Pós-Graduação em Ciência da Computação

JULIO VENÂNCIO DE MENEZES JÚNIOR

MEASURING RISKS IN SOFTWARE DEVELOPMENT PROJECTS

Federal University of Pernambuco
posgraduacao@cin.ufpe.br
www.cin.ufpe.br/~posgraduacao

Recife
2019

JULIO VENÂNCIO DE MENEZES JÚNIOR

MEASURING RISKS IN SOFTWARE DEVELOPMENT PROJECTS

A Ph.D. Thesis presented to the Center of Informatics of Federal University of Pernambuco in partial fulfillment of the requirements for the degree of Philosophy Doctor in Computer Science.

Concentration Area: Software engineering and programming languages

Advisor: Hermano Perrelli de Moura
Co-advisor: Cristine Martins Gomes de Gusmão

Recife
2019

Catalogação na fonte
Bibliotecária Monick Raquel Silvestre da S. Portes, CRB4-1217

M543m Menezes Júnior, Julio Venâncio de
Measuring risks in software development projects / Julio Venâncio de Menezes Júnior. – 2019.
145 f.: il., fig., tab.

Orientador: Hermano Perrelli de Moura.
Tese (Doutorado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2019.
Inclui referências e apêndices.

1. Engenharia de software. 2. Gerenciamento de projetos de software. 3. Gerenciamento de riscos. I. Moura, Hermano Perrelli de (orientador). II. Título.

005.1 CDD (23. ed.) UFPE- MEI 2019-061

Julio Venâncio de Menezes Júnior

Measuring Risks in Software Development Projects

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Doutor em Ciência da Computação.

Aprovado em: 20/02/2019.

_____

Orientador: Prof. Dr. Hermano Perrelli de Moura

BANCA EXAMINADORA

_____

Prof. Dr. Alexandre Marcos Lins de Vasconcelos
Centro de Informática /UFPE

_____

Prof. Dr. Sergio Castelo Branco Soares
Centro de Informática /UFPE

_____

Prof. Dr. Ricardo de Almeida Falbo
Departamento de Informática / UFES

_____

Prof. Dr. Carlos Eduardo Sanches da Silva
Instituto de Engenharia de Produção e Gestão / UNIFEI

_____

Prof. Dr. Jose Gilson de Almeida Teixeira Filho
Departamento de Ciências Administrativas / UFPE

To my family, friends, love, and professors who gave me the all necessary support to get here.

ACKNOWLEDGEMENTS

Well, finally getting to the end of my Ph.D. I can see how much I learned and experienced in these years of so much dedication. In the end, I realize that the most valuable thing for my life is the journey rather the results and contributions brought by this work.

In this long and winding road, although the act of research is often done lonely, everything that has been developed so far was possible due to several direct and indirect support.

First of all, thank God for allowing this plan to be realized.

Thank my advisors Hermano Perrelli de Moura and Cristine Martins Gomes de Gusmão for the opportunity, support and patience.

Thank my research colleagues from GP2 research group for the support and valuable help that were fundamental for important decision-making during this work.

Thank the Center of Informatics of the Federal University of Pernambuco for the institutional support and to provide an appropriate environment to develop the research.

Thanks a lot to my friends of "Ridiculinhos do CIn" laboratory for the reception in the final sprint.

I would like to thank SABER and SENAI Innovation Institute for Information and Communication Technologies for the comprehension so that I could reconcile labor and research activities. These environments certainly provided important insights for the development of the work.

I am thankful to Edmilson Luz Jr., my best friend, for the long-time friendship and long conversations about the life and important decisions that were taken in parallel to this work.

Thanks a lot to my friends Raphael D'Castro and Rodrigo Lins for the support, advice and for listening to me mainly during the final sprint.

I would like to especially thank Bruna Alves da Silva, my love, for the patience, cooperation, help, and friendship. Being to her side gave me the strength to keep going and not give up. Her presence in my life makes me a better person.

Finally, thank my family for the availability and basis that made it possible to achieve my Ph.D. goals.

ABSTRACT

There is a consensus that poorly managed adverse factors lead to project failures. Risk is an inherent part of any project. In other words risks are always present and may cause problems that lead to projects failure or poor performance. In software development projects, this scenario is not different. However, the practice of risk management in software development projects environments involves a high level of subjectivity, making it difficult to perceive the real impacts not only of risk management practices but also of the influence of risk on projects. A possible alternative for reducing this gap is through measurement-based approaches. In this sense, indicators are a relevant instrument for project's assessment and decision-making, since they are used to represent information in a clear and objective way. In this context, the objective of this thesis is to develop studies about software risk measurement and proposing and evaluating an indicator, called Project Risk Index (PRI), whose goal is to measure the risk level of a software development project at a given moment. This indicator starts from the premise that, in addition to the identified risk factors, some project's characteristics also contribute to raise the project risk levels. To achieve the research objectives, we started with an exploratory case study, seeking to resolve relevant issues in refining the proposed sources of information. Next, we performed a mapping of the most relevant risk factors in software development projects through a systematic literature review. Finally, two case studies were conducted with the proposed indicator, aiming to evaluate it and identify improvement points. The results show the existence of an indicator aiming to measure risk level of a software development project is crucial for better perception and clarity of the most critical project items. Regarding the proposed indicator, there are indications that its application may be useful and effective in risk management. Information about risk factors, combined with project's characteristics may be relevant for decision-making by the managers at the tactical and even strategic levels, enabling continuous and systematic monitoring of software development projects risk levels.

Keywords: Software project management. Risk management. Indicators. Measurement in Software Engineering.

RESUMO

Existe um consenso de que fatores adversos mal gerenciados levam a falhas em projetos. O risco é parte inerente de qualquer projeto. Em outras palavras: gerenciando ou não, os riscos sempre estão presentes e podem causar problemas que levam os projetos ao fracasso ou a uma má performance. Em projetos de desenvolvimento de software este cenário não é diferente. Contudo, as práticas de gerenciamento de riscos em projetos de desenvolvimento de software mais conhecidas envolvem um alto nível de subjetividade, dificultando na percepção da real noção dos impactos não só das práticas de gerenciamento de riscos, como também da influência dos riscos nos projetos. Uma possível alternativa para a redução desta lacuna é por meio de abordagens baseadas em medição. Neste sentido, indicadores constituem instrumentos relevantes para a avaliação de projetos e para a tomada de decisões, uma vez que são utilizados para representação da informação de forma clara e objetiva. Dentro deste contexto, o objetivo desta tese é desenvolver estudos acerca de medição de riscos em ambientes de desenvolvimento de software e propor e avaliar um indicador, chamado *Project Risk Index*, cujo objetivo é medir o nível de risco de um projeto de desenvolvimento de software em um dado momento. Este indicador parte da premissa de que, além dos fatores de riscos identificados, algumas características dos projetos contribuem para o aumento dos níveis de riscos do projeto. Para atingir este objetivo, esta pesquisa iniciou com a execução de um estudo de caso de caráter exploratório, buscando pontos relevantes para o refinamento dos parâmetros propostos. Em seguida, foi realizado um mapeamento de fatores de riscos mais relevantes em projetos de desenvolvimento de software por meio de revisão sistemática. Finalmente, dois estudos de caso foram realizados com o indicador proposto, buscando avaliá-lo e identificar pontos de melhoria. Os resultados apontam que a existência de indicador que vise medir o nível de risco de um projeto de desenvolvimento de software é crucial para a melhor percepção e clareza dos itens mais críticos de projetos. Em relação ao indicador proposto, há indícios de que seu uso pode ser útil e eficaz para o gerenciamento de projetos e de riscos. Informações sobre os fatores de riscos, combinados a características dos projetos, podem ser relevantes para a tomada de decisão por parte dos gestores nos níveis tático e estratégico, possibilitando o acompanhamento contínuo e sistemático dos níveis de riscos dos projetos de desenvolvimento de software.

Palavras-chave: Gerenciamento de projetos de software. Gerenciamento de riscos. Indicadores. Medição em engenharia de software.

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

CF	Characterization Factor
CMMI	Capability Maturity Model Integration
GQM	Goal-Question-Metrics
PC	Project Characterization
PCF	Project Characterization Factor
PSM	Practical Software Measurement
RBS	Risk Breakdown Structure
RE	Risk Exposure
RFE	Risk Factor Exposure
RP	Risk Points
SEI	Software Engineering Institute
UCP	Use Case Points
URPW	Unadjusted Risk Points Weight

CONTENTS

1 INTRODUCTION

An effective application of project management techniques is essential for projects to achieve their objectives. These project objectives are usually understood to be met when they are conducted successfully regarding schedule, costs, scope and controlled risks. We can infer that project management is not a trivial task, given the multiplicity of factors taken into account. In software projects it is not different. Software project management usually focuses on four pillars: Personnel, Product, Process and Project (PRESSMAN; MAXIM, 2014). That is, software engineering activities comprise human factors, associated with artifacts delivery, using a set of methods and tools that govern a project plan and must be stable. From this perspective, we can affirm that software projects are inherently risky ventures.

Another factor that increases the degree of risk of failures is the factor that is not common for an organization to have only one project being executed. Usually, we see multiple projects simultaneously inside the same environment, each at different stages of their respective life cycles, which further increases the management complexity , considering the primary objective is to maximize the level of the project's success. Therefore, an effective management that is driven by the factors that can harm the projects (risks), defining response plans correctly, although indirectly, is essential to achieve strategic objectives of organizations that develop software.

One of the instruments that support management, development, and maintenance of software projects are measurement-based approaches (RIFKIN; COX, 1991) since they optimize communication, support project monitoring, anticipate problems, as well as support the decision-making process (MCGARRY, 2002). In this sense, indicators are instruments of information representation that helps in organizing and synthesizing information and data to use it to support managerial activities, such as planning, goals setting and performance control (MOUSINHO, 2001). Hence, the use of indicators enables not only strategic decision-making but also its analysis. Considering that critical success factors of projects include risks, so they could be measured through the use of indicators, thereby allowing the well-founded decision- making. Therefore, as a managerial tool, the appropriate usage of metrics and indicators may contribute to improvement of project management practices.

This work addresses and deepens the issue of risk-driven project management and measurement through indicators. Its purpose is to perform studies aiming to improve software project risk management, considering measurement-based approaches are important to provide a better overview of success critical factors in environments of software development projects.

1.1 Motivation

There is a consensus in Software Engineering that, if adverse factors are not well managed, projects may fail. Researches show that only between 39% and 55% of software projects are considered successful (STANDISH GROUP, 2013; EL EMAM; KORU, 2008). It is

interesting to note that most times the cause of project failure is the poor management of adverse factors, i.e., risks.

Within the area of software project management, there are several elements to be managed. One of them is the risks, considered a fundamental part of project management. Some authors consider that managing project is managing risks (DE MARCO, 1997). This argument is valid, as software projects usually involve uncertainties of various natures during their development process, commonly related to factors such as innovation, constant changes, schedule, cost, among others. According to Barry Boehm, considered the father of risk management in software projects:

> "The most important thing for a project to do is to get focused on its critical success factors...The key contribution of software risk management is to create this focus on critical success factors - and to provide the techniques that let the project deal with them." (BOEHM, 1991)

The area of risk management in software projects, even considered necessary, still needs several advances, both in research and in practice. Approaches of risk management based on risk factors, on processes and modeling are evolving and being improved with sociocultural analysis, skills, and data analysis (BANNERMAN, 2015). Some studies show that the explicit practice of risk management contributes to performance improvement and increases the success of projects (WALLACE; KEIL; RAI, 2004a; RAZ; SHENHAR; DVIR, 2002). We can also affirm that when we are assuming risks, associated opportunities emerge, but, on the other hand, managing these factors requires specific skills too.

Risk management, due to abstract and subjective nature of the risk, is neglected by organizations that develop software because it does not bring any apparent immediate practical results (TRIGO; GUSMÃO; LINS, 2008). It happens due to cultural issues, associated with the fact that seeing risks can be understood as a signal of defeat. Besides, the gap between metrics and tools makes it difficult to have a more accurate and real analysis of return on investment about the usage of risk management.

It is important to note that organizations developing software tend to underestimate the importance of project risk management. For example, a research conducted by IBBS; KWAK (2000) with 30 organizations shows, among project management process areas, the one that presented the lowest level of maturity was risk management. In Brazil, for example, according to Secretariat of Planning in Informatics, only 11,43% of 446 Brazilian companies surveyed perform risk management (AUDY; PRIKLADNICKI, 2007).

Another problem that inhibits risk management is that it is a relatively recent area of study in software engineering, which, also, has little literature, its practical application is quite limited in organizational environments because it still involves a high degree of subjectivity (KONTIO, 2001). Some recent studies were aimed at minimizing this gap. However, they need most practical applications (D'CASTRO, 2013; MENEZES JR; GUSMÃO; MOURA, 2013).

The use of measurement techniques in Software Engineering allows the assessment of the progress of projects, products, processes, and resources, helping to control each one. It

is important to note that metrics and indicators act at both strategic and tactical levels of an organization, being effective at processes optimization, as well as helping in managerial decisions (LOPES, 2005). These techniques are supported by metrics and indicators. Software metric is a measure of some property of a piece of software or its specifications (SARAIVA, 2014), whereas indicators are metrics or a combination of metrics that provide additional information on estimates or assessments of specified attributes, being used as managerial tools for decision-making (D'CASTRO, 2013).

Metrics and indicators also support the continuous improvement of the environment. According to FENTON; PFLEEGER (1997), the use of metrics and indicators in projects increase the levels of:

- **Knowledge:** facilitates the discovery of what is going on in a project, enabling the assessment of the current situation, establishing more realistic goals and spreading such knowledge to stakeholders. Thus, the usage of metrics makes the aspects of products and processes more visible, giving a better idea about the relationship between activities and the resources involved;
- **Control:** gives conditions to select the best alternatives. This is done with basis on defined goals and monitoring of their compliance or noncompliance;
- **Improvement:** provides the measurement of the project evolution as a whole, based on adopted processes and generated artifacts or products.

In this context, we can state that metrics and indicators are critical managerial tools for software projects. On the other hand, its use specifically in software risk management has received little attention in literature. Differently, for example, from Economics, in which uncertainties and risks are explicitly measurable. In software development projects, many project decisions are complex once there are important paths to be determined. Therefore, decision-making in the context of uncertainties is very common in software projects (BOEHM, 1984; CHEN, 2001; SANGAIAH et al., 2018).

1.1.1 Problem statement

From the scenario previously described, we can highlight the following issues and opportunities for research related to the use of indicators that support risk management in software development projects:

Primary. The concept of risk in Software Engineering denotes high subjectivity.

The existing approaches in practice suggest that risk management, in particular risk identification and analysis, is biased due to low or wrong perception of risks by project members. It suggests that the recommended approaches to manage software development risks

using quantitative data must be reviewed because in practice only the experience and previous knowledge on risks by people and organization are taken into account (ARTTO; KAHKONEN, 2013). As stated by KONTIO (2001), there is a subjective nature on the understanding of risks in software projects. On the other hand, we cannot just ignore that the accumulated organizational and project data can be a rich source of information to support risk identification, risk strategy formulation, and decision-making. A way to reduce this degree of subjectivity is by using measurement-based approaches – i.e., metrics and indicators. However, the small amount of related works demonstrates that there are few proposals of metrics and indicators for software risk management, so it is a subject that needs more researches.

Secondary. Risk is a relevant subject, but its adoption in practice in software projects tends to be dropped away.

According to BANNERMAN (2015), software risk management research needs more practical studies, and the practice still lacks the prescriptions of research. In fact, in this context the practice of risk management is not wide or consistent (ROPPONEN; LYYTINEN, 1997; ROPPONEN, 1999; NYFJORD; KAJKO-MATTSSON, 2008a,b). From this perspective, it is crucial to assess the impacts of risk management application in environments of software development projects, seeking to extend the study and practices of risk management. Therefore, it is necessary to perform more studies of software risk management and evaluate them especially its practical application.

It is important to note that even if there is no explicit practice of risk management in the management of projects, the risks still exist and project managers should manage them. Evidence suggests that the use of risk management is positively related to project performance (JIANG; KLEIN, 2000; JIANG; KLEIN; DISCENZA, 2001; RAZ; SHENHAR; DVIR, 2002; WALLACE; KEIL, 2004; WALLACE; KEIL; RAI, 2004b; DE BAKKER; BOONSTRA; WORTMANN, 2010; HAN; HUANG, 2007). These studies show that at least the risk factors should be identified and well controlled so that projects may achieve their objectives. This is shown by the study of HAN; HUANG (2007), which concludes that risk factors associated with system requirements seriously affect the project performance. Thus, the identification of risk factors plays a crucial role in the success and the performance of software development projects.

Since there are factors and variables in an environment of software projects, and it is not always possible to execute all the action plans simultaneously, a risk-driven management model could help managers in prioritizing these actions. Therefore, one way of reducing these gaps is to incorporate the use of indicators as an instrument to support risk management in software development projects.

1.2 Research question and objectives

Considering the need of in-depth studies about risk management in practice, despite its recognized importance, and the need of less subjective and more efficient approaches, this work addresses indicators use to support risk management in software development projects. The proposal starts from the premise that indicators can contribute to a better perception of adverse factors and, as a consequence, for better decision-making to avoid unsuccessful projects.

The research assumes that two sources of information are fundamental to define a risk indicator: **risk factors** and **project characterization factors**. Risk factors are the source and cause of each hazard to a project. They are the internal and external factors that contribute to and influence the occurrence of the risk. Project characterization factors, in turn, consist of project characteristics that may directly contribute to increasing its risk levels regarding others in the same environment, and also enable the definition of similarity between projects, allowing the identification of risks and risk response plans using cumulative experience of previous projects. This perspective allows the assessment of one single project, but also multiple projects of an organization, looking at the tactical level of management, and allowing comparisons between projects in the same environment, which is the main focus of this thesis. Additionally, the obtained information also could be useful for project-portfolio management, allowing to maximize the benefits and to optimize the integrated allocation of resources for the organization.

Based on the previously presented context, the main research question investigated by this thesis is:

Is it possible to measure the risk level of a software development project using risk factors and the project characterization factors as source of information?

The main research objective of this thesis is to assess an indicator that aims to measure the risk level of a project in the context of software development projects.

It is important to note that this work also considers the project characterization factors as an important measure of project risk level. Additionally, this kind of information could be useful to identify similar projects, and to define better planning to anticipate the most prevalent risks.

Therefore, this thesis defines the following research specific objectives:

- To map risk factors in the context of software development projects, contributing to the standardization of risk assessment process;
- To evaluate software project characterization factors as an information source for risk measurement in projects;
- To evaluate the proposed indicator through case studies in software industry.

1.3 Methodology

The research design is based on the proposal of WOHLIN; AURUM (2015), that defines a specific model for research in Software Engineering, supporting researchers in decisions regarding the methodology from one or more research questions (Figure 1). The structure is composed by eight decision points: (i) Outcome, (ii) Logic, (iii) Purpose, (iv) Approach, (v) Process, (vi) Methodology, (vii) Data collection methods, and (viii) Data analysis methods. The authors organize these points in phases:

- **Strategic:** planning that guides the researcher to the other phases. It allows research to be carried out systematically and positions it regarding the other approaches. This phase involves decisions points (i) to (iv);
- **Tactical:** it encompasses decisions about how to operationalize the activities so that it will address the research questions in a more specific way. It involves decisions points (v) and (vi);
- **Operational:** it is the execution of the actions decided in the previous phases, this involves analysis and collection of data. The decisions points (vii) and (viii) are associated with this phase.

Given a problem or a research question, these phases are executed, not necessarily in a sequential way, generating research outcomes during and at the end of the work.

1.3.1 Research classifications

The research design according to the previously presented decision points is presented in Figure 1 (based on WOHLIN; AURUM (2015)). We conducted applied research, as this work is focused on the solution of a specific problem, to reduce the subjectivity levels present in processes of risk management of software development projects through the proposition of an indicator. However, we used the studies only to assess the proposal, once the assessed environments did used any kind of risk indicator in practice.

The research logic is inductive, because it focuses on the understanding of the indicator through some empirical studies, aiming to analyze its applicability, generating more general conclusions and theories.

The research purpose is exploratory and descriptive, as initially, it aims to deepen a theme that has few related works, and in the end, we performed experimental studies on a proposal elaborated in the exploratory step. It used an interpretivist approach and the primary purpose of this research is to understand the broader structure of phenomena recognizing the participants' perspective. Hence, the participants used their subjectivity and experiences to give meaning to the analyzed events.

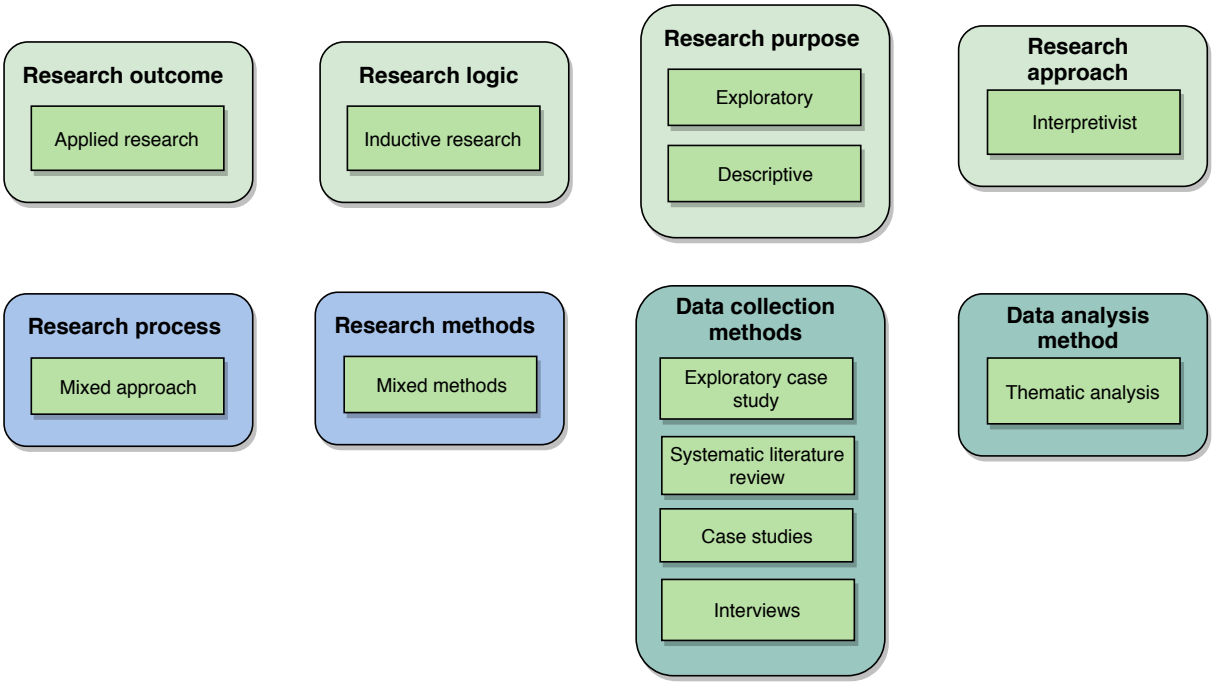

Figure 1 – Research classifications

Regarding the process, the research used a mixed approach, i.e., qualitative and quantitative methods, seeking to establish characteristics from the performed study. It is pertinent because the analysis of the indicator involves numerical and qualitative data. We also used a mixed approach in research methodology, once it was a research inquiry that employed more than one method of data collection from different sources to investigate a phenomenon. In this sense, we used four data collecting methods: an exploratory case study, a systematic literature review, two case studies, and two interviews. The exploratory case study presents the application of an existing proposal of an indicator and its variations in a real software project environment. Risk factors were mapped through a systematic literature review. Then, we performed case studies in environments of software development projects with the proposed indicator, and interviewed project members to get feedback and to evaluate the proposed indicator. Finally, we performed thematic analysis as data collection method, aiming to obtain a broad understanding of the data collected throughout this work.

Next section details the necessary steps to execute the proposed research method.

1.3.2 Steps

This work was organized in steps, as presented in Figure 2 as follows.

Step 1: Theoretical methodological positioning - This step defined the research methods that were adopted for this work. Several concepts and related works were studied as a way to better determine the paths to be traced throughout the research. Additionally, we explored the literature about research methodology in software engineering. This chapter can be defined as the main result of this step.

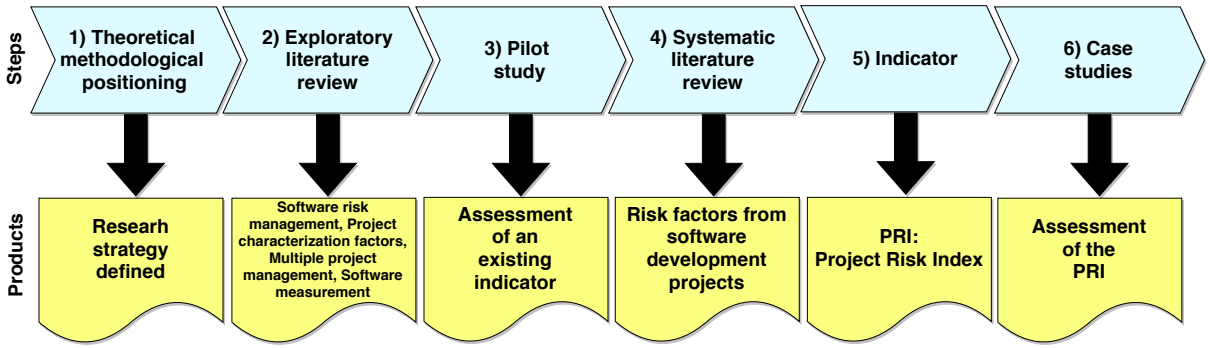

Figure 2 – Research steps

Step 2: Exploratory literature review - It consisted of a broader literature review of concepts related to this work, such as Metrics, Indicators, Risk Management, Software Project Management, Software Processes, and Software Process Improvement. It aimed to identify gaps and to characterize the problem statement better. It also involves a preliminary mapping of risk factors specific to environments of software development projects. Some results of this step were published as follows:

- PEREIRA, C. G.; MENEZES JR., J. V.; GUSMÃO, C. M. G. Proposta de estrutura analítica de riscos para ambientes de múltiplos projetos de software. In: INTERNATIONAL CONFERENCE ON INFORMATION SYSTEMS AND TECHNOLOGY MANAGEMENT. **Proceedings...** 2015
- WANDERLEY, M. et al. Proposal of risk management metrics for multiple project software development. **Procedia Computer Science**, v.64, p.1001–1009, 2015

Step 3: Pilot study - It consists of an exploratory case study, whose main goal was to use the Risk Point indicator (LOPES, 2005) in an environment of software development projects. We conduct a previous exploratory case study to get more insights about the risk factors and project characterizing factors, and to identify improvement points to perform in-depth studies to generate the proposal of PRI - Project Risk Index. The results of this step were published in:

- MENEZES JR., J. et al. Application of Metrics for Risk Management in Environment of Multiple Software Development Projects. In: INTERNATIONAL CONFERENCE ON ENTERPRISE INFORMATION SYSTEMS - VOLUME 1: ICEIS, 18. **Proceedings...** ScitePress, 2016. p.504–511

The results of the pilot study helped to deepen the proposal of an indicator using risk factors and project characterization factors as sources of information to measure risk level. The first is the main source and the second is the secondary source, that helped to better valuate the proposal, once there is a relationship between software risks and project characteristics in which these

characteristics may lead a project to failure (MCMANUS, 2012).

Step 4: Systematic literature review - It consists of a systematic literature review with the objective of identifying risk factors in environments of software development projects. With the result of this review, the risk factors were identified and categorized according to software development taxonomy developed by Software Engineering Institute (SEI) (CARR et al., 1993). The result also allowed the identification of the most relevant factors and the proposition of the indicator, defining weights for each risk factor based on its occurrence in the selected papers. The results of this effort were published at:

- MENEZES, J.; GUSMÃO, C.; MOURA, H. Risk factors in software development projects: a systematic literature review. **Software Quality Journal**, p.1–26, 2018

Step 5: Indicator - This step generates as the primary product a proposal of indicator, called Project Risk Index (PRI), whose goal is to measure the risk level of a software development project. The parameters defined for this indicator are composed by the multiplication of the sum of risk factors by project characterization factors values.

Step 6: Case studies - Aims to evaluate if the used parameters (risk factors and project characterization factors) contribute to the calculation of risk level of a software development project, as well as to identify points of improvements on the proposed indicator. At the end of the study, the participants were interviewed. Steps 5 and 6 generate the following publication:

- MENEZES, J.; GUSMÃO, C.; MOURA, H. Proposal of indicator to support risk management in environments of software development projects. **Journal of Convergence Information Technology**, v.13, n.3, p.41–52, 2018

1.4 Thesis structure

Besides this introductory chapter, the whole document is organized into five chapters as follows:

Chapter 2 - Theoretical foundation - this chapter presents theoretical foundations on risk management in environments of software projects, indicators and metrics in software engineering, related work, and a pilot study using a related work.

Chapter 3 - Risk factors in software development projects - this chapter presents a systematic literature review that aims to identify and to categorize risk factors in software development projects.

Chapter 4 - Assessment of the indicator - presents the risk indicator, called Project Risk Index (PRI), which uses two parameters: risk factors and project characterization factors. Additionally,

this proposal is evaluated through case studies.

Chapter 5 - Conclusions - this final chapter presents conclusions, including main contributions, limitations, and future work.

References used to build the thesis and appendices are found at the end of this document.

2 BACKGROUND

This chapter aims at presenting the central concepts that support this work: risk, risk management, multiple project management and software measurement.

2.1 Risks

Project management is not a trivial task. A set of factors, both quantitative and qualitative, are necessary to guarantee project success. One of the factors that practically permeates a whole life cycle of a project is risk. Moreover, its most prominent sources of information are the uncertainties. The term "risk" derives from the early Italian risicare, which means "to dare" (GERRARD; THOMPSON, 2002). As a science, the risk was born in 16th century, during the Renaissance, practically laying the foundations of probability theory (HALL, 1998). The main people that contributed to the modern definition of risk were Pascal and Bernoulli.

Pascal made contributions through solving gambling by a method to determine the probability of occurrence of possible outcomes (ROBINSON, 2013). Bernoulli, in turn, introduced the concept of utility, which is a function that measures the consequence of a given result to happen through risk assessment (COLMAN, 2016), so that his theory recognizes the importance of perception from people who is assessing risk. Utility theory has laid the foundations of the law of supply and demand in Economics. Combining both approaches, we can affirm that risk is associated to decision-making by the measure of uncertainty using analysis of probability related to possible losses (consequences), and these values can be dependent on the perception of who is measuring, depending on its degree of uncertainty.

The risk is strongly associated with achieving gains (achieving objectives) over a set of actions that needs to be previously analyzed and monitored. On the other hand, according to the Oxford Dictionary, a risk is a situation that involves exposure to danger (OXFORD, 2017). In fact, risk is associated not only with losses but also with opportunities, and project management community in general, recognizes this idea, as can be seen on Project Management Knowledge Guide – PMBOK (PMI, 2013), that defines a project risk as "event or uncertain condition that, if it occurs, will have a positive or negative effect on one or more project objectives".

Other authors, specifically in Software Engineering, focus on the perception that risk measure is associated with adverse factor measures (HALL, 1998; MCMANUS, 2012), using as parameters the probability of loss and the consequences in case of loss, in which the product of them we call Risk Exposure. Although the positive factors are not explicit in this logic of thought, it is important to note that the analysis of these factors contributes directly to the reduction of risk levels, which, in turn, maximizes the chances of success of projects.

PFLEEGER; HATTON; HOWELL (2001) and BOEHM (1989) suggest that three aspects are associated to software risk: (i) the loss associated to event, (ii) the probability of occurrence of the event, and (iii) the degree to which we can change the consequences of the event, being

defined as risk exposure, that is the product of the likelihood by the loss. Hence, the risk exposure would be determined quantitatively. However, the calculation of risk exposure is more an art than a science (PFLEEGER; HATTON; HOWELL, 2001), because there are no effective techniques for estimation of probability and loss of risks (FARIAS, 2002).

In project-based organization environments, this context is not different. All projects involve risks. Decisions are made daily in various managerial and individual levels to achieve project goals and objectives. Risks prevent us from achieving these goals and objectives (HELDMAN, 2010). According to HALL (1998), there are three categories of software risks:

- **Project risks:** associated with operational, organizational and contractual parameters. It is primarily tied to management and includes resource and contract constraints, external interfaces and supplier relationships;
- **Process risks:** adds aspects of management and technical procedures. In the first, it encompasses process activities, such as staffing, monitoring, quality assurance and configuration management. In technical procedures it is associated with software development activities, such as requirement analysis, design, and architecture, coding and testing;
- **Product risk:** encompasses the characteristics of software products, being primarily a technical responsibility. It is associated with requirements, architecture, coding and testing issues on a software product.

Figure 3 presents a hierarchy of responsibility for each category of software risk. As can be seen, it shows the relationships between them, as it considers two perspectives: management and technical, so that the first one derives from Project and Process risks, and the second one approach Product and also Process risks. For the correct management and controlling of risks, a set of actions and methods must be performed, as presented as a follows.

2.2 Risk management

Risk management is the art and science of identification, analysis, and response to risks during a project life cycle, with the interests of achieving the project objectives (SCHWALBE, 2015). The primary objective of risk management is to identify potential threats, analyze them to determine the ones that are most likely to occur, identifying the ones that can bring most significant impact to a project, in case of occurrence, and, finally, to define plans to mitigate or reduce the consequences or avoid risks, aiming to optimize project objectives (HELDMAN, 2010).

Most of the researches reported in literature about software risk management focuses on the identification and analysis of risk – i.e. the assessment of risks (BOEHM, 1991; FAIRLEY, 1994; DOROFEE et al., 1996; HALL, 1998; KONTIO, 2001; SIMPLEMAN et al., 1998;

VAN LOON, 2007). However, according to BANNERMAN (2015), few studies focus on the adoption of risk management in software development projects, since there is an awareness regarding the usage of its practice, but it is poorly applied. There is also a trend to start its practices, but over time it is neglected regardless of it being an integral part of management practices for software engineering. That is, in general, "risk management adoption lags the prescriptions of research" (BANNERMAN, 2015).

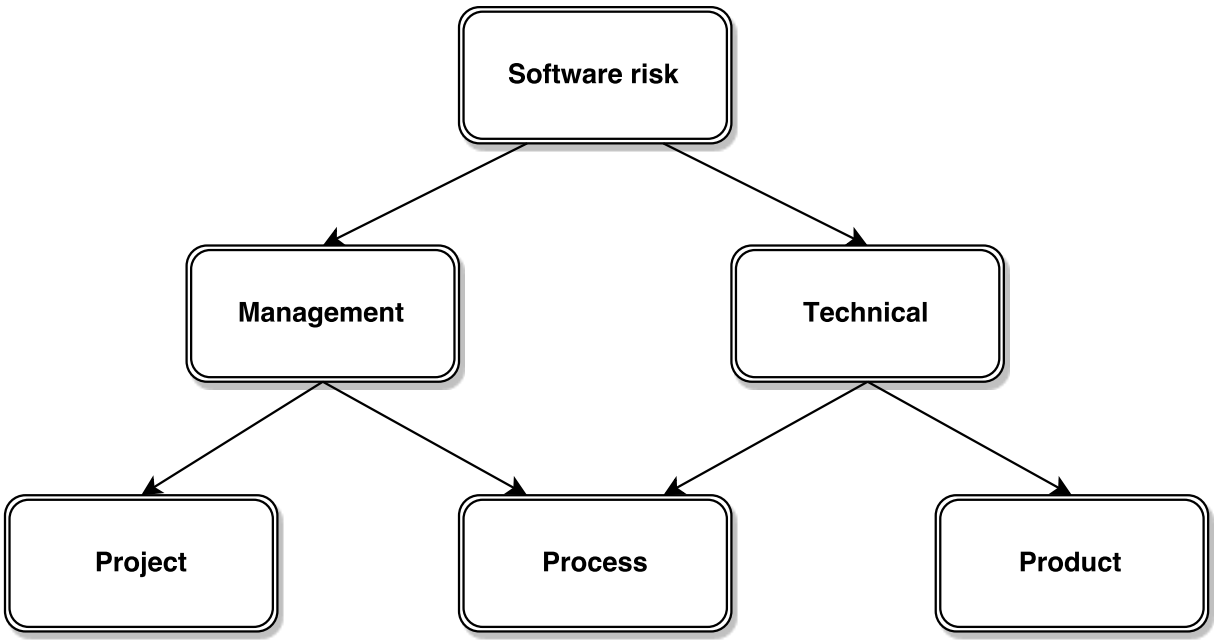

Figure 3 – Software risk classification (HALL, 1998)

If the research evidence suggests that risk management is of great value contributing to the success of the project results, why is the adoption of risk management practice so limited? The response can be based at least in part, on a set of barriers that can reduce or make the implementation of risk management unattractive in practice: cost, culture, uncertain benefits, capability, ownership, bad reviews, and reward structures. Therefore, the following enablers of risk management adoption and utilization may avoid these barriers: senior management commitment and support, governance, empowerment, culture, continuous improvement, and highlighting successes (BANNERMAN, 2015).

Risk management is the application of skills, knowledge, tools, and techniques to reduce threats to an acceptable level while maximizing opportunities (HELDMAN, 2010). Moreover, the risk management in software projects is the practice of assessment and control of risks that affect projects, processes, and products of software (HALL, 1998). An important point to be considered in software projects is communication, especially of technical risks, that are often known but poorly communicated (CARR et al., 1993).

In the last years, several works related to the application of artificial intelligence techniques in risk assessment have appeared, such as grey correlation (QINGHUA, 2009), (FU; LI; CHEN, 2012), fuzzy theory (TANG; WANG, 2010; SALMERON; LOPEZ, 2010), Bayesian

networks (FAN; YU, 2004) and case-based reasoning (TRIGO; GUSMÃO; LINS, 2008). Most of these techniques use risk factors as a source of information to predict risks.

The area of risk management in software projects, even considered important still needs several advances, both in research and in practice. Approaches to risk management based on risk factors, on the process and on modeling are evolving and being improved with sociocultural analysis, skills and data analysis (BANNERMAN, 2015). Some studies show that the explicit practice of risk management contributes to performance improvement and increases the success of projects (JIANG; KLEIN, 2000; JIANG; KLEIN; DISCENZA, 2001; RAZ; SHENHAR; DVIR, 2002; WALLACE; KEIL; RAI, 2004a,b; WALLACE; KEIL, 2004; DE BAKKER; BOONSTRA; WORTMANN, 2010; HAN; HUANG, 2007). We can also affirm that assuming risks, associated opportunities emerge, but, on the other hand, managing these factors requires specific skills too.

