Universidade de Pernambuco
Escola Politécnica de Pernambuco
Programa de Pós-Graduação em Engenharia da Computação

Carlos Henrique Maciel Sobral Timóteo

Uma Nova Proposta para a Análise Quantitativa de Riscos no Gerenciamento de Projetos de Software

Dissertação de Mestrado

Recife, Março 2014

UNIVERSIDADE
DE PERNAMBUCO

Universidade de Pernambuco
Escola Politécnica de Pernambuco
Programa de Pós-Graduação em Engenharia da Computação

Carlos Henrique Maciel Sobral Timóteo

Uma Nova Proposta para a Análise Quantitativa de Riscos no Gerenciamento de Projetos de Software

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Engenharia da Computação da Universidade de Pernambuco como requisito parcial para obtenção do título de Mestre em Engenharia da Computação.

Prof. Dr. Sérgio Murilo Maciel Fernandes
Orientador

Prof. Dr. Mêuser Jorge Valença
Coorientador

Recife, Março de 2014

REVIWER NOTES.
Recife, Mar 2014

Sumário

Capítulo 1

Introduction

How risky are software projects? Several studies about effectiveness of software cost, scope, schedule estimation techniques; surveys from software professionals in industry; and analysis of project portfolios have been done to answer this question [1]. However, there is not a consensus.

Every project involves risk. There is always at least some level of uncertainty in a project's outcome, regardless of what the Gantt chart on the wall seems to imply. High-tech projects are particularly risky, for a number of reasons. First, technical projects are high varied. These projects have unique aspects and objectives that significantly differ from previous work, and the environment for technical projects evolve quickly. In addition, technical projects are frequently "lean", challenged to work with inadequate funding, staff and equipment. To make matters worse, there is a pervasive expectation that however fast the last project may have been, the next one should be even quicker [2].

Projects that succeed generally do so because their leaders do two things well. First, they recognize that a few of the work on any project, even a high-tech project, is not new. For this work, the notes, records, and lessons learned on earlier projects can be a road map for identifying, and in many cases avoiding, many potential problems. Second, they plan project work thoroughly, especially the portions that require innovation, to understand the challenges ahead and to anticipate many of the risks [2].

Some benefits of good risk management of software projects are the reduction of costs associated with changes in software; the development of a response plan to unexpected events, i.e., a contingency risk plan, the prediction of likelihood of undesired events, tracking the baselines of cost, schedule and quality. Such factors may determine the success of projects [3] [4].

1.1 Motivation

In 2009, CHAOS Report [5] showed that 32% of projects achieved success - were delivered on time, on budget and with the promised requirements -; 44% of the projects were challenged; not least, 24% of projects failed and were canceled. That is due to the risks involved in project activities and to a absent or defective software risk management

[6].

Schmidt et al. [7] have noticed that many software development projects end in failure. They showed that around 25% of all software projects are canceled outright and as many as 80% of all software projects run over their budget, exceeding it by 50% in average. Paul Bannerman [8] states that industry surveys suggest that only a quarter of software projects succeed outright, and billions of dollars are lost annually through project failures or projects that do not deliver promised benefits. Moreover, the author shows evidences that it's a global issue, impacting private and public sector organizations [9].

How to predict possible events in short, medium and long term? By analyzing risks and uncertainties, project managers often rely on intuition rather than logic and analysis. However, intuitive thinking is often subject to delusions, causing predictable mental errors and eventually poor decisions. A way to balance the effect of these psychological illusions is a systematic risk analysis and efforts to mitigate them through analytical methods.

It is difficult to manage something that can not be quantified: project managers should quantify the probability of risk, the impact, and their cumulative effect on a project. Furthermore, it is important to evaluate the various mitigation options: the cost for each option and the required time to perform the mitigation [10]. Interpreting the concept of risk is a hard task. Especially regarding the application of this knowledge in the development and use of efficient procedures for risk analysis in software project management techniques. Managing risk and uncertainty in software projects is critical to project management discipline. However, in times of economic crisis becomes much more difficult to perform risk management, due to the costs incurred.

Since it is an area of research that is growing, new and better methodologies to identify, measure and control risk items of software need to be developed. Keshlaf and Riddle [11] conclude that even if there are many approaches there is still a large gap regarding what is practiced by software industries. Moreover, it is worth noticing that there is no unique solution to lead risk management appropriately [10].

