Sistemas & Gestão 14 (2019), pp 188-196

SISTEMAS & GESTÃO
Revista Eletrônica

RISK MANAGEMENT IN SOFTWARE PROJECTS: AN APPROACH BASED ON NON-FUNCTIONAL REQUIREMENTS

Ana Cristina da Silva Andrade
ana_cristinas@yahoo.com.br
Institute of Technological Education - Ietec, Belo Horizonte, Minas Gerais, Brazil.

José Luis Braga
zeluisdpiufv@gmail.com
Institute of Technological Education - Ietec, Belo Horizonte, Minas Gerais, Brazil.

André Luiz de Castro Leal
andrecastr@gmail.com
Federal Rural University of Rio de Janeiro - UFRJ, Rio de Janeiro, Rio de Janeiro, Brazil.

Fernando Hadad Zaidan
fhzaidan@gmail.com
Institute of Technological Education - Ietec, Belo Horizonte, Minas Gerais, Brazil.

ABSTRACT

At all stages of the software development process there are risks and these can present an opportunity or threat to the project. The early practice of risk management in software projects makes it possible to know and control the factors that impact the project, thus contributing to its quality and success. This article aims to propose a conceptual model composed of the main risk factors in software development projects that allows project managers to evaluate and monitor risks. In order to achieve results that meet the objectives of this work, activities were carried out in an interactive manner according to a previously developed mental map. Considering risk as a non-functional requirement, risk management models were proposed through the NFR (non-functional requirements) Framework and i* Framework. By way of example, it can be concluded that projects that deal in the right time with risk operations or part of them may have a greater chance of success.

Keywords: Software Project Management; Risk management; Non-Functional Requirements; Flexible Goals.

PROPPI / LATEC
DOI: 10.20985/1980-5160.2019.v14n2.1526

Revista Eletrônica Sistemas & Gestão
Volume 14, Número 2, 2019, pp. 188-196
DOI: 10.20985/1980-5160.2019.v14n2.1526

1. INTRODUCTION

In recent decades, flaws in software development projects have always been a matter of concern for software engineering. In the CHAOS Report of 2015 (Hastie; Wojewoda, 2015), published by Standish Group, the following figures were presented:

- 29% of the projects were successful (completed on time, within budget and with agreed scope).
- 52% of the projects were not executed as agreed (delay in delivery, budget overflow or reduction of scope).
- 19% of projects failed (canceled or unused).

These percentages, when expressed in monetary amounts, represent a significant amount for organizations and, in a software development organization, it is a corporate risk that can mean its survival.

As a result, software organizations seek new strategies to achieve project success and risk management has been adopted in a way that minimizes the emergence of impediments that lead to declining productivity and quality of the software generated (Silva, 2013). A software development project needs to meet the goals (quality, performance, environment and others) that are usually modeled as non-functional requirements (RNF).

RNFs are those that are not related to the specific services offered by the software (what the software does), but rather to the properties of the software, such as reliability and response time (as the software does) (Sommerville, 2011).

Based on Chung et al. (2000), Leite (2009), Supakkul et al. (2010) and Cappelli et al. (2010), who frame transparency as a quality requirement (not functional), in this work, the risk will be considered an RNF, or a softgoal, using the terminology of intentional modeling, since this is a subjective factor, dependent on the field of application and difficult to assess by stakeholders.

Using the NFR (non-functional requirements) Framework, it is possible to visualize the development of risk subjectivity in software development and, through the i* model, actions and responsibilities for risk mitigation will be operationalized.

This work considers risk as RNF and defines its concrete operations in order to minimize these risks. The aim is to obtain a model, composed of the main risk-related variables in software development that allows software engineers and project managers to consider including risk treatment in projects earlier in the development process, acting in a preventive manner and increasing the chances of success of the project.

Initially this article presents concepts of risk management, nonfunctional requirements and intentional models (NFR Framework and Framework i*). Next, the main risk factors identified in the software development process are presented and the models elaborated using such factors. Finally, the final considerations and opportunities for future work are presented.

2. LITERATURE REVIEW

2.1 Risk management

According to Project Management Body of Knowledge (PMBOK, 2017), risk is an event or uncertain condition that, if it occurs, will have a positive (opportunity) or negative (threat) effect on at least one project objective involving time, cost, scope or quality. Macedo and Salgado (2015), based on Charette (2005), define risk as an event or state that can cause damage, loss or delay in a software project. Risk management is fundamental for project management, being one of the ten areas of knowledge of the PMBOK and also handled by quality assessment models of software processes such as ISO/IEC15504 and MPS.BR.