From this point of view, the risk management becomes crucial, since software industry continually deals with resource shortages, technological advances and the increasing demand for complex systems in environments that undergo continuous modifications. In this scenario, considering the difficulties of organizations to stay in an increasingly globalized economy, with uncertain market conditions, and at the same time pressed by rapid technological advances, the use of risk management is fundamental among software engineers and project managers.

For effective management of risks in an environment of projects, it is necessary the support of the senior management for the implementation of risk management processes (MCMANUS, 2012). To perform good risk management, MCMANUS (2012) says that an organization must:

- Set the tune and the influence of the culture on risk management in the environment;
- Determine the tolerance levels to risks;
- Has active participation in major decisions in establishing risk profile or exposure;
- Manage the most relevant risks, aiming to reduce the probability and undesired surprises, through risk indicators.

In general, there is a consensus among the researchers and practitioners that the following activities comprise the risk management process (CARR et al., 1993; HALL, 1998; PURDY, 2010; PMI, 2013; SCHWALBE, 2015):

- **Risk management planning:** involves deciding how to approach and plan the risk management activities. It generates as main artifact a risk management plan;
- **Risk identification:** determines the risks and risk factors that can affect a project and documents the characteristics of each one. It generates a register of risks;
- **Qualitative risk analysis:** prioritizes the identified risks according to probability and impact of their occurrence;

- **Quantitative risk analysis:** numerically estimates the risk degree of prioritized risks on qualitative risk analysis;
- **Risk response planning:** involves performing of tasks to reduce threats, increasing opportunities with the goal to achieve project objectives;
- **Risk controlling:** consists on monitoring identified and residual risks, identifying new risks, performing the planning of new risk responses, and assessing the effectiveness of adopted risk management strategies.

Another mentioned, and essential activity is the risk communication because it permeates all phases of risk management. It is a critical component of successful management, and its role is primarily to provide, to exchange and to obtain information among stakeholders, directly supporting all other activities.

2.3 Risk factors

Fundamentally, risk factors are the source and cause of each hazard (SILVA, 2011; BRASILIANO, 2009). They are the internal and external factors that contribute to and influence the occurrence of the risk. In this perspective, risk is the sum of a set of risk factors that, if materialized, constitute a hazard. The risk statement, then, is composed by a condition and a consequence. Therefore, risk is an abstract and complex concept and one way to make it more concrete is by risk factors, so that help to reduce the level of abstraction of risk.

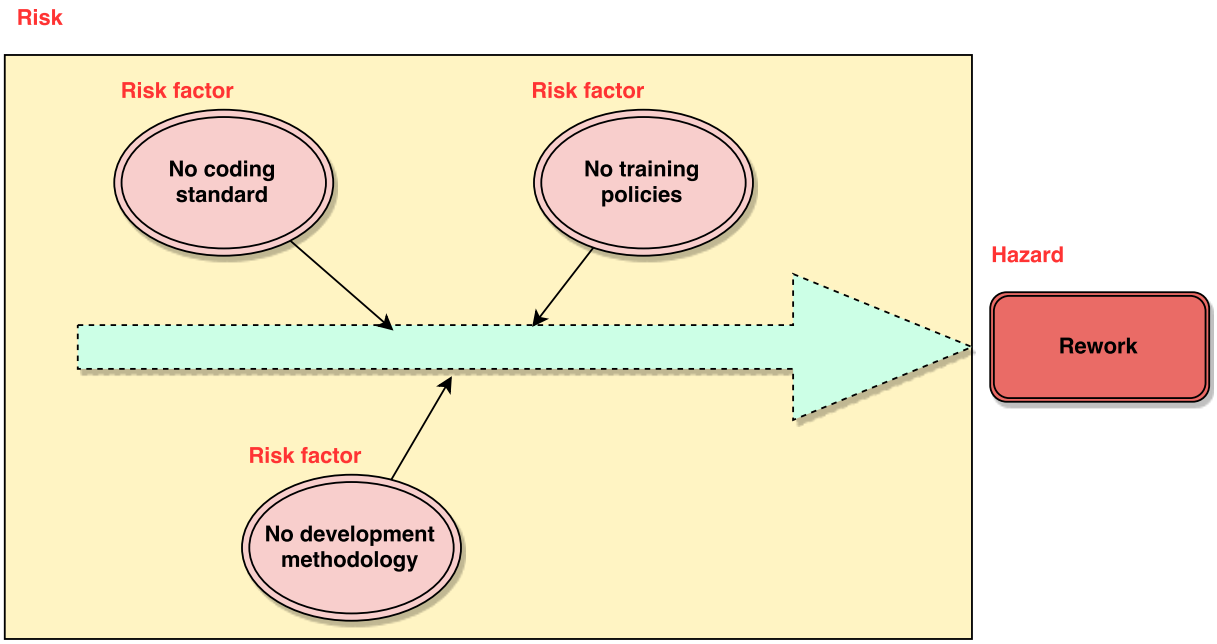

Figure 4 – Risks versus Risk factors versus Hazards

For example, a particular organization develops software without following a development methodology, or a coding standard, much less there are training policies. These are

conditions that lead to a hazard called rework. That is, the risk can be understood as a set of factors that lead to a certain hazard. This situation is better presented in Figure 4.

A risk factor alone may not adequately aggregate value in project management, once it only makes sense when there is a clear notion of probability of occurrence and the associated impacts and consequences. For this reason, a risk classification may provide a better identification of risks sources. A Risk Breakdown Structure (RBS) is an exciting tool, since it is a risk-oriented grouping, which organizes in a structured way, classifies and defines the exposure of the identified project risks, providing a hierarchical structure of potential risk factors (HILLSON, 2002). One known proposal of RBS for software projects is the risk taxonomy of CARR et al. (1993), developed by Software Engineering Institute – SEI, which provides a framework to support the identification of risks in software development projects.

2.4 Project characterization factors

MCMANUS (2012) suggests that there is a relationship between software risks and the characteristics of the project. Such characteristics include size, complexity, timeframe, technological innovation, uncertainty, governance, and management. They are related to the nature of software development projects, which involve multidisciplinary teams, with a high interaction between software, hardware, and people. Another point to highlight is the usefulness of these characteristics to identify similar projects, promoting more optimized strategies for risk monitoring, considering that similar projects tend to have similar risks in the same environment.

In this light, we can say that certain characteristics of a project may contribute to a higher level of risk than others. For example, CASH; MCKENNEY; MCFARLAN (1992) makes these factors explicit:

1. The project size such as staffing levels, duration, cost, and number of departments affected;

2. The technology such as level of technical expertise, emergent technology to be used, technical difficulty of design and experience of similar projects;

3. The project structures such as clear requirements, fixed and certain outcomes, and the absence of changes in specification during the lifetime of the project.

In the view of project characterization for risk assessment, COSTA (2005) defined one characteristic: project size, determined by the product between project duration and an average number of people in the project. These characteristics are used to group projects in the process of assessment of project risk level from a portfolio and to determine project size. The author does not detail this characteristic but mentions that there is a gap regarding project characterization in Software Engineering, which led the author to adopt a simplified characterization in the study.

The work of TRIGO; GUSMÃO; LINS (2008) uses project characterization for identifying similar projects. The authors started from the premise that projects with similar characteristics

lead to similar risks. Thus, they developed a risk-identification method using case-based reasoning technique (KOLODNER, 2014). Given some characteristics of a previously informed project, the method identifies similar past projects and raises risks of these projects. The authors use the following information to characterize software development projects:

- **Team size:** the composition of the number of people in the project team;
- **Geographical distribution of the team:** based on the location of the development team members and their work environments;
- **Experience of the development team:** which mixes experience and technical knowledge in software development;
- **Type of the project:** used to represent the project class, and;
- **Technological platform:** which determines the used software development technology, considering that the choice of platform limit resources in a general way.

The authors use an attribute-value representation, except for the characteristics "Type of project" and "Technological platform". Therefore, for purposes of calculating project similarity, these characteristics are not considered, but only for grouping projects, because they were not numeric values.

It is worth mentioning that the characteristics used by TRIGO; GUSMÃO; LINS (2008) are based on the model proposed by COELHO (2003), which proposes a characterization model to make feasible the comparison between projects based on similarity. It aims to apply similar development processes for similar projects. For this, three types of characteristics are defined:

- **Development characteristics:** which take into account only aspects related to software development, without considering organizational, contractual or market factors. The following characteristics are analyzed:
    1. **Team size:** which the relationship with the form of communication is emphasized, which becomes more complex as the team grows;
    2. **Geographical distribution of the team:** which, according to the author, it also has a strong relationship with the form of communication and the quality of the developed artifacts during the project life-cycle, affecting the team productivity;
    3. **Team experience:** as it is a complex characteristic, the author divided it into three - technical, application domain and development process;
    4. **Software criticality:** it refers to the severity of the consequences of a possible system failure to the project;

5. **Project size:** the author reinforces the importance of this characteristic, once large projects are less likely to be successful than small projects (JONES, 1998; ROYCE, 1998). For this, the author uses the project budget as a measure of project size.

- **Restrictive characteristics:** they impose constraints on a software development process. They are usually associated with tools, patterns, contractual requirements, and time and budget constraints;

- **Priorities of the project:** they are analyzed based on the relationship between the desired software quality and the necessary time to develop the software. It starts from the idea that both are conflicting attributes, being necessary that the project manager decide if the project priority is the quality or the time, considering that the higher is the product quality, the longer time it takes for its development.

A well-known model in the area is COCOMO II (BOEHM et al., 2000), which is used to estimate cost, effort, and schedule during the planning phase of a software project. For this, among other information, some characteristics of the project are raised:

- **Software scale drivers:** defined as the characteristics that may amplify the effort:
  - Precedentedness
  - Development flexibility
  - Architecture / risk resolution
  - Team cohesion
  - Process maturity

- **Software cost drivers:** that focus on resources of software development, being organized in four groups:
  - *Product*
    - Required software reliability
    - Database size
    - Product complexity
    - Developed for reusability
    - Documentation match to lifecycle needs
  - *Personnel*
    - Analyst capability
    - Programmer capability

        - Personnel continuity
        - Application experience
        - Platform experience
        - Language and toolset experience
    - *Platform*
        - Time constraint
        - Storage constraint
        - Platform volatility
    - *Project*
        - Use of software tools
        - Multisite development
        - Required development schedule

VIJAYASARATHY; BUTLER (2016) present a survey comparing some projects characteristics present in the leading software development methodologies. These are: (i) budget, (ii) project criticality, and (iii) team size. The purpose of these characteristics was to characterize software development projects according to the most common software development methodologies – agile, traditional, iterative, and hybrid.

Table 1 presents the characteristics that were mentioned more than once considering the works of COELHO (2003), COSTA (2005), TRIGO; GUSMÃO; LINS (2008), and VIJAYASARATHY; BUTLER (2016). As can be seen, only four characteristics presented this behavior.

Table 1 – Comparative analysis of software project characteristics

Work	**Characteristics**			
	Team size	**Geographical distribution**	**Team experience**	**Project size**
COELHO (2003)	X	X	X	X
COSTA (2005)	X			X
TRIGO; GUSMÃO; LINS (2008)	X	X	X	X
BOEHM et al. (2000)		X	X	
VIJAYASARATHY; BUTLER (2016)	X			X

It is important to notice that team size can vary throughout the project life-cycle, so this characteristic indicates the average team size. The characteristic “project size”, cited by three works, is defined by the budget. According to COELHO (2003), the usage of project costs is a way to contemplate the product size and the required effort to develop the project, so that the higher the cost, more critical is the project for the organization, as well as its monitoring.

In an environment with several simultaneous projects, the characterization contributes to the grouping and comparisons between the projects. It is crucial when considering the comparison of risk levels between different projects since project characterization factors are information that, alone, can make the project riskier.

2.5 Multiple project management

Managing one single project is not often enough in organizations because many are managed in multiple project environments, in which projects are executed simultaneously. This scenario is necessary since it aims to raise more financial resources to be reinvested, to cover expenses, and to bring profit to the organization (GUSMÃO, 2007). Even if a separate project presents an appropriate planning and effective control during its life cycle, its execution does not always happen as planned. Therefore, the existence of other projects becomes crucial to cover expenses and unforeseen, as well as give greater security to the organization (FREITAS, 2005).

The management of multiple projects within the same environment sees a set of projects as a single entity, respecting their interdependencies and aiming at common goals (ALENCAR; SANTANA, 2010). Approximately 90% of project resources are used in some context of multiple projects (ALENCAR; SANTANA, 2010; DANILOVIC; BÖRJESSON, 2001; FREITAS, 2005). Therefore, the need to manage various projects arose from the need to optimize resources, such as time, costs and people, within a view of increasing competition among organizations.

In general, an organization is structured in three levels of management: Strategic, Tactical and Operational (WYSOCKI, 2011). Senior management is part of the strategic level, encompassing global goals in a medium and long-term, according to organization strategy. Besides, the strategic management defines how project objectives achieve the business objectives and strategies (FRIGENTI; COMNINOS, 2002).

The management of multiple projects occurs in tactical level, and it consists of the definition of tasks to be executed to ensure that the defined objectives by strategic management are achieved. Tactical management focuses on checking the accomplishment of planned activities of the projects, and the necessary allocation of resources – human, financial and operational infrastructure – allowing the progress of activities (FREITAS; MOURA, 2004).

Hence, the tactical level seeks to optimize the usage of resources aiming to achieve the strategic objectives of the organization. Finally, the operational level sees the management of one single project, commonly considering the team and constraints of scope, cost, schedule, and quality.

Managing multiple projects allow the organization to manage risks, obtaining values in an integrated and holistic way (ALENCAR; SANTANA, 2010). In this sense, metrics and indicators are useful instruments in capturing information for the generation of these values. It is important to note that at this level of management, the projects will be in different stages in their respective life cycles, in which favors the continuous reallocation of resources. For example, in

the final phases of a project, there is no longer a need to maintain the same development team, so part of it can be reallocated to a project that is in the implementation phase.

Another typical situation in the environment of multiple projects is the projects reprioritization due to strategic and organizational issues. That is, this kind of scenario must be perceived straightforwardly, allowing managers the correct decision making, considering that there is a pool of projects that should not stop. For this, it is needed to have clarity and knowledge of the resources that are being used in the same environment, as well as specific skills. These skills primarily involve the separation of responsibilities with project managers, without having full control over details of each project (ALENCAR; SANTANA, 2010), as well as to carefully look at feedbacks from external influences (ARITUA; SMITH; BOWER, 2009).

DANILOVIC; BÖRJESSON (2001) mention three different types of multiple project environments: converging, diverging and parallel. The converging multiproject environment has a subproject and may in another case be an independent or a significant project containing other subprojects, but always converging to the development of one single product or service. This kind of environment is common in car and aircraft development/manufacturing corporations.

The diverging multiproject presents an environment in which different projects share the same background technology, i.e., platform, and product or business decision. Examples of this environment: car industry and software product lines, in which the same platform/chassis/engine/framework are used to develop a variety of car models or software products or services.

Finally, the parallel multiproject environment presents different independent projects, even if they may share specific resources, such as people, knowledge basis, among others. This environment focuses on the management of resources aiming to optimize projects and tasks.

2.6 Measurement in software engineering

The use of measurement is encouraged in Software Engineering community. According to ISO/IEC/IEEE 24765 (2010), Software Engineering is defined as “the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software”. As can be seen, measurement is an inherent activity of software engineering itself. Hence, metrics and indicators are used as key instruments in the measurement process, at the level of processes, projects and products (PRESSMAN; MAXIM, 2014; SINGH; SINGH; SINGH, 2011):

- Process measurement collects information about all projects life cycle in an environment. Its main goal is to provide a set of indicators to perform process improvement. This domain acts on both strategic and tactical levels, contributing especially to organizational success through quality models.

- Project measurement allows project managers: i) to assess the project status, ii) to monitor potential risks, iii) to find out areas or problems before they become

critical, iv) to make adjustments on workflows or tasks, and v) to assess project team capability to control the quality of work packages of software. It is helpful to tactical and operational levels of management, allowing the better control of projects, identifying opportunities and points of improvement considering main constraints – cost, time, scope and quality.

- Product measurement uses the software as the source of data and information. The main goal of this domain is to check product quality, including the final product (the software) and generated artifacts during its life cycle.

FENTON; PFLEEGER (1997) define measurement as "...the process by which numbers or symbols are assigned to attributes of entities in the real world in such a way as to describe them according to clearly defined rules".

In this sense, we understand that the measurement process essentially gives information about some attribute of an entity, in which:

- **Entity** is the object or event of interest which will be measured;
- **Attribute** is a characteristic or property of the entity to be measured.

For example, an entity might be a software artifact, a team or a person, a phase of a software development process, software process, among others. Attributes extract information from an entity according to defined aspects, such as estimated cost (of a project), elapsed time (of the testing phase), effort (to develop a set of requirements from an iteration, among others.

It is worth highlighting that measurement, if it is well done, gives information that generates knowledge for decision-making, as well as provides better control and tracking of the undertaken activities. Therefore, the measurement process contributes to continuous software process improvement.

According to PARK; GOETHERT; FLORAC (1996), there are four reasons to perform measurement in software engineering: i) to **characterize** for better understanding of processes, products, resources, and environment, establishing reference models for further evaluations; ii) to **evaluate** the status according to what was planned; iii) to **predict**, in other words, get enough basis to estimate, minimizing risks, and; iv) to **improve** the product quality and the performance of process and project through concrete quantitative data, which helps in visualizing improvement points more clearly.

In project management, the use of measurement can bring more consistency and formality. Therefore, it brings objectivity to the tools for monitoring project progress, also allowing uniformity, accuracy, and repeatability (RAD; LEVIN, 2006).

For the Software Engineering Institute (SEI), the act of measurement provides support to management, development, and maintenance of software projects (MCGARRY, 2002). According to IEEE (1998), the use of metrics reduces subjectivity in quality software assessment by providing quantitative information about product and process, making software quality more

visible. For PSM - Practical Software Measurement (MCGARRY, 2002), software measurement provides objective information to support project managers in the following aspects:

- Effective communication, through objective information;
- Follow-up of project objectives;
- Identification and correction of problems in advance;
- Support for more critical decision-making;
- Justify decision-making.

On the other hand, software measurement is still a young technology (SARAIVA, 2014; ABRAN, 2010). There are still difficulties in setting measurable goals regarding the products. For example, how can we measure that a product is easy to use, reliable and easy to maintain? Another point that makes difficult the use of metrics is the low precision in quantifying software components costs. We can affirm that there is still a low level of care in software organizations regarding controlled studies that could determine the effectiveness of new technologies and processes. Therefore, the use of measurement in software engineering aims to improve the understanding, and to control and improve processes, products and projects (FENTON; PFLEEGER, 1997; PRESSMAN; MAXIM, 2014).

HUMPHREY (1989) says that the primary motivations for using software measurement are: to understand, to evaluate, to control and to predict, always in the context of processes, products, and services. In this viewpoint, the use of software measurement is fundamental for an organization, since it can be executed at different management levels.

According to FENTON; PFLEEGER (1997), the most beneficial software engineering activities derived from the application of software measurement are: (i) estimation of costs and efforts, (ii) models and measures of productivity, (iii) data collection, (iv) measures and models of software quality, (v) reliability models, (vi) performance assessment, (vii) structure and complexity, (viii) assessment of capacity and maturity, (ix) management, and (x) assessment of methods and tools.

In the field of software project management, metrics and indicators can be essential decision-making instruments. PRESSMAN; MAXIM (2014) indicates that project metrics and indicators are tactical, that is, they can be used by project managers and respective teams for adapting workflow and technical activities of the project. Hence, measurement at the project level is useful to perform estimations, as well as to monitor and to control project progress.

2.6.1 Indicators and metrics

The difference between the terms “Metrics” and “Indicators” often is not so clear in software engineering. The fundamental differences lie in the purpose and the refinement of

information. KERZNER (2013) says that “an indicator is a measure that provides insight to an information requirement and supports decision making”.

An indicator can consist of one or more metrics. Figure 5 represents the main concepts of metrics and indicators. It is based on the standard ISO/IEC 15939 (2017), which proposes a measurement process applicable to software organizations. It also provides main concepts and terms about the subject. CMMI maturity model (CMMI-DEV, 2010) uses this standard as the basis for the process area Measurement and Analysis.

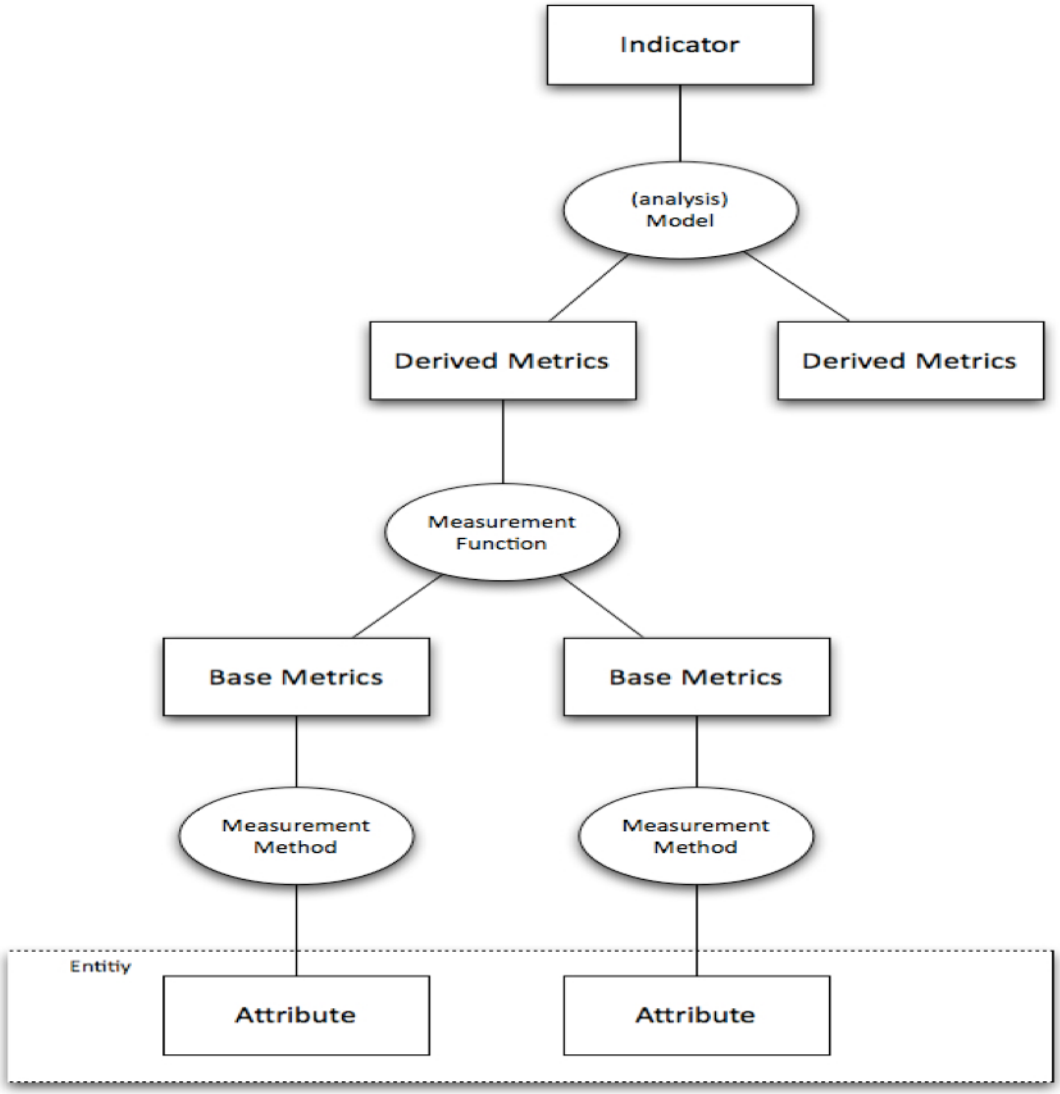

Figure 5 – Measurement information model (adapted from ISO/IEC 15939 (2017))

Software metrics can be defined as the continuous application of measurement-based techniques to the software development process and its product to supply meaningful and timely management information, together with the use of those techniques to improve that process and its products (GOODMAN, 2004).

There are two types of metrics: base and derived. A base metric is defined regarding a

single attribute and the method for quantifying it. It is functionally independent of other metrics. The measurement method comprises the logical sequence of operators to quantify an attribute to a specified scale. There are two types of measurement methods: subjective and objective. The first one quantifies an attribute through human judgment, whereas the objective method is based on numerical rules.

A derived metric is defined as a function of two or more values of base metrics. The function to characterize a derived metric must be a mathematical function between two or more base metrics (measurement function). The derived metrics capture information about more than one attribute or the same attribute from multiple entities. The basis of metrics is the measurement, which can be defined as a process by which symbols and numbers are assigned to attributes of entities from the real world so that they can be described through clearly defined rules (FENTON; PFLEEGER, 1997).

Indicators, in turn, are metrics that provide additional information on estimations or assessment of specified attributes. They are the basis of activities of analysis or decision-making process. Indicators are defined from a model of analysis that establishes the relationship between two or more metrics through its behavior over time and as a decision criterion. An indicator compares metrics to an expected result, allowing the decision-making through the view of the real situation of project aspects.

An indicator can be used to represent information in a clear and objective way (CARDOSO, 2004). It is also useful to understand the current situation (where you are), the way to be followed (how to get there) and how far you can go to achieve the defined goal (where you want to go).

Indicators were developed with the goal of treating the information to make it accessible, allowing the understanding of complex phenomena. They make these phenomena quantifiable and understandable so that they can be analyzed and used with the appropriate policies planning. They also allow the organizational and process improvement by optimizing the management of information.

Indicators should be as specific as possible to the subject treated; sensitive to specific changes in conditions of interest; scientifically reliable; unbiased and representative and they should provide maximum benefit and utility. Some criteria must be considered to select, define and make effective use of indicators (KLIGERMAN et al., 2007):

- Existence of database of attributes and parameters that comprise the metrics;
- Possibility of intercalibration, such as weights, constant values, or improvements;
- Total number of selected indicators, according to measurement objectives of the organization or environment;
- Type of information transmitted, including the nature of information, the functional processes related to the system and which public will receive this information;

- Possibility of comparison with another criterion or existing standards and goals;
- Cost of implementation and;
- Quick update.

Figure 6 describes the measurement process in software engineering. It is important to notice that it is a continuous process, so it is necessary to provide an infrastructure to collect and maintain data. Therefore, the calculation and assessment of these metrics are possible, contributing to controlling, assessment and improvement of processes, projects, and products.

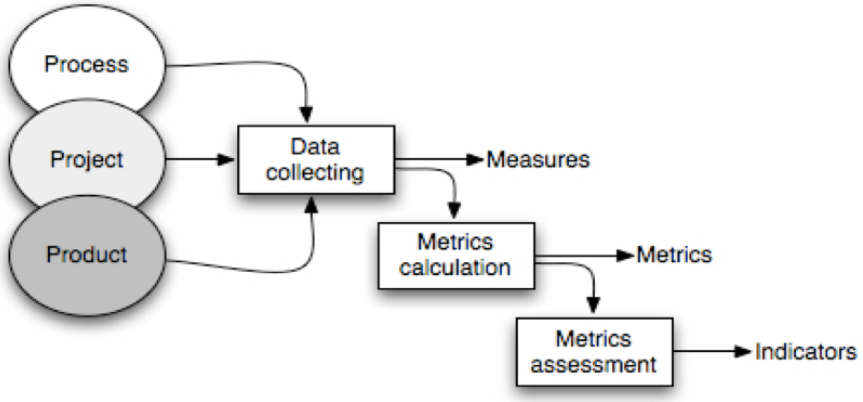

Figure 6 – Process to collect software metrics
(PRESSMAN; MAXIM, 2014)

The data collecting process uses information from process, project or product, where its calculation provides software metrics. These metrics must be evaluated and applied during the estimation, technical work, project control and process improvement. The assessment of metrics produces a set of indicators that gives guidelines to project or process.

2.7 Related work

There are few references in Software Engineering regarding the use of metrics and indicators for risk measurement in software projects. Another limiting point is the little execution of experimental studies that evaluate the proposals. Amongst some proposed metrics for software risks, two are worth mentioning given their pioneering and relevance: Risk Exposure and Risk Leverage (BOEHM, 1989).

Risk Exposure (RE(risk)) is defined as follows:

$$RE(risk) = Probability(risk) * Impact(risk) \tag{2.1}$$

The Probability defines the probability of occurrence of a risk, and the Impact is the consequence of risk if it occurs. This metrics is used for risk prioritization.

The Risk Leverage (RL) is defined as follows:

$$RL = \frac{RE_{\text{before}} - RE_{\text{after}}}{RRC} \tag{2.2}$$

RE_{before} is the risk exposure before the action plan execution and RE_{after} is the risk exposure after the action plan execution, and RRC means the Risk Reduction Cost, i.e., the cost of the action for risk exposure reduction. The RL measures the cost/benefit rate of the decisions made, aiming to reduce the chances of a problem occurring.

Some works did not propose metrics or indicators but point to the importance of using measurement-based techniques. One of them is the work of BECHTOLD (1997), which discusses the importance of metrics for risk management, showing examples of how some measures can help managers in the process of risk identification. For example, the risk factor "team qualification" could be measured through a combination of the following data: experience and knowledge level on a specific technology. On the other hand, the author did not present any practical application using these proposals.

FONTOURA; PRICE; PHIL (2004) use the Goal-Question-Metrics paradigm (GQM) as a tool for the definition of metrics for risk monitoring. However, the proposal was not put into practice. In the same light, HYATT; ROSENBERG (1997) propose a metrics program for risk assessment of software projects. A set of metrics is defined for software development life-cycle, but they are very restricted to the organization in which the authors worked, making it difficult to use in different projects managed by different teams.

LOPES (2005) developed an indicator called Risk Points (RP), which is defined as follows:

$$RP = URPW * PCF \tag{2.3}$$

In which URPW means Unadjusted Risk Points Weight and PCF means Project Characterization Factor. URPW is inspired on Unadjusted Use Case Weight (CLEMMONS, 2006), whereas PCF is based on the project characterization model proposed by COELHO (2003). According to the author, the objective of this indicator is to assess software project complexity based on the number of identified risks. However, this proposal has never been evaluated by the author.

COSTA (2005) uses some concepts of risks in Economics, more specifically in credit risk analysis, to evaluate the risk level of software projects. The author proposes a way of calculating how much capital a software development organization can gain or lose due to risks in a selected set of projects. To support this approach, the author also implemented it in a tool called RISICARE (COSTA; BARROS; TRAVASSOS, 2007a). An experimental study using this approach was performed and, according to the author, there were improvements regarding the reduction of subjectivity levels on risk analysis. However, this approach was not evaluated in a real organization.

SOUZA et al. (2009) proposed a set of metrics for controlling and measuring of risk-

based software testing. The objective of the proposed metrics is to indicate information related to controlling of test cases and test activities through product risk analysis. However, these metrics were not thoroughly evaluated, because there was only one simulation of the application of some metrics in a risk-based testing process. As a way to prioritize the most critical requirements of a project for testing, AMLAND (2000) proposes a method of calculating the risk level of functional requirements. That is, both works focus on technical risks, limiting its usage in project management. Moreover, they need more empirical studies in real or simulated environments to evaluate their effectiveness.

D'CASTRO (2013) advanced in a practical application about the use of indicators for risk monitoring by the proposition and implementation of a data mining and process mining. The objective of this work was the extraction of performance and quality indicators from configuration management tools, to monitor events and undesired condition during the software development process.

Given the limitation of related work, MENEZES JR; GUSMÃO; MOURA (2013) performed a systematic mapping study with the objective of mapping works that discuss metrics and indicators in risk management. Only three works in fact presented some metrics and indicators for risk management in software projects. One of them was previously mentioned in this section (SOUZA et al., 2009). The second work is BENAROCH; APPARI (2010), who uses concepts of Economics to propose a risk-pricing model in software development projects. This work focuses on the cost analysis for assessment of project sensitivity to a determined risk or risk category. Another identified work (HOSSEINGHOLIZADEH, 2010) uses information of source code of software products for analyzing technical risks, with the goal of mapping the riskiest component and functionalities of a specific system.

The related work closer to risk analysis of projects in the context of multiple software projects is LOPES (2005), through the indicator Risk Points. The indicator has the following characteristics:

- **Simplicity:** it uses elements present in software development activities, being adherent to most software development process, which does not end up being expensive to existent processes in an organization;
- **Applicability to the context of multiple software projects:** given its scope, whose focus is to define the risk level of a project in any stage of software life cycle, Risk Points also allows to build a database and knowledge about project risks;
- **Scalability:** basically, Risk Points use factors to characterize a project quantitatively and qualitatively, and the information of found risk factors. This scopes for risk analysis in different contexts, including allowing calibrations and improvements in the quality of necessary parameters for Risk Points calculation. In this case, there are two parameters: (i) Unadjusted Weight Risk Points, based on identified risk factors and (ii) Project characterization factor, that characterizes a project so that it allows

characterization of project factors to establish comparisons between different projects in the same environment;

- **Generality:** unlike other proposals, it is possible to use Risk Points indicator covering the evaluation of any category of risks, whether a project, process or product.

It is important to note other related work also considers information about project characteristics and the identified risks, but on different ways so that it is challenging to instantiate them for different contexts and multiple projects view. Therefore, the parameters of the indicator Risk Point were the starting point for proposing a risk indicator for environments of software development projects, because its proposal aims to define in one value the risk level of a project considering not only the identified risks but also some characteristics of a project as something that contributes to increasing its value.

Considering the potential of Risk Points indicators, next section presents a pilot study of this indicator application in a real environment of software development projects. We performed an exploratory case study in a environment of five software projects and its main goal was to evaluate the indicator and its effectiveness in a context of multiple software projects, providing information to identify the potential of the indicator, as well as improvement points.

2.8 Pilot study

Risk Points (RP) is an indicator that aims to represent the overall risk exposure level of a project (LOPES, 2005). RP is defined regarding the amount of identified risks, where these risks are defined concerning its probability and estimated impact, as the concept of Risk Exposure (RE) (BOEHM, 1989), defined as:

$$RE(risk) = Probability(risk) * Impact(risk) \tag{2.4}$$

RP is based on estimation technique of project size from Use Case Points metrics (UCP). This technique defines a way of quantifying the size of a software project based on defined use cases, also considering technical and environmental factors (KARNER, 1993; CLEMMONS, 2006). In summary, by UCP, it is possible to determine in one single value the project size, using as the basis the use cases and related technical and environmental factors.

Similarly, RP allows quantifying the project regarding its identified risks. It is necessary to estimate the Risk Exposure value, i.e., Probability versus Impact, for each identified risk, so, for a specific data collection about the current risks of a project, it is possible to determine a value of a project Risk Points (RP), as follows:

$$RP = PCF * URPW \tag{2.5}$$

In which PCF means **Project Characterization Factor** and URPW means **Unadjusted**

Risk Point Weight. PCF is a value for giving the project weight and adjust the final indicator value based on technical and environmental factors (COELHO, 2003). This value is defined through the answers of a questionnaire, which was developed from an empirical study with software project managers and management students. Then, PCF is defined as:

$$PCF = 1.05 + (0.015 * CF) \tag{2.6}$$

CF means **Characterization Factor**, it is determined by answering the 8 questions of a questionnaire with scores between 0 and 4, and then the answer is multiplied by the defined weighted value for each question. Finally, these 8 products are summed, resulting in the CF value (COELHO, 2003). Table 2 presents the project characterization factors and their respective weights. Therefore, CF is defined as:

$$\text{CF} = \sum_{i=1}^{8} (Question(i) * Weight(i)) \tag{2.7}$$

URPW, in turn, is the **Unadjusted Risk Point Weight**, composed by the identified risks during a data collection, in terms of their Risk Exposure. In this study, the estimation adopted has values in the range of $\{0.1, 0.2, \ldots, 0.9\}$. URPW value is composed by the summation of the weights of each identified risk, being this weight defined according the Risk Exposure (RE) value, as can be seen in Table 3. Thus, for n identified risks, the URPW value follows the rule:

$$URPW = \sum_{i=1}^{n} (Weight(risk(i)) \tag{2.8}$$

Briefly, a given data collection (even in a subjective way, with values in a scale of 5 levels of Probability and Impact) about the current risks of a project yields a value which represents the overall evaluation concerning the known risks of a project in a specific moment in its life cycle. This value allows a broad risk assessment of the risk exposure level of a project in different moments, and also allows a way to compare various projects in the same environment just based on their identified risks.

2.8.1 Study objectives

The primary objective of this pilot study was to evaluate the applicability of the proposed indicators and their effectiveness in risk assessment in an environment of multiple software projects. We collected information about risks each week in five projects in the same environment. For each project, risk factors were identified and analyzed using predefined scales of probability and impact for each risk.

Table 2 – Project characterization factors and respective weights (LOPES, 2005)

Factors	Answers	Weight
Team size	1 = small (7 to 20 people) 2 = average (21 to 50 people) 3 = big (51 to 100 people) 4 = very big (more than 100 people)	1.75
Multisite development	0 = same room 1 = same building, different rooms 2 = same city, same company, different buildings 3 = same city, different companies 4 = different cities	1.28
Team experience on the process	0 = no project 1 = 1 project 2 = 2 to 3 projects 3 = 4 to 5 projects 4 = more than 5 projects	1.13
Team experience on the application	0 = no project 1 = 1 project 2 = 2 to 3 projects 3 = 4 to 5 projects 4 = more than 5 projects	1.20
Team experience on the development technology	0 = no project 1 = 1 project 2 = 2 to 3 projects 3 = 4 to 5 projects 4 = more than 5 projects	1.50
Project criticality (possible consequence of a system failure)	1 = low losses, easily recoverable losses 2 = moderate losses, recoverable losses 3 = high losses, unrecoverable losses 4 = life threatening	1.20
Project size (investment)	0 = less than US$ 50.000,00 1 = between US$ 50.000,00 and US$ 150.000,00 2 = between US$ 150.000,00 and US$ 1.000.000,00 3 = between US$ 1.000.000,00 and US$ 3.000.000,00 4 = more than US$ 3.000.000,00	0.68
Project priority (quality/schedule)	0 = 10/90 1 = 30/70 2 = 50/50 3 = 70/30 4 = 90/10	1.13

Table 3 – Unadjusted Risk Point Weight (URPW) values

Classification	RE(risk)	Weight(risk)
Very low	[0.0, 0.2)	1
Low	[0.2, 0.4)	2
Average	[0.4, 0.6)	3
High	[0.6, 0.8)	4
Very high	[0.8, 1.0]	5

2.8.2 Method

To execute the study, we instantiated an agile risk management process called GARA (RIBEIRO et al., 2009), consistent with agile development methodologies, such as Scrum (SCHWABER, 2004), focused on multiple projects and simple enough for the risk management

activities, such as the data collecting. The indicators were applied in a software development environment from a research laboratory at a University, specialized in educational technologies, in which data were collected on information about risks weekly during two months. All the projects involved software applications on educational technologies.

