Uma Rede Bayesiana para calcular riscos arquiteturais em projetos de software

Karlos Almeida
Instituto Federal da Paraíba - Campus Monteiro
São José dos Cordeiros, Brasil
karlos.macedo@academico.ifpb.edu.br

Emanuel Filho
Instituto Federal da Paraíba - Campus Campina Grande
Campina Grande, Paraíba
emanuel.filho@ifpb.edu.br

Patrícia Pereira
Instituto Federal da Paraíba - Campus Monteiro
Monteiro, Brasil
patricia.pereira@academico.ifpb.edu.br

Tamires Xavier
Instituto Federal da Paraíba - Campus Monteiro
Sertânia, Brasil
tamires.xavier@academico.ifpb.edu.br

Ademar Neto
Universidade Federal de Campina Grande
Campina Grande, Paraíba
ademar.souto@virtus.ufcg.edu.br

Resumo— As Redes Bayesianas são modelos gráficos capazes de representar relações de causalidade entre variáveis. Muitos trabalhos, no estado da arte, utilizam Redes Bayesianas para apoiar o gerenciamento de riscos. Neste contexto, os nós representam fatores de riscos e, através das inferências probabilísticas, o risco do projeto é calculado. As abordagens se distinguem pela escolha dos fatores usados na construção da rede. Uma lacuna observada no estado da arte é a ausência de Redes Bayesianas para apoiar o gerenciamento de riscos utilizando fatores relacionados à arquitetura do projeto de software. Neste sentido, o objetivo deste trabalho é propor uma Rede Bayesiana para modelar fatores relacionados à segurança, desempenho e carga com a finalidade de calcular o risco arquitetural do projeto. Para validar nossa abordagem, um experimento foi realizado com três equipes de desenvolvimento em uma fábrica de software. Instanciamos a Rede Bayesiana proposta e calculamos o risco arquitetural do projeto. Como resultado 85,71% dos participantes concordaram com o risco calculado e acreditam que a abordagem é útil para auxiliar o gerenciamento de riscos.

Palavras-chave— gerenciamento de riscos, Redes Bayesianas, riscos arquiteturais

A Bayesian Network to calculate architectural risks in software projects

Abstract— ***Bayesian Networks are graphical models capable of representing causal relationships between variables. Many works in state of art use Bayesian Networks to support risk management. In this context, the nodes represent risk factors, and probabilistic inference calculates project risks. The choice of factors is what distinguishes approaches in building the network. A gap observed in state-of-the-art is the absence of Bayesian Networks to support risk management using factors related to the architecture of the software project. In this sense, this work proposes a Bayesian Network to model safety, performance, and load factors to calculate the project's architectural risk. A software factory experiment was carried out with three development teams to validate our approach. We instantiate the proposed Bayesian Network***

and calculate the architectural risk of the project. As a result, 85,71% of respondents agreed with the calculated risk and believed that the approach helps to manage risk.

Keywords— ***risk management, Bayesian Networks, architectural risks***

1. Introdução

Na indústria de software, os processos para gerenciamento de riscos estão presentes em projetos que utilizam tanto metodologias tradicionais quanto ágeis [Odzaly and Des Greer 2014]. Um risco é um evento ou uma condição incerta que, se ocorrer, tem efeito em pelo menos um objetivo do projeto [Boehm 1989]. O efeito pode ser positivo (oportunidade) ou negativo (ameaça) [PMI 2019].

As principais técnicas existentes na indústria para auxiliar o gerenciamento de riscos são baseadas em *Brainstorming*, listas de verificação, análise de matriz de probabilidade e impacto [Xu et al. 2003]. Além disso, métodos baseados em Inteligência Artificial são encontrados na literatura em abordagens promissoras. Fenton et al. [2008] utilizaram Redes Bayesianas para mensurar fatores de riscos e inferir densidade de defeitos do sistema.