1.2 Problem Description

Recent perceptions about risk management and the inherent challenges posed by the nature of software projects contributes to the lack of project stability from majority of software project organizations [12]. Kwak and Ibbs [13] identified risk management as the least practiced discipline among different project management knowledge areas. The authors mention that, probably, a cause for it is that software developers and project managers perceive managing uncertainty processes and activities as extra work and expense.

Barry Boehm [14] defined risk as the possibility of loss or injury. That definition can be expressed by risk exposure formula. Even Boehm cites risk exposure as the most effective technique for risk priorization after risk analysis, Paul Bannerman [8] considers this definition limited and unsuitable. In classical decision theory, risk was viewed as reflecting variation in the probability distribution of possible outcomes, whether ne-

gative or positive, associated with a particular decision. This study takes into account Project Management Institute [4] definition whereupon project risk is a certain event or condition that, if it occurs, has a positive or negative effect on one or more project objectives. A complementary definition proposed by Haimes [15] is also considered, which express risk as a measure of the probability and severity of adverse effects.

A risk factor is a variable associated with the occurrence of an unexpected event. Risk factors are correlational, not necessarily causal and, if one of them occurs, it may have one or more impacts. According to Haimes [15], risks can often arise as the result of an underlying stochastic process occurring over time and space, but also, can occur based on deterministic risk factors. Risk estimation can be achieved based on historical information and knowledge from previous similar projects and from other information sources [4].

Risk is a concept that many find difficult to be understood [16]. A limitation in this definition is the practical difficulty of estimating the probability and impact of various risk factors, especially in software projects. Probabilities can only be defined with significance for activities that are repeated many times under controlled conditions, however, the unique nature of many software projects activities do not allow accurate estimation of their probabilities. A classic approach in decision theory [8] defines risk as a variation in the probability distribution of a outcome, not a probable outcome.

Other limitation of that definition is that it only encompasses known or foreseeable threats, providing limited options to manage unnoticed threats , but also does not recognize unforeseeable threats. That is a consequence of the definition of risk in terms of probability and impact; since, to assess likelihood and impact is necessary to be able to predict an eventuality [8].

In addition, there is another issue: the best decisions are based on numerical quantification - determined by patterns from the past - or on subjective evaluation of the uncertainties? It is not possible to quantify the future with certainty, but through likelihood, it is possible to predict it based on the past. Although it is difficult to find a standard software project, it is possible to classify activities and set patterns that enable the estimation. To Paul Bannerman [8], the usual solution to that problem in software projects is to observe the risk in a broad way, in terms of uncertainty, and evaluate it qualitatively.

Haimes [16] considers two premises in research of risk analysis, which will also be considered during this study. First is that risk is usually quantified by mathematical formula of expectation. However, even though that formula enables a valuable measure of risk, it fails to recognize and or exacerbate the consequences of extreme events. Tom Kendrick presents in his book [17] a framework to identify and manage disasters. Second, states that one of the most difficult tasks in systems analysis is to know how to model it. Therefore, new proposals for quantitative analysis and modeling of systems taking into account its risks will contribute to the scientific advances in the field.

Even though risk management in software project management is a healthy process, its adoption is still far from expectations. A few causes are the overloading of responsibilities on project managers, the low importance attributed to the area, the lack of knowledge of risk management, the costs incurred in risk management activities, the lack of technical skill and familiarity with specific tools. Consequently, the project is

prone to the negative influence of risks without a contingency plan, which may lead to project failure. According to the benchmarking conducted in 2009 by the Project Management Institute, in 20% of the projects their managers do not perform all the planning processes and in only 35% of the projects, risk management is conducted according to a formal methodology, structured by policies, procedures and forms. Also, 46% of managers carry out management activities part time.

The need to manage risks (undesired events) increases exponentially with system complexity. Managing those events in such complex systems becomes difficult to identify and predict undesirable expected or unexpected events occurrence because of huge amount of risk factors involved and their relations. There is an increasing need for more systematic methods and tools to supplement individual knowledge, judgment and experience. These human traits are often sufficient to address less complex and isolated risks. For example, a portion of the most serious issues encountered in system acquisition are the result of risks that are ignored, due to its low likelihood, until they have already created serious consequences [18].