Five projects were monitored between May 2015 and July 2015 together with their leaders. The projects were related to software development like web platforms – front and back-end, web services and mobile application. The following steps were executed:

1. **Risk identification:** through brainstorming, using the Risk Taxonomy from Software Engineering Institute (CARR et al., 1993) as the trigger to conduct the meetings. Three project leaders participated in this stage, in which one was responsible by two projects, another one by two, and the last one by only one project.

2. **Risk assessment:** for each identified risks, values of probability and impact are calculated, resulting in Risk Exposure calculation. For this work, we adopted the values presented at Table 4, that is, a scale of 5 levels, to make easier to the participant assess the risks.

Table 4 – Values of probability and impact used

Name	Value
Very low	0.1
Low	0.3
Average	0.5
High	0.7
Very high	0.9

3. **Data processing:** with the raised information, the identified risks were categorized as from project and from environment. Project risks appeared in only single project and environment risks appeared on more than one project. With the information collected in the previous steps, the indicator was calculated.

4. **Risk controlling and monitoring:** consists of meetings to present the results and identified risks weekly.

It is important to notice that these steps were performed weekly. Next, we present some information about each project used in this study - product description, number of participants and duration.

Project 1: web system to support students' subscription in post-graduate and extension courses, including management of data and reports generation.

- Product: system information in web platform, front-end and back-end.

- Team: software development (2 members) and design (3 members).
- Duration: 6 months

Project 2: information system for management of academic works for undergraduate and graduate courses. This project has three important sub-products: term paper elaboration and discussion forum, management reports, and requirements of creating, retrieving, updating and deleting (CRUD).

- Product: information system in web platform, front-end, and back-end.
- Team: software development (3 members) and design (3 members).
- Duration: 10 months

Project 3: mobile system to access educational contents about healthcare stored in external repositories. The system demands an external authentication server, and another institution develops the server side of the system.

- Product: mobile application developed with Android platform.
- Team: mobile development (4 members) and design (3 members).
- Duration: 12 months

Project 4: support-components for a distance course about primary health care that includes virtual learning environment and a web portal.

- Product: a web portal for access to the course, front-end, including visual and usability adjustments.
- Team: web design (4 members), design (3 members) and virtual learning environment (1 member).
- Duration: 3 months

Project 5: development of information system, whose goal is to evaluate students present in educational platforms (Moodle) and management of them.

- Product: information system in web platform, front-end and back-end.
- Team: web design (4), design (3) and virtual learning environment (1).

- Duration: 5 months

Although this chapter has presented the definition of Risk Points and its alternatives as the product of PCF and URPW, we adopted the fixed value of PCF = 1.0, that is, each project has the same level of importance. The reason for this decision was the need for more detailed investigation about how to specify PCF value, especially regarding the formation of a questionnaire and respective weights. Also, the fact that PCF is in the range (1.05, 1.6422), which imply in a tiny variation of the final value of the indicator. Finally, the analyzed projects presented similar characteristics, so that PCF values did not influenced the study, nor the RP behavior.

2.8.3 Results

During eight weeks, we applied the previously presented method. Table 5 summarizes results about the number of identified risks. 31 different risks were identified in five projects, so there are same risks present in different projects. Considering the average Risk Exposure (RE), most of the identified risks have low value. Table 6 presents the top ten risks from the environment, i.e., the ones with highest risk exposure value (average).

Table 5 – Pilot study: general results

Total of identified risks	**31**
Total of Risk Exposure Mean	0.14
Number of identified risks – Project 1	30
Number of identified risks – Project 2	30
Number of identified risks – Project 3	22
Number of identified risks – Project 4	25
Number of identified risks – Project 5	26

Table 6 – Pilot study: top ten risks

Risk	**Average Risk Exposure**
Failures on deployment	0,25
Dependences of other teams	0.22
Dependence of specialists	0.22
Urgent demands, new demands raises	0.20
Conflicts with external activities of team members	0.20
Requirements changes	0.18
Team member absence	0.16
Team member unavailability	0.16
Exit of team member	0.16
Software testing process problems	0.16

It is important to notice that the project leader is the responsible for evaluating probability and impact according to information presented on Table 3. For each project, all the identified risks (for respective risk exposure values) are synthesized in one single value. Therefore, the idea is to represent the overall risk level of each project in a specific moment. Figure 7, for example, presents the results of the application of Risk Points/Number of identified risks. The x-axis represents the number of weeks, whereas Y-axis represents the indicator value.

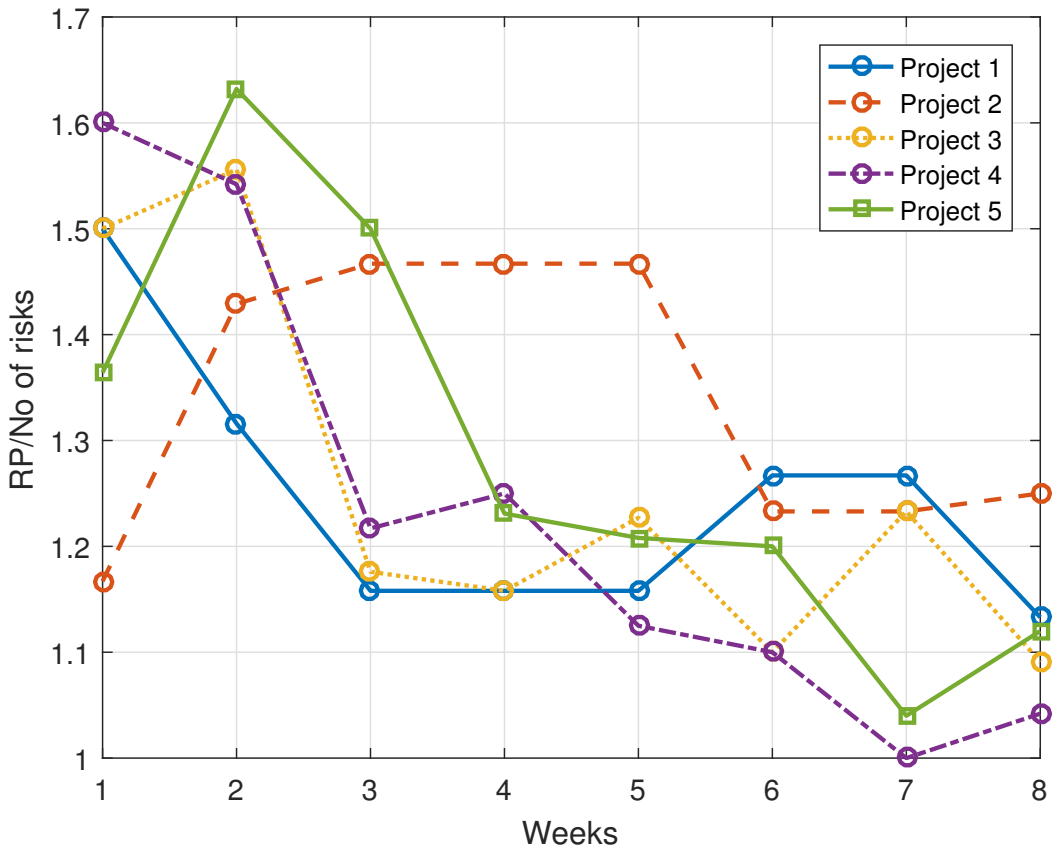

Figure 7 – Pilot study: Risk Points / Number of risks

We adjusted RP based on the number of identified risks for each project because it is very dependent on the number of risks so that the greater is the number of risks, the greater the indicator value. To allow comparison between projects, in this work we divided the metrics by the number of identified risks of each project. With this adjustment, it is possible to evaluate the values of the metrics directly, independently of the number of identified risks of each project. For example, consider that we have two projects: A and B. Project A has 100 risk factors, and Project B has only 10. Therefore, we have Indicator (project A)/100 and Indicator (project B)/10, reducing the influence of the quantity of identified risks.

Project 1 presented a considerable risk level in the last weeks because of an important release was approaching. Before that, this project had a successful delivery. The data collecting of Project 2 started when it was beginning a new development cycle after a significant milestone. At

that moment, requirements and new demands have grown. After the 5th week, the requirements were well defined, so that the risk level decreased. Even though this project was considered as riskier in the environment after eight weeks.

In the first month of data collecting, Project 3 delivered a critical release, which justifies the decrease of risk level during this period. In the second month, the team got test results with new requirements, adjustments, and bugs to be fixed. It can explain the oscillation that happened in the second month of this project.

Project 4 started as the riskier project and finished as the less risky. In fact, this project was considered successful and did not present problems during its life-cycle. Project 5 also was being finished. so it presented a decrease during the period of assessment, just waiting for evaluation, feedback and final approval of the testers.

In general, we realized that, after a significant milestone, the risk level presents accentuate decrease. After the feedback, bugs identification and adjustments on the scope, the values start to grow and remains increasing until the next milestone or delivery of a release.

At the beginning of the study, Project 4 presented the highest level of criticality and its value reduced to zero at the 8th week. In fact, this project had a relatively short schedule, and it was finishing successfully, just waiting for a final evaluation before the system deployment.

Project 2 was considered the most critical and riskier one. It means that there were risks classified as average or higher value. In the 8th week, the project was close to an important release.

Project 5 was delivering a release, and it was close to the end. The main functionalities were finished as it was agreed and it was just waiting for final feedback from an acceptance test. According to the leader of this project, the presented values were pertinent once it was facing a critical phase between the third and sixth week.

Finally, the high value in the second week of Project 3 was expected, once at that moment a critical delivery was being finished. However, the difference between the others values needs a more in-depth investigation because it could be a bias of the project leader experience or knowledge.

2.8.4 Discussion

An important characteristic identified in this work is: most of the identified risks are classified as very low or low. The impact of this in the indicators is the fact that the risks with high values are not well explored.

Another point to be considered is that the processed values of the indicator had potential to determine the risk level of a project through a single value in a certain moment. This information is crucial for a better comprehension of the context of the indicator application because different people can perform different estimations in the same project.

The risk list used was built using the information given by project leaders during the first

weeks. To guide the process of risk identification, we used the risk taxonomy of SEI (CARR et al., 1993), but only to make the brainstorming more focused. We did not use a predetermined risk list. It also means that the indicators values are an estimation of the general level of project risk exposure.

The main positive points of the assessed indicators show that they are capable of telling us, in only one single value, the general level of a software project risk exposure in a specific moment. Second, the indicators allow an assessment in environments of multiple projects, providing direct and indirect comparisons between different projects through their life-cycle.

The main improvement point about the indicators is their sensitivity to the experience level of the project manager and their accuracy level. In other words, the same project may have different values at the same moment when more than one person assesses it.

The analysis of FCP in the context of RI needs in-depth investigations. The tiny variation of the values did not allow its assessment, so that it is an improvement point in the proposition of an indicator that may consider project characteristics as one important source of information. As discussed earlier, there is a relationship between project risks and project characteristics, but the proposal of LOPES (2005) still not represents represent the influence of project characteristics in risk levels. In fact RP seemed to be too much dependent of the number of the identified risks.

This work gathered data weekly through online tools and meetings. At the end of the study, the project leaders made an assessment of the process and its effectiveness to improve knowledge about the project's risks. In general, the project leaders considered the study necessary for the process of project risk management, providing a better perception of risks.

In the first month of the study, all the data collection was face-to-face. This approach was efficient for the understanding of the projects, risk factors and, mainly, to make the process clearer. All the project leaders said that the presence of a risk manager is important to conduct risk identification and to make better estimations. However, this kind of meeting could be expensive, because it demands more face-to-face meetings with approximately one hour in duration.

In the second month (last four gatherings), we applied an online questionnaire with the project leaders. The positive point of this approach was the flexibility and agility. However, we observed difficulties in assuring that the project leaders answer the questionnaires on time. One proposal for the process could be an intermediate approach, using both online and face-to-face approach to take advantage of the positive points of each one, using toggled iterations.

By analyzing RP indicator, we realized two weaknesses: i) the calculation of unadjusted risk points weights is very dependent of the number of risks, and it did not consider any difference between risk factors categories, making no advantage regarding the usage of risk exposure; ii) the calculation of project characterization factors is incipient, because its weights were defined on a very small survey and the values range is very small so that it is not possible to see differences between RP values because of that.

Finally, this study performed weekly meetings to collect data only for proposal assess-

ment issues. In an real environment, where there would be dozens or even hundreds of projects, the process of indicator usage may be so expensive so that would be hard to use it. A strategy that could be useful for the analyzed environment from this exploratory case study is by defining mitigation strategies only for riskier projects, and then to check the levels again after their application .

2.9 Chapter summary

In this chapter, we provided the necessary background to comprehend the remaining of this thesis. Firstly we presented concepts about risk management in software engineering, showing the main challenges, activities and classifications. Then we sought to clearly define risk factors and project characterization factors, providing a comparison study of the last ones.

We also, presented concepts of multiple project management, once we focused on management of not only one single project, but on tactical management of resources in a environment of multiple projects, making the use of an indicator more appropriate to get different sources of information and summarize risk level in one single value. Additionally, we provided background about metrics and indicators in software engineering.

Finally, we presented related work and we select one to perform an exploratory case study - Risk Points (LOPES, 2005), which helped to better understand it and define paths to be followed during this work: the refinement of Risk Points, the proposal of Project Risk Index and its assessment though two case studies.

Next chapter presents a systematic literature review, whose goal was to better define risk factors using weights to make its calculation more coherent to software development projects environments.

3 RISK FACTORS IN SOFTWARE DEVELOPMENT PROJECTS

Even with the increasing importance of risk management in software engineering (GUSMÃO, 2007; DE BAKKER; BOONSTRA; WORTMANN, 2010), there are still some barriers to overcome, according to a study performed by BANNERMAN (2015):

- Costs: usage of risk management practice tends to create overhead in projects;
- Culture: specifically, the culture of risk aversion, which can influence the extent to which risk management is practiced;
- Uncertain benefits: it is necessary more empirical studies that demonstrate that the use of risk management improve project performance;
- Capability: the adoption and effectiveness of risk management practices may be constrained by limited organizational and individual capabilities in applying risk management.

Software quality is directly influenced by the quality of software development processes (GUSMÃO; MOURA, 2004). In this light, risk management is crucial to identify and to control potential problems during projects life cycle. Risk management is a practice that comprises processes, methods, and tools to control project risks (HIGUERA et al., 1994). Therefore, the use of risk management aims to increase the quality of the product and the software development process. On the other hand, quality also contributes to the reduction of project risk (SARIGIANNIDIS; CHATZOGLOU, 2014). We can observe that software development projects, in general, present delays in schedule, costs that overruns and functionalities that are below of expectations. These problems, while considered inherent to software development by many authors, can be minimized and controlled by continuous project risk management.

In risk management, an essential activity is the identification of risks or risk factors, being considered the most influent activity of risk management (DE BAKKER; BOONSTRA; WORTMANN, 2010; LÓPEZ; SALMERON, 2012) and widely used in both agile and traditional software development methods (NEVES et al., 2014). A risk factor is defined as a condition that can pose a severe threat to the successful completion of a software development project (MARCH; SHAPIRA, 1987). We can affirm that risk factor is the smallest unit of information that makes all risk management activities feasible, being the most valuable information, once they are the elements managed during the life cycle of one or more projects.

The inefficiency in risk identification process in the development of complex systems is considered one of the leading causes of project failures (REEVES et al., 2013). Several studies point out that in fact there is a direct relationship between risk management and success or improvement of the performance of software development projects (JIANG; KLEIN, 2000; JIANG; KLEIN; DISCENZA, 2001; RAZ; SHENHAR; DVIR, 2002; WALLACE; KEIL; RAI,

2004a; WALLACE; KEIL, 2004; DE BAKKER; BOONSTRA; WORTMANN, 2010; HAN; HUANG, 2007). These studies show that at least the risk factors should be identified and well controlled for projects to achieve their objectives. For instance, the study of (HAN; HUANG, 2007) concludes that risk factors associated with system requirements seriously affect the project performance. Thus, the identification of risk factors plays a crucial role in the success and the performance of software development projects. Besides, any risk management process consists mainly of first identifying the factors that may hurt project objectives.

In this context, this chapter aims to map risk factors for software development projects through a systematic literature review. The main goal of this work is to find and to categorize evidence of risk factors of software development projects through a comprehensive and rigorous literature review. The usage of a systematic literature review allows extending the reach of results, as well as to bring more in-depth insight into state of the art in the context of software development projects. So, the conjecture we want to investigate this systematic literature review is that it is essential to get a comprehensive view of risk factors in environments of software development projects.

After this introductory section, this chapter is organized as follows: First, it presents related works. Then we describe in detail the conducted research method, specifying the protocol, which includes search strings, research questions, the used sources, the search strategy, and inclusion and exclusion criteria. Also, this section presents how this protocol was refined. Therefore, next section displays the answers to the research questions, in which includes an extensive data analysis, with analysis of rigor and relevance and the evidence of identified risk factors. So we discuss in depth the findings and raise questions and implications for researchers and practitioners. Finally, we present final considerations, which includes the main contributions, threats to validity and future work.

3.1 Related work

In general, the area of risk management in software development projects still requires greater maturity regarding systematic literature reviews or mapping studies. After an ad hoc literature review, there is little evidence of secondary studies application in this area. One of them is the work of NURDIANI et al. (2011), which presents a systematic review with the objective of identifying risk mitigation strategies in global software development. ALAM; KHAN; ALI (2012) performed a systematic review seeking to identify adverse factors in knowledge sharing management, in the context of offshore software outsourcing.

PA; JNR (2015) searched for activities, processes, frameworks, and models that can support the decision-making process for risk mitigation in software projects. Finally, KHAN; BASRI; DOMINIC (2014) identified communications risks in the context of global software development.

Deduced from the above, therefore, systematic literature reviews related to the identifi-

cation and categorization of risk factors in environments of software development projects are not readily available. In other words, no systematic review has been done yet. Therefore, it would be interesting to perform a comprehensive overview of risk factors in this context, and the application of a systematic literature review can be useful and insightful to provide this work.

3.2 Research method

According to KITCHENHAM; CHARTERS (2007), the use of systematic literature reviews is appropriate to: (i) summarize evidence concerning a treatment or technology, (ii) identify gaps, and (iii) provide a background to appropriately position new opportunities of research and an overview of a particular subject or theme. It completely fits the objectives of this work, as its primary goal is to collect evidence of risk factors in the context of software development projects, identifying any gap regarding the area and using an empirical method to define the research background.

For conducting this systematic literature review, we used the guidelines provided by BIOLCHINI et al. (2005) and KITCHENHAM; CHARTERS (2007). The following activities comprised the research method:

1. **Planning the review:** it is the most important activity of the systematic literature review. It consists of the definition and assessment of the research protocol that defines all the necessary procedures for conducting the review (next activity). The research protocol in this work is composed by the following elements: (i) research questions (Section 3.2.1); (ii) search strategy (Section 3.2.2); (iii) database selection (Section 3.2.3); (iv)Inclusion and exclusion criteria (Section 3.2.4); and (v) strategy for data extraction (Section 3.2.5). The research protocol defines what to do and how to do it. It is important to notice that the planning is an iterative activity, in which each element of the protocol is assessed and calibrated until it is ready for the next activity.

2. **Conducting the review:** it is the complete execution of the planning process. It consists of the most exhaustive activity of the systematic literature review, comprising the database search, selection of studies that answer the research questions, and the extraction of data for analysis and posterior synthesis and discussion.

3. **Reporting the review:** it is just the writing up of the systematic literature review for dissemination among the potentially interested parties.

3.2.1 Research questions

The main objective of this systematic literature review is to identify and to categorize evidence of risk factors evident in the context of software development projects. Furthermore,

we seek to use evidence to assess the studies through rigor and relevance analysis. According to KITCHENHAM; CHARTERS (2007), the specification of the research questions is the most crucial part of any systematic literature review. We defined the following Research Questions to perform the study:

RQ 1. What are the risk factors present software development projects?
RQ 2. In which class do the found risk factors fit?
RQ 2.1. What are the rigor of the reported studies?
RQ 2.2. What are the relevance of the reported values?

RQ 1 can be considered the main question of the review and seeks to map risk factors of software development projects. So, after the selection of studies that answer this research question, the presented risk factor of each study is collected. RQ 2 seeks to categorize the identified risk factors using a taxonomy of risks for software development projects (CARR et al., 1993) and to perform a deeper analysis of the risk factors. Finally, RQ 2.1 and RQ 2.2 analyzes, respectively, the application of research methods and the applicability in the industry of each study, using the checklist proposed by IVARSSON; GORSCHEK (2011). Section 3.2.5 explains how data was collected, aiming to answer each research question.

3.2.2 Search activities

It defines the necessary activities to be performed to answer the research questions. These are:

1. Database search and extraction of studies using search strings
2. Duplicate elimination
3. Selection of works by applying inclusion-exclusion criteria
4. Extraction of risk factors from selected works
5. Categorization of risk factors
6. Rigor-relevance assessment of the selected works
7. Synthesis

To define the search strings, we combined the experience and knowledge of the authors in the domain and dataset of known studies, called Initial Quasi-Gold Standard (Initial QGS), as can be seen at Table 7.

The initial QGS is a set of studies known by the author before starting the planning of the systematic literature review, and that also answers the research questions. This reduced set

of studies is used to evaluate the research protocol (ZHANG; BABAR; TELL, 2011), once it is expected the studies that compose the initial QGS are present in the results of the systematic review, providing higher reliability to the research. In addition, it is expected that the studies that make up the initial QGS are from recognized conferences or journals by the community.

Table 7 – Initial Quasi-Gold Standard (QGS)

QGS 1	BOEHM, B. W. Software risk management: principles and practices. **IEEE software**, v.8, n.1, p.32–41, 1991
QGS 2	BARKI, H.; RIVARD, S.; TALBOT, J. Toward an assessment of software development risk. **Journal of management information systems**, v.10, n.2, p.203–225, 1993
QGS 3	ROPPONEN, J.; LYYTINEN, K. Components of software development risk: how to address them? a project manager survey. **IEEE transactions on software engineering**, v.26, n.2, p.98–112, 2000
QGS 4	SCHMIDT, R.; LYYTINEN, K.; MARK KEIL, P. C. Identifying software project risks: an international delphi study. **Journal of management information systems**, v.17, n.4, p.5–36, 2001
QGS 5	WALLACE, L.; KEIL, M.; RAI, A. How software project risk affects project performance: an investigation of the dimensions of risk and an exploratory model. **Decision Sciences**, v.35, n.2, p.289–321, 2004
QGS 6	TAKAGI, Y.; MIZUNO, O.; KIKUNO, T. An empirical approach to characterizing risky software projects based on logistic regression analysis. **Empirical Software Engineering**, v.10, n.4, p.495–515, 2005

This systematic literature review was executed between January 2017 and April 2017. We analyzed studies before December 2016 only on digital libraries. After the application of inclusion and exclusion criteria, some works were selected, and then we started to extract the necessary data to answer the research questions, analyzing and synthesizing them.

3.2.3 Search strategy

To perform searches in the selected sources, we defined a set of keywords using PICO tool (acronym for Population, Intervention, Comparison and Outcomes). However, this systematic literature review did not make Comparison studies, because we did not make any comparison of different interventions. Therefore, the strings and search strategy were determined according to Table 8. We used these strings to search title, abstract and keywords fields.

- **Population:** researches in the management of software development projects.
- **Intervention:** works related to risk management.
- **Outcomes:** factors, components, lists, dimensions, taxonomies, classes, classifications and categories of risk. Some aspects of classification were considered because some papers also suggest risk factors.

Until this set of strings was obtained, it was refined, as shown in Figure 8. This process was executed in January 2017. To achieve a reliable result, we performed an analysis of sensitivity,

which is defined as the number of relevant reports identified divided by the total number of existing relevant reports (HIGGINS; GREEN, 2011). Therefore, the sensitivity of the used strings to proceed with the selection of works was calculated by dividing the found Initial QGS by the total number of papers from Initial QGS. In this work, the sensitivity was 83.3% (5 of 6 papers), a value considered satisfactory for the analysis of the studies, according HIGGINS; GREEN (2011), that consider at least 70%. Only the QGS 1 (Table 7) was not found, because its focus is to present a risk management approach and the risk factor list is a secondary subject. However, the author is the first researcher that suggested a risk management process for software projects (BOEHM, 1989).

Table 8 – Search strategy

Population	"Software project management" OR "Software development project" OR "Software development" OR "Software project"
Intervention	"Risk management" OR "software risk" OR "software development risk" OR "software risk management" OR "software Project risk"
Outcomes	"risk factors" OR "risk Component" OR "risk List" OR "list of risks" OR "risk Dimension" OR "risk Taxonomy" OR "risk Class" OR "risk Category" OR "Ranking" OR "definition" OR "measure" OR "risk items" OR "characterization"
Search strategy: Population AND Intervention AND Outcomes	

For paper search, we used the following reference electronic libraries:

- ACM digital library - `https://dl.acm.org/advsearch.cfm?coll=DL&dl=ACM`
- Engineering village - `https://www.engineeringvillage.com/search/expert.url`
- IEEExplore - `https://ieeexplore.ieee.org/search/advsearch.jsp?expression-builder`
- Science direct (advanced search) - `https://www.sciencedirect.com/search/advanced`
- Scopus - `https://www.scopus.com/search/form.uri?display=basic`

The combination of strings presented in Table 8 was applied at databases Engineering village, Science Direct, and Scopus. However, due to database limitations, the strings had to be adjusted to match ACM digital library and IEEExplore. For ACM digital library, we performed a search that matches any of these strings: *"risk factor", "risk component", "risk list", "list of risks", "risk dimension", "risk taxonomy", "risk class", "risk category"*. This adaptation was necessary because the database did not accept combine boolean characters, such as AND, OR.

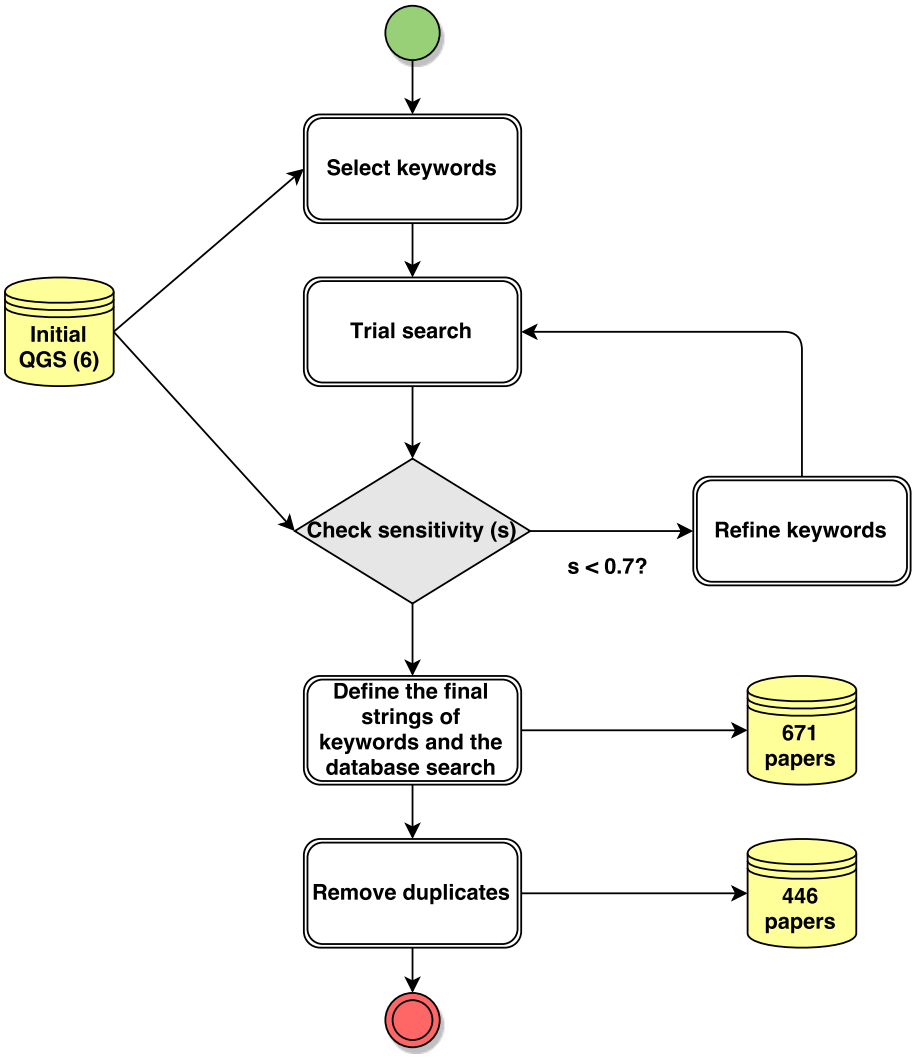

Figure 8 – Process of string refinement and final dataset gathering

IEEExplore library has a limitation regarding the maximum number of search terms: 15, so we defined the following keywords:

("Software development" OR "Software project") AND ("Risk management" OR "software risk" OR "software development risk" OR "software Project risk") AND ("risk factors" OR "risk Component" OR "risk List" OR "risk Category" OR "risk items" OR "characterization")

To minimize the chances of wrongly eliminating relevant papers, we used strings in ACM digital library and IEEExplore to be as comprehensive as possible in terms of population and intervention strings. For all databases, we looked for the strings at title, abstract and keywords. After elimination of duplicates, 446 papers were finally added to the initial database search to be analyzed according to inclusion and exclusion criteria by reading title, abstract, and keywords. If there were any doubts, the paper was read entirely, seeking for risk factors.

3.2.4 Inclusion and exclusion criteria

For the selection of papers that address the research questions, some criteria must be defined. The paper was included if it satisfied two conditions: (i) It discussed project risk management in software development project, and (ii) It proposed or mapped a list of risk factors for software development projects.

Additionally, if at least one of these exclusion criteria was satisfied, the paper was excluded: (i) it was not in English, (ii) it was not a primary study, (iii) it was not available to download, (vi) it presented risk factors whose description do not allow their understanding, and (v) It presented only risk categorization, classification or taxonomy proposals.

Initially, this filtering was performed by title, keywords, and abstract analysis. Then, we conducted a full reading of each paper, checking if it supports the defined inclusion and exclusion criteria. Therefore, all the selected documents satisfied all inclusion criteria and did not fit any exclusion criteria. It is important to notice that the database search was conducted without any set limits regarding publication year.

3.2.5 Data extraction strategy

Data analysis was performed with the selected papers after the application of inclusion-exclusion criteria. The Research Questions (RQs) drive the data extraction, which it is performed using as the basis the full reading of the selected papers (KITCHENHAM; CHARTERS, 2007). For each selected paper, the following information was extracted:

1. **Risk factors of software development projects:** each identified paper lists risk factors. These factors were extracted and cataloged. They were associated with RQ 1 and RQ 2;

2. **Classification of risk factors found:** after cataloging, the listed factors were classified according to the risk taxonomy proposed by SEI (CARR et al., 1993). This classification allowed to group the factors in common and to perform the counting of occurrences of each one. It was associated with RQ 2;

3. **Year:** the publication year. It was associated with RQ 1, because we searched for risk factors by period of time;

4. **Impact factor of publications:** h-index of publications. It is a metric of productivity and citation impact of publications and was associated with RQ 2.1 and 2.2;

5. **Rigor-relevance analysis:** consists of the application of an assessment checklist proposed by IVARSSON; GORSCHEK (2011), which allows the assessment of the quality of works using two criteria: rigor and relevance. The first one allows assessing the methodological rigor of works, whereas the second one assesses their relevance for the industry. It was associated with RQs 2.1 and 2.2.

The SEI risk taxonomy provides an organized way to categorize risk factors, and it is a consolidated report in software risk management (CARR et al., 1993). Thus, the defined indicators are adherent to software development environments, and they are not based only on theoretical aspects of a bibliographical review, but also on the practical application of consolidated risk classification.

This taxonomy a framework to structure and organize risks in software development projects so that it can be used as a basis for the elaboration of methods and activities of risk management. The taxonomic Classes are divided into Elements, which are characterized by Attributes. Figure 9 presents how SEI risk taxonomy is structured.

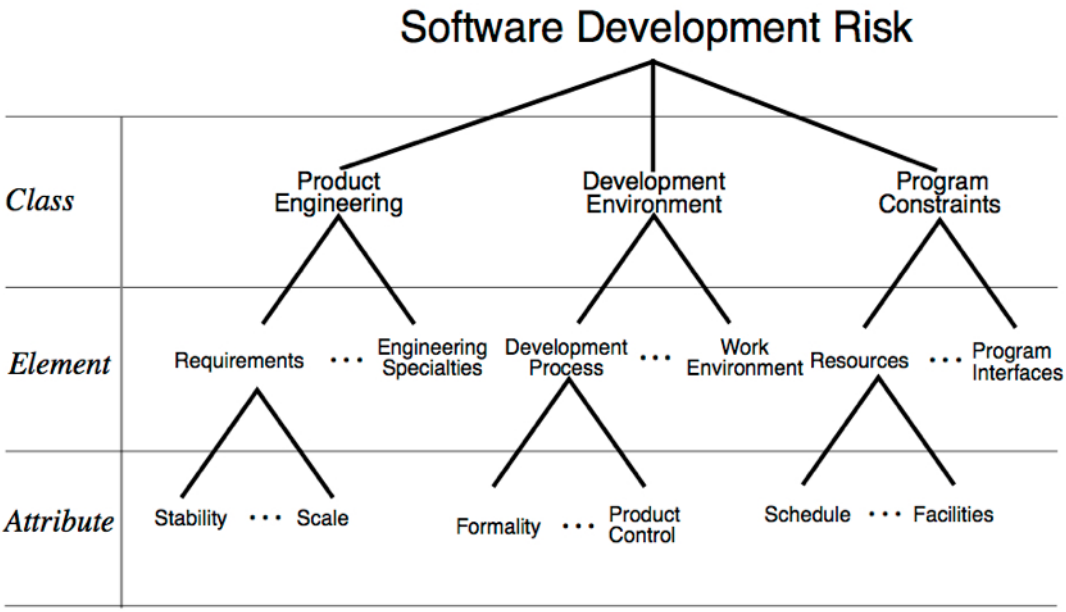

Figure 9 – SEI risk taxonomy (CARR et al., 1993)

As can be seen, the taxonomy classifies the risks into three big classes:

- **Product engineering:** it consists of technical aspects to be performed in the development of a product to be delivered to the client. It refers to activities of software and system engineering, such as analysis and requirements specification, design and implementation, integrations of software and hardware components, and tests. Engineering product risks commonly derive from requirements that are hard to implement, inappropriate assessment of quality requirements or design specification and poor quality of code implementation. This class has the following elements: Requirements, Design, Code and Unit Test, Integration and Test, and Engineering Specialties;

- **Development environment:** this class involves risks related to methods, procedures, and tools used in product development. It also includes management issues. This class has the following elements: Development Environment, Development System, Management Process, Management Methods, and Work Environment;

- **Program constraints:** it is composed of external risk factors, i.e., they are outside the control of the management, such as contractual, organizational and operational factors. The following elements are described in this class: Resources, Contract, and Program Interfaces.

Hence, this thesis used the classes and elements from SEI risk taxonomy to group the factors present in the identified works.

Rigor and relevance analysis, in turn, was based on (IVARSSON; GORSCHEK, 2011), which present an assessment methodology of empirical studies in software engineering. Rigor is the precise application of scientific methods, being defined through three aspects:

- **Context (C):** it assesses the level of detail of the presented information about the context in which the assessment was performed, including development process, contract, market, organization characteristics, among others, so that comparison with other contexts is possible;
- **Study design (S):** it assesses the level of detail of information about research method, including information such as dependent and independent variables, metrics, treatments, type of sample, among others;
- **Validity (V):** it assesses the level of detail of information about a discussion regarding threats to validity and limitations.

For each aspect, there are three possible values: strong description (1), medium description (0,5) and weak description (0). The rigor of each work is quantified by the sum of these aspects (C + S + V).

The relevance analysis aims to evaluate the impact of the work from the industry viewpoint, serving as a calibration point for rigor analysis. For this, the following aspects are taken into account:

- **User/Subject (U):** it assesses the representativeness of the subjects used in the work assessment. The use of industry professionals contributes to relevance;
- **Context (C):** it assesses if the study is performed in a setting representing real-world situations. The relevance is greater when the work is assessed in real contexts;
- **Scale (S):** it assesses if the scale of the applications is of realistic size. The relevance exists in this case when the work uses applications of scale similar to industry;
- **Research method (RM):** it assesses if the work mentioned the use of some research method, such as action research, case study, interviews, and surveys, among others.

For each aspect there are two possible values: contribute to relevance (1) and do not contribute to relevance (0). The study is considered relevant if it presents a relationship with the real world and if there is any research method being applied in the assessments. The relevance is quantified by the sum of these aspects (U + C + S + RM).

3.3 Results

This section shows the results found, including the analysis of the selected papers. The initial database search retrieved 671 studies, and the duplication elimination removed 226 studies. Therefore, 446 studies composed the initial dataset, as can be seen in Table 9. In the end, 41 papers consisted the final dataset, and the Figure 10 summarizes the performed work.

