutaish, R. E. (2010), 'Quality models in software engineering literature: an analytical and comparative study', *Journal of American Science* **6**(3), 166–175.

Alase, A. (2017), 'The interpretative phenomenological analysis (IPA): A guide to a good qualitative research approach', *International Journal of Education and Literacy Studies* **5**(2), 9–19.

Alba, M. & Hurtado, S. (2011), Validation and calibration of quantitative models for software development effort and size estimation, *in* '6th Colombian Computing Congress (CCC)', IEEE, pp. 1–6.

Alsaqaf, W., Daneva, M. & Wieringa, R. (2017), Agile quality requirements engineering challenges: First results from a case study, *in* '2017 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)', IEEE, pp. 454–459.

Alvaro, A., Santana, E. & Lemos, S. (2007), Towards a software component certification framework, *in* 'Seventh International Conference on Quality Software (QSIC 2007)', IEEE, pp. 298–303.

Alvaro, A., Santana, E. & Lemos, S. (2010), 'A software component quality framework', *ACM SIGSOFT Software Engineering Notes* **35**(1), 1–18.

Andreou, A. S. & Tziakouris, M. (2007), 'A quality framework for developing and evaluating original software components', *Information and software technology* **49**(2), 122–141.

Arciniegas, J., Fernandez, M., Hormiga, M., Tulande, A. & Collazos, C. (2010), 'Architectural patterns regarding web application domain usability', *Ingenieria e Investigacion* **30**(1), 52–55.

Baggen, R., Correia, J. P., Schill, K. & Visser, J. (2012), 'Standardized code quality benchmarking for improving software maintainability', *Software Quality Journal* **20**(2), 287–307.

Baquero, L., Gil, C. & Hernández, M. (2018), 'The usability and accessibility as quality factors of a secure web product', *International Journal of Applied Engineering Research* **13**(23), 16288–16294.

Barney, S. & Wohlin, C. (2009), Software product quality: Ensuring a common goal, *in* 'International Conference on Software Process', Springer, pp. 256–267.

Bassam, A., AL, B., Selamat, M. H., Din, J., Jabar, M. A. & Turaev, S. (2011), 'Software quality evaluation: user's view', *International Journal of Applied Mathematics and Informatics* (3), 200–207.

Bawane, N. & Srikrishna, C. (2010), 'A novel method for quantitative assessment of software quality', *International Journal of Computer Science and Security (IJCSS)* **3**(6), 508.

Bazeley, P. (2013), *Qualitative data analysis: Practical strategies*, Sage.

Behkamal, B., Kahani, M. & Akbari, M. K. (2009), 'Customizing iso 9126 quality model for evaluation of b2b applications', *Information and software technology* **51**(3), 599–609.

Berger, T. (2008), 'Concepts of national competitiveness', *Journal of International Business and Economy* **9**(1), 91–111.

Bertoa, M. F. & Vallecillo, A. (2002), 'Quality attributes for cots components', *I+D Computation* **1**(2), 128–143.

Bevan, N., Carter, J., Earthy, J., Geis, T. & Harker, S. (2016), New iso standards for usability, usability reports and usability measures, *in* 'International Conference on Human-Computer Interaction', Springer, pp. 268–278.

Boehm, B. W., Brown, J. R., Kaspar, H., Lipow, M. L. & Mc Cleod, G. (1978), *Characteristics of software quality*, TRW Software Technology, CERN Acelerating science, Amsterdam.

Burger, E. & Reussner, R. (2011), ‘Performance certification of software components’, *Electronic Notes in Theoretical Computer Science* **279**(2), 33–41.

Burmeister, E. & Aitken, L. M. (2012), ‘Sample size: How many is enough?’, *Australian Critical Care* **25**(4), 271–274.

Calero, C., Moraga, M. & Piattini, M. (2008), ‘Handbook of research on web information systems quality’.

Carvalho, F., Meira, R. L., Freitas, B. & Eulino, J. (2009), Embedded software component quality and certification, *in* ‘35th Euromicro Conference on Software Engineering and Advanced Applications’, IEEE, pp. 420–427.