Project risk management, according to the PMBOK (2017), is composed of the processes illustrated below to increase the likelihood and impact of positive events and decrease the likelihood and impact of adverse events to the project.

Revista Eletrônica Sistemas & Gestão
Volume 14, Número 2, 2019, pp. 188-196
DOI: 10.20985/1980-5160.2019.v14n2.1526

Figure 1. Overview of project risk management processes
Source: Adapted from Project Management Institute (2017).

2.2 Non-functional requirements

In software engineering, requirements are defined as the descriptions of what the system should do, the services it offers and the constraints on its operation, reflecting the needs of customers (Sommerville, 2011).

Software requirements are classified in:

- Functional requirements: Describe "what" the system should do, how the system should react to specific inputs, and how the system should behave in certain situations;

- RNFs: fix restrictions on "how" the functional requirements will be implemented, that is, restrict "how" the system performs the "what", and includes constraints on cost, performance, portability, robustness, and others.

RNF implementation can spread throughout the software. These requirements define global constraints of the software, the development process and the deployment process, and are considered global in that they arise from all parts of the system and their interactions (Xavier et al., 2009), and can affect whole system architecture and not just individual components.

RNFs are critical in terms of software development. In software design, if a given system functional requirement is not implemented, users may find a way around its absence. However, if an RNF is not met, it may compromise the functioning of the entire system.

2.3 NFR framework

The NFR Framework was proposed by Chung et al. (2000), focusing on the modeling of RNF and its operations, through the construction of a Softgoal Interdependency Graph (SIG), which describes the dependencies between softgoals (flexible targets) and how they are decomposed (Serrano, 2011). Flexible goal, synonymous with softgoal, are qualities (safety, performance, reliability, and others) desired by the actors that do not have clear criteria for their satisfaction, that is, they are subjective and dependent on the points of view of stakeholders (Oliveira et al., 2007).

In this framework the RNFs are treated as flexible targets (softgoals), which will be identified and refined, represented by a graphic structure inspired by the And / Or trees (Xavier et al., 2009). A softgoal is refined to the point where the operations are achieved, thus generating functional requirements in function of the need to detail the RNF.

For Chung et al. (2000), the goals are related to the intentionality of the actors, while requirements (functional and nonfunctional) are characteristics implemented by software functions.

By constructing the dependency graph, it is possible to evaluate the goals and determine if a particular nonfunctional requirement is being achieved in a specific project. However, according to Xavier et al. (2009), the goals represent RNF and these can rarely be considered totally "satisfied".

2.4 Framework i*

The i* model is intentional and aims to describe processes that involve several actors, reflecting the motivations and interests of these actors, as well as the relationship between them. Modeling is based on actors, goals, beliefs, skills, and commitments, and represents mutual dependence on goals, tasks, and resources. Unlike the other modeling techniques, it expresses the reason for certain action or decision making (Yu, 1995).

The i* (i-star) Framework, proposed in 1995 by Eric Yu, is a conceptual modeling technique for describing processes involving multiple actors (Serrano, 2011). This technique concentrates on the relationship between actors and their dependencies, focusing on the reasons or motivations that are associated with the behaviors (the why).

Revista Eletrônica Sistemas & Gestão
Volume 14, Número 2, 2019, pp. 188-196
DOI: 10.20985/1980-5160.2019.v14n2.1526

In the Framework i* actors depend on each other to achieve their goals, perform tasks, and provide resources. Through cooperation an actor can achieve goals that would be difficult if he were alone.

The i* has graphical representation in the form of a network of relationships, and is formed by two basic models: Model SD (strategic dependency), which describes relations of dependence between the actors, and the SR model (strategic reason), which explains how the actors achieve their goals.

3. RISK MANAGEMENT BASED ON NON-FUNCTIONAL REQUIREMENTS

The existing literature on risk management in software development projects indicates that one of the major reasons for failures in this type of project is the inadequate or even nonexistent assessment of risk factors.

Software risk management supporters say that actions to reduce the chances of a project failure can be made from identifying and analyzing the threats to project success throughout the entire development process cycle (Schmidt, 2001).

In order to evaluate the risks of a project, it is necessary to identify what these risks are, and to know those who deserve more attention from the project manager. However, project managers find it difficult to identify the most common risks in a software project.

Given this scenario, the first step of this work was to identify, through the literature, the main risk variables that impact the software development process. Among the options found in the available bibliography, Schmidt (2001) presents an extensive list of risk factors in software projects.