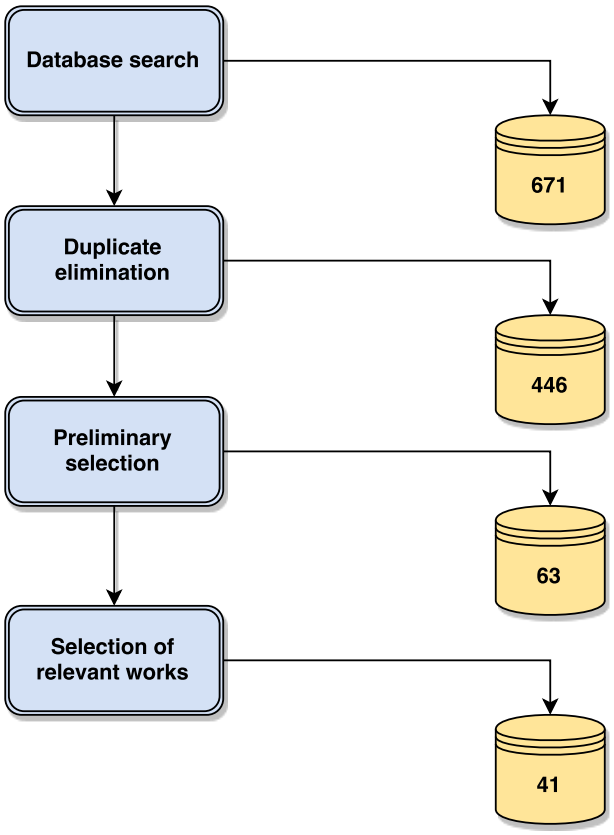

Figure 10 – Summarization of the search

After duplicate elimination, we applied the inclusion and exclusion criteria through the reading of titles, abstracts, and keywords, resulting in 63 pre-selected studies. After fully reading these studies, 40 were considered as the ones that in fact answered the research questions. One study from initial QGS that did not appear in the search results was included for data extraction process (BOEHM, 1991), that presents only a top 10 risk list, resulting in 41 selected studies. Therefore, 41 studies composed the final dataset and were analyzed by data extraction. (**Appendix A**) lists the selected papers in chronological order (S1 to S 41). All the process presented in Figure 10 was performed between February and April 2017. Next sections present

the extracted data as over viewed in Section 3.2.5.

Table 9 – Database search results

Database	**Results**
ACM digital library	147
Engineering village	164
IEEExplore	65
Science direct	28
Scopus	267
Total	671
Initial dataset	*446*

3.3.1 Evidence of risk factors of software development projects and their classification

A total of 148 different risk factors were mapped in this work. It represents an average of approximately 9 occurrences for each risk factor. 51 risk factors belong to "Product engineering" class, 41 belong to "Development environment" class, and 56 belong to "Program constraints" class. Table 25, Table 26, and Table 27 (**Appendix B**) present all these factors organized by class and element, according to SEI risk taxonomy (CARR et al., 1993). Table 10 presents the ten most mentioned risk factors for all classes.

Table 10 – Top 10 risk factors – all classes

Risk factor	**No. of occurrences**	**Class**	**Element**
1. Staff does not have required skills	55	Program constraints	Resources
2. Requirement ambiguity	44	Product engineering	Requirements
3. Bad commitment of the user / customer	37	Program constraints	Program interfaces
4. Requirement changes	32	Product engineering	Requirements
5. Introduction of new technology	30	Product engineering	Requirements
6. Unstable organizational environment	30	Program constraints	Program interfaces
7. Shortfalls in externally furnished components / Bad interfaces	27	Product engineering	Design
8. Technical complexity	25	Product engineering	Requirements
9. No planning or inadequate planning	25	Development environment	Management process
10. Incomplete requirement	24	Product engineering	Requirements
10. Quality of the specifications / documentation	24	Product engineering	Engineering specialties

For example, the risk factor "Staff does not have required skills" had 55 occurrences. It means that among the selected studies we found 55 occurrences of factors related to the same thing so that they were grouped into one single factor. Some examples of occurrence of the

factor “Staff does not have required skills” are: Personnel shortfalls (S1), lack of development expertise in team, (S2), Lack of technical expertise (S8), Lack of available skilled personnel (S9) and Team members lack specialized skills required by the project (S12).

As can be observed, each factor was identified and cataloged according to the number of occurrences of the evidence found. Therefore, all 41 studies present lists of risk factors. These factors were organized according to SEI risk taxonomy (CARR et al., 1993), by the identification of classes and respective elements. We can observe a predominance of factors associated with “Product engineering” class, being five associated with “Requirement” element. On the other hand, the most mentioned factor is related to lack of project team qualification, which was mentioned 55 times. It is important to highlight that a total of 1414 occurrences were identified, of which 353 (approximately 25%) were mentioned in Table 10.

Table 11 summarizes the number of occurrences of risk factor by class and element. 483 of the factors are in “Product engineering” class, and within this class, 244 are associated with “Requirements” element. In the class “Development environment”, that has 416 occurrences, there is a greater trend to equilibrium, but with a high concentration of factors on the element “Management process”. Finally, the class “Program constraints” presents 515 occurrences and privileges the elements “Resources” and “Program interfaces”.

Table 11 – Occurrences of risk factors by class and respective element

Class/Element	No. of occurrences	% (by class)
Product engineering	**483**	
Requirements	244	51%
Design	77	16%
Code and unit test	40	8%
Integration and test	62	13%
Engineering specialties	60	12%
Development environment	**416**	
Development process	91	22%
Development system	47	11%
Management process	138	33%
Management methods	55	13%
Work environment	85	20%
Program constraints	**515**	
Resources	260	50%
Contract	23	5%
Program interfaces	232	45%

Table 12, Table 13 and Table 14, in turn, present the top 10 risk factors of each class. In “Product engineering” class, the issue of bad clarity or ambiguity of requirements is cited as the most recurring factor among the found studies.

There is some uniformity in the distribution of risk factors of the class “Development environment”, with a higher slope to the factor associated with the bad planning of a project. In

the class “Program constraints” issues associated with team qualification is the most relevant risk factors according to evidence found in research. The factors of low commitment of user or client, as well as negative influences of the organizational environment, are highlighted. The element “Resources” and “Program interfaces” are the highlighting ones in top ten presented in Table 14.

Table 12 – Top 10 risks factors - Product engineering class

Risk factor	**No. of occurrences**	**Element**
1. Requirement ambiguity	44	Requirements
2. Requirement changes	32	Requirements
3. Introduction of new technology	30	Requirements
4. Shortfalls in externally furnished components / Bad interfaces	27	Design
5. Technical complexity	25	Requirements
6. Incomplete requirements	24	Requirements
7. Quality of the specifications / documentation	24	Engineering specialties
8. Lack of internal system Integration	18	Integration and test
9. Developing the wrong functions and properties	17	Requirements
10. Lack of proper tests	15	Integration and test

Table 13 – Top 10 risks factors - Development environment class

Risk factor	**No. of occurrences**	**Element**
1. No planning or inadequate planning	25	Management process
2. Low commitment of staff	20	Work environment
3. Insufficient Discipline and Standardization	19	Development process
4. Unclear project objectives	19	Management process
5. Ineffective Communications between team members	19	Work environment
6. Scope changes	18	Management process
7. Project progress not monitored closely enough	18	Management methods
8. Development Methodology was inappropriate for the project	16	Development process
9. Conflicts between team members	15	Work environment
10. Lack of effective project management methodology	14	Development process
10. Inappropriate CASE tools	14	Development system
10. Project manager’s experience	14	Management process
10. Low morale	14	Work environment

3.3.2 Study distribution by year

Figure 11 illustrates the study distribution by year. The oldest study dates from 1991, written by Barry Boehm (BOEHM, 1991), who is considered the father of risk management

in software engineering. The results reinforce this statement, as well as demonstrate that risk management in software projects is a relatively new area.

Table 14 – Top 10 risks factors - Program constraints class

Risk factor	**No. of occurrences**	**Element**
1. Staff does not have required skills	55	Resources
2. Bad commitment of the user/customer	37	Program interfaces
3. Unstable organizational environment	30	Program interfaces
4. Bad estimation of resources	22	Resources
5. Unrealistic schedule	21	Resources
6. Staff inexperience	19	Resources
7. High turnover	17	Resources
8. Disagreement with customer	17	Program interfaces
9. Lack of top management commitment / support to the project	16	Program interfaces
10. Unstable budget	15	Resources
10. Unreasonable customers	15	Program interfaces

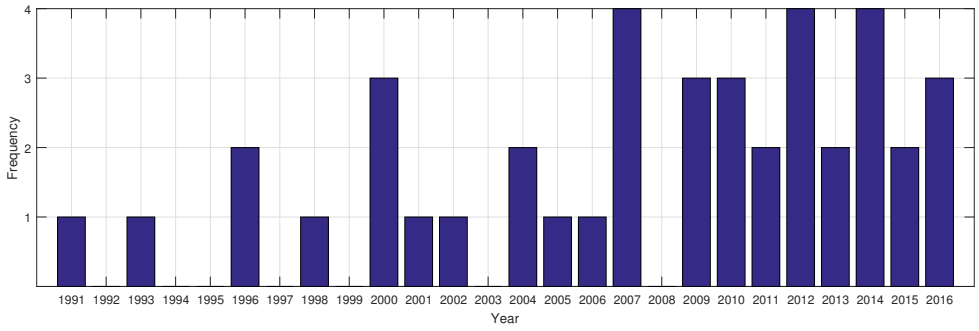

Figure 11 – Study distribution by year

Regarding the risk factors per year of published works, analyzing the top 5 most mentioned risk factors by stages, we observe the following results:

- **1991 to 2000:** i) Requirement changes, ii) Shortfalls in externally furnished components, iii) Staff does not have required skills, iv) Technical complexity, v) Unrealistic schedule;
- **2001 to 2010:** i) Bad commitment of the user/customer, ii) Staff does not have required skills, iii) Shortfalls in externally furnished components, iv) Requirement ambiguity, v) Introduction of new technology;
- **2011 to 2016:** i) Requirement ambiguity, ii) Staff does not have required skills, iii) Quality of the specifications, iv) Unstable organizational environment, v) Developing the wrong functions and properties, Lack of internal system Integration.

As can be seen, the factor related to staff skills is the most critical, regardless of the analyzed stage. We also realize that requirement risk factors are always recurrent. Finally, as the stages go by, there is evidence that there is greater emphasis on organizational and environmental aspects as factors that can influence the success or performance of software development projects.

3.3.3 Impact factor of publications

Table 15 presents the impact factor of publications, organized by frequency, h-index, and type (journal or conference). 35 different publications vehicles were found, which demonstrates a high variability of studies. H-index sorts the publications. Another point is the predominance of studies published in journals, which commonly have a more significant impact than conferences. Only eight conference publications (proceedings) compose the selected studies. The h-index is defined according to Scimago Journal & Country Rank (SJR, 2017), which is a known portal for evaluating impact factor of publications. As can be seen, good part of the studies present h-index greater than 30, evidencing a reasonable impact factor.

Table 16 presents the top 10 studies that have the highest number of citations according to Google Scholar (GOOGLE, 2017). It is noteworthy that studies S1, S2, S6, S9, and S11 also belong to Initial QGS. This fact gives greater reliability to this set, contributing to quality assurance of the found evidence. Only the studies S21 (2009), S28 (2012), S31 (2013), S34 (2014) and S40 (2016) do not present any citation, but, except for S21, all the others are recent.

Finally, Figure 12 presents the citation graph. The highlighted studies compose the Initial QGS. It is quite noticeable how these studies are strongly related. It occurs, because the most recent works often refer to the older ones, with the result that most of the studies mention the oldest study (S1). The most cited studies among the final dataset are, in this order: S1 (cited by 29 different studies), S2 (16), S5 (15) and S9 (13).

3.3.4 Rigor-relevance analysis

Aiming to answer to RQ2.1 and RQ 2.2, a study of rigor and relevance analysis was carried out (based on IVARSSON; GORSCHEK (2011)). The purpose was to consider the evaluation from a methodological perspective but also looking at the industrial relevance. Figure 13 summarizes the analysis of rigor (x-axis) and relevance (y-axis).

The rigor rubric average was 1.9, whereas relevance rubric average was 3.4. **Appendix C** shows the grades for rigor and relevance of each selected study. Eight studies presented the highest rigor and relevance grades (rigor = 3 and relevance = 4), whereas only one study presented the lowest grade (rigor and relevance = 0). To facilitate the visualization of classification of these studies, we used the distribution presented by MUNIR; WNUK; RUNESON (2016) and VASCONCELLOS et al. (2017), which describes four possible combinations:

- **High rigor and high relevance:** studies with rigor scores from 2 or higher, and relevance scores of 3 or 4;

Table 15 – Impact factor of publications - H-index (SJR, 2017)

Publication	**No. of studies**	**H-index**	**Type**
Communications of the ACM	2	157	Journal
IEEE Transactions on Software Engineering	1	128	Journal
Information and Management	1	119	Journal
Journal of Management Information Systems	3	107	Journal
International Conference on Software Engineering	1	89	Conference
IEEE Software	1	84	Journal
Decision Sciences	1	82	Journal
Journal of Systems and Software	1	72	Journal
IEEE Transactions on Engineering Management	1	69	Journal
Information and Software Technology	3	67	Journal
Journal of Information Technology	1	55	Journal
Empirical Software Engineering	1	45	Journal
Journal of Computer Information Systems	1	43	Journal
ACM SIGMIS Database	1	39	Journal
Benchmarking	1	38	Journal
Studies in Computational Intelligence	1	32	Journal
AT&T technical journal	1	30	Journal
Hawaii International Conference on System Sciences	1	28	Conference
International Journal of Information Technology and Decision Making	1	28	Journal
International Journal of Physical Sciences	1	21	Journal
Communications in Computer and Information Science	1	19	Journal
International Symposium on Empirical Software Engineering and Measurement	1	16	Conference
International Journal of Services and Standards	1	16	Journal
Project Management Journal	1	16	Journal
International Review on Computers and Software	1	13	Journal
Journal of Information and Knowledge Management	1	12	Journal
International Journal of Software Engineering and its Applications	1	9	Journal
Journal of Computing and Information Technology	2	4	Journal
Americas Conference on Information Systems	1	3	Conference
Workshop on Embedded Systems Education	1	2	Conference
Malaysian Software Engineering Conference	1	1	Conference
ACM SIGCPR Conference on Computer Personnel Research	1	0	Conference
Procedia Technology	1	0	Journal
International Conference on Computer and Information Sciences	1	0	Conference
International Journal of Hybrid Information Technology	1	0	Journal

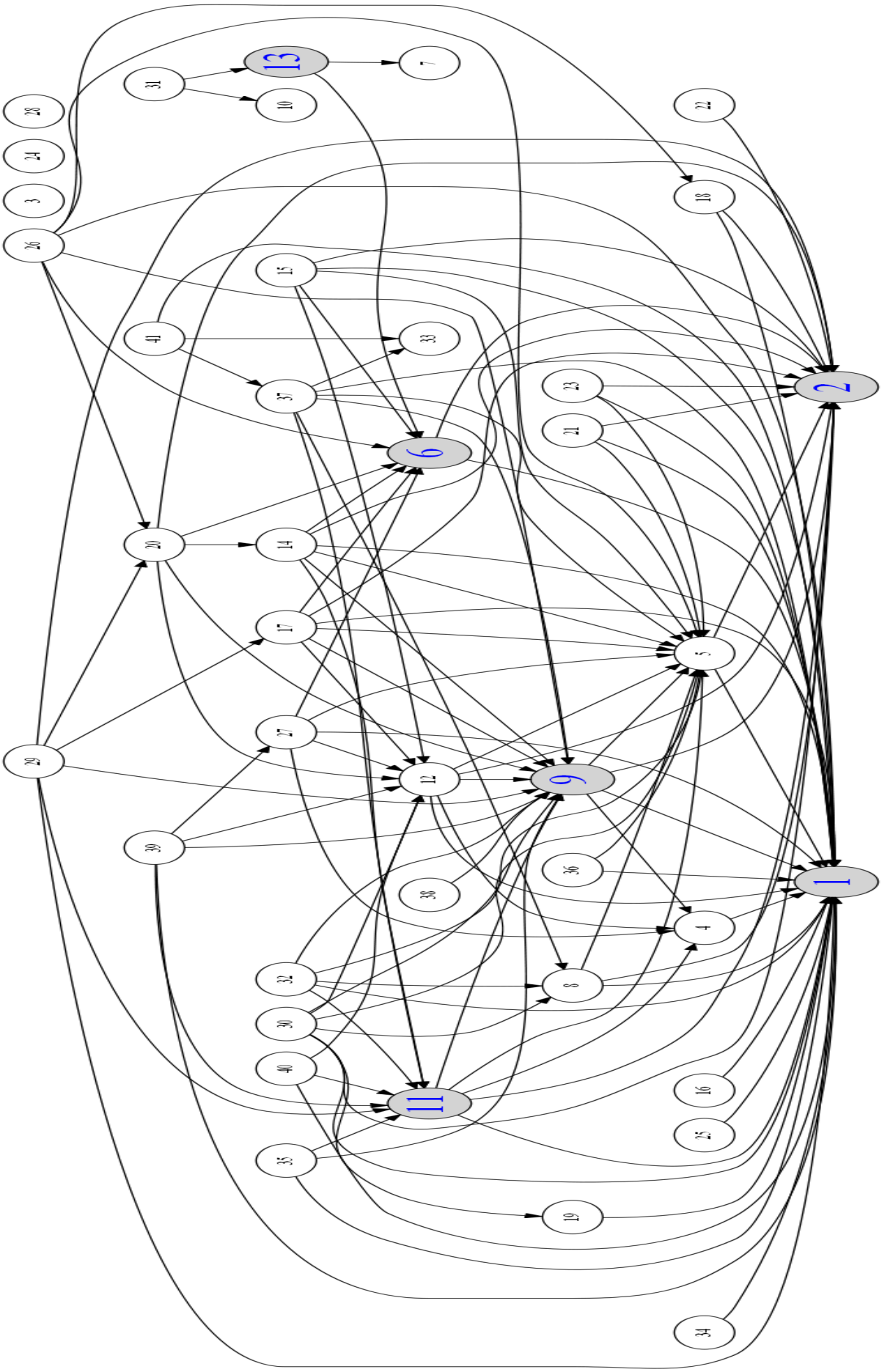

Figure 12 – Citation graph. Highlighted nodes compose the initial QGS.

Table 16 – Top ten cited studies

Study	No. of citations	Publication	Publication type	Year
S1	1953	IEEE Software	Journal	1991
S9	1180	Journal of Management Information Systems	Journal	2001
S5	873	Communications of the ACM	Journal	1998
S2	861	Journal of Management Information Systems	Journal	1993
S11	464	Decision Sciences	Journal	2004
S6	420	IEEE Transactions on Software Engineering	Journal	2000
S12	417	Information and Management	Journal	2004
S20	189	Communications of the ACM	Journal	2009
S10	170	Information and Software Technology	Journal	2002
S17	114	Journal of Management Information Systems	Journal	2007

- **High rigor and low relevance:** studies with rigor scores from 2 or higher, and relevance scores from 0 to 2;
- **Low rigor and high relevance:** studies with rigor scores from 0 and 0.5, and relevance scores of 3 or 4;
- **Low rigor and low relevance:** studies with rigor scores from 0 and 0.5, and relevance scores from 0 to 2;

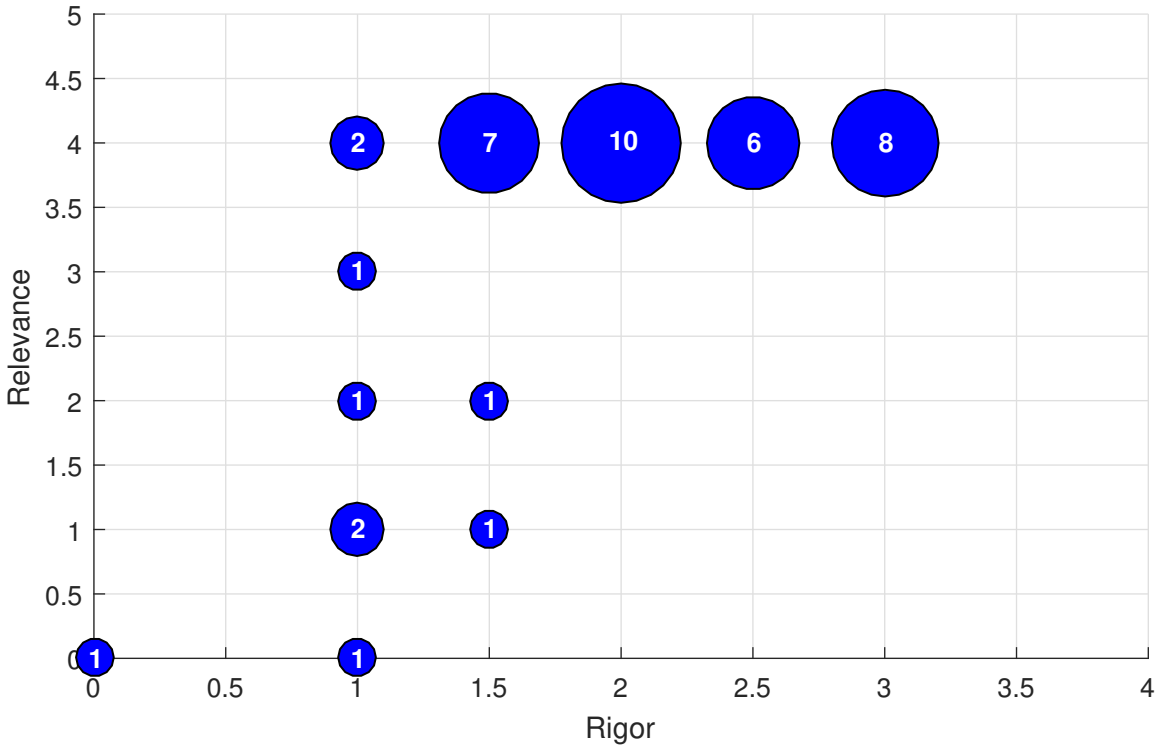

Figure 13 – Rigor-relevance assessment results

Table 17 shows the study distribution according to the presented classification. As can be

observed, the evidence point to works with high relevance. It occurs because the great majority of the studies were carried out in real environments and with industry professionals.

Table 17 – Rigor-relevance study distribution

(Rigor; Relevance)	Studies	No. of studies
High rigor and high relevance		
(3;4)	S6, S9, S17, S18, S26, S30, S34, S38	8
(2.5;4)	S2, S11, S12, S23, S29, S41	6
(2;4)	S4, S5, S7, S10, S15, S19, S32, S33, S37, S40	10
High rigor and low relevance		
None		0
Low rigor and high relevance		
(1;3)	S3	1
(1;4)	S1, S8	2
(1.5;4)	S16, S20, S21, S28, S31, S35, S39	7
Low rigor and low relevance		
(0;0)	S36	1
(1;0)	S25	1
(1;1)	S13, S27	2
(1;2)	S24	1
(1.5;1)	S14	1
(1,5; 2)	S22	1

In the selected papers the following research methods were used to identify risk factors, and respective frequencies: case study (2), database from past projects (5), Delphi (3), focus group (2), interview (7), literature review (17), and survey (23). Figure 14 summarizes the results by research method.

As can be seen, the survey method represents almost 40% of the total frequency, so that it is a relatively well-accepted method to identify risk factors in software development projects. Another interesting result is that several works combine more than one method: from the 41 selected papers, 14 present this scenario, and among them literature review is present in 9 occurrences.

3.4 Discussion

A high number of risk factors were found in the literature. It shows a certain maturity from theoretical viewpoint regarding the main risk factors in software projects.

Analyzing the 11 most mentioned factors, it is clear that requirement risks are the most relevant because they are present in seven occasions. Ambiguity, changes, new technologies, complexity, and incompleteness are pointed out as the risk factors that deserve more attention in software project management. Also, the most mentioned risk factor relates to team skills to develop their activities.

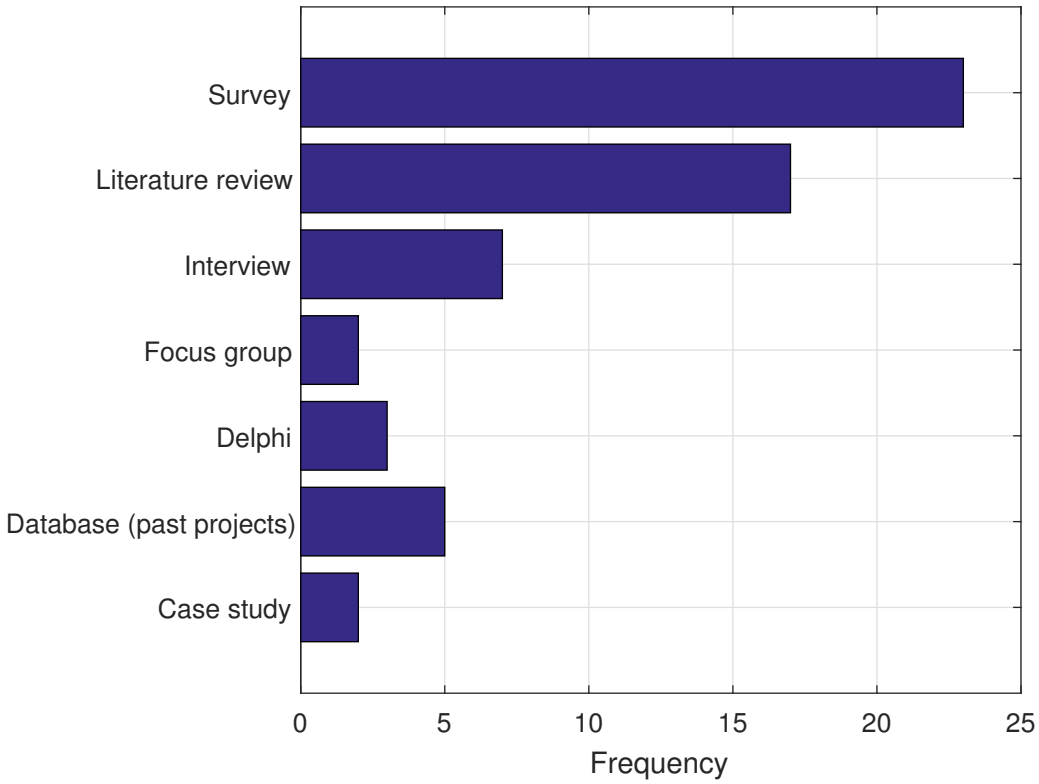

Figure 14 – Used research methods

On the other hand, the most related management factors were: Bad commitment of the user/customer, Unstable organizational environment e No planning or inadequate planning. The first two are external factors to a project that directly impact it, and the last one has a substantial relation to the quality of the project planning, especially regarding scope.

For better reading, next subsections discuss the results by class according to SEI Taxonomy (CARR et al., 1993). Finally, a discussion about the quality of the studies is presented.

3.4.1 Product engineering class

In this class, the element "Requirement" is the one with the highest concentration of identified factors. There are 238 occurrences, distributed in 16 different risks factors. The factor "requirement ambiguity" has the highest number of occurrences and concerns misunderstanding, unclear, inaccurate and vague requirements.

Still, in this class, we can highlight the high occurrence of factors related to: i) changes in requirements, ii) technology that was not used before, iii) technical complexity and iv) incompleteness of requirements. According to the evidence found, the importance of risk analysis associated with requirements in software development projects is evident.

"Design" element, which deals with risks associated with software architecture issues, presents 85 occurrences distributed in 11 different risk factors. The most mentioned one is

related to problems with external component and bad interfaces. Also in the line of interfaces, the factor relating to bad interfaces with legacy systems was mentioned, since the evolution of software often makes it difficult the integration between systems that must not stop.

Regarding the factors that belong to the element “Code and unit test”, we can highlight that the ones that presented greater relevance are: (i) bad quality of code, and (ii) poor adequacy between the hardware and the software. Regarding the first item, questions associated with inconsistency, poor modularization, lack of coding standards, and difficult to understand the code are mentioned. The second item calls attention to the selection of the target hardware so that the software is appropriate. In this element, we found 40 occurrences, distributed in 7 different risk factors.

The element “Integration and test” presents 54 occurrences, distributed in 8 different risk factors. Of these, two stands out. The first one is the bad internal integration of the system, which is related to difficulties in closing the initial release version due to problems in integrating system components. The second one is the lack of proper tests, which reinforces the importance of software testing at the similar level of significance about software development processes.

Finally, the element “Engineering specialties” brought 66 occurrences, distributed in 9 different risk factors. It is clear that the most mentioned factor is related to the low quality of documentation and specifications of the development of software artifacts. Next, factors related to two non-functional requirements (security and usability) are considered necessary for the good progress of software development projects.

The class “Product engineering” presents a good variability of factors that permeate all stages of software development lifecycle. On the other hand, it highlights the importance of software requirements as the most valuable source of information on risk factors.

3.4.2 Development environment class

The element “Development process” presents 91 occurrences, distributed in 9 different risk factors. The most mentioned ones are related to lack of development process, its low quality, and inappropriate. Issues related to lack of project and risk management methodologies are also mentioned. Additionally, there are issues of process control, version control, and requirements traceability process. Thus, the importance of minimally specified process is evident to minimize the chances of project failure.

The element “Development system” has a total of 47 occurrences, distributed in 5 different risk factors. The presented factors concern the importance of appropriate development environments to software projects, including software and hardware. Another important factor considering the found evidence is associated to support to team, users, and vendors. The relative low quantity of factors and occurrences in this element shows that these factors are less important concerning others. It occurs because nowadays there is a better availability of hardware and development environments for most of software applications.

The most recurring risk factor in this class belongs to element “Management process”: No planning or inadequate planning. The findings related to this factor bring information such as: unclear milestones, optimistic or pessimistic planning, bad scope, and unclear or bad identification of the most critical activities. The other factors have a strong relation to bad planning because they involve scope changes and unclear project objectives. We also found factors related to low experience in project management, risk management, and leadership. The element “Management process” presents 138 occurrences, distributed in 12 risk factors, and it is the most cited element in “Development environment” class. In the perspective of risk management, this result makes sense because a well-planned project, with well-trained managers and leaders, and with roles clearly defined is capable of minimizing the chances of project failure.

In the element “Management methods” stands out the risk factor related to low control and monitoring of projects. In this sense, the use of metrics and indicators is cited as an important support-tool, because it allows a less subjective assessment of project progress. It is also worth mentioning the existence of permanent education program so that the team is always prepared for the designated roles. The element “Management methods” brought 55 occurrences, distributed in 7 different risk factors.

Finally, the element “Work environment” presents 85 occurrences, distributed in 8 different risk factors. Three factors stand out: i) Low commitment of staff, ii) Bad communication, and iii) Low morale and conflicts between project staff. Also, we found factors related to cultural differences and language, resistance to changes ad ineffective meetings. It is worth that the low commitment of the team is the second most cited risk factor of the class. Therefore, we can affirm that team motivation and communication are fundamental items for the good progress of software development projects.

3.4.3 Program constraints class

The element “Resources” plays an important role in the risk identification process. We found 260 occurrences, distributed in 24 different risk factors. The factor “Staff does not have required skills”, from element "Resources", presented the most occurrences of all classes. It is a fact that software development projects still have a very high dependence on technical skills of the team. Added to this, we have other factors, such as: low experience, turnover, team and project size relationship, the dependence of few people and low staff availability. Much of this is because an organization usually deals with multiple projects simultaneously, and effective management of people for projects optimizing is necessary. Besides, according do findings, issues related to estimation quality of schedule and budget also play an important role in software project risks. Additionally, we found factors related to work environment, especially the issue of lack of disaster prevention. Hence, we can affirm that the element “Resources” requires the tactical level of management to aid the mitigation of risks, once this level of management analyze the allocation of resources between multiple projects.

The risk factors related to the element “Contract”, in turn, have little apparent influence on software development project. Only 23 occurrences were found, distributed in 5 different risk factors. Much of the studies discuss or presents few risk factors of this element, possibly because software development environments seek to focus more on technical aspects of management and software development.

Finally, the “Program interfaces” element present 232 occurrences, distributed in 27 different risk factors. It is a very representative number, which demonstrates its importance and relevance in software development projects. An interesting result points to the importance of the client participation as a fundamental element for reducing project failure rates. Much of the studies treat the customer as the final user or their representatives. Other important factors, given the number of occurrences, are the influence of the organizational environment and policies on projects. Top management support also is placed as one of the most critical success factors, as well as organizational changes and micromanagement.

3.4.4 Quality of the studies

The results of rigor-relevance analysis demonstrate the good quality of the selected studies. Only five (5) of our reviewed papers used only literature review as a method to raise and assess risk factors. Survey with professionals of the industry is the most used research method, being used by 25 studies. Other mentioned methods are: Delphi, case studies, interviews, focus groups and database of past projects.

In this way, we can state that there is evidence that the found risk factors, if combined, are relevant for the industry and practitioners. The high number of occurrence of factors also can indicate saturation of data. On the other hand, within the evidence, there is no related systematic literature reviews or mapping studies. The studies tend to present a list of factors from ad-hoc literature review, followed by a specific validation study, making it difficult to generalize.

3.5 Limitations and threats to validity

Despite the recommended application of research methods to perform the systematic literature review, it is important to highlight the limitations of this research process so that we can identify points of improvements for future work in this area. In general, we can list the following threats to validity:

- Only one author performed the selection of works and rigor-relevance analysis, under the supervision of the advisors. It is not ideal but follows the recommendations of (KITCHENHAM; CHARTERS, 2007), which accepts that one single researcher can perform Ph.D. works (the case of this study) since the supervisor checks the consistency of the findings and evaluates the research protocol. The researchers have advanced experience and knowledge about project and risk management, which helps minimize flaws;

- Given the high occurrence of risk factors and the lack of a tool that automates data extraction process, there may have been an influence of the researchers in the identification of risk factors. The use of SEI Risk Taxonomy (CARR et al., 1993) as reference model was essential for reducing errors, once it presents a detailed description of classes, elements, and attributes;

- Bias in the quality of the found evidence is also a limitation. To minimize this, we used recommended guidelines for evidence-based software engineering (KITCHENHAM; CHARTERS, 2007; BIOLCHINI et al., 2005);

- We did not perform any snowballing search, and it may have biased the work. The authors chose to not perform due to the high amount of information found only with the selection of papers, which indicated a data saturation. For this work specifically, the point of saturation is achieved due to a high repetition of information and few novel risk factors in most recent studies.

- The databases ACM and IEEExplore search engines presented limitations for systematic literature review. Therefore, we have to make adjustments, trying to be as comprehensive as possible regarding the number of papers about the subject. However, after the end we can state that both databases were not necessary to conduct the review

- The classification of data extracted using the SEI Risk Taxonomy (CARR et al., 1993) may have errors or be biased because we defined it without a tool or statistical method. We just trusted on our experience on the subject to fit the risk factors to existing classes and elements. Although it is a consolidated risk-breakdown structure, if there would be disagreements regarding some classes and elements, it reflects in the results too, being necessary the analysis considering other types of risk classifications.

3.6 Chapter summary

Aiming to map the main risk factors in software development projects, this work performed a systematic literature review. As a results, 41 studies were selected, among which 1414 occurrences of risk factors were identified. After removal of duplication, 148 different risk factors were identified. Next, these factors were categorized according to the classification proposed by SEI Taxonomy (CARR et al., 1993), and finally, as a way of evaluating the quality of the selected studies, we performed a rigor-relevance analysis.

The results shows the relevance of requirement and resources risks. They were crucial to identify the most prevalent risk factors and used to define the indicator Project Risk Index - PRI, that will be presented in the next chapter.

4 PROPOSAL AND ASSESSMENT OF THE INDICATOR

This chapter aims to present and assess a proposal of an indicator called Project Risk Index, whose primary goal is to measure the risk level of a software development project. This indicator uses two main sources of information: risk factors and project characteristics. Firstly we present the proposal, and then two case studies, including the applied methodology, discussions and results.

4.1 The indicator: Project Risk Index (PRI)

Project Risk Index (PRI) is an indicator that seeks to establish the overall risk level of a software development project based on risk factors and project characteristics at a certain moment in a specific environment. Its usage is recommended to improve risk perception, especially in an environment of multiple projects. This indicator can be applied at all stages of a software development lifecycle. Its usage requires the existence of a risk management process, regardless of formalism degree. On the other hand, it is possible to use PRI in an organization that does not have formalized risk management processes, requiring at least risk identification, analysis and monitoring activities.

PRI uses two sources of information: project characterization factors and risk factors. This proposal considers that, in addition to risk factors to be assessed, information about project characteristics can increase the project risk level within the environment.

Risk factors are defined by risk analysis based on risk exposure calculation proposed by BOEHM (1989), which consists of the the product of probability of an unsatisfactory outcome by the consequence of the such an outcome (FAIRLEY, 1994). The Risk Exposure is commonly used to determine the degree of attention to each risk factor to be given, allowing a in-depth analysis and the definition of strategies for mitigation and controlling of most critical risks. PRI proposal uses these information, but also assigns weights for certain classes of risks, bringing a more comprehensive analysis for software development projects' environments.

In the context of multiple projects, it is necessary defining ways of performing an holistic analysis by the valuation of risk levels for each project (and not only for each risk), i.e., defining which projects are riskier. A strategy based on summation of risk exposures only will show the projects that have more identified risks, disregarding the most critical factors. Another alternative would be using the average of risk exposures of a project. However, the usage of average tends not to represent data that reveals extreme trends, since they standardize the dataset (MAGALHÃES; LIMA, 2002), which may not highlight the risk factors with the highest values. Thus, the calculation of risk factors exposure for each project also uses weights to seek to highlight specific classes of risk factors that tend to be more relevant in the context of software development projects.

Some projects, because they present certain characteristics more intensely, may be riskier

regarding others. Additionally, project characteristics give data for analysis of project similarity, i.e., projects with similar characteristics tend to have similar risks. In other words, given its importance, project characteristics are information that must be analyzed inside the organizational environment, regardless the process of risk factor exposure analysis.

4.1.1 Origins of PRI

As stated before, PRI proposal is based on Risk Factors Exposure and Project Characterization Factors. The first one was defined after an mapping of risk factors occurence (chapter 3) and a comparative study of project characteristics (chapter 2). However, its proposal in fact uses and adapt information of other proposals, notably those that are presented in Figure 15.