Castellanos, H. A. (2016), ‘Personality recognition applying machine learning techniques on source code metrics’, *Working notes of FIRE* **18**(1), 7–10.

Chung, L. & do Prado Leite, J. C. S. (2009), On non-functional requirements in software engineering, *in* ‘Conceptual modeling: Foundations and applications’, Springer, pp. 363–379.

Colomo, R., Fernandes, E., Soto, P. & Sabbagh, M. (2011), ‘Software product evolution for intellectual capital management: The case of Meta4 PeopleNet’, *International Journal of Information Management* **31**(4), 395–399.

Creswell, J. W. (2013), ‘Research design: qualitative, quantitative, and mixed methods approaches’.

Creswell, J. W. & Creswell, J. D. (2017), *Research design: Qualitative, quantitative, and mixed methods approaches*, Sage publications.

Crotty, M. (1998), *The foundations of social research: Meaning and perspective in the research process*, Sage.

Curcio, K., Malucelli, A., Reinehr, S. & Paludo, M. A. (2016), ‘An analysis of the factors determining software product quality: A comparative study’, *Computer Standards & Interfaces* **48**, 10–18.

Darmawan, N., Chong, A. Y., Ooi, K. B. & Tan, B. I. (2010), ‘Factor strategy model: proofs of prototype concept for software quality evaluation’, *The Journal of Computer Information Systems* **50**(3), 139–149.

Delgado, A. F., Paz, D. E., Arciniegas, J. L. & Pino, F. J. (2016), 'Architectural model from the information view to support interoperability of software tools that support software process improvement', *Ingenieria y competitividad* **18**(2), 35–52.

Deming, W. E. (1982), 'Quality, productivity and competitive position. MIT Press, Cambridge'.

Diaz, M., Garcia, F. & Piattini, M. (2010), 'Mis-pyme software measurement capability maturity model–supporting the definition of software measurement programs and capability determination', *Advances in Engineering Software* **41**(10), 1223–1237.

Dromey, R. G. (1995), 'A model for software product quality', *IEEE Transactions on Software Engineering* **21**(2), 146–162.

Duque, A., Fernandez, J. T., Stevens, R. & Aussenac, N. (2011), 'Oquare: A square-based approach for evaluating the quality of ontologies', *Journal of Research and Practice in Information Technology* **43**(2), 159–176.

Eatough, V. & Smith, J. A. (2008), 'Interpretative phenomenological analysis', *The Sage handbook of qualitative research in psychology* **179**, 194.

Escobar, V. & Linares, M. (2012), A model for measuring agility in small and medium software development enterprises, *in* 'XXXVIII Conferencia Latinoamericana en Informatica (CLEI)', IEEE, pp. 1–10.

Farah, G., Tejada, J. S. & Correal, D. (2014), Openhub: a scalable architecture for the analysis of software quality attributes, *in* 'Proceedings of the 11th Working Conference on Mining Software Repositories', ACM, pp. 420–423.

Farr, W. (1996), 'Software reliability modeling survey. Technical Foundations'.

Fayad, M. E., Laitinen, M. & Ward, R. P. (2000), 'Thinking objectively: software engineering in the small', *Communications of the ACM* **43**(3), 115–118.

Febrero, F., Calero, C. & Moraga, M. A. (2016), 'Software reliability modeling based on ISO/IEC SQuaRE', *Information and Software Technology* **70**(2), 18–29.

Fedesoft (2015), Caracterización del sector teleinformática, software y TI en Colombia, Report.

Ferdows, K. & De Meyer, A. (1990), 'Lasting improvements in manufacturing performance: In search of a new theory', *Journal of Operations Management* **9**(2), 168–184.

Fernandez, C, M. & Piattini, M. (2012), Modelo para el gobierno de las TIC basado en las normas ISO, Report.

Fernandez, Y., Cruz, C. & Verdegay, J. L. (2018), 'A new model based on soft computing for evaluation and selection of software products', *IEEE Latin America Transactions* **16**(4), 1186–1192.