For this work, the list published by Schmidt (2001) was then compared with publications by Lopes (2014) and Barki (1993), which also present risk factors in software projects. Based on this comparison, the factors related to Planning and Communication were added to the Schmidt list, thus defining the set of risk factors for software development projects, presented in the column "Risk factors list" in table 1, which served as base for the study presented.

Identified the main risk factors of a software project, the next step was the creation of a risk management model through the NFR Framework. Note that risk factors were appropriately renamed to be treated as softgoals, as presented in the "Softgoals" column of Chart 1, and thereafter the softgoal risk was refined.

Chart 1. Risk factors of software development project x softgoals

List of risk factors	Softgoals
1. Corporate environment	**Corporate environment**
Change in business and organizational environment	Corporate Environment Volatility
Incompatibility between business culture and new processes	Incompatibility between business culture and new processes
Lack of business value and support	Lack of business value and support
Unstable corporate environment (competitive pressures radically change user requirements)	Corporate Environment Instability
Property and/or top management changes	Modifiability of ownership and/or top management
Non-existent Strategic Alignment	Non-existent Strategic Alignment
2. Sponsorship / property	**Property**
Lack of high management commitment	Absence of high management commitment
Lack of project acceptance	Absence of project acceptance
Lack of user commitment	Absence of user commitment
Conflict between departments	Incompatibility between departments
Lack of approval from all parties	Absence of approval from all parties
3. Relationship management	**Relatability**
Failure to manage user expectations	Lack of management of user expectations
Inappropriate user involvement	Inappropriate user involvement
Lack of user cooperation	Lack of user cooperation
Failure to identify / involve all stakeholders	Failure to identify / involve all stakeholders
Increased user expectations	Increased user expectations
Managing multiple relationships with stakeholders	Manageability of multiple stakeholder relationships
Lack of adequate user experience for key users	Inexperience of key users
4. Project management	**Manageability**
Not managing or managing changes improperly	Lack of management / inadequate change management
Lack of skill / power to manage project	Absence of ability / power to manage project
Non-existent / inadequate methodology	Non-existent / inadequate methodology
Inadequate definition of roles and responsibilities	Inefficient definition of roles and responsibilities
Poor or non-existent control	Non-existent / inadequate control

Revista Eletrônica Sistemas & Gestão
Volume 14, Número 2, 2019, pp. 188-196
DOI: 10.20985/1980-5160.2019.v14n2.1526

Non-existent / inadequate risk management	Non-existent / inadequate risk management
Choice of wrong development strategy	Absence of assertiveness in the choice of development strategy
5. Scope	**Scope**
Misunderstood and/or poorly defined objectives / scope	Inefficient definition / understanding of scope and objectives
Changes of scope / objectives	Modifiability of scope / objectives
Poor definition or incomplete definition	Inefficient / Incomplete definition
Technological Focus Only / Ignore Business Requirements	Exclusively technological focus
Many lines of communication	Variability of communication lines
6. Requirements	**Requirements**
Lack of Frozen Requirements (Changes)	Instability
Poor definition / understanding	Inefficient definition / understanding
Lack of domain / subject knowledge	Absence of domain / subject knowledge
7. Financing	Cost
Poorly estimated development cost	Poorly estimated development cost
Lack of budget for maintenance cost	Lack of budget for maintenance cost
8. Chronogram (scheduling)	**Development time**
Estimated time frame	Poorly estimated development time
Priority lower than other projects	Priority lower than other projects
9. Development Process	**Methodology**
Non-existent / inadequate methodology	Non-existent / inadequate methodology
New methodology / technology	Immaturity of methodology / technology
10. Personnel	**People**
Lack of knowledge / expertise	Lack of knowledge / expertise
Lack of competence / ability to manage	Absence of competence / ability to manage
Bad team relationship	Low team affinity
11. Staffing	**Staffing**
Insufficient / inappropriate staff involved	Insufficient / inappropriate staff involved
Rotativity of persons	Rotativity of persons
Excessive use of third parties	High number of third parties
Lack of knowledge / competence and availability of those involved	Lack of knowledge / competence of those involved
12. Technology	**Technology**
New technologies	New technologies
Instability of technical architecture	Instability of technical architecture
13. External dependencies	**External dependencies**
External dependencies not met	External dependencies not met
Multiple Suppliers	Multiple Suppliers
Lack of control over third parties / suppliers	Lack of control over third parties / suppliers
14. Planning	**Planning**
Non-existent / inadequate planning	Inexistent / Inadequate
15. Communication	**Communication**
Non-existent / inadequate communication	Inexistent / Inadequate

Source: Prepared by the authors.