The Guide to the Project Management Body of Knowledgment [4] presents Monte Carlo Simulation as a good practice method to project risk analysis. However, there are some limitations in the adoption of this approach that makes it unfeasible [19]. Simulations can lead to misleading result if inappropriate inputs, derived from subjective parametrization, are entered into the model. Commonly, the user should be prepared to make the necessary adjustments if the results that are generated seem out of line. Moreover, Monte Carlo can not model risks correlations. That means the numbers coming out in each draw are random and in consequence, an outcome can vary from its lowest value, in one period, to the highest in the next. Therefore, alternative approaches must be considered to predict risk likelihood and impact, taking into account project risk characteristics and Monte Carlo Simulation limitations. Thus, risk analysis should be a more accurate and easier task, from users point of view. This work finds artificial neural networks as a valuable alternative to be considered in software project risk analysis.

1.3 Objectives

1.3.1 Research Questions

How to define and model risks in software project management considering disasters?
How quantitatively analyze risks in software project management?
Which data of risk registers of software project are available to perform the study?
How to develop a model to predict the risks of software project management in order to efficiently support decision making?

1.3.2 General Objective

The main purpose of this dissertation is to analyze which is a more efficient approach to software project risk analysis: Monte Carlo Simulation (MCS) technique, Linear

Regression Models (LRM's) or Artificial Neural Networks (ANN's) alternatives - Multilayer Perceptron (MLP), Support Vector Machine (SVM) and Radial Basis Function (RBF) - to improve accuracy and decrease the error prone risk impact estimation.

1.3.3 Specific Objectives

- Develop a technique to predict risk impacts through a artificial neural network to manage risks in software project
- Evaluate traditional approaches to risk impact estimation in terms of decreasing prediction error
- Develop a small risk database from project coordinator experiences supported by PERIL database

To reach the first specific objective the following neural networks are analyzed: Multilayer Perceptron (MLP), Support Vector Machine (SVM), Radial Basis Function (RBF). Moreover, a Neuro-fuzzy System (NFS) is included in the study. The second specific objective is related the analysis of Monte Carlo Simulation (MCS), Multiple Linear Regression Model (MLRM) and Regression Tree Model (RTM). Lastly, the third specific objective is achieved through the collection of risk registers from a software engineering experience in small and medium size projects supported by knowledge of PERIL database and Kendick book [2].

In summary, the methodology adopted in this study is to make statistical experiments to evaluate the prediction error of risk impact from PERIL dataset [2], a framework to identify risks in software project management. The seven selected techniques will estimate the outcome of risk impacts. Mean Absolute Error (MAE) will be calculated thirty times for each approach, and then a hypothesis test may be necessary to assert which is a more accurate method that fits this data. More details about this method is presented in Chapter 4.

It is concluded that a MLP variation called MLPReg, was the successful approach to estimate risk impacts. Besides that, all artificial neural networks alternatives are better than , both, linear regression models and monte carlo simulation. Therefore, it could not be found any reason to assign monte carlo simulation a recommended method to risk analysis according to statistical experiments conducted.

The rest of the dissertation is organized in the following chapters: Chapter 2 address project risk management, qualitative and quantitative risk analysis concepts, monte carlo simulation, linear regression models and artificial neural networks concepts and characteristics. Chapter 3 describes PERIL database, data preprocessing methods to prepare the database to the study and defines the experiments to be undertaken. Chapter 4 describes each experiment. Chapter 5 presents the obtained results for each experiment. Chapter 6 presents a database of risk registers mentioned early. Finally, Chapter 7 presents the conclusions and suggestions of future works.

The works described in this dissertation had results published in the following papers:

- C. H. M. S. Timoteo, M. J. S. Valença, S. M. M. Fernandes, ”*Evaluating Artificial Neural Networks and Traditional Approaches for Risk Analysis in Software Project Management - A case study with PERIL dataset*”, ICEIS 2014: *16th International Conference on Enterprise Information Systems*, Abril, 2014.

Referências Bibliográficas

[1] Alexander Budzier and Bent Flyvbjerg. Double whammy-how ict projects are fooled by randomness and screwed by political intent. *arXiv preprint arXiv:1304.4590*, 2013.

[2] Tom Kendrick. *Identifying and managing project risk: essential tools for failure-proofing your project*. Amacom: New York, 2003.

[3] Y. Y. Higuera, R. P. e Haimes. Software risk management, 1996.

[4] Project Management Institute. *A Guide to the Project Management Body of Knowledge*. 2008.