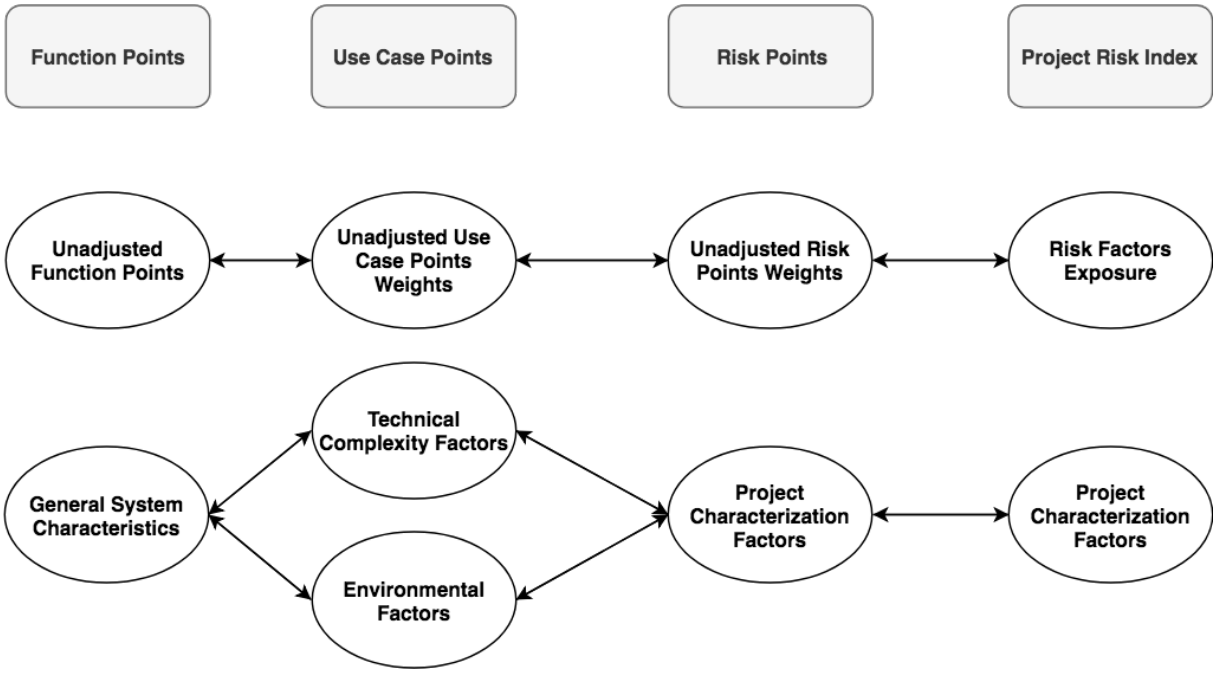

Figure 15 – Mapping between Function Points, UCP, RP, and PRI

Function Points is a consolidated product metric in the literature and software industry (JONES, 2017) whose main goal is to measure the functionality from user point of view using software requirements (ALBRECHT, 1979). Aside, its variations over time, basically there is two sources of information: Unadjusted Function Points and General System Characteristics. Briefly, the first one uses data and transaction functions of a software product, whereas the second one are adjustment factors that rates the general functionality of the software product.

From Function Points idea, the metric Use Case Points was defined (KARNER, 1993; CLEMMONS, 2006), whose main goal is to define a system size. It can be defined as a variation of Function Points using as the main basis the software use cases analysis. A slight difference in this case is the detailing of system characteristics, that were divided into two ones: technical complexity factors and environmental factors. The first one measures functional requirements, whereas the second measures the non-functional requirements. The calculation of

Unadjusted Use Case Point Weights analyzes each use case considering its complexity and the actor complexity.

Risk Points is based on Use Case Points (UCP) analysis (KARNER, 1993; CLEMMONS, 2006). However, adjustments were necessary to make it adherent to software risk management process, but trying to keep the consistency of the formula. We conducted an exploratory case study (chapter 2) using Risk Points in an real environment of software projects, aiming to identify strengths and weaknesses. Its analysis helped in the study of risk factors mapping and comparative analysis of project characterization factors. The Risk Factors Exposure aims to use information from risk factors identification and analysis process, using weights for different risk classes. Then, we use Project Characterization Factors as elements of calibration, once they aim to get information from the environment and the projects that are capable of directly influence the risk factors.

Therefore, the proposal of Project Risk Index (PRI) combined the exploratory case studies results, a analysis of other related works, a comparative analysis of project characteristics (chapter 2) and a mapping of risk factors (chapter 3). There is a stronger relation between PRI and RP, since they bring some similarities, and the calculation idea was based on consolidated previous works, considering risks factors and project characteristics as adjustment factors.

4.1.2 Composition of PRI

PRI indicator is defined as follows:

$$PRI = PCF * RFE \tag{4.1}$$

Where, PCF = **Project Characterization Factor** and RFE = **Risk Factor Exposure**. PCF value is calculated as follows:

$$PCF = \frac{\sum PC}{N} \tag{4.2}$$

Where, PC = Project Characteristic, which represents the corresponding value in a scale of 1 to 5 for each project characteristic, and N represents the total number of used characteristics, considering that the higher is the value, the more its characteristic is risky. The chosen scale follows the recommendation of BOEHM et al. (2000), COELHO (2003), and TRIGO; GUSMÃO; LINS (2008) to measure project characteristics. In this work, PCF is just an arithmetic mean of the Project Characteristics, and we use the following project characteristics and respective scales:

- **PC1: Team size** - defines the composition of team in terms of number of people of a project team. The team size of a project may affect the definition of processes and strategies for managing of the project. It may have impact specially on communication, planning and configuration management, since as the team grows, it requires more formal processes and tools.
    - 1 to 6 people (1)

    - 7 to 20 people (2)
    - 21 to 50 people (3)
    - 51 to 100 people (4)
    - More than 100 people (5)

- **PC2: Geographical distribution** - defines the location of team members and their work environments. The existence of distributed teams through different places and sometimes different organizations may affect their productivity. Distributed teams commonly require more formals ways of communication, as well it must be more clear and detailed.

    - Same room (1)
    - Same building and different rooms (2)
    - Same city and same organization with different buildings (3)
    - Same city and different organizations (4)
    - Different cities (5)

- **PC3: Team experience in the process** - it is one aspect of team experience that considers the usage of a specific software development process previously. This characteristic affects directly on the project management, because an inexperienced team in the process may need to be followed-up closer.

    - Very low (5)
    - Low (4)
    - Average (3)
    - High (2)
    - Very high (1)

- **PC4: Team experience in the application** - it is another aspect of team experience that considers its previous contact with project with similar domains (example of domain: health, education, games, law, etc.). This characteristic may affect mainly in the definition of software requirements.

    - Very low (5)
    - Low (4)
    - Average (3)
    - High (2)
    - Very high (1)

- **PC5: Team experience in the development technology** - this third characteristic related to experience considers the knowledge and experience regarding technical aspects, such as programming languages, frameworks, tools and adopted paradigms. This characteristics affects mainly design and development activities.
    - Very low (5)
    - Low (4)
    - Average (3)
    - High (2)
    - Very high (1)
- **PC6: Project size** - it measures the size of the project in terms of investment. We consider that smaller projects has more chance of success than bigger projects. It may affect several aspects of development process, such as number of activities (which requires a more formal management and better tools), configuration management formalism and project prioritization.
    - Until R$ 50.000,00 (1)
    - Between R$ 50.000,00 and R$ 150.000,00 (2)
    - Between R$ 150.000,00 and R$ 1 million (3)
    - Between R$ 1 million and R$ 3 million (4)
    - Over R$ 3 million (5)

These characteristics were defined based on the works of COELHO (2003); COSTA (2005); TRIGO; GUSMÃO; LINS (2008); BOEHM et al. (2000); VIJAYASARATHY; BUTLER (2016), discussed at chapter 2, to be information capable of minimally characterizing a software development project. Other characteristics could be easily included, removed or adjusted, depending on the environment in which the analyzed projects are inserted. However, it is essential that the same characteristics are used in the same environment, since they may be useful information for the identification of similar projects.

The calculation of the Risk Factors Exposure (RFE) is defined as:

$$RFE = \sum_{i=1}^{n} RE\_weight(risk) * Weight(e_i) \tag{4.3}$$

Where n is the number of identified risk factors and RE_weight(risk) is defined by the result of the calculation of Risk Exposure (RE(risk)) of each identified risk factor, as shown in Table 18. We used a scale from "very low" to "very high" to make the assessment more intuitive.

Table 18 – RE(risk) definition

Classification	Risk Exposure (RE (risk))	RE_ weight (Risk)
Very Low	[0.0, 0.2)	1
Low	[0.2, 0.4)	2
Average	[0.4, 0.6)	3
High	[0.6, 0.8)	4
Very High	[0.8, 1.0]	5

The calculation of RE(risk) for each identified risk is defined according to (BOEHM, 1989) proposal:

$$RE(risk) = Probability(risk) * Impact(risk) \tag{4.4}$$

The Probability defines the probability of occurrence of a risk, and the Impact is the consequence of risk if it occurs. Both probability and impact are assigned on a scale of values between 0 and 1.

Finally, the Weight(e_i) value corresponds to the weight of each risk factor according to elements presented at software development risk taxonomy defined by CARR et al. (1993). The e_i can assume the following values:

- If e_i = requirements, Weight(e_i) = 0.172560113
- If e_i = design, Weight(e_i) = 0.054455446
- If e_i = code and unit test, Weight(e_i) = 0.028288543
- If e_i = integration and test, Weight(e_i) = 0.043847242
- If e_i = engineering specialties, Weight(e_i) = 0.042432815
- If e_i = development process, Weight(e_i) = 0.064356436
- If e_i = development system, Weight(e_i) = 0.033239038
- If e_i = management process, Weight(e_i) = 0.097595474
- If e_i = management methods, Weight(e_i) = 0.038896747
- If e_i = work environment, Weight(e_i) = 0.060113154
- If e_i = resources, Weight(e_i) = 0.18387553
- If e_i = contract, Weight(e_i) = 0.016265912
- If e_i = program interfaces, Weight(e_i) = 0.16407355

The risk factors related to requirements, resources and program interfaces have greater weight concerning others. All these weights were defined based on the number of occurrences of risk factors identified in the systematic literature review (chapter 3). We normalized the values of Table 11, by dividing the number of occurrences of each element by the total number of occurrences (1414).

As well as the project characteristics, these weights can be adjusted according to the environment needs. The list of risk factors considered in this thesis and case studies is available in Table 25, Table 26, and Table 27 (Appendix B).

4.1.3 Guidelines for PRI application

The application of PRI in a real software development environment is simple. The minimum requirements are:

- Risk identification process: it is recommended the usage of a predefined list of risks to be used on all projects inside the organization. Appendix B presents a full version, that may be adapted according to the organization needs;
- Risk analysis process: PRI uses the calculation of risk exposure proposed by BOEHM (1989) for analysis of identified risks

Figure 16 presents a workflow of PRI usage in a project for each collect. As can be seen the only necessary artifact is a risk list to support the process of risk identification and analysis, defining RFE value. The attribution of values for project characteristics for PCF calculation can be performed independently of the risk identification and analysis processes. Then, PRI value is calculated and these set of activities can be repeated periodically.

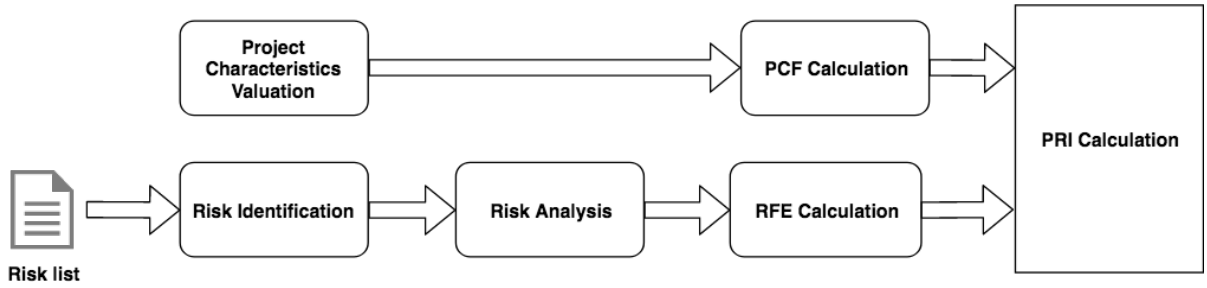

Figure 16 – Steps for PRI calculation in a project for each collect

Another relevant point is that PRI does not depend of the existence of a given process of software development or project management to work. However, it requires at least the project leader to feed the required information for the calculation, as well as to follow-up the most critical risk factors more closely.

PRI usage is more recommended for multiple projects environments, for two reasons: it allows the comparison between projects and allows the definition of a database to establish baselines for the organization, as well as the definition PRI of a new project based on past ones.

By looking PRI in isolation, its value may not make much sense, but its usage in an environment of multiple projects in progress makes it possible to compare projects to identify the riskiest ones. Besides, it makes it possible to assess the overall risk level of a project, identifying possible causes for the variation of the values, allowing the definition of appropriate action plans.

Project characteristics can also be used to categorize projects, i.e., the project with similar characteristics tend to have similar risks. This assumption is useful within an organization with a database of projects and identified risk factors among their respective life cycles. Thus, we can, for example, determine the necessary actions to mitigate the most critical risk factors of a particular project using as the basis the identified risk factors of a project with similar characteristics within the same organizational environment.

Based on the performed studies, considering a scenario of high amount of information obtained over time and the quantity of analyzed projects, it is recommended to use PRI mainly in critical and strategic projects for the organization. Regarding the periodicity of collections, it may vary, but the collecting process in the beginning and in the milestones seems to be coherent, since these moments can reveal changes in project characteristics, as well as in the risk exposure values.

4.1.4 Example of PRI application in practice

To better explain and understand the indicator and how it would be used in practice, this section presents a fictional project called “Toy project”, whose objective is to develop a front-end module of a system for health professionals training. The project is composed of 6 members, distributed in different buildings, and with an average experience in the used technology.

To calculate the Project Characterization Factor (PCF), we consider the following information:

- PC1: Team size: 1 to 6 people (1)
- PC2: Geographical distribution: same city and same organization with different buildings (3)
- PC3: Team experience in the process: Average (3)
- PC4: Team experience in the development technologies: Average (3)
- PC5: Team experience in the application: Average (3)
- PC6: Project size: Between R$ 150.000,00 and R$ 1 million (3)

Therefore, PCF value is:

$$PCF = \frac{PC1 + PC2 + PC3 + PC4 + PC5 + PC6}{6}$$

$$PCF = \frac{1+3+3+3+3+3}{6}$$

$$PCF \approx 2.67$$

Let us assume that in this project we identified and analyzed the risks presented in Table 19. Suppose that the values of probability and impact informed by project leader were told in values on a scale of 1 to 5, in which 1 represents the value very low, and 5 is very high. The value 0, in this case, implies that the risk factor does not exist.

Table 19 – Identified and analyzed risk factors - Toy project

Risk factors	**Probability**	**Impact**	**Product**	**RE (risk)**	**RE_weight(risk)**
Incomplete requirements	3	2	6	0.24	2
Requirements that are difficult to implement	2	1,5	3	0.12	1
Dependence on few people	5	5	25	1	5
Poor code	3.5	3	10.5	0.42	3
Infeasible/incorrect design	2.5	2	5	0.2	2

The Risk Exposure of each risk factor is then calculated (RE(risk)), and as can be seen, the result of the product of the probability by the impact can vary between values 1 to 25 (Product). This result was converted to lie in a range between 0 and 1, dividing each RE(risk) by 25. Finally, following the rule presented in Table 18, the weights based on RE are presented in column RE_weight(risk).

For the calculation of the Risk Factors Exposure (RFE), we did the sum of the products of the RE values by the weight of each risk factor. This weight is defined according to the element of which the risk factor is part. For better understanding, Table 20 shows the consolidated information for the calculation of RFE.

Table 20 – Risk factors and weights (e_i) - Toy project

Risk factors	**RE_weight(risk)**	**Element**	**Weight(e_i)**
Incomplete requirements	2	Requirements (e_1)	0.172560113
Requirements that are difficult to implement	1	Requirements (e_1)	0.172560113
Dependence on few people	5	Resources (e_{11})	0.18387553
Poor code	3	Code and unit test (e_3)	0.028288543
Infeasible/incorrect design	2	Design (e_2)	0.054455446

Therefore, we have:

$$RFE = 2*0,172560113+1*0,172560113+5*0,18387553+3*0,028288543+2*0,054455446$$

$$RFE \approx 1.63$$

Getting the product of PCF by RFE, according to the definition of PRI, the following final value is obtained:

$$PRI = 2.67 * 1.63$$

$$PRI = 4.35$$

The value of PRI corresponds to the risk level of the "Toy project" at a given moment. That is, the higher the PRI, the riskier the project. It is important to emphasize that PRI starts from the premise that, in addition to the identified risk factors, the project characterization factors also contribute to raising the risk level of the project.

4.2 Case studies

Case studies were conducted in two different organizations that have environments of software development projects. The primary objective of these studies was to assess the indicator PRI under distinct aspects, which will be presented in the next sections.

As a way of justifying the adopted research design, it is essential to highlight the inherent challenges of empirical studies in software engineering. Several involved variables are complicated to control, as they involve human and organizational factors. Also, the results are almost dependent on the practitioners, so different people can lead to different results (JURISTO; MORENO, 2013).

Other factors make empirical studies in software engineering more challenging, as mentioned by KONTIO (2001), such as: limitation of resources often force empirical studies to be conducted on limited scope or artificial environments (CURTIS, 1980); software projects are under time pressure, limiting the number of data points and empirical arrangements to be used (GLASS, 1994); in industrial context it is difficult to control sufficient number of parameters (TICHY et al., 1995); technologies frequently change in software engineering, potentially making research results obsolete by the time enough scientific evidence has been collected (TICHY et al., 1995); constructs in software engineering are often large and complex, making their assessment and control in experiments difficult (FITZGERALD, 1991). Such challenges remain recurrent and lead to the need to clearly define the objectives of empirical studies to be conducted aiming to reduce these gaps.

More specifically in software risk management, the challenges regarding empirical studies are getting greater (KONTIO, 2001). We can highlight the following challenges:

- **High subjectivity:** risks perception may vary per participant context so that the data comparison becomes limited;
- **Low quality of data or no historical data:** risk estimation in software development projects are usually subjective, and commonly there is no database of past risks;
- **Projects are unique:** each set of events in a project is unique and not repeatable. It also happens in project risks, so the generalization of results from empirical studies in software projects is very limited since it is necessary to consider aspects and situations in time;
- **Difficulties in assessing the accuracy of methods, processes, indicators or frameworks of risk management:** the single occurrence of a risk cannot be used to draw

any conclusions about the accuracy of the object of study. The ideal would be to have a large number of data points to analyze results more reliably. On the other hand, as previously mentioned, this ideal scenario is almost impossible due to a high degree of subjectivity involved and human and organizational factors that directly influence the study. Hence, the best alternative for this kind of research is to privilege qualitative methods for data collection and analysis;

- **Tendency to modify participants' behavior:** the single introduction of a risk management method inevitably changes the participants' behavior. The participants' sensitivity to risk becomes greater, causing a possible bias in the obtained results;
- **Difficulty to perform empirical studies in real projects:** software projects tend to be long and costly. Because of these limitations, it is often not possible to perform empirical studies, or in other cases the number of data points is limited.

However, these challenges do not prevent the systematic application using scientific principles in the studies. Recognizing such constraints makes it possible to propose a research design so that it can provide more reliable results.

One of the most commonly used empirical approaches in software engineering is case studies, which consists of studies of contemporary phenomena in natural contexts (RUNESON; HÖST, 2009). Case study is a research strategy that focuses on the understanding of the dynamics of a phenomenon in a specific setting (EISENHARDT, 1989). It is widely applied in areas such as: psychology, sociology, political science, social work, business, and community planning. In software engineering, as with the mentioned areas, the application of case studies aims to increase knowledge about individuals, groups, and organizations about social, political, and related phenomena (RUNESON; HÖST, 2009).

In fact, case studies do not generate accurate results like an experiment, but they are capable of providing a better understanding of the studied phenomenon (SEAMAN, 1999). One of the reasons for this is the possibility of mixing quantitative and qualitative data in the analysis of results, although the focus is more significant on qualitative data because it provides a broader and deeper description. Thus, case studies tend to use a mixed approach.

According to PERRY; SIM; EASTERBROOK (2006), a case study should follow the following criteria:

- Have research questions;
- Data is collected in a planned and consistent manner;
- Inferences are made from the data to answer the research questions;
- Explore a phenomenon, or produces an explanation, description, or casual analysis of it;

- Threats to validity are addressed systematically.

As discussed by RUNESON; HÖST (2009), a case study has an exploratory purpose, seeking to find out what is happening, new insights, and generating new ideas and hypotheses. Also, case studies may also be descriptive, if the generality of the phenomenon has secondary importance. The use of an exploratory and descriptive approach makes this study interpretivist (or constructivist), i.e., tries to understand the events through participants' interpretation in their respective contexts. Such characteristics make this research design more flexible so that the key parameters of the study can change over its course.

This chapter reports two case studies that were conducted using the same empirical design. Two software development environments from two different organizations applied the PRI indicator, aiming to capture information and impressions from team leaders' viewpoint. Next sections present objectives, design arrangements and results.

4.2.1 Study goals

The goals of the case studies were to characterize the indicator pros and cons, assess its feasibility, effectiveness, usefulness, and to characterize its applicability in two different environments. Besides, these studies were used to provide practical feedback on the proposed indicator PRI under distinct aspects. The idea is to provide a comprehensive and more in-depth understanding of PRI.

The study goals were defined based on an adaptation of the Goal-Question-Metric paradigm (GQM), which helps to articulate and focus the research questions and refine them in a set of questions, broken down with metrics (BASILI; CALDIERA; ROMBACH, 1994). GQM paradigm is widely accepted and recommended to support empirical studies in software engineering (JURISTO; MORENO, 2013; WOHLIN et al., 2012; TRAVASSOS; GUROV; AMARAL, 2002; BASILI; SELBY, 1991), involving the definition of study goals.

Regarding the mentioned adaptations in GQM paragigm, this work primarily defined goals and questions. However, most of information were got trough interview answers and qualitative date, because we sought to get impressions of the participants according to the defined goals and questions. Table 21 present the GQM statements of the case studies.

From the goals, a set of questions is generated to determine the proposed goals. Finally, each question is analyzed regarding which metrics we need to know to answer each research question. In this work, in addition to metrics, we primarily used qualitative data through the collection of interviews impressions.

As can be seen the GQM objectives of the studies were formulated into five GQM statements:

- **Goal 1: Characterize pros and cons of PRI** - it aims to comprehend PRI and analyze regardings its pros and cons.

Table 21 – GQM Statements of the case studies

Goal 1: Characterize pros and cons of PRI	
Analyze	The PRI indicator
In order to	Characterize it
With respect to	Its pros and cons
From the perspective of	Project leaders
In the context of	Environments of software development projects
Goal 2: Evaluate the feasibility of PRI	
Analyze	The PRI indicator
In order to	Evaluate it
With respect to	Its feasibility in the analyzed environments
From the perspective of	Project leaders
In the context of	Environments of software development projects
Goal 3: Evaluate the effectiveness of PRI	
Analyze	The PRI indicator
In order to	Compare it to Average Risk Exposure
With respect to	Effectiveness
From the perspective of	Project leaders
In the context of	Environments of software development projects
Goal 4: Characterize applicability of PRI in different environments	
Analyze	The PRI indicator
In order to	Characterize it
With respect to	Its applicability in different environments
From the perspective of	Researcher
In the context of	Environments of software development projects
Goal 5: Evaluate the usefulness of PRI	
Analyze	The PRI indicator
In order to	Evaluate it
With respect to	Usefulness
From the perspective of	Project leaders
In the context of	Environments of software development projects

1.1 How much complexity does the use of the indicator causes to your management activities?

1.1.1 Interview (subjective questions)

1.2 Are people confused?

1.2.1 Interview (subjective questions)

1.3 What is the understanding people have about the indicator?

1.3.1 Interview (subjective questions)

1.4 Subjective evaluation of the advantages and disadvantages of using the indicator in the study

1.4.1 Interview (subjective questions)

- **Goal 2: Evaluate the feasibility of PRI** - analyzes PRI in terms of effort and trade-off of using it.

2.1 What was the required effort throughout the process?

2.1.1 Hours log from the case studies

2.2 Was the indicator feasible in the evaluated environments?

2.2.1 Interview (subjective questions)

2.2.2 Number of identified risks

2.2.3 Number of resolved risks

- **Goal 3: Evaluate the effectiveness of PRI** - makes a comparison study between PRI and the Average Risk Exposure.

3.1 What was the collected impressions after comparison of project risk levels using PRI, Average Risk Exposure and some variations of PRI?

3.1.1 Interview (subjective questions)

3.1.2 Observation

3.2 What were the impressions captured about the assessed indicators

3.2.1 Interview (subjective questions)

3.3 Was there any difference between the riskiest project rankings (PRI and ARE - Average Risk Exposure)? Does the ranking presented by the calculation of the indicator make sense?

3.3.1 Comparison of the rankings of the analyzed projects. Participants, before the start of data collection, will set the order of priority of the project risk level. This order will be compared with the calculation performed by the indicator

- **Goal 4: Characterize applicability of PRI in different environments** - tries to get impressions of the author about the application of PRI in both environments.

4.1 What is the context of each environment?

4.1.1 Fill-in form with data from each environment

4.1.2 Interview (subjective questions)

4.2 What are the individual characteristics and background of each participant?

4.2.1 Interview (subjective questions)

4.3 What are the impressions from comparative analysis of indicator performance in both environments?

4.3.1 Triangulation of the researcher's impressions with those of the participants

- **Goal 5: Evaluate the usefulness of PRI** - aims to get impressions from the participants regarding the aspects that make PRI useful.

5.1 In what aspects is the indicator useful for your environment?

5.1.1 Interview (subjective questions)

5.2 What was the degree of influence of risk factors in the calculation of the indicator?

5.2.1 Data analysis of the indicator

5.3 Degree of influence of the project characterization factors in the indicator

5.3.1 Data analysis of the indicator

Each goal was decomposed into characterization questions, and a set of metrics were defined for each question. Some metrics are interview questions, conducted at the end of the study, and other metrics are items that were measured during the study or counted from artifacts generated during the study.

4.2.2 Study design arrangements

The design of the case studies consists of the application of an instance of an agile process of risk management in software development projects (RIBEIRO et al., 2009). This process was chosen for two reasons: i) it is rather a lightweight process, which generates only a single artifact – risk matrix: composed by the list of identified risk factors per project, with respective risk exposure and status, and; ii) the evaluated organizations does not have any explicit risk management process.

Figure 17 shows the set of activities conducted to carry out the case studies. Each case study involves two participants: i) **the researcher**, responsible for planning and following-up of the activities, and; ii) **the project leader**, responsible for selection and characterization of projects, identification and analysis of risks. The studies were carried out in two distinct

environments, where each project leader participated in the research. Both studies were conducted between October 2017 and February 2018.

We applied a sequential set of activities, in which activity 3 has repetitions and took most of the time of both case studies. The details of the activities according to previous figure are presented as follows.

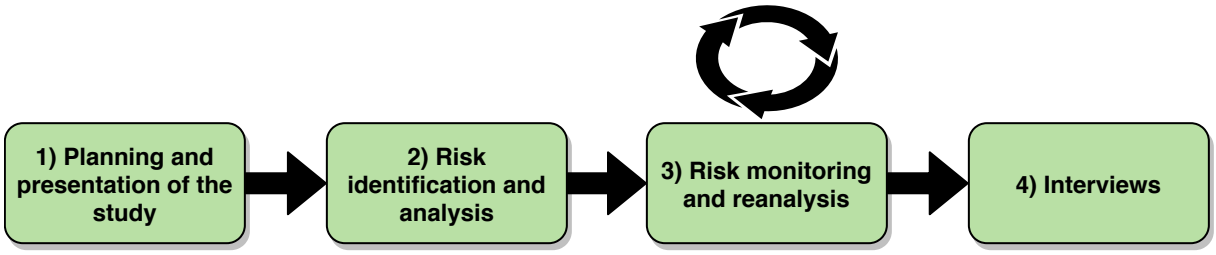

Figure 17 – Study design workflow

1 – Planning and presentation of the study: consists of the first meeting with the project leader, whose objective was to present the process, the indicator PRI and the case study, as well as, its goals and schedule. At this point, the project leaders selected the projects to analyze, as long as these projects were still in progress;

2 – Risk identification and analysis: consists of the project characterization, identification of risk factors and their analysis, through the definition of probability and impact of each risk factor. We applied a form requesting information about each project, its characteristics, risk factors and respective values of probability and impact (in a scale of 1 to 5). This activity was carried out through an online form by the project leaders, without the researcher's influence;

3 – Risk monitoring and reanalysis: consists of following-up of the identified risks through the update of probability and impact values. This activity happened more than once. At that moment, to facilitate the process of data collection and the subsequent assessment of the indicator, we considered only the highest risk factors values of probability and impact. This activity was performed through face-to-face meetings between the project leader and the researcher, in which the values of probability and impact of each risk factor were reanalyzed. Also, we asked the project leaders if they identified new risk factors that were worth consideration in the study. This activity was performed in two rounds. A pause of one month was taken before the beginning of the second round of risk monitoring and reanalysis. No data were collected during the break, and its purpose was to allow the leaders a greater accentuation on the understanding of the indicator, as well as to make a comparison between the moments in which data were collected;

4 – Interviews: were conducted with the project leaders seeking to assess the indicator PRI from the theirs viewpoint regarding the previously established criteria, namely: characterization, feasibility, effectiveness, and usefulness. We performed semi-structured interviews at the end of each study with the project leaders of each environment. The Appendix D presents the interview template used to conduct this activity. All the interviews were recorded, and the audio was transcribed verbatim (transcriptions in Portuguese are available in Appendix E).

Data from both case studies were analyzed considering the study goals, while important issues were also identified and highlighted. When an item was underscored, the experiences from the other case study were compared to it, and rationale and explanations were discussed. In addition to the interviews, the data collection was performed considering multiple sources, which allows data triangulation and can improve the reliability of results (EISENHARDT, 1989; SEAMAN, 1999).

4.2.3 The environments and context

This section presents scenarios used to perform the case studies. As previously mentioned, two distinct organizations participated in the study. Both organizations have ongoing software development projects and well structured IT sectors. Each organization had the participation of projects' leaders, in which they had theoretical and practical knowledge in project management, and they had an in-depth knowledge of the analyzed projects. Next, each environment is detailed, as well as respective projects leaders and examined projects.

4.2.3.1 Organization A

It is a research group, situated in Recife, Brazil, specialized in the development of technologies for education. This organization develops teaching, research and extension projects for different areas of knowledge, but focused on continuing education of professionals. It has about 150 collaborators. This organization, which has existed since 2010, has a group of information technology and design, composed of 17 members, who participate in several software development projects. The environment runs hybrid methods, more focused on Scrum, but the projects are planned using traditional methods. Two projects were used in this study:

- **Project A1:** it aims to develop a web platform for elaboration and management of academic works. The project lasted for twelve months and uses Java and PrimeFaces technologies;
- **Project A2:** it aims to develop an online forum plug-in to a virtual learning environment, using artificial intelligence techniques. The project lasted for six months and uses PHP and jQuery technologies.

The leader of these projects works in this organization since 2012 and is currently responsible for software development and design teams. He has two years of experience in leadership and project management but has a low level of expertise in risk management.

4.2.3.2 Organization B

It is a government organization specialized in law issues. Situated in Recife, Brazil, it is a traditional institution, established in 1822 by the federal government, currently acting

in Pernambuco state. Within this organization, there is a board responsible for actions of information and communication technology. The organization has about 5000 employees, in which the IT sector has more than 100 and approximately 100 systems are developed, which defines the priority actions as projects. The environment runs hybrid methods, in which the most important projects use waterfall methods, combined agile methods, especially kanban. Inside this context, three projects were analyzed:

- **Project B1:** its main objective is to develop tools for management of a certain type of lawsuit, integrated to judicial systems. This project lasted six months;
- **Project B2:** it aims to implement tools to control the frequency of the employees of the organization through access control in turnstiles. This project lasted six months;
- **Project B3:** it aims to develop a support system to send electronic orders for payment of court orders. The project lasted nine months.

All the analyzed projects in this organization used the same technology: Java Enterprise Edition with Oracle database. The project leader has been an employee of the organization for seven years, and currently is a development manager, that follows-up all software development projects life-cycles of the organization. She has four years of practical experience in project management and has a high level of knowledge in risk management.

4.3 Case studies results

We performed five iterations in Organization A. Figure 18 and Figure 19 present the graphs of results presented to the project leaders: PRI and Average Risk Exposure, in which the second one is just the average of risk exposures of each risk factor. 97 risk factors were identified in project A1 and 96 in project A2. For a better understanding of PRI and better comprehension of the factors PRI indices, we followed-up only the risk factors with high and very high-risk exposure. Thus, 19 risk factors were monitored in project A1, and 16 in the project A2. In fact, 27 different risk factors were observed, once some of them co-occurred in both projects.

During the five iterations, project A1 always presented higher values than project A2. According to the project leader, this behavior makes sense, since project A1 was in fact considered as the most critical. This difference is easily perceived because project A1 presents higher PCF and higher risk exposure. Table 22 shows the identified risk factors in organization A and respective Average Risk Exposure (ARE) of each factor. In this study, PCF values did not change, presenting the values 2.2 (project A1) and 1.6 (project A2), respectively. In this light, project A1 present characteristics that make it riskier than project A2.

In both projects, there is a downward trend in PRI values over time. This behavior reflects the fall of Average Risk Exposure over time, as can be seen in Figure 19. Additionally, it can be noticed that the differences between the indices of each project became more accentuated

when we consider the PCFs.Over time, the project A2 was being deprived by the organization because of other projects that become more priority. Project A1 was one of these, and besides, it was close to a significant milestone: the first release version in the production environment. This release was done at the end of 2018 January, and after that, the values fell.

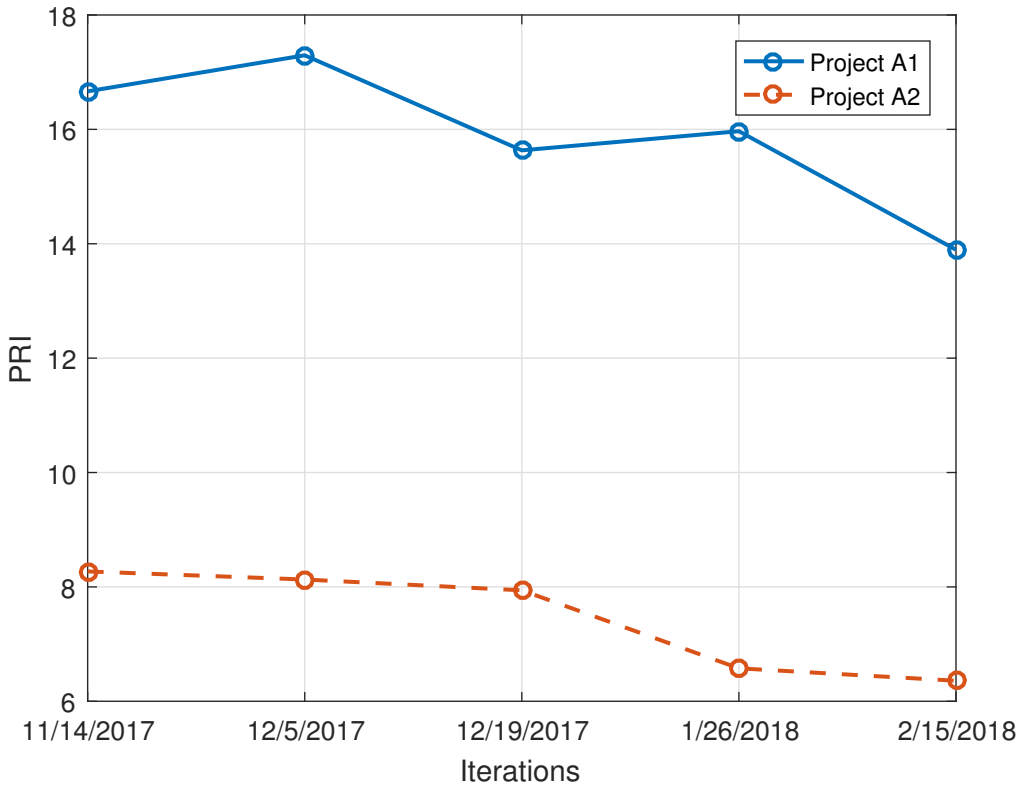

Figure 18 – PRI - Organization A

PRI values for organization B are presented in Figure 20. We performed four data collections (iterations), and 54, 55, and 54 risks factors were identified in projects B1, B2, and B3, respectively. Among them, to keep some risk factors that enabled the monitoring and understanding of the indicator, we monitored only the ones with high or very high-risk exposure. Therefore, we followed-up 10, 12, and 9 risk factors for projects B1, B2, and B3, respectively. In fact, we monitored 19 different risk factors in this organization. Table 23 shows all identified and monitored risk factors for projects B1, B2, and B3, and several average risk exposure of each one.

Regarding the Project Characterization Factors (PCF), they did not vary, presenting the following values: Project B1: 2.4, B2: 2.2, and B3: 2. As can be seen in Figure 20, at the first moment, the project B1 was considered the riskiest but ended up as the least risky at the end of the study. The project B2 had a slight increase, and at the end, it declined, whereas the project B3 showed some regularity until the end of the study. It is very noticeable that all the projects presented a fall in the PRI values during the study, and in the end, we found a situation under control regarding project risks.