Note that the refinements performed according to Table 1 will compose the software risk catalog used in this work.

With the elaborated risk GIS, it is possible to understand that, when managing risks in a corporate environment, ownership, relationality, manageability, scope, cost, development time, methodology, people, project resources, technology, external dependencies, and planning and communication, project risks will be managed. In this case, there is a positive contribution between dependencies and, if all dependencies are met, then the root will also be.

Risk SIG allows you to visualize softgoals, or flexible goals, for the domain you are trying to manage, the first step being for software risk management. In addition to showing the consequences of risks, it also presents the interrelationship between various softgoals, as well as between operations, and the negative and positive impacts between them.

The tree shown above can be used by project managers as a framework at the time of risk identification of a software development project. Through its applicability, it is possible to verify if the most common risk factors in software projects are being managed and also to generate a complete and detailed Risk Breakdown Structure (RBS), since the tree includes technical, organizational, management and external factors.

RBS is a risk management tool to be developed according to each project. According to Hillson et al. (2006), the RBS can be defined as a grouping that organizes and defines the risks of the project, and makes possible the understanding of the risks assumed by the project.

As the risk SIG presented in Figure 2 does not detail the operations of all the targets, Figure 3 presents the SIG risk cut with the insertion of the operations that will be treated in this study for the case of scope management. For example, following the graph, the operationalization for exclusi-

Revista Eletrônica Sistemas & Gestão
Volume 14, Número 2, 2019, pp. 188-196
DOI: 10.20985/1980-5160.2019.v14n2.1526

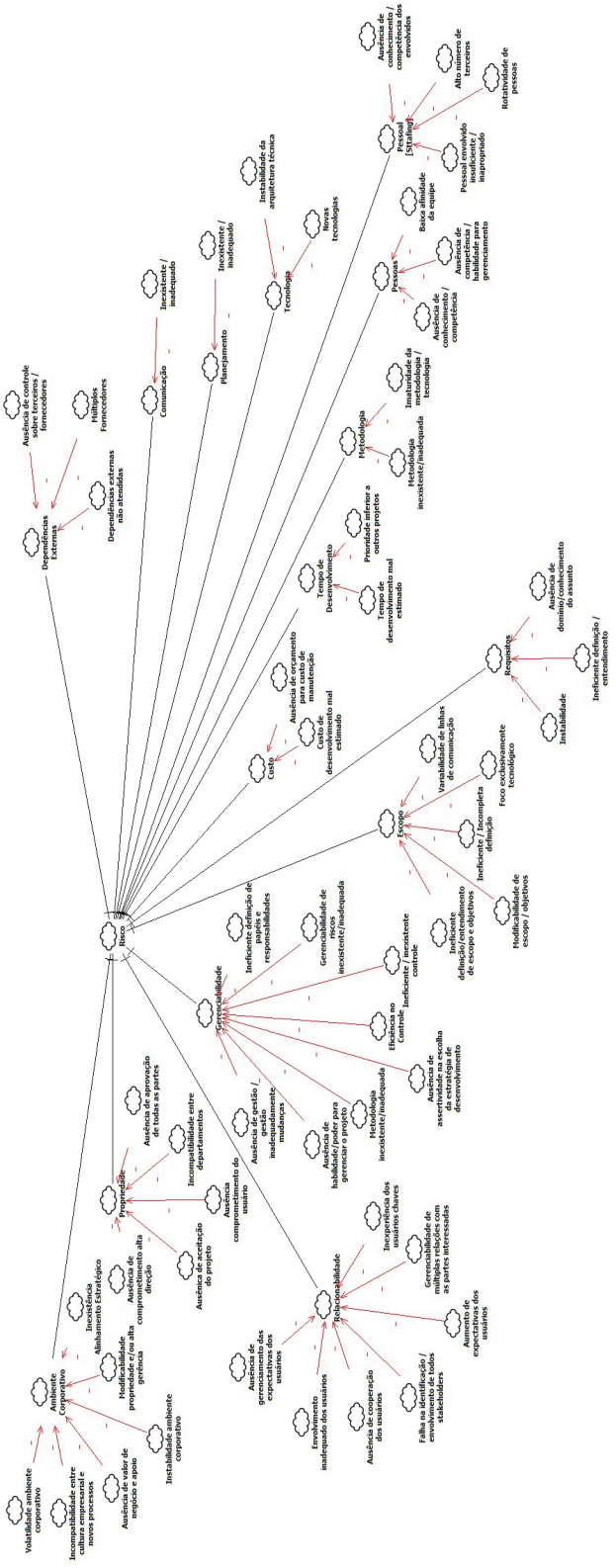

Figure 2. SIG risk management in software projects.
Source: Prepared by the authors (2018).