Fusch, P. I. & Ness, L. R. (2015), 'Are we there yet? data saturation in qualitative research', *The qualitative report* **20**(9), 1408.

Gall, C., Lukins, S., Etzkorn, L., Gholston, S., Farrington, P., Utley, D., Fortune, J. & Virani, S. (2008), 'Semantic software metrics computed from natural language design specifications', *IET software* **2**(1), 17–26.

GARTNER, I. (2015), 'Hype cycle for emerging technologies identifies the computing innovations that organizations should monitor'.

Garzas, J., Pino, F. J., Piattini, M. & Fernandez, C. M. (2013), 'A maturity model for the spanish software industry based on iso standards', *Computer Standards and Interfaces* **35**(6), 616–628.

Gasca, G. P., Manzano, J. C., Giraldo, L. M. & Echeverri, J. A. (2013), Statistical analysis of the implementation for best practices in software development organizations, *in* '8th Iberian Conference on Information Systems and Technologies (CISTI)', pp. 1–8.

Georgiadou, E. (2003), 'Gequamo: a generic, multilayered, customisable, software quality model', *Software Quality Journal* **11**(4), 313–323.

Giorgi, A. (1994), 'A phenomenological perspective on certain qualitative research methods', *Journal of phenomenological psychology* **25**(2), 190–220.

Giraldo, F. D., Espana, S., Pineda, M. A., Giraldo, W. J. & Pastor, O. (2014), Integrating technical debt into mde, *in* 'CAISE 14 (Forum/Doctoral Consortium)', Citeseer, pp. 145–152.

Goulding, C. (2005), 'Grounded theory, ethnography and phenomenology: A comparative analysis of three qualitative strategies for marketing research', *European journal of Marketing* **39**(3/4), 294–308.

Grady, R. B. (1992), *Practical software metrics for project management and process improvement*, ACM, USA.

Guana, V. & Correal, D. (2013), 'Improving software product line configuration: a quality attribute-driven approach', *Information and Software Technology* **55**(3), 541–562.

Guest, G., Bunce, A. & Johnson, L. (2006), 'How many interviews are enough? an experiment with data saturation and variability', *Field methods* **18**(1), 59–82.

Haigh, M. (2010), 'Software quality, non-functional software requirements and it-business alignment', *Software Quality Journal* **18**(3), 361–385.

Hatcliff, J., Heimdahl, M., Lawford, M., Maibaum, T., Wassyng, A. & Wurden, F. (2009), 'A software certification consortium and its top 9 hurdles', *Electronic Notes in Theoretical Computer Science* **238**(4), 11–17.

Heck, P., Klabbers, M. & van Eekelen, M. (2010), 'A software product certification model', *Software Quality Journal* **18**(1), 37–55.

Hering, D., Schwartz, T., Boden, A. & Wulf, V. (2015), Integrating usability-engineering into the software developing processes of SME: a case study of software developing SME in Germany, *in* 'Proceedings of the 8th International Workshop on Cooperative and Human Aspects of Software Engineering', IEEE Press, pp. 121–122.

Hofman, R. (2010), *Software Quality Perception*, Springer Netherlands, Dordrecht, pp. 31–37.

Hyatt, L. E. & Rosenberg, L. H. (1996), A software quality model and metrics for identifying project risks and assessing software quality, *in* 'Product Assurance Symposium and Software Product Assurance Workshop', p. 209.

Idri, A., Bachiri, M. & Fernández-Alemán, J. L. (2016), 'A framework for evaluating the software product quality of pregnancy monitoring mobile personal health records', *Journal of medical systems* **40**(3), 50.

IMF (2017), The growth return of infrastructure in latin america, Report, International Monitoring Fund.

Inayat, I., Salim, S. S., Marczak, S., Daneva, M. & Shamshirband, S. (2015), 'A systematic literature review on agile requirements engineering practices and challenges', *Computers in human behavior* **51**, 915–929.