[5] The Standish Group. Chaos report. 2009.

[6] S. Islam. Software development risk management model - a goal driven approach. *Proc. Of ESEC FSE Doctoral Symposium 09 Amsterdam The Netherlands*, pages 5–8, 2009.

[7] Roy Schmidt, Kalle Lyytinen, Mark Keil, and Paul Cule. Identifying software project risks: an international delphi study. *Journal of management information systems*, 17(4):5–36, 2001.

[8] Paul L Bannerman. Risk and risk management in software projects: A reassessment. *Journal of Systems and Software*, 81(12):2118–2133, 2008.

[9] KPMG. Global it project management survey. 2005.

[10] L. Virine. Project risk analysis: How to make better choices in the uncertain times. *Proceedings of PMI Global Congress*, 2009.

[11] Riddle S. Keshlaf, A. Risk management for web and distributed software development projects. *The Fifth International Conference on Internet Monitoring and Protection*, 2010.

[12] YH Kwak and J Stoddard. Project risk management: lessons learned from software development environment. *Technovation*, 24(11):915–920, 2004.

[13] Young Hoon Kwak and C William Ibbs. Calculating project management's return on investment. *Project Management Journal*, 31(2):38–47, 2000.

[14] B. W. Boehm. Software risk management: principle and practices. *IEEE Software*, 8:32–41, 1991.

[15] Yacov Y Haimes. *Risk modeling, assessment, and management*. John Wiley & Sons, 2011.

[16] Yacov Y(University of Virginia) Haimes. *Risk Modeling, Assesment and Management*. John Wiley & Sons, Inc., 3rd edition edition, 2009.

[17] T. Kendrick. *Identifying and Managing Project Risk: Essential Tools for Failure-Proofing your Project*. 2003.

[18] Ronald P Higuera and Yacov Y Haimes. Software risk management. Technical report, DTIC Document, 1996.

[19] Ibbotson Product Support. *Monte Carlo Simulation*, 2005. IbbotsonAssociates, 225 North Michigan Avenue Suite 700 Chicago, IL 60601-7676.

[20] Osamu Mizuno, Takuya Adachi, Tohru Kikuno, and Yasunari Takagi. On prediction of cost and duration for risky software projects based on risk questionnaire. In *Quality Software, 2001. Proceedings. Second Asia-Pacific Conference on*, pages 120–128. IEEE, 2001.

[21] Xishi Huang, Danny Ho, Jing Ren, and Luiz Fernando Capretz. A neuro-fuzzy tool for software estimation. In *Software Maintenance, 2004. Proceedings. 20th IEEE International Conference on*, page 520. IEEE, 2004.

[22] Yong Hu, Jiaxing Huang, Juhua Chen, Mei Liu, and Kang Xie. Software project risk management modeling with neural network and support vector machine approaches. In *Natural Computation, 2007. ICNC 2007. Third International Conference on*, volume 3, pages 358–362. IEEE, 2007.

[23] Iman Attarzadeh and Siew Hock Ow. A novel soft computing model to increase the accuracy of software development cost estimation. In *Computer and Automation Engineering (ICCAE), 2010 The 2nd International Conference on*, volume 3, pages 603–607. IEEE, 2010.

[24] Dorota Dzega and Wieslaw Pietruszkiewicz. Classification and metaclassification in large scale data mining application for estimation of software projects. In *Cybernetic Intelligent Systems (CIS), 2010 IEEE 9th International Conference on*, pages 1–6. IEEE, 2010.

[25] PEI Yu. Software project risk assessment model based on fuzzy theory. *Computer Knowledge and Technology*, 16:049, 2011.

[26] Urvashi Rahul Saxena and SP Singh. Software effort estimation using neuro-fuzzy approach. In *Software Engineering (CONSEG), 2012 CSI Sixth International Conference on*, pages 1–6. IEEE, 2012.

[27] Zhang Dan. Improving the accuracy in software effort estimation: Using artificial neural network model based on particle swarm optimization. In *Service Operations and Logistics, and Informatics (SOLI), 2013 IEEE International Conference on*, pages 180–185. IEEE, 2013.

[28] Beatrice Lazzerini and Lusine Mkrtchyan. Analyzing risk impact factors using extended fuzzy cognitive maps. *Systems Journal, IEEE*, 5(2):288–297, 2011.