Legend:

Center figure: Risk

Line 1 – Top left *(anticlockwise)*: Corporate environment *(Center)*; Volatility of the corporate environment *(Top left)*; Incompatibility between business culture and new processes *(Far left)*; Lack of business value and support *(Lower left)*; Instability in the corporate environment *(Lower center)*; Modifiability of ownership and/or top management *(Lower right)*; Lack of strategic alignment *(Upper right)*.

Line 2 – Left: Property *(Center)*; Absence of senior management commitment *(Top left)*; Absence of project acceptance *(Lower left)*; Absence of user commitment *(Lower center)*; Incompatibility between departments *(Lower right)*; Absence of approval from all parties *(Upper right)*.

Line 3 – Left: Relatability *(Center)*; Lack of management of user expectations *(Top left)*; inappropriate user involvement *(Upper left)*; Lack of user cooperation *(Mid left)*; Failure to identify / involve all stakeholders *(Lower left)*; Increased user expectations *(Lower center)*; Manageability of multiple stakeholder relationships *(Lower right)*; Inexperience of key users *(Upper right)*.

Line 4 – Left: Manageability *(Center)*; Lack of management / inadequate change management *(Upper left)*; Absence of ability / power to manage project *(Mid left)*; Non-existent / inadequate methodology *(Lower left)*; Absence of assertiveness in the choice of development strategy *(Lowest left)*; Efficiency in control *(Lower center)*; Inefficient / non-existent control *(Lower right)*; Manageability of nonexistent / inadequate risks *(Mid left)*; Inefficient definition of roles and responsibilities *(Upper right)*.

Line 5 – Right: Scope *(Center)*; Inefficient definition / understanding of scopes and objectives *(Upper left)*; Modifiability of scopes / objectives *(Lower left)*; Inefficient / incomplete definition *(Lower center)*; Exclusively technological focus *(Lower right)*; Variability of communication lines *(Upper right)*.

Line 6 – Right: Requirements *(Center)*; Instability *(Lower left)*; Inefficient definition / understanding *(Lower center)*; Absence of domain / subject knowledge *(Lower right)*.

Line 7 – Right: Cost *(Center)*; Poorly estimated development cost *(Lower right)*; Lack of budget for maintenance cost *(Upper right)*.

Line 8 – Right: Development time *(Center)*; Poorly estimated development time *(Lower left)*; Property inferior to other projects *(Lower right)*.

Line 9 – Right: Methodology *(Center)*; Non-existent / inadequate methodology *(Lower left)*; Immaturity of methodology / technology *(Lower right)*.

Line 10 – Right: People *(Center)*; Lack of knowledge / expertise *(Lower left)*; Absence of competence / ability to manage *(Lower center)*; low team affinity *(Lower right)*.

Line 11 – Right: Personnel / Staffing *(Center)*; Staff involved insufficient / inappropriate *(Lower left)*; turnover of persons *(Lower right)*; High number of third parties *(Mid left)*; Lack of knowledge / competence of those involved *(Upper right)*.

Line 12 – Right: Technology *(Center)*; New technologies *(Lower right)*; Instability of technical architecture *(Upper right)*.

Line 13 – Right: Planning *(Center)*; existing / inadequate *(Right)*.

Line 14 – Right: Communication *(Center)*; existing / inadequate *(Right)*.

Line 15 – Right: External dependencies *(Center)*; External dependencies not met *(Lower right)*; Multiple Suppliers *(Mid right)*; Lack of control over third parties / suppliers *(Upper right)*.

Revista Eletrônica Sistemas & Gestão
Volume 14, Número 2, 2019, pp. 188-196
DOI: 10.20985/1980-5160.2019.v14n2.1526