Apesar de existirem técnicas amplamente conhecidas na indústria e na academia, gerenciar riscos ainda é considerada uma tarefa complexa e crítica [Arumugam et al. 2017]. Riscos são eventos abstratos caracterizados pela subjetividade [Islam 2009], o que torna difícil uma análise precisa para auxílio à tomada de decisão [Van Scoy 1992, Min et al. 2017, Mendes et al. 2018].

De forma geral, as abordagens baseadas em Redes Bayesianas buscam identificar variáveis (i.e., fatores de riscos) e em seguida realizar o cálculo do risco no projeto. Estes fatores podem estar relacionados ao escopo (p.e., clareza dos requisitos, detalhes do escopo), ao time (p.e., experiência dos desenvolvedores, habilidades técnicas) ou as tecnologias (p.e. produtividade com uso de linguagens, integração de bibliotecas). Portanto, cada Rede Bayesiana presente na literatura [Fenton et al. 2008, Kumar and Yadav 2017, Khodakarami and Abdi 2014, Nguyen et al. 2018] faz inferência com base nos fatores utilizados na modelagem da abordagem.

Uma lacuna observada na literatura foi a presença de Redes Bayesianas com foco em definir o risco arquitetural de um projeto de software. Neste contexto, este trabalho propõe uma Rede Bayesiana formada por nós relacionados à segurança, desempenho, carga e demais restrições técnicas de um projeto de software. Para definição da estrutura foram utilizadas buscas em fóruns da área e opiniões de especialistas.

Para validar a Rede Bayesiana proposta, foi realizado um experimento com três equipes de desenvolvimento que desenvolvem software de diferentes contextos. Ao todo, quatorze participantes avaliaram a abordagem. Os resultados foram considerados promissores, 85,71% dos participantes concordaram com o risco calculado e consideraram a abordagem útil para auxiliar o gerenciamento de riscos.

O restante do artigo está estruturado da seguinte forma: na Seção 02 está descrito o referencial teórico; na Seção 03 a metodologia desta pesquisa é discutida; por fim, na Seção 04 é apresentado a Redes Bayesiana e na Seção 05 são discutidos os resultados da validação.

2. Fundamentação Teórica

2.1. Gerenciamento de Riscos

O guia PMBOK (Project Management Body of Knowledge) define sete processos para o gerenciamento de riscos [(PMI) 2019], com atividades do planejamento ao monitoramento dos riscos. Por outro lado, a norma ISO 31000 apresenta princípios e recomendações em cinco atividades que abordam riscos [International Standards Organisation 2018]. De forma geral, o gerenciamento de

riscos está dividido em três atividades: identificar, mensurar e monitorar os riscos [Kajko-Mattsson and Nyfjord 2008].

O objetivo do processo de identificação é detectar os eventos que podem afetar o projeto [Chadli and Idri 2017]. Com os riscos identificados, deve-se criar uma lista com o registro desses eventos [Ancveire et al. 2015] contendo uma descrição de suas características e um plano de respostas para cada evento identificado.

A mensuração de riscos consiste em avaliar e medir [Hearty et al. 2005] os eventos previamente identificados [Smith and Merritt 2002]. Inicialmente uma análise qualitativa é realizada, considerando a probabilidade de ocorrência dos riscos e seus impactos nos objetivos do projeto [Gupta and Sadiq 2008]. Em seguida, busca-se quantificar numericamente os eventos para uma avaliação com maior precisão [Kojima et al. 2008].

Na etapa de monitoramento, os riscos são inspecionados periodicamente para avaliar indicações de que o risco ocorreu ou está prestes a ocorrer [Kerzner 2017]. Esta etapa é também uma fase de controle, onde são implementados planos de respostas como forma de mitigação desses eventos [Kumar et al. 2018]. O objetivo é evitar ou reduzir o impacto e assim melhorar o controle das atividades do projeto [Emam and Koru 2008].

Os riscos de um projeto de software podem estar relacionados a aspectos do processo de desenvolvimento (p.e., escopo e requisitos, comunicação com o cliente, equipe de desenvolvimento) ou do produto (p.e., tipo de software, aspectos arquiteturais e restrições tecnológicas).