ISO14598 (2000), ‘Software engineering - product evaluation - part 3: Process for developers (ISO/IEC:14598)’.

ISO24765 (2010), ‘Systems and software engineering - Vocabulary (ISO/IEC:24765)’.

ISO25000 (2014), ‘Systems and software engineering - systems and software quality requirements and evaluation (SQuaRE) - guide to SQuaRE (ISO/IEC:25000)’.

ISO25023 (2016), ‘Systems and software engineering - systems and software quality requirements and evaluation (SQuaRE) - measurement of systems and software product quality (ISO/IEC:25023)’.

ISO9126 (2001), ‘Software engineering - Product Quality - Part 1: Quality model (ISO/IEC:9126)’.

Izzat, A. (2013), *Website performance measurement: process and product metrics*, Website Performance Measurement, Jordan, pp. 275–301.

Jacobson, I., Ng, P. W., McMahon, P. E., Spence, I., Lidman, S. & Zapata, C. M. (2013), ‘La esencia de la ingenieria de software: el nucleo de semat’, *Revista Latinoamericana de Ingenieria de Software* **1**(3), 71–78.

Jacobson, I., Ng, P. W., McMahon, P., Spence, I. & Lidman, S. (2012), ‘The essence of software engineering: the semat kernel’, *Queue* **10**(10), 40.

Jacoby, J. & Matell, M. S. (1971), ‘Three-point likert scales are good enough’, *Journal of Marketing Research* **8**(4), 495–500.

Jamwal, R. S. & Jamwal, D. (2009), Issues and factors for evaluation of software quality models, *in* ‘Proceedings of the 3rd National Conference’, Computing for Nation Development, pp. 1–6.

Jinzenji, K., Hoshino, T., Williams, L. & Takahashi, K. (2013), 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)’, IEEE, pp. 310–319.

Jones, C. (2013), ‘Function points as a universal software metric’, *SIGSOFT Softw. Eng. Notes* **38**(4), 1–27.

Jones, C. (2017), Errors and omissions in software historical data: separating fact from fiction, Report, Namcook Analytics.

Kalaimagal, S. & Srinivasan, R. (2010), ‘Q’Facto 12: an improved quality model for COTS components’, *ACM SIGSOFT Software Engineering Notes* **35**(2), 1–4.

Khosravi, K. & Gueheneuc, Y.-G. (2004), On issues with software quality models, *in* ‘The Proceedings of the 11th Working Conference on Reverse Engineering (GEODES)’, pp. 172–181.

Kitchenham, B. (2004), ‘Procedures for performing systematic reviews’, *Keele, UK, Keele University* **33**(2004), 1–26.

Kitchenham, B. & Pfleeger, S. L. (1996), ‘Software quality: the elusive target’, *IEEE software* **13**(1), 12–21.

Kumar, A., Grover, P. & Kumar, R. (2009), ‘A quantitative evaluation of aspect-oriented software quality model (AOSQUAMO)’, *ACM SIGSOFT Software Engineering Notes* **34**(5), 1–9.

Kumar, D., Gupta, V. & Kapur, P. (2015), Assessment of quality factors in enterprise application integration, *in* ‘2015 4th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO) (Trends and Future Directions)’, IEEE, pp. 1–6.

Lampasona, C., Heidrich, J., Basili, V. R. & Ocampo, A. (2012), Software quality modeling experiences at an oil company, *in* ‘Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement’, ACM, 2372296, pp. 243–246.

Lampasona, C. & Kläs, M. (2011), Supporting the adaptation of software quality models–an empirical investigation, *in* ‘Proc. of the 4th Workshop on Modeling and Assessment of Software Quality SQMB’, Vol. 11.

LeCompte, M. D. (2000), ‘Analyzing qualitative data’, *Theory into practice* **39**(3), 146–154.

Liberatore, M. J. & Pollack-Johnson, B. (2013), ‘Improving project management decision making by modeling quality, time, and cost continuously’, *IEEE Transactions on Engineering Management* **60**(3), 518–528.