[29] A. Osundahunsi. Effective project risk management using the concept of risk velocity, agility and resiliency. *CAMERON International, Houston, TX, USA*, page 13, 2012.

[30] Ray C Williams, George J Pandelios, and Sandra G Behrens. *Software Risk Evaluation (SRE) Method Description: Version 2.0*. Carnegie Mellon University, Software Engineering Institute, 1999.

[31] Chris Chapman and Stephen Ward. *Project risk management: processes, techniques and insights*. John Wiley, 1996.

[32] Richard Fairley. Risk management for software projects. *Software, IEEE*, 11(3):57–67, 1994.

[33] Kakoli Bandyopadhyay, Peter P Mykytyn, and Kathleen Mykytyn. A framework for integrated risk management in information technology. *Management Decision*, 37(5):437–445, 1999.

[34] Project Management Institute. Practice standard for project risk management. 2009.

[35] Young Hoon Kwak and Lisa Ingall. Exploring monte carlo simulation applications for project management. *Risk Management*, 9(1):44–57, 2007.

[36] Luis Torgo. Data mining with r. *Learning by case studies. University of Porto, LIACC-FEP. URL: http://www. liacc. up. pt/ltorgo/DataMiningWithR/. Accessed on*, 7(09), 2003.

[37] Jiawei Han, Micheline Kamber, and Jian Pei. *Data mining: concepts and techniques*. Morgan kaufmann, 2006.

[38] Lee C.S.G. Lin, C.T. Neural fuzzy systems: A neuro-fuzzy synergism to intelligent systems. 1996.

[39] M. J. S. Valença. Aplicando redes neurais: um guia completo. *Livro Rápido, Olinda–PE*, 2005.

[40] Pitts W. MCCulloch, W. A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophisics*, pages 115–133, 1943.

[41] Donald O Hebb. *The organization of behavior: A neuropsychological theory*. New York: Wiley, 1949.

[42] Bernard WIDROW, Marcian E HOFF, et al. Adaptive switching circuits. 1960.

[43] Frank Rosenblatt. Perceptron simulation experiments. *Proceedings of the IRE*, 48(3):301–309, 1960.

[44] P. Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. PhD thesis, Harvard University, Cambridge, MA, 1974.

[45] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, DTIC Document, 1985.

[46] M. J. S. Valença. *Fundamentos das Redes Neurais - Exemplos em JAVA*. 2a edicao edition.

[47] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Macmillan, New York, 1994.

[48] S. Haykin. *Redes Neurais: Princípios e Práticas*. 2007.

[49] Mohamad T Musavi, Wahid Ahmed, Khue Hiang Chan, Kathleen B Faris, and Donald M Hummels. On the training of radial basis function classifiers. *Neural networks*, 5(4):595–603, 1992.

[50] Sheng Chen, CFN Cowan, and PM Grant. Orthogonal least squares learning algorithm for radial basis function networks. *Neural Networks, IEEE Transactions on*, 2(2):302–309, 1991.

[51] Russell Beale and Tom Jackson. *Neural Computing-an introduction*. CRC Press, 2010.

[52] Sidney Siegel. Nonparametric statistics for the behavioral sciences. 1956.

[53] S Amari, Noboru Murata, Klaus-Robert Müller, Michael Finke, and H Yang. Statistical theory of overtraining-is cross-validation asymptotically effective? *Advances in neural information processing systems*, pages 176–182, 1996.

[54] Shun-ichi Amari, Andrzej Cichocki, Howard Hua Yang, et al. A new learning algorithm for blind signal separation. *Advances in neural information processing systems*, pages 757–763, 1996.

[55] N. Taleb. *Fooled by randomness*. Nwe York: Random House, 2001.

[56] William N Venables, David M Smith, and R Development Core Team. An introduction to r, 2002.

[57] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The weka data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18, 2009.

[58] A.J. Smola and B. Schoelkopf. A tutorial on support vector regression. Technical report, 1998. NeuroCOLT2 Technical Report NC2-TR-1998-030.

[59] Kevin L Priddy and Paul E Keller. *Artificial Neural Networks: An introduction*, volume 68. SPIE Press, 2005.

[60] S.K. Shevade, S.S. Keerthi, C. Bhattacharyya, and K.R.K. Murthy. Improvements to the smo algorithm for svm regression. In *IEEE Transactions on Neural Networks*, 1999.