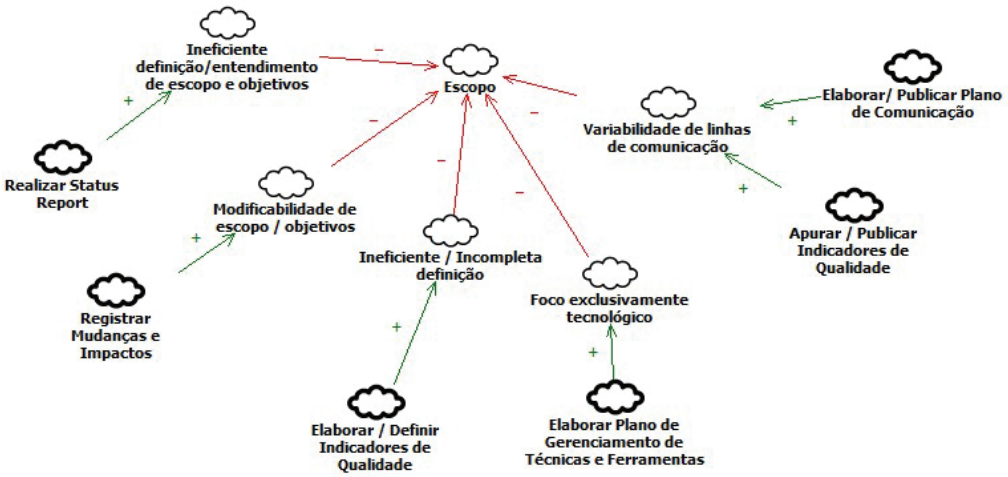

Figure 3. Risk SIG: operations of the objectives of scope management

Source: Prepared by the authors (2018).

Legend:

Center figure: Scope

Line 1 *(anticlockwise)*: Inefficient definition / understanding of scope and objectives *(Upper left)*; Perform Status Report *(Far upper left)*.

Line 2: Modifiability of scope / objectives *(Mid left)*; Record changes and impacts *(Far mid left)*.

Line 3: Inefficient / incomplete definition *(Lower left)*; Elaborate / define quality indicators *(Far lower left)*.

Line 4: Exclusively technological focus *(Lower right)*; Elaborate management plan of techniques and tools *(Far lower right)*.

Line 5: Variability of production lines *(Upper right - Center)*; Determine / publish quality indicators *(Far lower right)*; Elaborate / publish communication plan *(Far upper right)*.

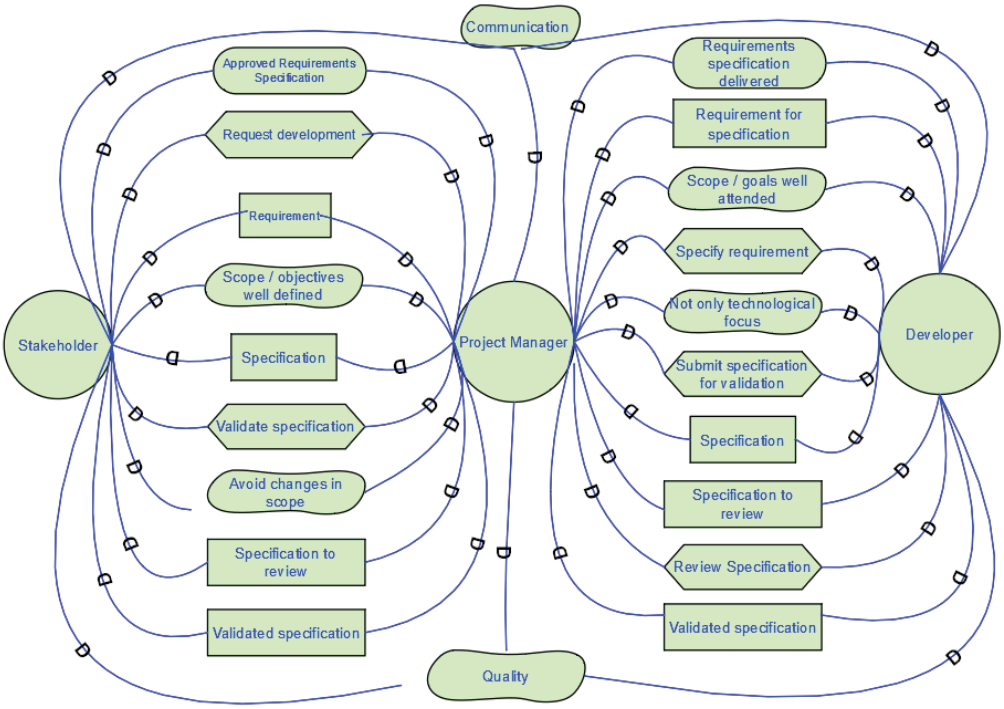

Figure 4. SD Model of Scope Management

Source: Prepared by the authors (2018).

Legend: (Figure 5)

Title: Proposed organizations in the Risk SIG and Scope Management SD dependencies explicitly inserted in the business process.

Column 1 – Left: Scope Management

Column 2 – Left: Developer; Stakeholder; Project Manager

Line 1 A *(Project Manager)*: Develop scope management plan; Develop requirements management plan; Develop techniques and tools management plan; Define / publish quality indicators; Activities of the process of planning management according to PMBOK; Activity of creating EAP according to PMBOK; Activity of controlling scope according to PMBOK.