Littlewood, B. & Strigini, L. (2010), Software reliability and dependability: a roadmap, *in* 'Proceedings of the Conference on the Future of Software Engineering', ACM, pp. 175–188.

Maibaum, T. & Wassyng, A. (2008), 'A product-focused approach to software certification', *Software Technologies* **41**(2), 91–93.

Marcos, J., Arroyo, A., Garzas, J. & Piattini, M. (2008), 'La norma ISO/IEC 25000 y el proyecto kemis para su automatizacion con software libre', *Revista Espanola de Innovacion, Calidad e Ingenieria del Software (REICIS)* **4**(2), 133–144.

Marin, H. A. & Bedoya, A. E. (2015), Una adopcion de pmbok al ciclo de vida de desarrollo de proyectos software en pequenas empresas, *in* 'Congreso Iberoamericano de Ingenieria de Proyectos', Red Iberoamericana de Ingenieria de Proyectos, pp. 1–19.

Marshall, C., Brereton, P. & Kitchenham, B. (2015), 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', ACM, p. 26.

Mason, M. (2010), Sample size and saturation in phd studies using qualitative interviews, *in* 'Forum qualitative Sozialforschung/Forum: qualitative social research', Vol. 11.

McCall, J. A., Richards, P. K. & Walters, G. F. (1977), Factors in software quality: final report, Report, Information Systems Programs, General Electric Company.

Mellado, D., Rodriguez, M., Verdugo, J. & Piattini, Mario amd Fernandez, E. (2010), 'Evaluacion de la calidad y seguridad en productos software'.

Metaute, P. & Serna, A. (2016), '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.

Miles, M. B., Huberman, A. M. & Saldana, J. (2014), *Qualitative data analysis: A Method Sourcebook*, Sage.

MinTIC (2015), Caracterizacion del sector teleinformatica, software y ti en colombia, Report, MinTIC-Sena-Fedesoft.

Moreno, J. J., Bolaños, L. P. & Navia, M. A. (2010), 'Un acercamiento a las prácticas de calidad de software en las mipymesps del suroccidente colombiano'.

Morris, J., Lee, G., Parker, K., Bundell, G. A. & Lam, C. P. (2001), 'Software component certification', *IEEE Computer* **34**(9), 30–36.

Moustakas, C. (1994), *Phenomenological research methods*, Sage.

Nakai, H., Tsuda, N., Honda, K., Washizaki, H. & Fukazawa, Y. (2016), A square-based software quality evaluation framework and its case study, *in* '2016 IEEE Region 10 Conference (TENCON)', IEEE, pp. 3704–3707.

Ortega, M., Perez, M. & Rojas, T. (2003), 'Construction of a systemic quality model for evaluating a software product', *Software Quality Journal* **11**(3), 219–242.

Oviedo, J. R., Rodriguez, M. & Piattini, M. (2015), Certification of ipavement applications for smart cities a case study, *in* 'International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE)', IEEE, pp. 244–249.

Pelaez, L. E., Hurtado, R. A. & Franco, J. A. (2011), 'Certificacion de la calidad del proceso y producto: ruta para pymes colombianas que fabrican software', *Ventana Informatica* **25**(2), 41–61.

Perdomo, W. (2019), '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, PhD Thesis', *Nacional de Colombia University, Computer and Decision Sciences Department, Faculty of Mines* pp. 1–120.

Perdomo, W., Prior, J. & Leaney, J. (2020), How do colombian software companies evaluate software product quality?, *in* 'Proceedings of the 30th international Workshop on Software Measurement (IWSM) and the 15th international Conference on Software Process and Product Measurement (MENSURA)', CEUR-WS, pp. 1–16.

Perdomo, W. & Zapata, C. (2015), Identificacion de criterios para relacionar la usabilidad con el alfa sistema de software del nucleo de semat, *in* 'Latin American Software Engineering Symposium', LASES, pp. 17–22.

Perdomo, W. & Zapata, C. M. (2021), 'Software quality measures and their relationship with the states of the software system alpha', *Ingeniare. Revista chilena de ingeniería* **29-2**, 1–23.