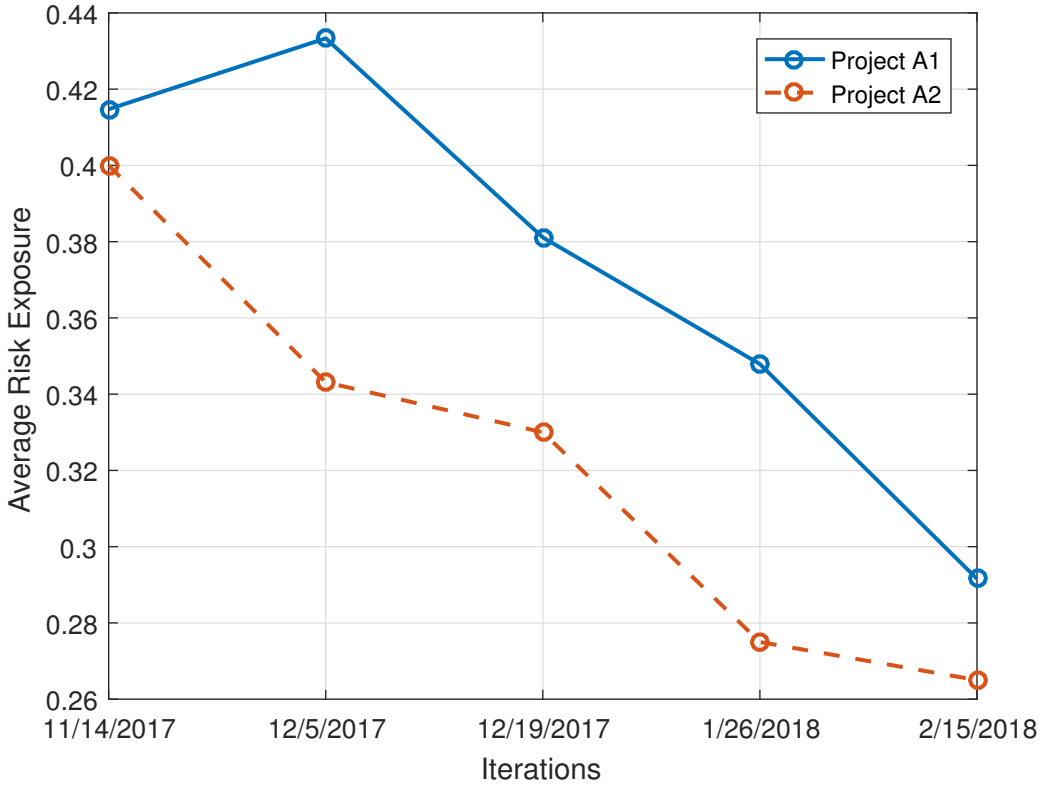

Figure 19 – Average Risk Exposure - Organization A

An important point to be mentioned is that project B2, in the leader's view, was the riskiest. This was not reflected in Figure 20, but there is a reason: before the study started, project B2 team already knew that the senior manager defined it as a project with high priority and critical. So, it made the team aware of this project regarding others. Regarding project B1, when the study started, the leader did not give it due attention, and after the first iteration, we realized that several risk factors were being neglected.

This behavior of fall in the risk levels was also noted in the average risk exposure of each project, as can be seen in Figure 21. In fact, these projects were at a critical moment in the beginning, in which the teams had to perform a task-force because they internally presented severe performance issues. For this reason, the period between the first and second iteration was high.

It is important to notice that organizational factors influenced these projects, more specifically project B2. A change in senior management was planned, and at the penultimate iteration, the project B2 was threatened, due to priority reasons. This situation is clearly visible in Figure 20, but not in Figure 21, because of that risk factor is associated with an element that presents a high weight (Program interfaces). However, the new senior management chose to keep it, so that the oscillations of project values perceive this behavior at the end.

Next sections discuss the results according to study goals (section 4.2.1). The data collection tools used to discuss these results were forms, interviews, and impressions of each

meeting held.

Table 22 – Analyzed risk factors – Organization A (ARE means Average Risk Exposure)

Risk factor	Project A1	Project A2	ARE A1	ARE A2
Dependence on few people	X	X	0.564	0.78
Memory or storage constraints	X		0.332	
Bad availability of testing infrastructure	X		0.364	
Wrong or no stress testing	X	X	0.42	0.244
Lack of or poor risk management	X		0.372	
Lack of experience in risk management methods	X	X	0.344	0.33
Staff does not have required skills	X		0.374	
High technical complexity of the software	X		0.372	
Incomplete requirements	X	X	0.44	0.308
Team is not used to deal with projects of this magnitude	X		0.264	
Presence of requirements that are difficult to implement	X	X	0.288	0.28
Bad adequacy between hardware and specified system	X		0.28	
The team that tests is the same one that implements	X	X	0.44	0.388
Design specification is incomplete	X	X	0.328	0.252
Insufficient discipline and standardization of processes	X	X	0.344	0.332
Staff inexperience	X		0.284	
Delay in recruitment and resourcing	X		0.648	
Technology has not been understood by the project team	X		0.256	
Team workload is very limited	X		0.38	
Poor code		X		0.36
Bad design standards		X		0.248
Unavailability of data needed for testing		X		0.184
Insufficient consideration of reliability/availability		X		0.256
Team's inability to follow the processes		X		0.268
Team size is too small for the project		X		0.53
Lack of project management skills and training for the project manager		X		0.22
The project manager was not given full authority to manage project		X		0.182

Table 23 – Analyzed risk factors - Organization B (ARE means Average Risk Exposure)

Risk Factor	Project B1	Project B2	Project B3	ARE B1	ARE B2	ARE B3
Incomplete requirements	X	X	X	0.3	0.47	0.385
Bad availability of testing infrastructure	X	X	X	0.495	0.4075	0.395
Project progress not monitored closely enough	X			0.59		
Poor infrastructure/user/team support or infrastructure is unavailable	X	X	X	0.4675	0.4825	0.4025
Client's lack of knowledge about the software requirements	X			0.5475		
Unclear project objectives	X			0.25		
Lack of effective project management methodology	X	X		0.295	0.31	
Dependence on few people	X	X	X	0.28	0.3775	0.47
Failure to manage end user expectations	X			0.29		
Bad estimation of schedule	X			0.3325		
The project manager was not given full authority to manage project	X			0.3		
Shortfalls in externally furnished components/Bad interfaces		X	X		0.6	0.3125
Requirement changes		X	X		0.2975	0.23
Lack of proper tests		X			0.29	
Difficulty in system release management and deployment		X	X		0.2	0.37
Scope changes		X			0.29	
Changes in senior management		X			0.12	
Poor interface with customer. contractors. and other departments		X	X		0.3175	0.25
High technical complexity of the software			X			0.22

4.3.1 Characterization of pros and cons

The focus of this result consists of understanding and analyzing pros and cons of PRI, considering the performed case studies. Regarding complexity and effort of application, we realized tha the first data collection process, which consisted of PCF and assessment of risk factors, was a little expensive due to the high amount of factors to be identified and analyzed. However, there was no difficulty in its day-to-day use, and the graphical representation over time has proved to be very important to understand how the indicator worked.

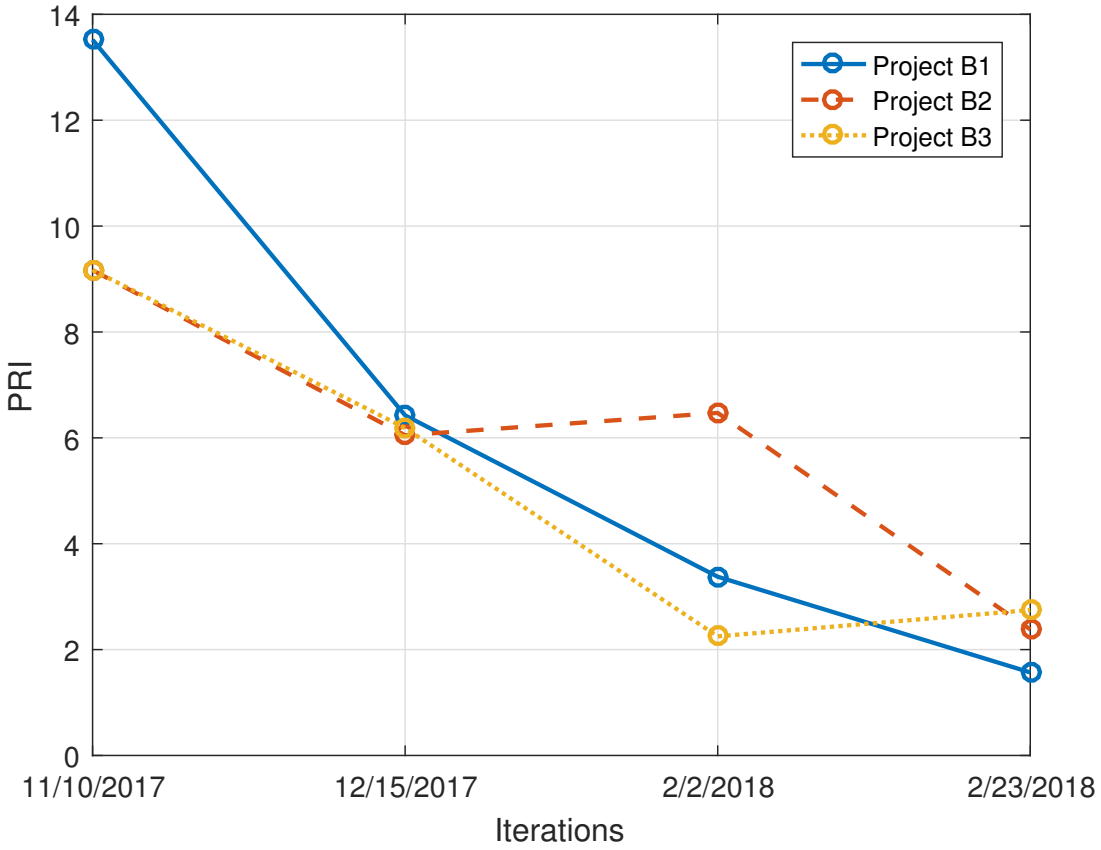

Figure 20 – PRI - Organization B

The understanding about PRI during the studies was evident: it states in a timeline view the risk level of a project in a given environment. It also makes possible to better understand these values when there is more than one project to be analyzed, allowing comparisons.

As positive points, we can highlight that PRI provides better visibility of the risk factors and a holistic understanding of the issues that may vary the values. The indicator provides a snapshot of project risks that reflect their reality regarding risk levels. In other words, the strength of the indicator is its clarity about the presented results.

Improvements in the perception of risks were noted. The project ranking makes the project leader aware of risk factors, aiming to reduce PRI values. Additionally, causes of changes in PRI values is better realized over time. The presented information seemed to be coherent

according to the participants impressions, that is, the combination of risk factors and project characterization factors provide better perception and clarity regarding risks.

In addition to the issue to relatively high effort to assess risks at the first moment, other improvement points were mentioned: to automate the data collecting process and provide additional ways to measure probability and impact, making the process less expensive and more productive.

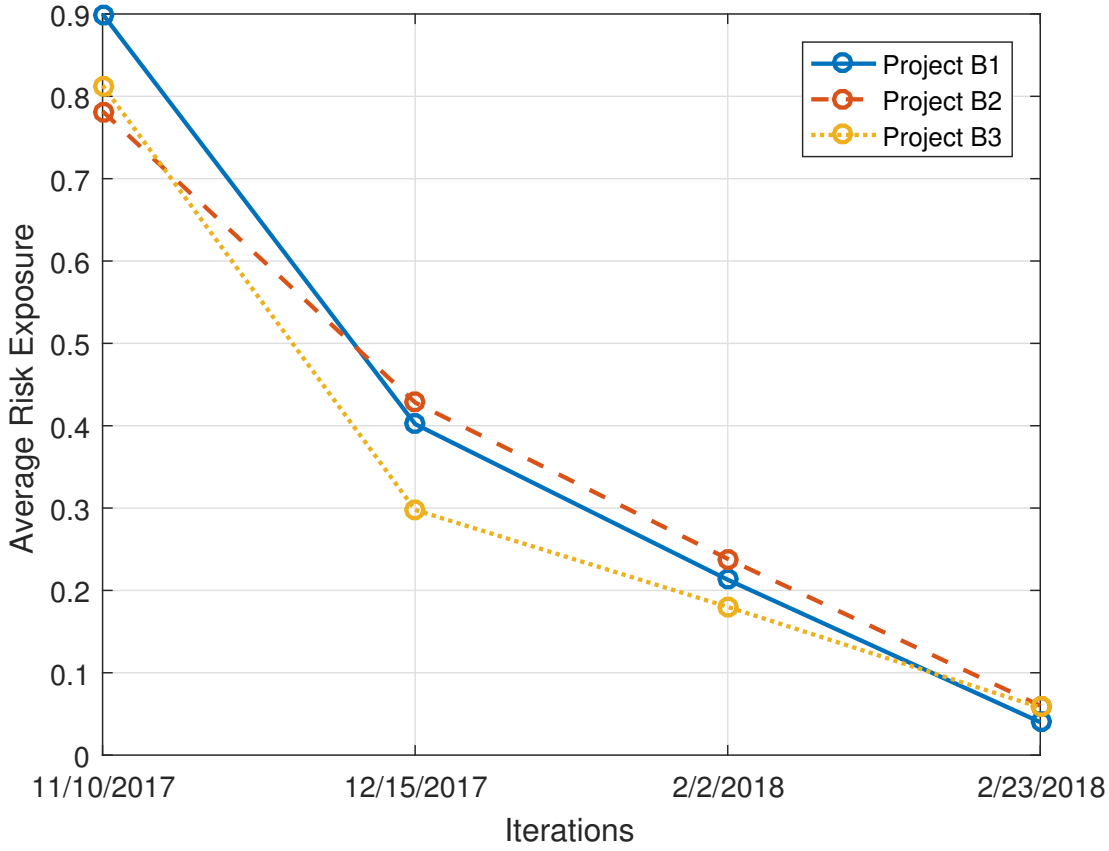

Figure 21 – Average Risk Exposure - Organization B

Another important point to be analyzed in the future is a trade-off analysis of PRI application in real world projects. In this study, for example, we had to follow-up only the highest risks due to the high amount of unimportant identified ones. Therefore, in the real world the frequency of collections should not be weekly, for example. A suggestion to increase PRI strengths is by performing collection only in the beginning and during milestones.

4.3.2 Feasibility

We performed a feasibility analysis of the indicator considering its application in the studied environments. Regarding the effort in hours from leader's point of view, the study took 6 hours and 7.75 hours in organizations A and B, respectively. The effort in organization B was greater because one more project was analyzed.

Considering the performed studies, we can infer that the application of the indicator is feasible, as long as two issues are addressed: (i) the existence of support tools, and (ii) the culture of risk management within the environment. The project characterization factors used in the studies made sense for the participants, but one of them suggested another factor that indicates the priority level of a project for the organization. Regarding risk factors and used weights, both participants consider that they are coherent and comprehensive in the context of software development projects.

Unfortunately, it was not possible to define and execute risk response plans, due to scheduling issues and to the fact that both organizations did not have formal risk management processes. Even so, it was possible to observe that in both organizations the risk levels fell. However, there is no evidence to affirm that the indicator was decisive for this fact. What can be stated is that the indicator helped to improve the perception of risks in the analyzed projects, which indirectly led the project leaders to look more carefully at the identified risk factors. Moreover, in organization B, there were cases of resolved risks: five in project B1 and four in project B2.

Another issue to note is the fact that there is some clarity in the perception of the factors that led a given project to vary its values. It is more evident when the variation is high, so that tends to indicate a kind of noise in the analyzed project. For example, in project A1 there was a tendency for growth in the values when a critical delivery was approaching. And after its delivery, the values began to decrease. In organization B, in general, we realized that the factors that most impacted the projects were related to requirements and external interference.

Finally, it is important to reflect on the application of the indicator on a larger scale. The higher number of projects, the higher the effort required to collect and monitor data, which leads to the need for more in-depth studies about a systematized application within an organization, especially in cases in which there is more than one project leader involved.

4.3.3 Effectiveness

A comparison study of PRI with Average Risk Exposure was performed. This comparison was presented with the support of the project leaders. Additionally, we analyzed PRI variations, mainly regarding calculation of Risk Factor Exposure (RFE) and Project Characterization Factors (PCF).

Figure 19 and Figure 21 present the average risk exposure level of each project. We realize that PRI presented better visibility of project risk levels regarding the other scenarios. The difference between the projects is not so clear using Average Risk Exposure because the values in most cases are very close. During the study, we realized that there is a tendency that the values of risk exposure of each risk factor are close to each other, making it difficult to identify project items that in fact need a closer follow-up due to situations of risks.

Aiming to deepen the effectiveness analysis, we assessed a few other variations of PRI :

- **Scenario 1:** it does not consider Project Characterization Factors, i.e., PCF = 1
- **Scenario 2:** considers that Weight(e_i) = 1;
- **Scenario 3:** it does not consider Project Characterization Factors, i.e., PCF = 1, and considers that Weight(e_i) = 1.

Figure 22 and Figure 23 present the results using these scenarios. Observing the Organization A, we realize that the variations did not produce changes in the ranking. In scenario 1, there was only a reduction in the distancing of risk levels between the projects. In the scenario 2, even with same weights of risk factors, the fact that average risk exposure of project A1 is always greater than that of project A2 contributed to the presented behavior. Finally, the scenario 3 presented the highest decreasing values.

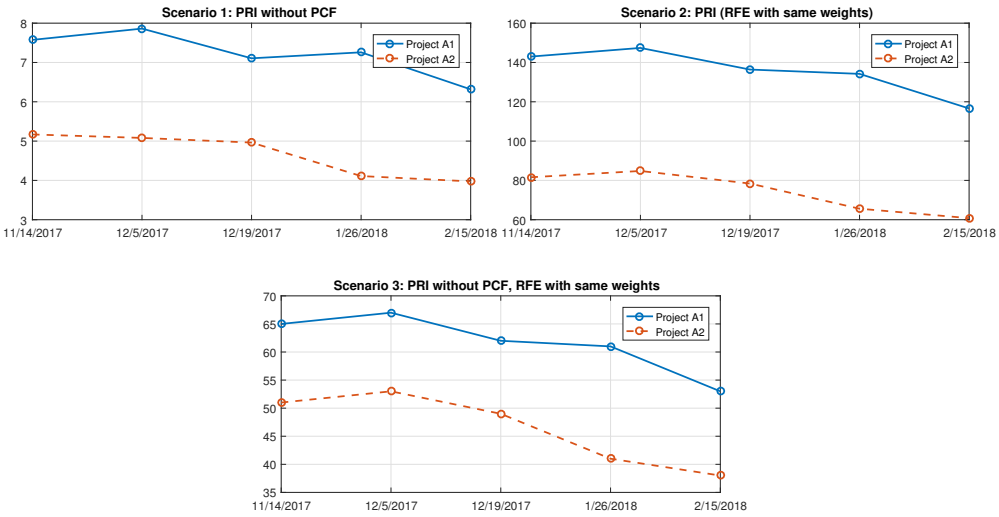

Figure 22 – Variations of PRI - Organization A

Analyzing the variations of PRI in organization A, we realize that the presence of PCF in the calculation is directly proportional to the distance of the values between the projects. So, the difference between PRI values becomes greater when PCF is considered, highlighting that some project characteristics may be enough to determine which project is riskier within the environment.

Observing the Organization B, in some cases, there were changes in the ranking of about PRI. It happened mainly due to the proximity between values in the assessed projects. In this case, we can infer that PCF value influences the distance between the projects, providing greater visibility to the riskier ones. Also, the presence of weights of the risk factors presents greater clarity, once it privileges the most critical factors of software development projects.

Through this analysis, it is noticed that the sensitivity to PCF value is low, only helping to accentuate differences when project characteristics diverge. On the other hand, PRI is very sensitive to RFE, especially if the factor has a high weight. The studies pointed out that Risk

Exposure values were relatively close, which makes it difficult to identify critical points that are often not perceived. Thus, the weights of risk factors are important to highlight the most commonly recurring factors in software development projects.

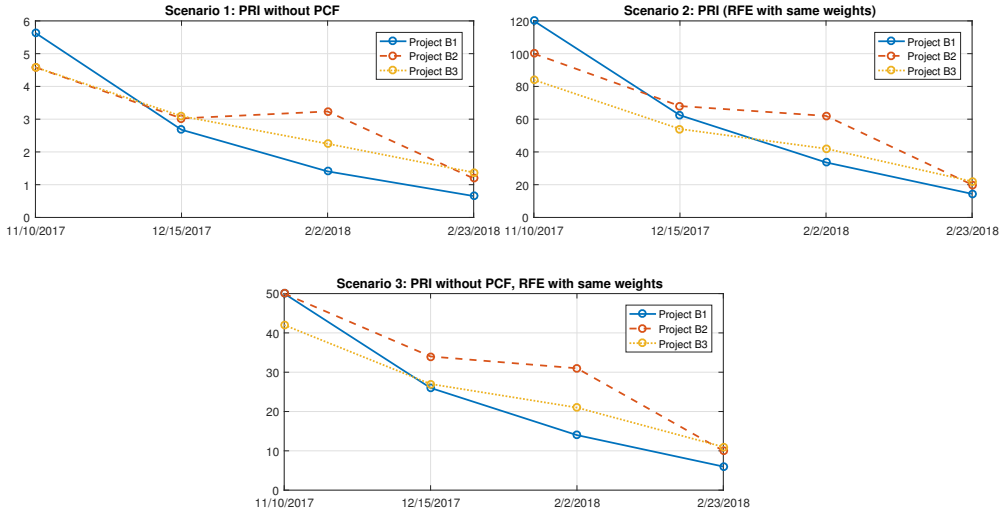

Figure 23 – Variations of PRI - Organization B

Therefore, according to performed case studies, we can say that PRI provide a better view of riskier projects than the average risk exposure. The main benefit of PRI is to clearly understand which projects are most risky and which factors contribute most to the variation of the values. It is reflected in the expectation of the participants in both organizations, whose project rankings were as expected.

4.3.4 Applicability in different environments

As described before, the studies were performed in two very different environments: one is strongly project-oriented and is located in a higher education institution (A), and another is purely from the public sector (B). It is important to note that both organizations do not carry out a formal risk management process. Only organization B identifies some risk factors, but the monitoring is not done.

The Organization A has, in its majority, projects in the area of health education, which demand the development of support tools. The IT team has areas of web and mobile development and design. The vast majority of the employees are temporary, paid for projects (professionals and trainees), and the main limitation of the environment is the low use of tests. From IT infrastructure point of view, the environment has undergone changes, which has made it more robust and reliable.

The Organization B, in turn, is fully sectorized, using a matrix structure. The main difficulty with the execution of projects, according to the project leader, is the distance between team members, both physical and communication issues. This distancing also happens with the

sponsors and the client. Another problem is the high relationship between the developed systems, in which many of these are legacy.

In these environments, there is high applicability of the proposed indicator. The spent effort to identify and monitor risk factors was considered quite acceptable. One positive point is that its use helps to anticipate problems, as well as to handle the most critical aspects of the projects. It happens due to the higher values assigned that can give this prominence, highlighting the most relevant risk factors. The clarity of the presentation of the information in a synthesized way is a strength on the application of PRI in these environments. it is essential that there should be support tools and processes to guarantee its use in a continuous and systematized way.

Therefore, considering only these environments, we see that PRI can be applied. That is, PRI does not depend of the development or management processes. However, it requires a minimum process of risk identification and analysis to be performed.

Finally, it is worth mentioning that there is no evidence that it is possible to compare PRI values in different environments. Each organization has its information, which is strongly dependent on human judgment, which is also influenced by organizational factors.

4.3.5 Usefulness

The participants say that PRI is a useful indicator. However, to generate a real impact, there would have to be a gradual deployment in the organization. One exciting point that could make the usefulness better noticed is the possibility of using data from past projects with similar characteristics to predict possible risk factors in new projects. According to the participants, the view brought by the indicator was useful to anticipate some factors whose attention was not being given, allowing a better approximation of the most critical points of the projects.

The main aspects that make PRI useful, according the participants, were:

- Clarity: the information provided by PRI was clear regarding its purpose. It allowed a proactive view of items that could jeopardize a project;
- Coherence: the project rankings seemed to be coherent, specially when compared to ARE. PRI provided a better comprehension of which projects are riskier than others;
- Resources management: once more than one project was analyzed for each organization, and they shared resources and risks, the application of PRI allowed the perception of resources that are necessary to avoid or mitigate the risks, The view of factors makes the project leader be aware of the identified risks and conduct strategies to mitigate them, especially the risks from the riskier projects.

Hence, there is evidence that the use of the indicator was useful to manage the projects' resources better, directing efforts to the most critical and important points. The visibility and perception of risks became more evident, making the risk management process more robust.

4.4 Threats to validity

This section summarizes some threats that may interfere with the case studies, such as the internal, external, construct and conclusion validity (WOHLIN et al., 2012).

Internal validity. Software development activities are essentially risky ventures because the processes are strongly dependent on knowledge and even on the motivation of individuals. In the same way, empirical studies in projects present several situations and variables whose control is challenging to be done. Aiming to minimize this scenario, both organizations were submitted to the same research design and the same instruments of data collection. Besides, there was no interference of the researcher during the studies, because he acted only as a mediator to collect data and to resolve doubts throughout the process.

External validity. The limited number of participants, data and projects limit the generalizability of the results. Hence, during the selection of projects and environments, we tried to use common scenarios in the context of software development projects, to guarantee some representativeness of the obtained results. Also, the selection of projects to be analyzed is a limitation, because it is strongly dependent of the participants' availability and could not be the best options to be studied. The same may happen in the selection of project leaders.

Construct validity. As discussed earlier, conducting a study in a real environment induces the participants to have a positive attitude regarding risks. It is a strong limiting factor in risk management studies, because the simply adoption of a risk management process in environments that do not have it influences in the results. Therefore, it is tough to dissociate the object of study of the context in which the projects and participants are involved. To minimize this bias, we use a data analysis that is more qualitative, seeking to evaluate the indicator under different aspects to extract as much information as possible, integrating the analysis with the extracted quantitative data.

Conclusion validity. The studies did not allow statistical analysis, because the volume of data would need to be greater in terms of number of projects and participants. Besides, the data collected were dependent on human judgment. For this reason, the studies sought to perform qualitative analysis, aiming to obtain a very detailed understanding of the indicator, its implications and information for calculation within specific contexts. Finally, we used the same instrument for data collection in all projects to reduce bias.

The result analysis was primarily qualitative, and we sought to treat other perspectives to generate better results, especially aiming to minimize the subjective bias of the researcher (MERRIAM; TISDELL, 2015):

- **Credibility:** the most crucial point of credibility is to provide evidence that the findings are credible as the data are presented. For this, we seek to use different data sources, as well as to use the same instrument for data collecting and same process in different environments;
- **Consistency:** is the data consistency regarding the presented results. We tried to

minimize the interference of the research, defining several study goals in order to get more qualitative information to analyze;

- **Transferability:** consists of the definition of what extent the findings can be applied to other situations. We sought to encourage the participants to also analyze the applicability of the indicator in the assessed environments. Certainly, the generalizability of the results is something difficult in empirical studies that involve projects, but there are indications that the characteristics of the analyzed organizations concerning projects cannot be treated as outliers.

4.5 Case studies conclusion and lessons learned

Based on the results reported above, we developed a set of lessons learned that we consider essentials of our case studies. There is evidence that they are independent of the project and as such can be applied to other projects as well.

- **PRI may require low effort to be implemented:** its use does not require a formal risk management process, and it also can be a starting point to encourage its continuous application. The whole process did not take more than 8 hours in a frame of 3 months. The required data are collected so that the result evidences high potential of application in software development projects. Also, the used information to calculate PRI value makes sense;

- **PRI is easy-to-understand:** according to the participant impressions, the objective of PRI and the presented results were consistent with the generated expectation. In fact, PRI helps to note the risk levels of each project within the same environment, and the results are coherent;

- **Importance of support tools:** with a continuous monitoring and increasing volume of data during projects lifecycle, it is crucial to implement software support tools with the objective of automating the process of data collection and presentation;

- **Improved project risk perception:** the synthetic graphical display of risk levels helped project leaders to have a better perception of risks. The perspective of separating the most critical projects from the others brought a better vision of the necessary action plans to improve the performance of the analyzed projects. The reasons for the variation in values were also very evident, especially when it was abrupt;

- **Risk-driven culture:** the use of the indicator would be more effective from the moment the organization starts to have a risk-driven culture. That is, to provide conditions to raise information to anticipate the most critical project factors, as

well as to use information from past projects to generate inference in new projects with similar characteristics. In this sense, it is evident that risk factors and project characterization factors are in fact useful information to assist in the measurement of the risk level of a software development project.

4.6 Chapter summary

This chapter presented the main contribution of this thesis: the indicator PRI - Project Risk Index. Its main objective is to define the risk level of a project using two source information: risk factors and project characterization factors. PRI was assessed through two case studies in two different organizations.

The results indicate that PRI may be a important measure of risk level in environments of software development projects. Its usage is more recommended for multiple project environments, because it can be useful to compare projects and use stored data for inferences in new projects using the similarity of project characteristics.

However, there it is necessary to perform more studies considering different contexts and collecting more data. As they were the first case studies in real projects, the results still present so much qualitative data to analyze and are strongly dependent of the participants profile. So, this study can be considered only an initial kick-off to perform more in-depth studies. Even so, the results were considered positive, as PRI were easy-to-use, easy-to-understand, and independent of any management or development software project.

5 CONCLUSIONS

The existence of projects necessarily involves the presence of risks. The assessment and control of risk factors are essential for the projects to be executed as planned, present good performance, and be successful. Unsuccessful projects or projects with poor performance are not interesting for project-based organizations, because they bring several kinds of losses, and can negatively impact at all levels of management. However, the application of processes, methods or techniques of risk management in software development projects still requires more research strongly related to practice.

One of the possible reasons for this problem is that risk management approaches are very subjective because they are based on human judgment, making it difficult to apply them continuously. One way to minimize this situation is through the use of indicators to support software risk management. Through indicators, it would be possible to synthesize the risk level of a project, make decisions and identify critical points within the environment.

This work addressed this gap, investigating the use of indicators to measure the risk level of software development projects. Despite the main proposal of this thesis - Project Risk Index (PRI) is based on subjective analysis of project leaders, we defined weights and project characteristics that are adherent to software development projects. Considering the scenario of project risk management, it is impossible to dissociate the subjective analysis to the object of the study, once the human judgment is crucial to the acting of management. However, it is still necessary to define indicators to generate useful information for decision-making and assessment considering the amount of identified risks and project characteristics.

We performed several studies that culminate in the proposal and assessment of an indicator that uses two parameters: risk factors and project characterization factors. That is, by using the result of risk identification and analysis to attribute weights, and consider that some project's characteristics may influence directly on project risk level.

In comparison with related works, previously discussed in Section 2.7, the present research performed empirical studies that could analyze the proposal from different perspectives, something that was little explored in previous works. These studies allowed the identification of possibilities of improvements and gathering of other insights not previously explored due to this gap. From this research, we seek to define an indicator also considering the perspectives of the related work, so that it is least dependent on specific processes or format of data, providing a more general view of risk levels in projects and allowing the comparison between then within the same environment.

This simplicity of application makes the use more adherent to different software development processes. In addition, the proposal is customizable, that is, it is possible to adjust weights and use certain characteristics according to the organization features. Therefore, its use can be adapted and extended according to the organizational characteristics as well as pre-established knowledge base.

This proposal was based on evidence synthesized in three empirical studies, supported by literature review: experience report, systematic literature review, and two case studies. The experience report and the systematic literature review supported the proposal, while the case studies assessed it.

The experience report consisted of the application and assessment of an existing proposal that never had been assessed before, in a real environment of software development projects - Risk Points (LOPES, 2005). The results guided the research to identify improvement points, highlighting the importance of better quantifying risk factors and to better investigate the relevance of project characterization factors.

Considering that the identification of risk factors is the most essential activity of the management of risks, the information brought by these factors is crucial and could be parameterized, providing indicators. Thus, risk factors were mapped through a systematic literature review, which brought 41 selected papers, 1414 primary risk factors, which were grouped and categorized, and the duplicate was eliminated, resulting in 148 different risk factors. The findings of this review point out that requirements, resources and program interfaces are the most critical risk elements in software development projects.

Through these results, an indicator called PRI – Project Risk Index was proposed. This proposal was assessed through two case studies, in which we used the same research design. Within the analyzed contexts, the evidence suggests that PRI is consistent, feasible, effective and useful. Both organizations where the case studies were performed have environments of software projects, which allow us to infer that in other settings the results may also be positive.

The findings of this thesis answered positively the research question stated in Chapter 1. Both the risk and the project characterization factors are conceptually essential information to establish the risk level of a project at a given moment. Therefore, there is evidence that the risk level of a software development project can be measured using as sources of information the risk factors and the project characterization factors. Additionally, observing the research objectives also stated in Chapter 1, we realize that they were also achieved in the following way:

- To map risk factors of software development projects in the context of multiple projects environments, contributing to the standardization of risk assessment process: achieved mainly through systematic literature review, whose results contributed to categorization and identification of the most relevant factors;

- To evaluate software project characterization factors as an information source for risk measurement in projects: the view of project's characteristics as additional information to measure its risk level makes sense. The usage of PCF in the case studies, especially in chapter 4 helped to clarify the differences in the project risk levels. However, further research is necessary regarding which factors are significant and what would be the most effective way to parameterize them regarding the risks;

- To evaluate the proposed indicator through case studies in the software industry:

achieved with case studies in two organizations and the capture of participant's impressions

The proposal of PRI is the main characteristic of originality of this work. Related work did not provide an in-depth analysis for environments of multiple software projects, in which there is more information present to justify the need of an indicator that combines it and generate a single value that may represent relevant insights for the environment and for the assessed projects. Additionally, the it was assessed in real projects and the results indicated paths and improvement points to be addressed more clearly.

Therefore, the findings of this thesis can be considered significant for advancing the state of the art in risk management in software projects. Even so, we believe that this research is an initial step for the definition of better ways to measure risk levels of a project, contributing to better understanding and deepening of views of risk management in a slightly different way.

5.1 Research contributions

This work sought to investigate an instrument to support risk management of software development projects: we started with the idea that the risk factors present in most processes of risk identification and project management, in conjunction with project characterizing factors could generate useful information about risk level at a given time. The main contributions of this thesis are organized by themes, as follows:

1. **The usage of an indicator that use risk factors and project characterization factors can contribute to the improvement of risk perception and clarity:** although the use of risk management processes already helps in the levels of risk perception and clarity, the use of indicators may improve them. The good perception in this research means that it is possible to understand which factors are determinant for the variation of the values throughout the time. The good clarity, in turn, concerns the understanding of the indicator's objective, as well as the contribution of its application in environments of software development projects. The visibility of the most critical items and the synthesized view of the risk level of a project allow a rapid reading of project risk exposure, being useful in decision-making to improve the values. It is also important to note that indicators help to observe items or projects that are being neglected;

2. **The usage of an indicator that use risk factors and project characterization factors seems to be coherent:** the parameters used to calculate the indicator allowed to observe its feasibility in real software projects. The project rankings appear to be closed as expected, especially in cases where a project is more risky than others. The use of risk factors and project characterization factors makes sense when used as useful information to measure risks in software development projects;

3. **The used indicators are sensitive to critical moments of a project:** there is a trend that the average value of the risk exposures of each risk factor tends to be low. It means that mechanisms are necessary to ensure that the most critical items are highlighted. According to the conducted studies, there is evidence that the capture of these most critical items is possible through the proposed indicator. That is, when a project presents very high values of PRI regarding others, it means that action plans are necessary to mitigate the risk factors that are harming a project at a given moment. When these most critical factors are controlled, there is a marked downward trend over time;

4. **The assessment of risk factors exposure is a good source of information to indicate the risk level of a project:** since that is possible that risk assessment can become inexpensive, it is interesting to use the result of this activity to generate indicators without generating bottlenecks in the processes of software development. Furthermore, the findings indicate that risk factors related to requirements, external factors, and resources management are the most important in environments of software development;

5. **There are indications that project characteristics contribute to indicating the risk level of a project:** when comparing projects, we can state that some project's characteristics are capable of contributing to one project being riskier than others. The results of this research show that there is evidence that projects' characteristics may help to show the differences between the projects regarding risks. However, this topic lacks more in-depth studies, since such characteristics must be aligned with the organization and the adopted software development processes;

5.2 Limitations

This thesis has limitations and improvement points that must be better analyzed in the future. Here is the highlight of such areas:

- The generalizability is limited. Further empirical studies would be necessary to assess the proposed indicator in practice;

- The use of a systematic literature review to raise and categorize risk factors may be a limitation of the research. Although the literature brought well-developed works methodologically and with good relevance to the industry, that would be necessary the application of data collection through the industry, complementing the view brought by a literature analysis;

- The project characterization factors need further investigations. In this thesis, we decided to simplify the model, based on the arithmetic average of a set of characteris-

tics that can be easily parameterized. However, it requires more in-depth analysis because it was not possible to test the variations of project characterization factors values throughout the time, neither to analyze if the used characteristics were, in fact, the most appropriate;

- It is not possible to state that the assessed indicators present the same results in different environments. What can be stated is that conceptually it makes sense and there is evidence, based on the carried out studies, that it can add value to risk management activities.

- We did not analyzed PRI impact during risk monitoring and controlling activities. This kind of analysis require more dedication from the participants, something that were not possible to perform. Therefore, it is necessary to conduct in-depth studies to effectively assess PRI in terms of risk controlling, i.e., measuring the impact of PRI application in real-world projects after controlling of critical risks and the cost of these actions in risk response.

5.3 Future work

Due to the time constraints to prepare a thesis and the small number of studies in this area, this work can be seen as initial research towards the use of indicators to support risk and project management in the context of software development. Thus, the following issues should be investigated as future work:

- Assessment of risk and project characterization factors among more researchers and practitioners, seeking to reinforce the indicated results, as well as to identify strengths and weaknesses of these results;

- Another relevant point not addressed in this work is the relationship between risk factors. It is important to mention that one or more risk factors may cause a hazard, so an important investigation to be studied is the relationship between risk factors;

- Analyzing the influence of software development environment on risk factors;

- In-depth investigation of risk factors related to software requirements, program interfaces, and resources;

- Considering the increasing importance of the area, once it has been even more common in the software industry, conduct studies focused on Global Software Development (GDS) and Distributed Software Development (DSD);

- Assess the proposed indicator through more empirical studies, allowing the comparison within different environments and further evaluations. Furthermore, the results of

these studies can help in the calibration of the indicator considering specific contexts or the type of the project;

- Investigate other ways to improve the indicator. For example, an item not considered in this proposal is the project manager's experience as a factor that can influence the proposal;
- Development of support tools for risk measurement;
- Investigation of artificial intelligence techniques, aiming to make inferences about new coming projects based on past or ongoing projects, allowing to anticipate risk factors and take preventive actions;
- Definition of a framework for risk-driven software project management.

REFERENCES

ABRAN, A. **Software metrics and software metrology**. John Wiley & Sons, 2010.

ALAM, A. U.; KHAN, S. U.; ALI, I. Knowledge sharing management risks in outsourcing from various continents perspective: a systematic literature review. **Int. J. Digital Content Technol. Appl**, v.6, n.21, p.27–33, 2012.

ALBRECHT, A. J. Measuring application development productivity. In: JOINT SHARE, GUIDE, AND IBM APPLICATION DEVELOPMENT SYMPOSIUM, 1979. **Proceedings...** 1979.

ALENCAR, L. H.; SANTANA, M. O. Análise do gerenciamento de múltiplos projetos na construção civil. **Revista de Gestão e Projetos-GeP**, v.1, n.1, p.74–92, 2010.