Line 1 B *(Project Manager)*: Develop stakeholder management plan; Elaborate / publish communication plan; Perform status report - check objectives / service / techniques and tools to be used / measure indicators; Develop EAP; Control the scope; Record changes and impacts; Perform status report; Determine / publish quality indicators.

Line 2 *(Stakeholder)*: Request new product; Activities of the process of collecting requirements according to PMBOK; Eliciting requirements; Yes; Validate requirements; Yes *(top)*; Requirement validated? *(Mid)*; No *(Lower)*.

Line 3 *(Project Manager)*: Collect requirements; Conduct feasibility analysis; Requirement feasible?; No (Lower); Analyze requirements; Specify requirements; Review requirements; Activities of the process of validating the scope according to PMBOK.

Revista Eletrônica Sistemas & Gestão
Volume 14, Número 2, 2019, pp. 188-196
DOI: 10.20985/1980-5160.2019.v14n2.1526

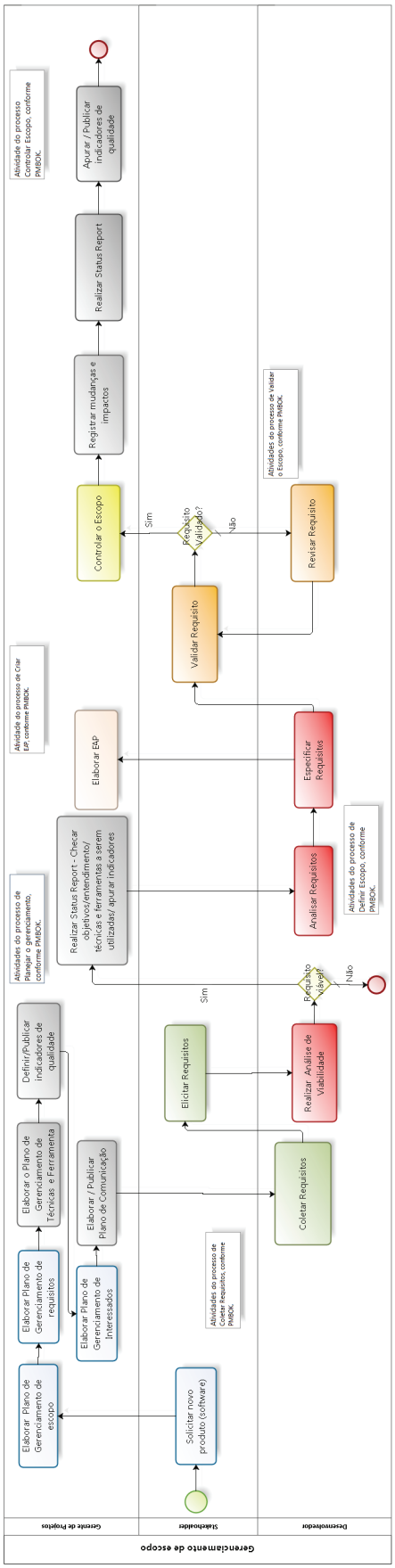

Figure 5. Business process model of scope management with insertion of risk elements Operacionalizações propostas no SIG de Riscos e as dependências do SD de Gerenciamento de Escopo inseridas de forma explícita no processo de negócio.
Source: Prepared by the authors (2018).

vely technological focus (that influences scope) aims to elaborate a Management Plan of Techniques and Tools.

The following is an example of agent interaction applied on the scope management domain, in which process actors were identified and the SD model presented in Figure 4 was developed.

When analyzing the SD model, note that flexible goals and targets are interconnected through dependencies, correlations, and contributions. It is possible to visualize the process actors, their goals (goals), and the flexible goals.

Figure 5 presents an example of applicability of the elaborated models. For this, the process model was developed based on the notation Business Process Modeling Notation (BPMN), referring to the management of the scope of a project. In it, the RNFs were explicitly inserted in the business process, using the proposed SIG risk operations and SD dependencies of scope management.

Note that it was possible to achieve a process model with a greater level of detail, since risk management activities, which until then were part of the tacit knowledge of those involved in the process, were inserted explicitly in the process without affecting the efficiency of the original process.

4. FINAL CONSIDERATIONS

The main contribution of this work was the conceptual (qualitative) risk model in a project, grouped in a single graph structure, to facilitate its understanding and practical application in projects. It was possible to gather the main contributions of all the works cited in a single structure that, over time, could be considered a conceptual framework to help designers and project managers to better see the risk situations in each project, and to treat them properly.