Pham, H. & Pham, M. (1991), Software reliability models for critical applications, Report, Idaho National Engineering Laboratory.

Phillips, L. B., Aurum, A. & Svensson, R. B. (2012), Managing software quality requirements, *in* '2012 38th Euromicro Conference on Software Engineering and Advanced Applications', IEEE, pp. 349–356.

Pietkiewicz, I. & Smith, J. A. (2014), 'A practical guide to using interpretative phenomenological analysis in qualitative research psychology', *Psychological journal* **20**(1), 7–14.

Pino, F. J., Garcia, F. & Piattini, M. (2008), 'Software process improvement in small and medium software enterprises: a systematic review', *Software Quality Journal* **16**(2), 237–261.

Pino, F. J., Pardo, C., Garcia, F. & Piattini, M. (2010), 'Assessment methodology for software process improvement in small organizations', *Information and Software Technology* **52**(10), 1044–1061.

Pino, F. J., Ruiz, F., Garcia, F. & Piattini, M. (2012), 'A software maintenance methodology for small organizations: Agile-mantema', *Journal of Software: Evolution and Process* **24**(8), 851–876.

Porter, M. E. (2001), 'The value chain and competitive advantage', *Understanding business: Processes* pp. 50–66.

Pressman, R. S. (2005), *Software engineering: a practitioner's approach*, Palgrave Macmillan, United States.

Procolombia (2017), Investment environment and buisiness opportunities in colombia, Report, Procolombia.

Putnam, L. H. & Myers, W. (2004), 'Five core metrics: the intelligence behind successful software management', *Software Quality Professional* **6**(2), 44.

Radatz, J., Geraci, A. & Katki, F. (1990), 'Ieee standard glossary of software engineering terminology', *IEEE Std* **61**(12), 3.

Radjenović, D., Heričko, M., Torkar, R. & Živkovič, A. (2013), 'Software fault prediction metrics: A systematic literature review', *Information and software technology* **55**(8), 1397–1418.

Rahmani, H., Sami, A. & Khalili, A. (2016), 'CIP-UQIM - A unified model for quality improvement in software SMEs based on CMMI level 2 and 3', *Information and Software Technology* **71**, 27–57.

Ralph, P. & Kelly, P. (2014), The dimensions of software engineering success, *in* 'Proceedings of the 36th International Conference on Software Engineering', pp. 24–35.

Rawashdeh, A. & Matalkah, B. (2006), 'A new software quality model for evaluating cots components', *Journal of Computer Science* **2**(4), 373–381.

Rodriguez, J. I., Sierra, E. & Jaramillo, L. K. (2015), 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.

Rodriguez, M., Oviedo, J. R. & Piattini, M. (2016), 'Evaluation of software product functional suitability: a case study', *Software Quality Professional* **18**(3), 18–29.

Rodriguez, M., Pedreira, O. & Fernandez, C. M. (2015), 'Certificacion de la mantenibilidad del producto software: un caso practico', *Revista Latinoamericana de Ingenieria de Software* **3**(3), 127–134.

Rodriguez, M. & Piattini, M. (2012*a*), 'Revision sistematica sobre la certificacion del producto software', *Computer Science and Engineering* **20**(3), 16–24.

Rodriguez, M. & Piattini, M. (2012*b*), Systematic review of software product certification, *in* '7th Iberian Conference on Information Systems and Technologies (CISTI),', IEEE, pp. 1–6.

Salcedo, E. & Gil, C. (2012), 'Sistema difuso para la evaluacion de la calidad externa de software orientado a la web', *Revista Educacion en Ingenieria* **7**(13), 91–101.

Sánchez-Gordón, M.-L. & O'Connor, R. V. (2016), 'Understanding the gap between software process practices and actual practice in very small companies', *Software Quality Journal* **24**(3), 549–570.

Schwandt, T. A. et al. (1994), 'Constructivist, interpretivist approaches to human inquiry', *Handbook of qualitative research* **1**, 118–137.