2.2 Redes Bayesianas

As Redes Bayesianas são modelos gráficos capazes de representar as relações de causalidade entre variáveis [Fan and Yu 2004]. Pertencem à família de modelos probabilísticos usados para representar incertezas de um domínio [Ben-Gal 2008]. Segundo Neapolitan et al. [Neapolitan et al. 2004], a técnica surgiu para representar contextos em que há um grande número de variáveis, e o objetivo de verificar a influência probabilística que uma ou mais variáveis exercem sobre outras.

Uma Rede Bayesiana pode ser dividida em termos qualitativos e quantitativos [van der Gaag 1996]. A parte qualitativa, o chamado aprendizado estrutural, é a representação gráfica das variáveis em forma de um Grafo Acíclico Dirigido. A parte quantitativa, chamada de aprendizado de parâmetros, define relações de dependência como funções de probabilidade condicional entre variáveis, usando relações de causa e consequência [Khodakarami and Abdi 2014].

O conhecimento de uma Rede Bayesiana é extraído da probabilidade condicional de um conjunto de variáveis de consulta, de acordo com as variáveis de evidência. [Ben-Gal 2008]. Esta probabilidade é calculada utilizando o Teorema de Bayes, onde a probabilidade de um evento é definido dado que outro evento ocorreu [Neapolitan et al. 2004].

Com o grafo estruturado e as tabelas de probabilidades definidas, o raciocínio para uso das Redes Bayesianas pode ser realizado de quatro maneiras distintas [Norvig 2009]. O raciocínio de diagnóstico está dirigido em sentido oposto das arestas da rede, implicando: se existe um efeito, a causa pode ser estimada. O raciocínio preditivo ou prognóstico calcula efeitos futuros seguindo as causas apresentadas. A direção desse raciocínio coincide com a direção das arestas.

3. Metodologia

A construção da Rede Bayesiana proposta nesta pesquisa foi dividida em dois passos: pesquisas em fontes públicas de informações técnicas (p.e., sites, blogs, fóruns, revistas eletrônicas) e elicitação de conhecimento através de entrevistas com especialistas.

Inicialmente foi realizada a pesquisa a fim de reunir e analisar fatores de riscos recorrentes nos projetos de software no que se refere às características arquiteturais do software.

Sobre a fonte de pesquisa, os autores deste artigo navegaram em sites na intenção de coletar fatores de riscos que os profissionais da área se depararam durante o desenvolvimento de software.

Ao final desta etapa, uma lista com trinta e dois fatores de riscos foi criada. São exemplos: criptografia assimétrica, decodificação de token, acesso a banco em nuvem, acesso a API de terceiros e integração com plataformas externas. A lista de fatores de riscos criada com a pesquisa em sites é a entrada da segunda e última etapa para construção da Rede Bayesiana.

A lista foi apresentada a dois especialistas da área com mais de 10 anos de experiência em desenvolvimento de software. Com base no KEN (Emília Mendes, 2014), os especialistas inicialmente agruparam os fatores que compartilhavam de um mesmo contexto (p.e., segurança, desempenho). Em seguida, para cada grupo de fatores foram analisadas as dependências dos fatores para criação do grafo (p.e., criptografia e decodificação são nós filhos de um nó pai "segurança de login").

Na segunda etapa, também foram avaliadas redundâncias (p.e., o fator de risco "acesso a múltiplos usuários" têm mesma finalidade de "granularidade de acesso"). Ao final, a Rede Bayesiana foi modelada com um total de doze fatores de riscos divididos em cinco níveis. Na seção a seguir, são apresentados detalhes de cada nó da Rede Bayesiana.