AMLAND, S. Risk-based testing:: risk analysis fundamentals and metrics for software testing including a financial application case study. **Journal of Systems and Software**, v.53, n.3, p.287–295, 2000.

ARITUA, B.; SMITH, N. J.; BOWER, D. Construction client multi-projects–A complex adaptive systems perspective. **International Journal of Project Management**, v.27, n.1, p.72–79, 2009.

ARTTO, K.; KAHKONEN, K. **Managing risks in projects**. Spon Press, 2013.

AUDY, J. L. N.; PRIKLADNICKI, R. **Desenvolvimento distribuído de software**. Elsevier, 2007.

BANNERMAN, P. L. A Reassessment of Risk Management in Software Projects. **Handbook on Project Management and Scheduling Vol. 2**, p.1119–1134, 2015.

BARKI, H.; RIVARD, S.; TALBOT, J. Toward an assessment of software development risk. **Journal of management information systems**, v.10, n.2, p.203–225, 1993.

BASILI, V. R.; CALDIERA, G.; ROMBACH, H. D. The Goal Question Metric Approach. In: **Encyclopedia of Software Engineering**. Wiley, 1994.

BASILI, V. R.; SELBY, R. W. Paradigms for experimentation and empirical studies in software engineering. **Reliability Engineering & System Safety**, v.32, n.1-2, p.171–191, 1991.

BECHTOLD, R. Managing risks with metrics. **A term paper for MJY Team Software Risk Management WWW Site**, 1997.

BENAROCH, M.; APPARI, A. Financial pricing of software development risk factors. **IEEE software**, v.27, n.5, p.65–73, 2010.

BHUTA, J.; MALLICK, S.; SUBRAHMANYA, S. A survey of enterprise software development risks in a flat world. In: EMPIRICAL SOFTWARE ENGINEERING AND MEASUREMENT, 2007. ESEM 2007. FIRST INTERNATIONAL SYMPOSIUM ON. **Proceedings...** 2007. p.476–478.

BIOLCHINI, J. et al. **Systematic review in software engineering**. System Engineering and Computer Science Department, COPPE/UFRJ, 2005.

BOEHM, B. W. Software engineering economics. **IEEE transactions on Software Engineering**, n.1, p.4–21, 1984.

BOEHM, B. W. Software risk management. In: ESEC '89: 2ND EUROPEAN SOFTWARE ENGINEERING CONFERENCE. **Proceedings...** 1989.

BOEHM, B. W. Software risk management: principles and practices. **IEEE software**, v.8, n.1, p.32–41, 1991.

BOEHM, B. W. et al. **Software cost estimation with Cocomo II with Cdrom**. Prentice Hall PTR, 2000.

BRASILIANO, A. C. R. **Gestão e Análise de Riscos Corporativos**: método brasiliano avançado. São Paulo: Editora Sicurezza, 2009.

CARDOSO, L. **Indicadores de Produção Limpa**: uma proposta para análise de relatórios ambientais de empresas. 2004. Mestrado Profissional em Gerenciamento e Tecnologia Ambiental no Processo Produtivo — Universidade Federal da Bahia.

CARR, M. J. et al. **Taxonomy-based risk identification**. Carnegie Mellon University, 1993.

CASH, J. I.; MCKENNEY, J. L.; MCFARLAN, F. W. **Corporate Information Systems Management**: text and cases. 3rd.ed. McGraw-Hill Professional, 1992.

CERPA, N.; VERNER, J. M. Why did your project fail? **Communications of the ACM**, v.52, n.12, p.130–134, 2009.

CHEN, S.-M. Fuzzy group decision making for evaluating the rate of aggregative risk in software development. **Fuzzy Sets and Systems**, v.118, n.1, p.75–88, 2001.

CLEMMONS, R. K. Project estimation with use case points. **The Journal of Defense Software Engineering**, p.18–22, 2006.

CMMI-DEV. **CMMI® for Development, Version 1.3**. Carnegie Mellon University, 2010.

COELHO, C. C. **Um Modelo de Adaptação de Processos de Software**. 2003. Dissertação (Mestrado em Ciência da Computação) — Universidade Federal de Pernambuco.

COLMAN, A. M. **Game theory and experimental games**: the study of strategic interaction. Elsevier, 2016.

COSTA, H. **Uma Abordagem Econômica Baseada em Riscos para Avaliação de uma Carteira de Projetos de Software**. 2005. Mestrado em Engenharia de Sistemas e Computação — Universidade Federal do Rio de Janeiro.

COSTA, H.; BARROS, M. O.; TRAVASSOS, G. H. RISICARE: uma ferramenta para apoiar a avaliação de riscos de uma carteira de projetos de software. In: XIV SESSÃO DE FERRAMENTAS - XXI SBES. **Anais...** 2007.

COSTA, H. R.; BARROS, M. d. O.; TRAVASSOS, G. H. Evaluating software project portfolio risks. **Journal of Systems and Software**, v.80, n.1, p.16–31, 2007.

CURTIS, B. Measurement and experimentation in software engineering. **Proceedings of the IEEE**, v.68, n.9, p.1144–1157, 1980.

DANILOVIC, M.; BÖRJESSON, H. Managing the multiproject environment. In: THIRD DEPENDENCE STRUCTURE MATRIX (DSM) INTERNATIONAL WORKSHOP. **Proceedings...** 2001.

DASGUPTA, J.; MOHANTY, R. Estimating operational risk indices for software services outsourcing industry: a case. **International Journal of Services and Standards**, v.6, n.1, p.43–61, 2010.

D'CASTRO, R. J. **Monitoramento online de riscos operacionais no desenvolvimento de software**: um framework baseado em mineração de processos e de dados. 2013. Dissertação (Mestrado em Ciência da Computação) — Universidade Federal de Pernambuco.

DE BAKKER, K.; BOONSTRA, A.; WORTMANN, H. Does risk management contribute to IT project success? A meta-analysis of empirical evidence. **International Journal of Project Management**, v.28, n.5, p.493–503, 2010.

DE MARCO, T. **The deadline**: a novel about project management. Dorset House, 1997.

DOROFEE, A. J. et al. **Continuous Risk Management Guidebook.** CARNEGIE-MELLON UNIV PITTSBURGH PA, 1996.

EISENHARDT, K. M. Building theories from case study research. **Academy of management review**, v.14, n.4, p.532–550, 1989.

EL EMAM, K.; KORU, A. G. A replicated survey of IT software project failures. **IEEE software**, v.25, n.5, 2008.

ELZAMLY, A.; HUSSIN, B. Managing Software Project Risks (Design Phase) with Proposed Fuzzy Regression Analysis Techniques with Fuzzy Concepts. **International Review on Computers and Software (IRECOS)**, v.8, n.11, p.2601–2613, 2013.

ELZAMLY, A.; HUSSIN, B. Managing software project risks (analysis phase) with proposed fuzzy regression analysis modelling techniques with fuzzy concepts. **CIT. Journal of Computing and Information Technology**, v.22, n.2, p.131–144, 2014.

ELZAMLY, A.; HUSSIN, B. Modelling and Evaluating Software Project Risks with Quantitative Analysis Techniques in Planning Software Development. **CIT. Journal of Computing and Information Technology**, v.23, n.2, p.123–139, 2015.

ELZAMLY, A.; HUSSIN, B.; SALLEH, N. M. Top Fifty Software Risk Factors and the Best Thirty Risk Management Techniques in Software Development Lifecycle for Successful Software Projects. **International Journal of Hybrid Information Technology**, v.9, n.6, p.11–32, 2016.

FAIRLEY, R. Risk management for software projects. **IEEE software**, v.11, n.3, p.57–67, 1994.

FAN, C.-F.; YU, Y.-C. BBN-based software project risk management. **Journal of Systems and Software**, v.73, n.2, p.193–203, 2004.

FARIAS, L. L. **Planejamento de Riscos em Ambientes de Desenvolvimento de Software Orientados à Organização**. 2002. Mestrado em Engenharia de Sistemas e Computação — Universidade Federal do Rio de Janeiro.

FENTON, N.; PFLEEGER, S. L. **Software Metrics - A Rigorous & Practical Approach**. PWS Publishing Company, 1997.

FITZGERALD, G. Validating new information systems techniques: a retrospective analysis. **Information systems research: Contemporary approaches and emergent traditions**, p.657–672, 1991.

FONTOURA, L. M.; PRICE, R. T.; PHIL, D. Usando gqm para gerenciar riscos em projetos de software. In: SIMPÓSIO BRASILEIRO DE QUALIDADE DE SOFTWARE (SBQS), 18. **Anais...** 2004. v.18, p.39–54.

FREITAS, B. C. C. **Um modelo para gerenciamento de múltiplos projetos de software**. 2005. Dissertação (Mestrado em Ciência da Computação) — Universidade Federal de Pernambuco.

FREITAS, B. C. C.; MOURA, H. P. GMP: uma ferramenta para a gestão de múltiplos projetos. In: SIMPÓSIO BRASILEIRO DE SISTEMAS DE INFORMAçÃO (SBSI). **Anais...** 2004.

FRIGENTI, E.; COMNINOS, D. **The practice of project management**: a guide to the business-focused approach. Kogan Page Publishers, 2002.

FU, Y.; LI, M.; CHEN, F. Impact propagation and risk assessment of requirement changes for software development projects based on design structure matrix. **International Journal of Project Management**, v.30, n.3, p.363–373, 2012.

GEMINO, A.; REICH, B. H.; SAUER, C. A temporal model of information technology project performance. **Journal of Management Information Systems**, v.24, n.3, p.9–44, 2007.

GERRARD, P.; THOMPSON, N. **Risk-based e-business testing**. Artech House, 2002.

GLASS, R. L. The software-research crisis. **IEEE Software**, v.11, n.6, p.42–47, 1994.

GOODMAN, P. **Software metrics**: best practices for successful it management. Rothstein Associates Inc, 2004.

GOOGLE. **Google scholar citations**. Online; accessed May 2017, `https://scholar.google.com/intl/en/scholar/citations.html`.

GUSMÃO, C. M. G. **Um Modelo de Processo de Gestão de Riscos para Ambientes de Múltiplos Projetos de Desenvolvimento de Software**. 2007. Tese (Doutorado em Ciência da Computação) — Universidade Federal de Pernambuco.

GUSMÃO, C. M. G.; MOURA, H. P. Gerência de risco em processos de qualidade de software: uma análise comparativa. In: SIMPõSIO BRASILEIRO DE QUALIDADE DE SOFTWARE. **Anais...** 2004.

HALL, E. M. **Managing risk**: methods for software systems development. Pearson Education, 1998.

HAN, W.-M.; HUANG, S.-J. An empirical analysis of risk components and performance on software projects. **Journal of Systems and Software**, v.80, n.1, p.42–50, 2007.

HEEMSTRA, F. J.; KUSTERS, R. J. Dealing with risk: a practical approach. **Journal of Information Technology**, v.11, n.4, p.333–346, 1996.

HELDMAN, K. **Project manager's spotlight on risk management**. John Wiley & Sons, 2010.

HIGGINS, J. P.; GREEN, S. **Cochrane handbook for systematic reviews of interventions**. John Wiley & Sons, 2011. v.4.

HIGUERA, R. P. et al. **An introduction to team risk management**. Carnegie Mellon University, 1994.

HIJAZI, H. et al. Identifying Causality Relation between Software Projects Risk Factors. **IJSEIA**, v.8, n.2, p.51–58, 2014.

HILLSON, D. The Risk Breakdown Structure (RBS) as an aid to effective risk management. In: EUROPEAN PROJECT MANAGEMENT CONFERENCE. CANNES, FRANCE, 5. **Proceedings...** 2002. p.1–11.

HOERMANN, S. et al. Comparing risks in individual software development and standard software implementation projects: a delphi study. In: SYSTEM SCIENCE (HICSS), 2012 45TH HAWAII INTERNATIONAL CONFERENCE ON. **Proceedings...** 2012. p.4884–4893.

HOSSEINGHOLIZADEH, A. A source-based risk analysis approach for software test optimization. In: INTERNATIONAL CONFERENCE ON COMPUTER ENGINEERING AND TECHNOLOGY (ICCET), 2. **Proceedings...** 2010. v.2, p.V2–601.

HUMPHREY, W. S. **Managing the software process**. Addison-Wesley, 1989.

HYATT, L. E.; ROSENBERG, L. H. Software metrics program for risk assessment. **Acta astronautica**, v.40, n.2-8, p.223–233, 1997.

IBBS, C. W.; KWAK, Y.-H. Assessing project management maturity. **Project Management Journal**, v.31, n.1, p.32–43, 2000.

IEEE. IEEE Std 1061-1998. **IEEE Standard for a Software Quality Metrics Methodology**, 1998.

ISO/IEC 15939. **Systems and software engineering – Measurement process**. International Organization for Standardization and International Electrotechnical Commission, 2017. International Standard. (15939).

ISO/IEC/IEEE 24765. **Systems and software engineering – Vocabulary**. ISO, IEEE and PMI, 2010. International Standard. (24765).

IVARSSON, M.; GORSCHEK, T. A method for evaluating rigor and industrial relevance of technology evaluations. **Empirical Software Engineering**, v.16, n.3, p.365–395, 2011.

JIANG, J. J.; KLEIN, G.; DISCENZA, R. Information system success as impacted by risks and development strategies. **IEEE transactions on Engineering Management**, v.48, n.1, p.46–55, 2001.

JIANG, J.; KLEIN, G. Software development risks to project effectiveness. **Journal of Systems and Software**, v.52, n.1, p.3–10, 2000.

JONES, C. Project management tools and software failures and successes. **Crosstalk**, p.13–17, 1998.

JONES, C. **Quantifying Software**: global and industry perspectives. CRC Press, 2017.

JURISTO, N.; MORENO, A. M. **Basics of software engineering experimentation**. Springer Science & Business Media, 2013.

KARNER, G. Resource estimation for objectory projects. **Objective Systems SF AB**, v.17, 1993.

KEIL, M. et al. A framework for identifying software project risks. **Communications of the ACM**, v.41, n.11, p.76–83, 1998.

KERZNER, H. **Project Management Metrics, KPIs, and Dashboards**: a guide to measuring and monitoring project performance. John Wiley & Sons, 2013.

KHAN, A. A.; BASRI, S.; DOMINIC, P. Communication risks in GSD during RCM: results from slr. In: INTERNATIONAL CONFERENCE ON COMPUTER AND INFORMATION SCIENCES (ICCOINS). **Proceedings...** 2014. p.1–6.

KIM, E. H.; PARK, Y. Prediction of IS project escalation based on software development risk management. **Journal of Information & Knowledge Management**, v.6, n.02, p.153–163, 2007.

KITCHENHAM, B.; CHARTERS, S. Guidelines for performing systematic literature reviews in software engineering. In: **Technical report, Ver. 2.3 EBSE Technical Report. EBSE**. sn, 2007.

KLIGERMAN, D. C. et al. Sistemas de indicadores de saúde e ambiente em instituições de saúde. **Ciência & Saúde Coletiva**, v.12, n.1, p.199–211, 2007.

KOLODNER, J. **Case-Based Reasoning**. Elsevier Science, 2014.

KONTIO, J. **Software engineering risk management**: a method, improvement framework, and empirical evaluation. Helsinki University of Technology, 2001.

KOOPMAN, P. Risk areas in embedded software industry projects. In: WORKSHOP ON EMBEDDED SYSTEMS EDUCATION, 2010. **Proceedings...** 2010. p.5.

LI, J. et al. A Bayesian Networks-Based Risk Identification Approach for Software Process Risk: the context of chinese trustworthy software. **International Journal of Information Technology & Decision Making**, v.15, n.06, p.1391–1412, 2016.

LOPES, S. **Análise e Definição de Métricas para o Processo de Gerência de Riscos para Projetos de Software**. 2005. Graduation Work in Computer Science — Universidade Federal de Pernambuco.

LÓPEZ, C.; SALMERON, J. L. Risks response strategies for supporting practitioners decision-making in software projects. **Procedia Technology**, v.5, p.437–444, 2012.

MAGALHÃES, M. N.; LIMA, A. C. P. de. **Noções de probabilidade e estatística**. Editora da Universidade de São Paulo, 2002. v.5.

MARCH, J. G.; SHAPIRA, Z. Managerial perspectives on risk and risk taking. **Management science**, v.33, n.11, p.1404–1418, 1987.

MCGARRY, J. **Practical software measurement**: objective information for decision makers. Addison-Wesley Professional, 2002.

MCMANUS, J. **Risk management in software development projects**. Routledge, 2012.

MENEZES, J.; GUSMÃO, C.; MOURA, H. Risk factors in software development projects: a systematic literature review. **Software Quality Journal**, p.1–26, 2018.

MENEZES, J.; GUSMÃO, C.; MOURA, H. Proposal of indicator to support risk management in environments of software development projects. **Journal of Convergence Information Technology**, v.13, n.3, p.41–52, 2018.

MENEZES JR., J. et al. Application of Metrics for Risk Management in Environment of Multiple Software Development Projects. In: INTERNATIONAL CONFERENCE ON ENTERPRISE INFORMATION SYSTEMS - VOLUME 1: ICEIS, 18. **Proceedings...** ScitePress, 2016. p.504–511.

MENEZES JR, J.; GUSMÃO, C.; MOURA, H. Defining indicators for risk assessment in software development projects. **Clei electronic journal**, v.16, n.1, p.11–11, 2013.

MERRIAM, S. B.; TISDELL, E. J. **Qualitative research**: a guide to design and implementation. John Wiley & Sons, 2015.

MIZUNO, O. et al. Characterization of risky projects based on project managers' evaluation. In: SOFTWARE ENGINEERING, 22. **Proceedings...** 2000. p.387–395.

MOORTHY, J. T. S.; IBRAHIM, S. bin; MAHRIN, M. N. Identifying usability risk: a survey study. In: MALAYSIAN SOFTWARE ENGINEERING CONFERENCE (MYSEC), 8. **Proceedings...** 2014. p.148–153.

MOUSINHO, P. O. **Indicadores de desenvolvimento sustentável**: modelos internacionais e especificidades do brasil. 2001. Mestrado em Ciência da Informação — Universidade Federal do Rio de Janeiro.

MUNIR, H.; WNUK, K.; RUNESON, P. Open innovation in software engineering: a systematic mapping study. **Empirical Software Engineering**, v.21, n.2, p.684–723, 2016.

NEVES, S. M. et al. Risk management in software projects through knowledge management techniques: cases in brazilian incubated technology-based firms. **International Journal of Project Management**, v.32, n.1, p.125–138, 2014.

NURDIANI, I. et al. Risk identification and risk mitigation instruments for global software development: systematic review and survey results. In: GLOBAL SOFTWARE ENGINEERING WORKSHOP (ICGSEW), 2011 SIXTH IEEE INTERNATIONAL CONFERENCE ON. **Proceedings...** 2011. p.36–41.

NYFJORD, J.; KAJKO-MATTSSON, M. Software risk management: practice contra standard models. In: SECOND INTERNATIONAL CONFERENCE ON RESEARCH CHALLENGES IN INFORMATION SCIENCE. RCIS 2008. **Proceedings...** 2008. p.65–70.

NYFJORD, J.; KAJKO-MATTSSON, M. Integrating risk management with software development: state of practice. In: IAENG INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, BROWNWALKER PRESS, BOCA RATON, USA. **Proceedings...** 2008.

OXFORD. **Definition of risk**. Online; accessed June 2017, https://en.oxforddictionaries.com/definition/risk.

PA, N. C.; JNR, B. A. A REVIEW ON DECISION MAKING OF RISK MITIGATION FOR SOFTWARE MANAGEMENT. **Journal of Theoretical & Applied Information Technology**, v.76, n.3, 2015.

PARK, R. E.; GOETHERT, W. B.; FLORAC, W. A. **Goal-Driven Software Measurement. A Guidebook**. Carnegie Mellon University, 1996.

PEREIRA, C. G.; MENEZES JR., J. V.; GUSMÃO, C. M. G. Proposta de estrutura analítica de riscos para ambientes de múltiplos projetos de software. In: INTERNATIONAL CONFERENCE ON INFORMATION SYSTEMS AND TECHNOLOGY MANAGEMENT. **Proceedings...** 2015.

PERRY, D. E.; SIM, S. E.; EASTERBROOK, S. Case studies for software engineers. In: SOFTWARE ENGINEERING, 28. **Proceedings...** 2006. p.1045–1046.

PFLEEGER, S. L.; HATTON, L.; HOWELL, C. C. **Solid Software**. Prentice Hall PTR, 2001.

PMI. **A guide to the project management body of knowledge (PMBOK guide)**. 5.ed. Project Management Institute, 2013.

PRESSMAN, R. S.; MAXIM, B. R. **Software engineering**: a practitioner's approach. 8.ed. McGraw-Hill Science/Engineering/Math, 2014.

PROCACCINO, J. D. et al. Case study: factors for early prediction of software development success. **Information and software technology**, v.44, n.1, p.53–62, 2002.

PURDY, G. ISO 31000:2009 - setting a new standard for risk management. **Risk Analysis**, v.30, n.6, p.881–886, 2010.

QINGHUA, P. A model of risk assessment of software project based on grey theory. In: INTERNATIONAL CONFERENCE ON COMPUTER SCIENCE & EDUCATION, 2009. ICCSE'09., 4. **Proceedings...** 2009. p.538–541.

RAD, P. F.; LEVIN, G. **Metrics for project management**: formalized approaches. Management Concepts Inc., 2006.

RAZ, T.; SHENHAR, A. J.; DVIR, D. Risk management, project success, and technological uncertainty. **R&D Management**, v.32, n.2, p.101–109, 2002.

REED, A. H.; KNIGHT, L. V. Differing Impact Levels from Risk Factors on Virtual and Co-Located Software Development Projects. **AMCIS 2009 Proceedings**, p.118, 2009.

REED, A. H.; KNIGHT, L. V. Project risk differences between virtual and co-located teams. **Journal of Computer Information Systems**, v.51, n.1, p.19–30, 2010.

REEVES, J. D. et al. Risk identification biases and their impact to space system development project performance. **Engineering Management Journal**, v.25, n.2, p.3–12, 2013.

RIBEIRO, L. et al. A case study for the implementation of an agile risk management process in multiple projects environments. In: PORTLAND INTERNATIONAL CONFERENCE ON MANAGEMENT OF ENGINEERING & TECHNOLOGY (PICMET 2009). **Proceedings...** 2009. p.1396–1404.

RIFKIN, S.; COX, C. **Measurement in Practice. Software Engineering Institute**. Carnegie Mellon University, 1991.

ROBINSON, E. **Pascal and Probability**: volume 1 in the scientist and science series. Createspace Independent Pub, 2013. (Scientist and Science).

ROPPONEN, J. Risk assessment and management practices in software development. **Beyond the IT Productivity Paradox. John Wiley & Sons, Chichester**, p.247–266, 1999.

ROPPONEN, J.; LYYTINEN, K. Can software risk management improve system development: an exploratory study. **European Journal of Information Systems**, v.6, n.1, p.41–50, 1997.

ROPPONEN, J.; LYYTINEN, K. Components of software development risk: how to address them? a project manager survey. **IEEE transactions on software engineering**, v.26, n.2, p.98–112, 2000.

ROYCE, W. **Software Project Management**: a unified framework. Addison-Wesley, 1998. (Object Technology Series).

RUNESON, P.; HÖST, M. Guidelines for conducting and reporting case study research in software engineering. **Empirical software engineering**, v.14, n.2, p.131, 2009.

SALMERON, J. L.; LOPEZ, C. A multicriteria approach for risks assessment in ERP maintenance. **Journal of Systems and Software**, v.83, n.10, p.1941–1953, 2010.

SAMANTRA, C. et al. Interpretive structural modelling of critical risk factors in software engineering project. **Benchmarking: An International Journal**, v.23, n.1, p.2–24, 2016.

SANGAIAH, A. K. et al. Towards an efficient risk assessment in software projects–Fuzzy reinforcement paradigm. **Computers & Electrical Engineering**, v.71, p.833–846, 2018.

SARAIVA, J. A. G. **Classifying metrics for assessing object-oriented software maintainability**: a family of metrics' catalogs. 2014. Tese (Doutorado em Ciência da Computação) — Universidade Federal de Pernambuco.

SARIGIANNIDIS, L.; CHATZOGLOU, P. D. Quality vs risk: an investigation of their relationship in software development projects. **International Journal of Project Management**, v.32, n.6, p.1073–1082, 2014.

SCHMIDT, R.; LYYTINEN, K.; MARK KEIL, P. C. Identifying software project risks: an international delphi study. **Journal of management information systems**, v.17, n.4, p.5–36, 2001.

SCHWABER, K. **Agile project management with Scrum**. Microsoft press, 2004.

SCHWALBE, K. **Information technology project management**. Cengage Learning, 2015.

SEAMAN, C. B. Qualitative methods in empirical studies of software engineering. **IEEE Transactions on software engineering**, v.25, n.4, p.557–572, 1999.

SHAHZAD, B.; AL-OHALI, Y.; ABDULLAH, A. Trivial model for mitigation of risks in software development life cycle. **International Journal of Physical Sciences**, v.6, n.8, p.2072–2082, 2011.

SHARIF, A. M.; BASRI, S. Risk assessment factors for SME software development companies in Malaysia. In: INTERNATIONAL CONFERENCE ON COMPUTER AND INFORMATION SCIENCES (ICCOINS). **Proceedings...** 2014. p.1–5.

SHARMA, A.; SENGUPTA, S.; GUPTA, A. Exploring risk dimensions in the Indian software industry. **Project Management Journal**, v.42, n.5, p.78–91, 2011.

SHRIVASTAVA, S. V.; RATHOD, U. Categorization of risk factors for distributed agile projects. **Information and Software Technology**, v.58, p.373–387, 2015.

SILVA, S. **Proposta de tratamento de fatores de riscos em desenvolvimento de software para uma organização no setor público**. 2011. Dissertação (Mestrado em Ciência da Computação) — Universidade Federal de Pernambuco.

SIMPLEMAN, L. et al. **Risk management guide for DOD acquisition**. DEFENSE SYSTEMS MANAGEMENT COLL FORT BELVOIR VA, 1998.

SINGH, G.; SINGH, D.; SINGH, V. A study of software metrics. **IJCEM International Journal of Computational Engineering & Management**, v.11, p.22–27, 2011.

SJR. **Scimago Journal & Country Rank**. Online; accessed May 2017, `http://www.scimagojr.com/aboutus.php`.

SOUZA, E. et al. Measurement and control for risk-based test cases and activities. In: LATIN AMERICAN TEST WORKSHOP (LATW´09), 10. **Proceedings...** 2009. p.1–6.

STANDISH GROUP. **Chaos manifesto**. Online; accessed May 2017, `http://www.immagic.com/eLibrary/ARCHIVES/GENERAL/GENREF/S130301C.pdf`.

SULGROVE, R. N. Scoping software projects. **AT&T technical journal**, v.75, n.1, p.35–45, 1996.

SUMNER, M. Risk factors in enterprise wide information management systems projects. In: ACM SIGCPR CONFERENCE ON COMPUTER PERSONNEL RESEARCH, 2000. **Proceedings...** 2000. p.180–187.

TAKAGI, Y.; MIZUNO, O.; KIKUNO, T. An empirical approach to characterizing risky software projects based on logistic regression analysis. **Empirical Software Engineering**, v.10, n.4, p.495–515, 2005.

TANG, A.-g.; WANG, R.-l. Software project risk assessment model based on fuzzy theory. In: INTERNATIONAL CONFERENCE ON COMPUTER AND COMMUNICATION TECHNOLOGIES IN AGRICULTURE ENGINEERING (CCTAE). **Proceedings...** 2010. v.2, p.328–330.

TICHY, W. F. et al. Experimental evaluation in computer science: a quantitative study. **Journal of Systems and Software**, v.28, n.1, p.9–18, 1995.

TIWANA, A.; KEIL, M. Functionality risk in information systems development: an empirical investigation. **IEEE transactions on engineering management**, v.53, n.3, p.412–425, 2006.

TRAVASSOS, G. H.; GUROV, D.; AMARAL, E. **Introdução à engenharia de software experimental**. UFRJ, 2002.

TRIGO, T. R.; GUSMÃO, C.; LINS, A. CBR Risk - Risk Identification Method using Case Based Reasoning. In: INTERNATIONAL CONFERENCE ON INFORMATION SYSTEMS AND TECHNOLOGY MANAGEMENT. **Proceedings...** 2008. v.5.

TSUNODA, M. et al. Analyzing Risk Factors Affecting Project Cost Overrun. **Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing 2012**, p.171–184, 2013.

VAN LOON, H. A management methodology to reduce risk and improve quality. **IT Professional**, n.6, p.30–35, 2007.

VASCONCELLOS, F. J. et al. Approaches to strategic alignment of software process improvement: a systematic literature review. **Journal of Systems and Software**, v.123, p.45–63, 2017.

VERNER, J. M.; ABDULLAH, L. M. Exploratory case study research: outsourced project failure. **Information and Software Technology**, v.54, n.8, p.866–886, 2012.

VIJAYASARATHY, L. R.; BUTLER, C. W. Choice of Software Development Methodologies: do organizational, project, and team characteristics matter? **IEEE Software**, v.33, n.5, p.86–94, Sept 2016.

WALLACE, L.; KEIL, M. Software project risks and their effect on outcomes. **Communications of the ACM**, v.47, n.4, p.68–73, 2004.

WALLACE, L.; KEIL, M.; RAI, A. How software project risk affects project performance: an investigation of the dimensions of risk and an exploratory model. **Decision Sciences**, v.35, n.2, p.289–321, 2004.

WALLACE, L.; KEIL, M.; RAI, A. Understanding software project risk: a cluster analysis. **Information & Management**, v.42, n.1, p.115–125, 2004.

WANDERLEY, M. et al. Proposal of risk management metrics for multiple project software development. **Procedia Computer Science**, v.64, p.1001–1009, 2015.

WARKENTIN, M. et al. Analysis of systems development project risks: an integrative framework. **ACM SIGMIS Database**, v.40, n.2, p.8–27, 2009.

WOHLIN, C.; AURUM, A. Towards a decision-making structure for selecting a research design in empirical software engineering. **Empirical Software Engineering**, v.20, n.6, p.1427–1455, 2015.

WOHLIN, C. et al. **Experimentation in Software Engineering**. Springer Berlin Heidelberg, 2012. (Computer Science).

WYSOCKI, R. K. **Effective project management**: traditional, agile, extreme. John Wiley & Sons, 2011.

ZHANG, H.; BABAR, M. A.; TELL, P. Identifying relevant studies in software engineering. **Information and Software Technology**, v.53, n.6, p.625–637, 2011.

ZHANG, T.; ZHANG, Y. Research on Project Development Key Risk Factors of Small and Medium-Sized Software Enterprises. **Emerging Research in Artificial Intelligence and Computational Intelligence**, p.139–146, 2012.

APPENDIX A – SELECTED PRIMARY STUDIES

ID	Reference
S1	BOEHM, B. W. Software risk management: principles and practices. **IEEE software**, v.8, n.1, p.32–41, 1991
S2	BARKI, H.; RIVARD, S.; TALBOT, J. Toward an assessment of software development risk. **Journal of management information systems**, v.10, n.2, p.203–225, 1993
S3	SULGROVE, R. N. Scoping software projects. **AT&T technical journal**, v.75, n.1, p.35–45, 1996
S4	HEEMSTRA, F. J.; KUSTERS, R. J. Dealing with risk: a practical approach. **Journal of Information Technology**, v.11, n.4, p.333–346, 1996
S5	KEIL, M. et al. A framework for identifying software project risks. **Communications of the ACM**, v.41, n.11, p.76–83, 1998
S6	ROPPONEN, J.; LYYTINEN, K. Components of software development risk: how to address them? a project manager survey. **IEEE transactions on software engineering**, v.26, n.2, p.98–112, 2000
S7	MIZUNO, O. et al. Characterization of risky projects based on project managers' evaluation. In: SOFTWARE ENGINEERING, 22. **Proceedings...** 2000. p.387–395
S8	SUMNER, M. Risk factors in enterprise wide information management systems projects. In: ACM SIGCPR CONFERENCE ON COMPUTER PERSONNEL RESEARCH, 2000. **Proceedings...** 2000. p.180–187
S9	SCHMIDT, R.; LYYTINEN, K.; MARK KEIL, P. C. Identifying software project risks: an international delphi study. **Journal of management information systems**, v.17, n.4, p.5–36, 2001
S10	PROCACCINO, J. D. et al. Case study: factors for early prediction of software development success. **Information and software technology**, v.44, n.1, p.53–62, 2002
S11	WALLACE, L.; KEIL, M.; RAI, A. How software project risk affects project performance: an investigation of the dimensions of risk and an exploratory model. **Decision Sciences**, v.35, n.2, p.289–321, 2004
S12	WALLACE, L.; KEIL, M.; RAI, A. Understanding software project risk: a cluster analysis. **Information & Management**, v.42, n.1, p.115–125, 2004
S13	TAKAGI, Y.; MIZUNO, O.; KIKUNO, T. An empirical approach to characterizing risky software projects based on logistic regression analysis. **Empirical Software Engineering**, v.10, n.4, p.495–515, 2005
S14	TIWANA, A.; KEIL, M. Functionality risk in information systems development: an empirical investigation. **IEEE transactions on engineering management**, v.53, n.3, p.412–425, 2006
S15	KIM, E. H.; PARK, Y. Prediction of IS project escalation based on software development risk management. **Journal of Information & Knowledge Management**, v.6, n.02, p.153–163, 2007
S16	BHUTA, J.; MALLICK, S.; SUBRAHMANYA, S. A survey of enterprise software development risks in a flat world. In: EMPIRICAL SOFTWARE ENGINEERING AND MEASUREMENT, 2007. ESEM 2007. FIRST INTERNATIONAL SYMPOSIUM ON. **Proceedings...** 2007. p.476–478
S17	GEMINO, A.; REICH, B. H.; SAUER, C. A temporal model of information technology project performance. **Journal of Management Information Systems**, v.24, n.3, p.9–44, 2007
S18	COSTA, H. R.; BARROS, M. d. O.; TRAVASSOS, G. H. Evaluating software project portfolio risks. **Journal of Systems and Software**, v.80, n.1, p.16–31, 2007
S19	CERPA, N.; VERNER, J. M. Why did your project fail? **Communications of the ACM**, v.52, n.12, p.130–134, 2009
S20	WARKENTIN, M. et al. Analysis of systems development project risks: an integrative framework. **ACM SIGMIS Database**, v.40, n.2, p.8–27, 2009
S21	REED, A. H.; KNIGHT, L. V. Differing Impact Levels from Risk Factors on Virtual and Co-Located Software Development Projects. **AMCIS 2009 Proceedings**, p.118, 2009

Continued on next page

Continued from previous page

ID	Reference
S22	DASGUPTA, J.; MOHANTY, R. Estimating operational risk indices for software services outsourcing industry: a case. **International Journal of Services and Standards**, v.6, n.1, p.43–61, 2010
S23	REED, A. H.; KNIGHT, L. V. Project risk differences between virtual and co-located teams. **Journal of Computer Information Systems**, v.51, n.1, p.19–30, 2010
S24	KOOPMAN, P. Risk areas in embedded software industry projects. In: WORKSHOP ON EMBEDDED SYSTEMS EDUCATION, 2010. **Proceedings...** 2010. p.5
S25	SHAHZAD, B.; AL-OHALI, Y.; ABDULLAH, A. Trivial model for mitigation of risks in software development life cycle. **International Journal of Physical Sciences**, v.6, n.8, p.2072–2082, 2011
S26	SHARMA, A.; SENGUPTA, S.; GUPTA, A. Exploring risk dimensions in the Indian software industry. **Project Management Journal**, v.42, n.5, p.78–91, 2011
S27	LÓPEZ, C.; SALMERON, J. L. Risks response strategies for supporting practitioners decision-making in software projects. **Procedia Technology**, v.5, p.437–444, 2012
S28	ZHANG, T.; ZHANG, Y. Research on Project Development Key Risk Factors of Small and Medium-Sized Software Enterprises. **Emerging Research in Artificial Intelligence and Computational Intelligence**, p.139–146, 2012
S29	HOERMANN, S. et al. Comparing risks in individual software development and standard software implementation projects: a delphi study. In: SYSTEM SCIENCE (HICSS), 2012 45TH HAWAII INTERNATIONAL CONFERENCE ON. **Proceedings...** 2012. p.4884–4893
S30	VERNER, J. M.; ABDULLAH, L. M. Exploratory case study research: outsourced project failure. **Information and Software Technology**, v.54, n.8, p.866–886, 2012
S31	TSUNODA, M. et al. Analyzing Risk Factors Affecting Project Cost Overrun. **Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing 2012**, p.171–184, 2013
S32	ELZAMLY, A.; HUSSIN, B. Managing Software Project Risks (Design Phase) with Proposed Fuzzy Regression Analysis Techniques with Fuzzy Concepts. **International Review on Computers and Software (IRECOS)**, v.8, n.11, p.2601–2613, 2013
S33	ELZAMLY, A.; HUSSIN, B. Managing software project risks (analysis phase) with proposed fuzzy regression analysis modelling techniques with fuzzy concepts. **CIT. Journal of Computing and Information Technology**, v.22, n.2, p.131–144, 2014
S34	MOORTHY, J. T. S.; IBRAHIM, S. bin; MAHRIN, M. N. Identifying usability risk: a survey study. In: MALAYSIAN SOFTWARE ENGINEERING CONFERENCE (MYSEC), 8. **Proceedings...** 2014. p.148–153
S35	SHARIF, A. M.; BASRI, S. Risk assessment factors for SME software development companies in Malaysia. In: INTERNATIONAL CONFERENCE ON COMPUTER AND INFORMATION SCIENCES (ICCOINS). **Proceedings...** 2014. p.1–5
S36	HIJAZI, H. et al. Identifying Causality Relation between Software Projects Risk Factors. **IJSEIA**, v.8, n.2, p.51–58, 2014
S37	ELZAMLY, A.; HUSSIN, B. Modelling and Evaluating Software Project Risks with Quantitative Analysis Techniques in Planning Software Development. **CIT. Journal of Computing and Information Technology**, v.23, n.2, p.123–139, 2015
S38	SHRIVASTAVA, S. V.; RATHOD, U. Categorization of risk factors for distributed agile projects. **Information and Software Technology**, v.58, p.373–387, 2015
S39	SAMANTRA, C. et al. Interpretive structural modelling of critical risk factors in software engineering project. **Benchmarking: An International Journal**, v.23, n.1, p.2–24, 2016