Serebrenik, A., Mishra, A., Delissen, T. & Klabbers, M. (2010), Requirements certification for offshoring using lspcm, *in* 'Seventh International Conference on the Quality of Information and Communications Technology (QUATIC)', IEEE, pp. 177–182.

Sharma, A., Kumar, R. & Grover, P. (2008), 'Estimation of quality for software components: an empirical approach', *ACM SIGSOFT Software Engineering Notes* **33**(6), 1–10.

Sibisi, M. & Van Waveren, C. C. (2007), A process framework for customising software quality models, *in* 'AFRICON 2007', IEEE, pp. 1–8.

Singh, S., Singh, P. & Rani, S. (2012), 'Challenges to development of standard software quality model', *International Journal of Computer Applications* **49**(10), 1–7.

Sjoberg, D. I., Dyba, T. & Jorgensen, M. (2007), The future of empirical methods in software engineering research, *in* '2007 Future of Software Engineering', IEEE Computer Society, pp. 358–378.

Smith, J. A. (2015), *Qualitative psychology: A practical guide to research methods*, Sage.

Soley, R. M. & Curtis, B. (2013), The consortium for IT software quality (cisq), *in* 'International Conference on Software Quality', Springer, pp. 3–9.

Solyman, A. M., Ibrahim, O. A. & Elhag, A. A. M. (2015), 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)', IEEE, pp. 123–128.

Srivastava, P. R. & Kumar, K. (2009), An approach towards software quality assessment, *in* 'International Conference on Information Systems, Technology and Management', Springer, pp. 150–160.

StandishGroup (2015), 'The Chaos report', *United States of America* .

Tamai, T. & Anzai, T. (2018), Quality requirements analysis with machine learning., *in* 'ENASE', pp. 241–248.

Terry, G., Hayfield, N., Clarke, V. & Braun, V. (2017), 'Thematic analysis', *The Sage handbook of qualitative research in psychology* pp. 17–37.

TradeMap (2015), Trade statistics for international business development, *in* 'Geneva, Switzerland: International Trade Centre (United Nations Conference on Trade and Development-World Trade Organization)). Accessed at several dates from: http://www.trademap.org'.

Trendowicz, A., Punter, T. et al. (2003), Quality modeling for software product lines, *in* '7th ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE'03)'.

Upadhyay, N., Despande, B. M. & Agrawal, V. P. (2011), 'Towards a software component quality model', *Advances in Computer Science and Information Technology* pp. 398–412.

Vaismoradi, M., Jones, J., Turunen, H. & Snelgrove, S. (2016), 'Theme development in qualitative content analysis and thematic analysis'.

Villavicencio, M. & Abran, A. (2015), 'Sugerencias para la inclusión de temas de medición de software en un currículo de ingeniería de software para estudiantes de pregrado', *Revista Latinoamericana de Ingeniería de Software* **1**(1), 117–126.

Wagner, S. (2013), *Software product quality control*, Springer, Germany.

Wenger, E. (1998), 'Communities of practice: Learning as a social system', *Systems thinker* **9**(5), 2–3.

Wirth, N. (2008), 'A brief history of software engineering', *IEEE Annals of the History of Computing* **30**(3), 32–39.

Xu, Q., Jiao, R. J., Yang, X., Helander, M., Khalid, H. M. & Opperud, A. (2009), 'An analytical kano model for customer need analysis', *Design Studies* **30**(1), 87–110.

Yahaya, J., Deraman, A. & Hamdan, A. R. (2010), Continuously ensuring quality through software product certification: a case study, *in* 'International Conference on Information Society (i-Society)', IEEE, pp. 183–188.

Yahaya, J. H., Deraman, A. & Hamdan, A. R. (2008), SCfM-PROD: A software product certification model, *in* '3rd International Conference on Information and Communication Technologies - From Theory to Applications-ICTTA', IEEE, pp. 1–6.

Zelkowitz, M. V. (2011), *Measuring and monitoring technical debt*, Vol. 82, Advances in Computers, p. 44.