The identified variables were grouped in the risk catalog, used to create the risk SIG, through the NFR Framework. This catalog is dynamic and represents the first step towards the elaboration of a more complete catalog. Risk SIG allowed showing a way to validate the risk requirements through the network analysis of flexible targets, using the risk catalog.

It is concluded that the models and examples presented can contribute to the project managers to identify and manage the risks of the project in the initial phase, which will generate a warning of the possible problems, enabling a preventive action and contributing positively to the quality and success of the software product.

Revista Eletrônica Sistemas & Gestão
Volume 14, Número 2, 2019, pp. 188-196
DOI: 10.20985/1980-5160.2019.v14n2.1526

There is still much work to be done regarding the risk management approach using non-functional requirements. As a future work, it is suggested to apply the model to the other PMBOK project management areas, since the example presented is only related to the scope management. And a little more challenging would be the application of the variables and models hitherto presented for the creation of a system of intentional agents.

REFERENCES

Barki, H. et al. (1993). Toward an assessment of software development risk. Journal of management information systems, Vol. 10, No. 2, pp. 203-225.

Chung, L. et al. (2000). Non-functional requirements in software engineering. Springer Science & Business Media, New York.

Hastie, S.; Wojewoda, S. (2015). Standish Group 2015 Chaos Report Q&A with Jennifer Lynch. Disponível em: https://www.infoq.com/articles/standish-chaos-2015. Acesso em 22 abr. 2019.

Hillson, D. et al. (2006). Managing project risks using a cross risk breakdown matrix. Risk Management, Vol. 8, pp. 61-76.

Leite, J. C. S. D. P. (2009). Software Transparency. In Dagstuhl Seminar Proceedings. Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

Lopes, J. D. S. (2014). Um modelo para apoio à decisão em avaliação de riscos em projetos de software utilizando simulação com dinâmica de sistemas. Dissertação de Mestrado, Universidade Federal Viçosa, Viçosa, MG, Brasil.

Macedo, M. H. B.; Salgado, E. G. (2015). Gerenciamento de risco aplicado ao desenvolvimento de software. Sistemas & Gestão, Vol. 10, No. 1, pp. 158-170.

Oliveira, A. et al. (2007). Engenharia de requisitos intencional: tornando o software mais transparente. XXI Simpósio Brasileiro de Engenharia de Software, 15-19 out. 2007, João Pessoa, PB.

PMBOK, G. (2017). Um guia do conjunto de conhecimentos em gerenciamento de projetos. In Project Management Institute.

Schmidt, R. et al. (2001). Identifying software project risks: an international delphi study. Journal of Management Information Systems, Vol. 17, No. 4, pp. 5-36.

Serrano, M. (2011). Desenvolvimento Intencional de Software Transparente Baseado em Argumentação. Tese de Doutorado, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, RJ.

Silva, F. L. A. D. (2017). Análise do Impacto do Gerenciamento de Riscos no Sucesso de Projetos: Um Estudo de Caso em uma Organização de Desenvolvimento de Software. Dissertação, Universidade Federal de Pernambuco, Recife, PE.

Sommerville, I. (2011). Engenharia de Software. 9 ed. Pearson Prentice Hall, São Paulo.

Supakkul, Sam et al. An NFR pattern approach to dealing with NFRs. In: 2010 18th IEEE International Requirements Engineering Conference. IEEE, 2010. pp. 179-188.

Xavier, L. et al. (2009). Integração de Requisitos não Funcionais a Processos de Negócios: Integrando BPMN e NFR. In: Anais do WER10 - Workshop em Engenharia de Requisitos, Cuenca, Ecuador, April 12-13, 2010.

Yu, E. (1995). Modelling strategic relationships for process reengineering. Tese, Universidade de Toronto, Toronto, Canadá.

Cappelli, C. et al. (2010, March). Transparency versus security: early analysis of antagonistic requirements. In Proceedings of the 2010 ACM symposium on applied computing, pp. 298-305.

Received: April 14, 2019

Approved: May 09, 2019

DOI: 10.20985/1980-5160.2019.v14n2.1526

How to cite: Andrade, A. C. S.; Braga, J. L.; Leal, A. L. C. et al. (2019), "Risk management in software projects: an approach based on non-functional requirements", Sistemas & Gestão, Vol. 14, No. 2, pp. 188-196, available from: http://www.revistasg.uff.br/index.php/sg/article/view/1526 (access day month abbreviated year).