Continued on next page

Continued from previous page

ID	Reference
S40	LI, J. et al. A Bayesian Networks-Based Risk Identification Approach for Software Process Risk: the context of chinese trustworthy software. **International Journal of Information Technology & Decision Making**, v.15, n.06, p.1391–1412, 2016
S41	ELZAMLY, A.; HUSSIN, B.; SALLEH, N. M. Top Fifty Software Risk Factors and the Best Thirty Risk Management Techniques in Software Development Lifecycle for Successful Software Projects. **International Journal of Hybrid Information Technology**, v.9, n.6, p.11–32, 2016

APPENDIX B – RISK FACTORS LIST

Table 25 – Risk factors - product engineering class

Risk factor	No. of occurrences	Element
Requirement ambiguity	44	Requirements
Requirement changes	32	Requirements
Introduction of new technology	30	Requirements
Shortfalls in externally furnished components/Bad interfaces	27	Design
Technical complexity	25	Requirements
Incomplete requirements	24	Requirements
Bad quality of the specifications/documentation	24	Engineering specialties
Lack of internal system Integration	18	Integration and test
Developing the wrong functions and properties	17	Requirements
Lack of proper tests	15	Integration and test
Bad Requirement verifiability	13	Requirements
Bad Requirement consistency	13	Requirements
Real-time performance shortfalls	12	Design
Project is too big	11	Requirements
Poor code	11	Code and unit test
Memory or storage constraints	10	Design
Security/safety constraints	10	Engineering specialties
Developing the wrong user interface	9	Engineering specialties
Requirement that is difficult to implement	8	Requirements
Infeasible/incorrect design	8	Design
Bad availability of testing infrastructure	8	Integration and test
Poor reuse	8	Integration and test
No design standards	7	Design
Bad adequacy between hardware and specified system	7	Code and unit test
Software difficult to understand due to human factors, such as cultural issues	7	Engineering specialties
Gold-plating	6	Requirements
Inappropriate programming language	6	Code and unit test
Over-optimistic technology perceives	5	Requirements
System size and project duration	5	Requirements
Difficult to implement algorithms	5	Code and unit test
Bad specification of unit testing	5	Code and unit test
Bad stability of technical architecture	5	Code and unit test
There is no specification of the software	5	Engineering specialties
Limited resources for testing	4	Design
Bad target-hardware availability	4	Design
Difficulty in system release management and deployment	4	Integration and test
Technology has not been understood by the project team	3	Requirements
Type of system may be a problem	3	Requirements

Continued on next page

Table 25 – *Continued from previous page*

Risk factor	**No. of occurrences**	**Element**
High project complexity	3	Requirements
Bad availability of data needed for testing	3	Integration and test
No software update plan	3	Engineering specialties
Technological discontinuity	2	Requirements
The development involves legacy applications	2	Design
Insufficient consideration of system reset approach	2	Design
Bad regression testing	2	Integration and test
Bad stress testing	2	Integration and test
Testers are different from development team	2	Integration and test
Some algorithms that may not satisfy the requirements	1	Design
Design specification incompleteness	1	Code and unit test
Non-modifiable requirement document	1	Engineering specialties
Insufficient consideration of reliability/availability	1	Engineering specialties

Table 26 – Risk factors - development environment class

Risk factor	**No. of occurrences**	**Element**
No planning or inadequate planning	25	Management process
Low commitment of staff	20	Work environment
Insufficient discipline and standardization	19	Development process
Unclear project objectives	19	Management process
Ineffective communications between team members	19	Work environment
Scope changes	18	Management process
Project progress not monitored closely enough	18	Management methods
Development methodology was inappropriate for the project	16	Development process
Conflicts between team members	15	Work environment
Lack of effective project management methodology	14	Development process
Inappropriate CASE tools	14	Development system
No project manager's experience	14	Management process
Low morale	14	Work environment
Poor infrastructure/user/team support or infrastructure is unavailable	13	Development system
The available hardware is not enough	12	Development system
Poor interface with customer, contractors, and other departments	12	Management process
Existence of requirement traceability	10	Development process
Lack of clarity of role definitions/responsibilities	10	Management process
Insufficient training of End-Users	10	Management methods
Bad familiarity with the development process	9	Development process
Not managing change properly	9	Management process
Ineffective project organization	9	Management process
Poor quality assurance	9	Management methods
Lack of or poor risk management	8	Development process
Wrong stakeholders management	8	Management process
Poor or nonexistent process control	7	Development process
Available or existing project management system	7	Development system
Poor project leadership	6	Management process
Lack of progress reporting.	6	Management methods
Cultural or language differences	6	Work environment
Poor Coordination between Multiple Teams	5	Development process
Bad commitment of stakeholders	5	Management process
Lack of data needed to keep track of a project.	5	Management methods
Poor configuration control	4	Management methods
Project team members resist change	4	Work environment
Excessive and inefficient meetings	4	Work environment
No version control	3	Development process
Experience in risk management methods	3	Management process
Team's ability to follow the processes	3	Management methods
Team does not follow quality procedures	3	Work environment
Insufficient Documentation of the development system	1	Development system

Table 27 – Risk factors - program constraints class

Risk factor	No. of occurrences	Element
Staff does not have required skills	55	Resources
Bad commitment of the user/customer	37	Program interfaces
Unstable organizational environment	30	Program interfaces
Bad estimation of resources	22	Resources
Unrealistic schedule	21	Resources
Staff inexperience	19	Resources
High turnover	17	Resources
Disagreement with customer	17	Program interfaces
Lack of top management commitment/support to the project	16	Program interfaces
Unstable budget	15	Resources
Unreasonable customers	15	Program interfaces
Team size is too small	14	Resources
Unrealistic budget	14	Resources
Dependence on few people	13	Resources
Corporate politics with negative effect on project	12	Program interfaces
Authoritative senior management	12	Program interfaces
Unstable schedule	10	Resources
Failure to manage end user expectations	10	Program interfaces
Low project priority	10	Program interfaces
Bad contract	9	Contract
Lack of project management skills and training	8	Resources
Bad estimation of budget	8	Resources
No awareness of disasters	8	Resources
Insufficient resources	7	Resources
Bad estimation of schedule	7	Resources
Poor vendors	7	Program interfaces
Poor coordination between multiple vendors	7	Program interfaces
Bad staff availability	6	Resources
Bad interface with subcontractors	6	Program interfaces
Bad commitment of the sponsor	6	Program interfaces
Intellectual property or litigation related issues	5	Contract
Dependencies (another project)	5	Contract
Inexperienced end users	5	Program interfaces
Lack of trust between the client and the team	5	Program interfaces
Client's lack of knowledge about the software requirements	5	Program interfaces
Large number of users (internals or externals) affected	4	Program interfaces
Client knowledge about organizational culture	4	Program interfaces
Bad interface with prime contractor	4	Program interfaces
Change in management	4	Program interfaces

Continued on next page

Table 27 – *Continued from previous page*

Risk factor	**No. of occurrences**	**Element**
Inability to acquire necessary hardware/software	4	Program interfaces
Low percentage of task accomplishment	3	Resources
Uncertainty about the legal environment	3	Contract
Bad interface with associate contractors	3	Program interfaces
Difficult to execute fixed price projects	2	Resources
Delay in recruitment and resourcing	2	Resources
Aggressive schedule affected team motivation	2	Resources
Human errors	2	Resources
The approval of funding took more time	2	Resources
Unsatisfactory facilities	2	Resources
High number of users representatives/customers	2	Program interfaces
User representative is outside of organization	2	Program interfaces
No sponsors or wrong sponsors	2	Program interfaces
The project manager was not given full authority to manage project	2	Program interfaces
Lack of cost and schedule control	1	Resources
Lack of control over consultants, vendors, and subcontractors	1	Contract
Sponsor goals are not clear	1	Program interfaces

APPENDIX C – RIGOR-RELEVANCE ASSESSMENT

Study	Context	Study design	Validity	Rigor	Subjects	Context	Scale	Research method	Relevance
S1	1	0	0	**1**	1	1	1	1	**4**
S2	1	1	0.5	**2.5**	1	1	1	1	**4**
S3	1	0	0	**1**	1	1	1	0	**3**
S4	1	1	0	**2**	1	1	1	1	**4**
S5	1	1	0	**2**	1	1	1	1	**4**
S6	1	1	1	**3**	1	1	1	1	**4**
S7	1	1	0	**2**	1	1	1	1	**4**
S8	1	0	0	**1**	1	1	1	1	**4**
S9	1	1	1	**3**	1	1	1	1	**4**
S10	1	1	0	**2**	1	1	1	1	**4**
S11	1	1	0.5	**2.5**	1	1	1	1	**4**
S12	1	1	0.5	**2.5**	1	1	1	1	**4**
S13	1	0	0	**1**	0	0	1	0	**1**
S14	1	0.5	0	**1.5**	0	0	1	0	**1**
S15	1	0.5	0.5	**2**	1	1	1	1	**4**
S16	1	0.5	0	**1.5**	1	1	1	1	**4**
S17	1	1	1	**3**	1	1	1	1	**4**
S18	1	1	1	**3**	1	1	1	1	**4**
S19	1	1	0	**2**	1	1	1	1	**4**
S20	1	0.5	0	**1.5**	1	1	1	1	**4**
S21	1	0.5	0	**1.5**	1	1	1	1	**4**
S22	1	0.5	0	**1.5**	1	0	0	1	**2**
S23	1	1	0.5	**2.5**	1	1	1	1	**4**
S24	1	0	0	**1**	1	1	0	0	**2**
S25	1	0	0	**1**	0	0	0	0	**0**
S26	1	1	1	**3**	1	1	1	1	**4**
S27	0.5	0.5	0	**1**	0	0	0	1	**1**
S28	1	0.5	0	**1.5**	1	1	1	1	**4**
S29	1	1	0.5	**2.5**	1	1	1	1	**4**
S30	1	1	1	**3**	1	1	1	1	**4**
S31	1	0.5	0	**1.5**	1	1	1	1	**4**
S32	1	1	0	**2**	1	1	1	1	**4**
S33	1	1	0	**2**	1	1	1	1	**4**
S34	1	1	1	**3**	1	1	1	1	**4**
S35	1	0.5	0	**1.5**	1	1	1	1	**4**
S36	0	0	0	**0**	0	0	0	0	**0**
S37	1	1	0	**2**	1	1	1	1	**4**
S38	1	1	1	**3**	1	1	1	1	**4**
S39	0.5	1	0	**1.5**	1	1	1	1	**4**
S40	1	1	0	**2**	1	1	1	1	**4**

Continued on next page

Continued from previous page

Study	Context	Study design	Validity	Rigor	Subjects	Context	Scale	Research method	Relevance
S41	1	1	0.5	**2.5**	1	1	1	1	**4**

APPENDIX D – INTERVIEW TEMPLATE

Interview guide

Presentation

- Thanks to the participant.
- Permission to record the interview audio.
- General data of the participant.
- Explain again how the indicator works.
- Presentation of the evolution of the risk levels of the projects in the environment, showing the values presented after the calculation of the indicator and after the calculation of the average risk exposure.
- Inform the time estimation for the interview.

Objective of the interview: To collect data about the impressions and experiences of the project managers regarding the proposed indicator to characterize, evaluate the feasibility, effectiveness, and usefulness of the indicator.

Interview script

Participant's background

1. Talk about your professional experience in software projects.
2. What role do you currently play in the organization?
3. How many years of experience do you have in project management?
4. How do you consider your level of knowledge in risk management?
5. What do you mean by risk in the context of software development project?

Characterize *PRI*

1. How much complexity and effort do you understand that the use of the indicator causes in your management activities?
2. What is your understanding about the indicator? Did you feel confused?
3. What indicator points do you consider positive?

4. What indicator points do you consider negative?

Evaluate feasibility of *PRI*

1. Do you consider that the application of the indicator is feasible in your environment?
2. In what aspects do you understanding that the project characterization factors contribute to project risk levels?
3. Looking at the project characterization factors, do they make sense? Would you have any suggestion of project characterization factor that would be interesting to include in your analysis?
4. Do the risk factors and their weights make sense to you?
5. Did you noticed any important risk that was not present on the used list?
6. Do you consider that the used process to raise information for the indicator is coherent?
7. Can you realize why indicator's values vary? What was decisive in each project?

Evaluate effectiveness of *PRI*

1. By comparing the project rankings raised by the proposed indicator and by the average risk exposure, can you see any advantage of the indicator concerning average risk exposure? Which are?
2. Did you notice any difference between the ranking of the riskiest projects?
3. Does the presented ranking make sense?
4. Talk about your perception of the use of the indicator about the period in which you did not use it.

Characterize applicability of *PRI* in different environments

1. Talk about the environment in which the projects are inserted - day-to-day, strengths, weaknesses, etc. What are your greatest difficulties?
2. What development methods are used in your environment?
3. Talk about the practices of risk management in your environment. Is there any process? How often is it used? If so, what techniques are used?
4. Did the application of the indicator bring any benefit? Which are?
5. Did the application of the indicator bring any harm? Which are?

Evaluate the usefulness of *PRI*

1. Do you consider the indicator useful? In what aspects?
2. In what ways do you understand that the indicator can add value to your activities?
3. Has the process in general taken more time than you expected? Did the time spent on the performed activities add value?

Final considerations

1. In general, what is the impact of the indicator in your environment?
2. What are the most critical points of the indicator?
3. What do you consider of the carried out study that was positive?
4. What points of improvement have you identified throughout the study?

APPENDIX E – INTERVIEW TRANSCRIPTIONS - IN PORTUGUESE

Organization A project leader

Background do participante

1. Fale um pouco da sua experiência profissional em projetos de software

a. Eu trabalho com o grupo XX e YY desde 2013. Comecei como estagiário tanto na parte de design e desenvolvendo em front-end e aí chegou um certo tempo que passei a ser líder técnico. E aí eu encabeçava alguns projetos, não totalmente, mas com a ajuda de outras pessoas, fui ganhando um pouco mais de liberdade até me tornar líder técnico das duas equipes.E aí hoje eu coordeno mais do que 2 projetos nas duas áreas. Hoje tem a área de desenvolvimento, que pode ter tanto a parte de back-end como de front-end, e a parte de design.

2. Quantos anos de experiência você possui em gerenciamento de projetos?

a. Acho que 2 anos.

3. Como você considera seu nível de conhecimento em gerenciamento de riscos?

a. Na verdade meu conhecimento nessa área é mais empírico, não tenho o conhecimento técnico, não parei em nenhum momento para estudar. O que acontecia é que um projeto ia se desenvolvendo e no olhômetro ia medindo o que era arriscado e o que não era.

4. O que você entende por risco no contexto de projetos de desenvolvimento de software?

a. Eu acho que existem alguns, não sei se o nome certo seria componentes, que para mim mesmo empiricamente fica mais visível, como por exemplo dependência de equipe, ou definição de um requisito de projeto e aí eu conseguia perceber mesmo que empiricamente que aquilo ali tinha um risco maior ou não. Dependendo do desenvolvimento do projeto, ia transformar num caos, alguns tinham um risco maior, outros tinham um risco menor. Na minha visão seria uma funcionalidade ou componente, talvez, que ele tem uma probabilidade de causar falhas no projeto ou dano bem maior, algo desse tipo.

Caracterização do indicador

1. Quanto de complexidade e esforço você entende que o uso do indicador causa às suas atividades de gerenciamento?

a. Inicialmente, logo no primeiro momento, de fato você tem vários pontos a tratar. Só que, de fato, o desgastante foi mais no primeiro momento onde eu tive que pegar a planilha e ter que ir mapeando. Acho que era mais de 50, bem mais...mais de 100...enfim, aí tive que passar um tempo relativamente alto, coisa de 1 hora analisando, Alguns eu senti certa dúvida, só que aí com o desenvolver das coletas acho que ficou mais fácil.

2. Qual é o entendimento que você tem do indicador? Você se sentiu confuso?

a. Como eu disse anteriormente, a minha ideia de risco de projeto é uma coisa muito empírica. E através das coletas e das entrevistas, acho que fica muito mais visível, analisando um

projeto e outro, como o Projeto A1 é muito arriscado em relação ao Projeto A2. Eu já sabia disso, só que é diferente de você ver da forma que está printado aqui na tela, através dessas coletas e todas as pesquisas.

3. Que pontos do indicador você considera positivos?

a. Acho que ficou mais visível, e dependendo do momento em cada coleta. Por exemplo, quando o Projeto A1 tava perto da entrega, eu acho que de alguma forma isso impactava mais. Eu como tava a frente, eu ficava com mais medo, logo toda pontuação que dava, eu dava um pouco mais alta. Então para mim era muito mais arriscado E aí dá para perceber claramente nos gráficos que ele vai diminuindo o risco de acordo com as entregas.

4. Que pontos do indicador você considera negativos?

a. Acho que para mim está muito claro. Acho que as coletas não tomaram tanto tempo assim. Não vejo nenhuma melhoria. para mim o gráfico está muito claro.

Avaliação da viabilidade do indicador

1. Você considera que a aplicação do indicador é viável no seu ambiente?

a. Sim, totalmente. Não sei se existe uma forma de fazer isso online, digital, ou ou coisa do tipo. Que a gente conseguisse amarrar a alguma ferramenta. Se tivesse seria uma maravilha. Porque de fato para mim e eu acredito que para um monte de, tanto o pessoal que foi desenvolvedor e acabou caindo num cargo de gerência de projetos, fica muito mais fácil perceber onde a coisa está pegando mesmo.

2. Em que aspectos você entende que os fatores caracterizadores de projeto contribuem para os níveis de riscos de um projeto?

a. Eu acho que ajuda.

3. Observando os fatores caracterizadores de projetos, eles fazem sentido para você? Você teria alguma sugestão de fator caracterizador que seria interessante incluir na análise?

a. Fazem total sentido.

4. Os fatores de riscos (e seus pesos) fazem sentido para você?

a. Acredito que sim.

5. Você notou algum risco importante que não esteve presente na lista utilizada?

a. Acho que a questão da hora (horário) do espaço, que foi levantado na penúltima coleta. Porque hoje a gente está dentro de um espaço que não é 100% nosso e como a gente está dentro da universidade e estão passando por um problema político e vamos trabalhar só até 17h. Se você juntar isso no final...

6. Você considera que o processo utilizado para levantar as informações para o indicador é coerente?

a. Acho que sim. Acho que é até importante que você consegue perceber, não ficar só na coleta, você tem um feedback, torna o processo mais prazeroso. você vê a coisa funcionando. A frequência (quinzenal) foi ótima.

7. Você consegue perceber por que o valor do indicador cresce ou decresce? O que foi determinante em cada projeto?

a. Acho que pelo momento do projeto, por exemplo, se a gente pegar o Projeto A2, que está na primeira versão e já está rodando, e a gente vai trabalhar na segunda versão. Então dá para perceber que ele está mais linear, não tem tanta curva, ele não desce e não sobe. Já o Projeto A1 tava exatamente num momento de entrega, e aí tinha justamente alguns pontos que faziam com que ele se tornasse mais arriscado ou menos. Por exemplo, no momento em que a gente tava chegando no final da entrega e tinha uma dependência de colaboradores, se saísse um deles era um risco muito alto. E aí ele ia subindo e descendo. para mim isso ficava muito visível.

Avaliação da eficácia do indicador

1. Comparando os rankings de projetos levantados pelo indicador proposto e pela média da exposição ao risco, você consegue enxergar alguma vantagem do indicador em relação a exposição média ao risco? Quais?

a. Agora eu consigo ter uma noção da diferença. Dá para perceber que o Projeto A1, baseado em todos aqueles pontos de coleta, é muito mais arriscado que o Projeto A2. Então eu vejo que o gráfico do PRI tem vantagem sim. Existe uma diferença muito grande que no gráfico do ERm não é representado. Eles aparentam ser muito parecidos, quando na realidade não são.

2. Você notou alguma diferença entre os rankings dos projetos mais arriscados?

3. O ranking apresentado após o cálculo do indicador faz sentido?

a. Faz total sentido.

4. Fale da sua percepção a respeito do uso do indicador em relação ao período em que você não o utilizou.

a. Acho que sim (houve diferença). Acho que o fato de você dá um certo intervalo, eu meio que me perdi na hora que eu voltei. Meio que eu me desliguei dos dados da avaliação que eu fiz. Acho que talvez, não sei se é interessante, diminuir. Porque tava tudo fresco na minha cabeça, e depois tive que relembrar. Teve uma pequena curva de aprendizagem de novo.

Caracterização da aplicabilidade do indicador no ambiente

1. Fale um pouco sobre o ambiente no qual os projetos estão inseridos. Dia-a-dia, pontos positivos, pontos negativos. Quais são suas maiores dificuldades?

a. A gente tem projetos que na verdade foram aprovados há um certo tempo atrás, e a gente trabalha principalmente com desenvolvimento de cursos que normalmente são ligados a saúde. E aí, para o desenvolver desse material a gente precisava de algumas ferramentas para conduzir outras situações. E aí, a equipe foi obrigada a...o grupo foi obrigado a contratar profissionais na área de desenvolvimento. Então há um tempo atrás a gente tinha um colaborador que só mexia na parte de projetos, e aí ele saiu e a gente está se virando com o que tem. A gente tem a nossa coordenação técnica, que é a equipe que trabalha tanto com a parte de

conteúdo, como a parte de gerenciar aluno, orientador, professor. E a gente a gente tem a parte de desenvolvimento. Hoje a gente tem em média 17 colaboradores, tanto CLTs como estagiários, e que são divididos em mais 3 vertentes: a parte de infraestrutura, desenvolvimento web (que é separada em mais duas: front e back-end), e design. De colaborador, a gente não tem do que reclamar. E aí algumas dificuldades que a gente encontra, pelo menos eu percebo, é a parte de testes. A parte de infraestrutura é muito tranquila. Há um tempo atrás a gente teve alguns problemas que conseguiu evoluir bastante. A parte de riscos, como eu tinha dito, foi uma parte que foi tudo empírico. Talvez a gente comece a adotar essa parte, ter uma atenção maior a parte de riscos. Outro problema é a parte de testes, que normalmente quem está desenvolvendo está testando.

2. Que métodos de desenvolvimento são utilizados no seu ambiente? (ágil, tradicional, híbrida, etc.)

a. Em algum momento foi caótico (inicialmente). Isso porque o grupo surgiu de um projeto que evoluiu de uma forma não organizada e quem tava na liderança teve que resolver. Depois de algum tempo houve a contratação de outros profissionais, foram rodados alguns cursos, e aí a gente começou a tentar puxar o Scrum. A gente tentou, não da forma tradicional, a gente não consegue. Então é meio que um tradicional e ágil.

3. Fale um pouco do uso de práticas de gerenciamento de riscos no seu ambiente. Existe algum modelo? Com qual frequência é utilizado? Se existe, que técnicas são utilizadas?

a. É tudo empírico.

4. A aplicação do indicador trouxe algum benefício? Quais?

a. Acho que só trouxe coisas boas, porque uma coisa é você ter essa noção empírica, outra coisa é deixar isso bem mais claro.

5. A aplicação do indicador trouxe algum malefício? Quais?

a. Se a gente conseguisse fazer isso em qualquer projeto de uma forma automatizada, acho que ficaria muito mais evidente onde aplicar mais energia ou não.

Avaliação da utilidade do indicador

1. Você considera o indicador útil? Em quais aspectos?

a. Para mim é totalmente útil. Uma questão é que de fato eu nunca utilizei algum. Se existe algum eu não sei. O que a gente fazia na verdade era em reuniões analisar, ver equipe, ver cronograma e tentar fazer uma média do risco e aplicar. A gente conseguia ver alguns, mapear, mas acredito que na cabeça de cada líder. Mas não é uma coisa tão técnica e tão direta assim.

2. De que formas você entende que o indicador pode agregar valor às suas atividades?

a. Eu acho que só em ficar bem claro, esse indicador ser analisado enquanto o projeto tivesse rodando e talvez com menos tempo, ia ficar muito mais fácil gerenciar o recurso, analisar o recurso. Como eu tinha dito, isso tudo é feito, mas é muito empiricamente, não é tão rico assim.

3. O processo em geral tomou mais tempo do que você imaginava? O tempo gasto nas atividades realizadas agregou valor?

a. Eu achava que no início seria mais custoso, mas depois de um tempo eu achei muito natural. E o tempo foi muito curto. A gente sentava aqui e gastava meia hora, acho que até menos. E o resultado, isso pensando num projeto muito pesado, que tem um alto custo, o que é meia hora a cada quinze dias?

Considerações finais

1. Em geral, qual é o impacto do indicador no seu ambiente?

a. O Projeto A1 é de fato um projeto que tem muito mais risco. O Projeto A2 é um plug-in que já está na primeira versão, já identificamos alguns problemas que não impedem ele de estar rodando. Já o Projeto A1 não, fora o peso místico que o projeto tem, que desde 3,4 anos que não sai é de fato um projeto que tem mais risco e isso ficou bem evidente. Na verdade o que todo mundo pensava, empiricamente, ficou evidente tecnicamente, usando os fatores fatores de riscos e pontos levantados.

2. Quais são os pontos mais críticos do indicador?

a. Acho que só talvez a forma de apresentar, que poderia mais simples. Não sei se essa forma é a melhor. Seria legal, por exemplo, eu não me lembro quais foram os pontos mais ou menos indicados em cada coleta, o que levou a subir, o que levou a descer. Por exemplo, por que esse ponto está menor que esse? O que mudou? O que variou? Por que na verdade cada ponto desse é um conjunto de outros pontos, de vários.

3. O que você considera do estudo realizado que foi ponto positivo?

a. Foi legal para, primeiramente pro meu conhecimento, para eu entender como funciona. Me fez repensar todos os pontos. Porque aquela lista é gigante, mas de fato faz você pensar em muitos riscos que você não leva em consideração. Normalmente você leva em consideração o tempo de projeto, tamanho de equipe, se algum membro da equipe vai sair ou não, e aí você vê aquele monte de coisas, aí você percebe que tem outros riscos.

4. Que pontos de melhoria você identificou durante todo o estudo?

a. A metodologia eu deixaria a mesma, eu acho que só a forma, se pudesse ser algo digital, dinâmico, automático ou outra coisa do tipo. Não sei se é possível mas acho que é interessante a conversa. No mais, nada a mudar.

Organization B project leader

Background do participante

1. Fale um pouco da sua experiência profissional em projetos de software

a. Eu atuo na área de gerenciamento de projetos de software há pelo menos...há alguns anos aqui e no antigo emprego por 4 anos e somando tudo tenho 11. Lá no antigo emprego atuei

mais na parte de requisitos e aqui mais voltado ao desenvolvimento de software, desde a parte de implementação até o final.

2. Qual função você atualmente desempenha na organização?

a. Hoje eu sou gerente de desenvolvimento de software, a minha gerência é responsável pela fase de implementação, testes, gerência de configuração e análise e projeto.

3. Quantos anos de experiência você possui em gerenciamento de projetos?

a. 4 anos

4. Como você considera seu nível de conhecimento em gerenciamento de riscos?

a. Entre baixo, médio e alto, eu ficaria de médio para alto. Quando eu atuei em gerenciamento de projetos, a gente fazia a gestão de riscos seguindo realmente as práticas do PMBOK.

5. O que você entende por risco no contexto de projetos de desenvolvimento de software?

a. considero qualquer evento possível de acontecer que possa interferir no projeto...eu posso interferir no escopo, no prazo, como na qualidade. Pode ser qualquer ameaça interna ou externa que possa vir a provocar alguma...

Caracterização do indicador

1. Quanto de complexidade e esforço você entende que o uso do indicador causa às suas atividades de gerenciamento?

a. Não teve esforço quase nenhum e nenhuma dificuldade. O painel gráfico é adequado e facilita bastante.

2. Qual é o entendimento que você tem do indicador? Você se sentiu confuso?

a. Diz claramente numa linha de tempo, comparando os 3 projetos aí expostos, quais os que estão com ranking com relação ao riscos, qual destes 3 projetos está com o maior risco ou não e isso numa linha de tempo, isso está bem claro.

3. Que pontos do indicador você considera positivos?

a. Eu diria “menos é mais”, eu acho muito importante isso. você consegue fazer uma leitura rápida, o que ajuda. O resultado em si ficou bem próximo do que a gente imaginaria se eu fosse fazer uma classificação individual. Como eu conheço bem os 3 projetos e acompanhei os 3, então eu tenho uma noção. De fato, o início foi bem...refletiu a realidade. Projeto B2 realmente estava no momento mais crítico, mais arriscado do que os outros 2. Então para mim, eu não teria nada a acrescentar do indicador em si, porque para mim ficou claro e o indicador. E no final deu essa mudança no ranking e teve realmente a ver com o que ocorreu aqui. Hoje em dia, Projeto B3 e Projeto B2 estão realmente sob controle e Projeto B1 sofreu uma alteração, mas dentro de um desvio padrão.

4. Que pontos do indicador você considera negativos? (por exemplo, em relação a esse gráfico apresentado, fica fácil perceber o que levou o valor a subir ou descer?)

a. Projeto B2 teve uma queda bem grande por conta da questão dos requisitos. Nós saímos

de uma obscuridade total em requisitos. Projeto B3 foi um dos projetos mais normais e isso reflete também no gráfico. Nós tínhamos requisitos já previamente mais ou menos estabelecidos. E Projeto B1 também iniciou de certa forma normal até acontecer essa peculiaridade que tem a ver com requisitos e com a interferência externa direta (mudanças na alta gestão). Então ele reflete bem a realidade dos projetos.

Avaliação da viabilidade do indicador

1. Você considera que a aplicação do indicador é viável no seu ambiente?

a. Considero viável, mas não a curto prazo só por uma razão do ferramental, porque como a gente não tem a cultura do gerenciamento do riscos (aqui não tem). Aí tem um passo a frente, um dever de casa a fazer para conseguir permitir a aplicação.

2. Em que aspectos você entende que os fatores caracterizadores de projeto contribuem para os níveis de riscos de um projeto? a. O que mais afeta nos riscos de projeto é a experiência da equipe no framework a ser utilizado e no negócio.

3. Observando os fatores caracterizadores de projetos, eles fazem sentido para você? Você teria alguma sugestão de fator caracterizador que seria interessante incluir na análise?

a. Acho que os fatores tem tudo a ver, em especial a experiência da equipe na aplicação e na tecnologia (nessa ordem). Experiência no processo tem uma importância menor aqui, pois os processos são bem definidos. Eu acho que o tamanho do projeto talvez pudesse ter ficado, porque realmente quando o projeto é muito grande, ele pode ter uma influência nos riscos, mas num projeto muito pequeno não. A gente não tem os custos, a gente faz a mensuração do tamanho, e acho que poderia ser utilizado. Outro ponto a olhar são as interferências externas, porque tivemos muitas, de priorização, o que afeta muito na velocidade porque você pode ter um forte patrocinador e isso tem influência. Aqui nem todo desenvolvimento vira projeto...esses 3 viraram.

4. Os fatores de riscos (e seus pesos) fazem sentido para você?

a. Fizeram sentido, porque incluiu desde a parte de processo, o que a gente sentiu muito para Projeto B2, na falta de uma gestão de projetos mais atuante e isso afeta. Acho que são coerentes.

5. Você notou algum risco importante que não esteve presente na lista utilizada?

a. Não.

6. Você considera que o processo utilizado para levantar as informações para o indicador é coerente?

a. Achei bem tranquilo e bem coerente. Também coincidiu com a esteira do desenvolvimento do projeto. Então conseguiu pegar o início, o meio e a conclusão. Porque talvez se não tivesse sido assim o resultado poderia ter sido incoerente e a própria coleta também.

7. Você consegue perceber por que o valor do indicador cresce ou decresce? O que foi determinante em cada projeto?

a. A percepção para mim fica bem claro para Projeto B2, mas nem para todos eu consigo ter essa clareza. Eu não consigo cobrir todos, mas alguns sim. Eu diria que meio a meio. Para alguns não parece ficar claro se alguma coisa específica que influenciasse, senão o curso natural do projeto, porque ao longo do tempo as coisas vão melhorando, então nem todos eu conseguiria, mas para alguns, como Projeto B2 está claro, a subida de Projeto B1 no final também. O Projeto B3 é que para mim não consigo perceber.

Avaliação da eficácia do indicador

1. Comparando os rankings de projetos levantados pelo indicador proposto e pela média da exposição ao risco, você consegue enxergar alguma vantagem do indicador em relação a exposição média ao risco? Quais?

a. Realmente eu vejo a diferença. No indicador PRI eu consigo identificar, não todos, mas eu consigo identificar em algumas situações a razão da descida ou da subida. Eu consigo distinguir o projeto em algumas situações. Nesse (ERm) eu não consigo, porque todos decrescem e ficam num nível mais ou menos próximo, não iria me fazer relembrar nenhum que ocorreu durante o projeto. Nesse daqui (PRI), quando você olha, como existe uma distanciação, isso te ajuda a olhar o porquê.

2. Você notou alguma diferença entre os rankings dos projetos mais arriscados?

3. O ranking apresentado após o cálculo do indicador faz sentido?

a. Faz todo o sentido.

4. Fale da sua percepção a respeito do uso do indicador em relação ao período em que você não o utilizou.

a. Não teve muito impacto porque pegou justamente o recesso, e além disso a mudança de gestão afeta também. Foi um período oportuno de parada, porque não andou muito.

Caracterização da aplicabilidade do indicador no ambiente

1. Fale um pouco sobre o ambiente no qual os projetos estão inseridos. Dia-a-dia, pontos positivos, pontos negativos. Quais são suas maiores dificuldades?

a. A maior dificuldade é realmente o distanciamento entre as equipes. Existe tanto fisicamente, quanto relacionamento (esse tem pouco), mas o distanciamento é um fator que interfere muito no dia-a-dia. E o distanciamento do desenvolvimento em geral com o patrocinador ou cliente. Além disso tem uma variável que é o relacionamento muito grande entre os sistemas. A gente tem mais de 100 sistemas, alguns legados. Isso é um dificultador.

2. Que métodos de desenvolvimento são utilizados no seu ambiente? (ágil, tradicional, híbrida, etc.)

a. É um híbrido, a gente usa para gerenciamento o modelo kanban. E as práticas ágeis a gente não tem formalizadas ainda, mas usamos o kanban. O normal é isso (ágil), mas como a gente tem projetos maiores onde a gente usa um modelo cascata.

3. Fale um pouco do uso de práticas de gerenciamento de riscos no seu ambiente. Existe algum modelo? Com qual frequência é utilizado? Se existe, que técnicas são utilizadas?

a. Existe, mas é muito tímido e eu acho que mais para início, mas efetivo não. Não existe uma técnica para o acompanhamento. Existe um levantamento de riscos do projeto, mas isso não é acompanhado. A gente faz pelo sentimento, mas não existe a técnica.

4. A aplicação do indicador trouxe algum benefício? Quais?

a. O que ele trouxe de positivo, eu poderia tentar me antever mais para resolver um problema. Por que com o indicador a gente percebe porque o valor sobe (esse não: Projeto B3), mas os demais eu poderia pegar um fator e tratar melhor esse fator específico. Nesse caso ele poderia trazer um benefício se fosse realmente utilizado de forma oficial. Teria que...isso aí é da maturidade, para mim vai ficar mais claro os quadrantes que tão afetando cada projeto. Aí sim seria cada vez mais efetivo.

5. A aplicação do indicador trouxe algum malefício? Quais?

a. Nenhum.

Avaliação da utilidade do indicador

1. Você considera o indicador útil? Em quais aspectos?

a. Eu consideraria ele útil numa visão de médio a longo prazo, porque para primeiro ter o indicador, eu teria que ter riscos. Tendo esse levantamento de riscos, aí a gente realmente...o indicador poderia ser útil até para prever os próximos passos desses mesmos sistemas, já que ele usa características do sistemas e dos pesos associados, aí ele poderia prever em um novo projeto, já que depois da primeira versão ele continua. Então em considerando todos os fatores aí sim ele seria útil no futuro e eu me antecipar a esses riscos não ocorridos. Então isso requer uma base muito grande de acompanhamento de riscos e características de projetos para chegar a esse nível de atividade. Não é algo fácil, é difícil, mas ele passaria a ser útil de fato, porque poderia identificar a razão da subida e da descida, ou da característica mesmo do projeto ou do risco. É bem difícil, mas para o início, se você for comparar o PRI com ERm eu vejo muito mais utilidade no PRI, por que ele mostra uma perspectiva de separar o joio do trigo.

2. De que formas você entende que o indicador pode agregar valor às suas atividades?

a. Ele me agregou porque a partir do momento que você para para pensar nos riscos ou monitorar aquele risco é o momento em que você vai tentar atacar aquele risco. Então essa questão da relação com o gerente de projeto do projeto Tema, isso me alertou para tratar com cuidado maior. Isso me ajudou nesse sentido de reforçar a mitigação de risco, de me antever. Então quando você me questiona de cada fator isso me faz refletir e isso me agregou a gestão dos projetos e eu tentei me antecipar a alguns dos problemas e me aproximei mais do gerente de projeto para suprir lacunas. E nesse sentido ele me ajudou.

3. O processo em geral tomou mais tempo do que você imaginava? O tempo gasto nas atividades realizadas agregou valor?

a. O esforço foi razoável. A coleta inicial foi muito longa, eu achei redundante, não sei se era proposital, mas eu achei repetitivo, sensação de repetição...mesma coisa (os fatores de riscos). Acho que poderia dar uma enxugada, mas tirando isso as todas as etapas foram tranquilas.

Considerações finais

1. Em geral, qual é o impacto do indicador no seu ambiente?

a. Em grande escala o impacto seria aumentado, com mais projetos, aí teria que rever para ficar mais enxuto ainda. Eu acho pesado para ser feito com mais projetos. A não ser que isso seja diluído para o pessoal do PMO, que aí cada um ficaria com 2 ou 3 projetos, o que seria mais factível.

2. Quais são os pontos mais críticos do indicador?

a. Além do mencionado, não.

3. O que você considera do estudo realizado que foi ponto positivo?

a. O envolvimento desde o início é fundamental. Achei o estudo bacana.

4. Que pontos de melhoria você identificou durante todo o estudo?

a. A primeira coleta e eu senti uma certa dificuldade na classificação da probabilidade e do impacto, porque o impacto normalmente é mais igual, mais normalizado (dentro do desenvolvimento de software). A probabilidade é mais fácil. Já o impacto eu ia mais ou menos na mesma média da probabilidade, se tirar ele ou ver outra forma de calcular ele ajudaria melhor.