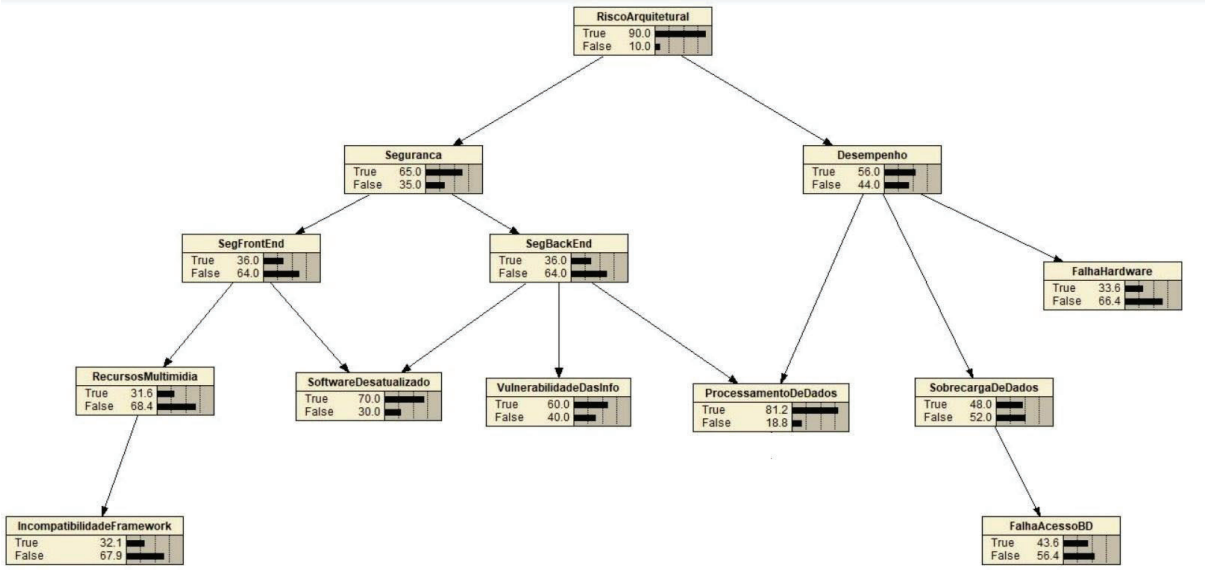

Figura 01 - Rede Bayesiana

4. Solução Proposta

Inicialmente para modelagem da Rede Bayesiana foram utilizadas consultas em fóruns, blogs e sites de desenvolvimento de software. O Objetivo foi identificar os principais riscos associados à Arquitetura de um software. Essa atividade foi realizada pelos autores deste artigo. Para a seleção dos riscos, durante as consultas ocorria a apuração dos riscos mais pertinentes, onde aparentemente o risco não havia solução, ou a solução viraria um dilema optativo entre um caminho a ser seguido.

Esta rede pode ser utilizada para a visualização de onde estão as principais vulnerabilidades dos riscos arquiteturais atualmente, além de servir de auxílio para as equipes de desenvolvimento obterem um norte de onde deve-se ter mais cautela.

Como forma de validar a rede construída e adicionar novos nós, dois especialistas de uma empresa de software foram consultados. Eles foram importantes principalmente para definir as relações de causalidade e construção da Rede Bayesiana apresentada na Figura 1.

A seguir um catálogo com a descrição de cada um dos nós é apresentado:

- **Segurança -** A segurança do software é a medida protetiva contra possíveis invasões indesejadas ou maliciosas no sistema em questão. Estas invasões geralmente ocorrem com a finalidade de encontrar dados de usuários, do software, ou da instituição para onde o software foi desenvolvido, podendo ocasionar na exposição destes dados, ou até mesmo serem utilizados para fins indeterminados. Para evitar este risco, há boas práticas no que diz respeito ao desenvolvimento, e tecnologias a serem analisadas antes de serem aplicadas em um determinado projeto.

- **Desempenho -** O desempenho é onde o tempo da execução das tarefas implementadas no software é levado em consideração. Estas tarefas partem desde a abertura, até a realização de operações presentes no software, como a pesquisa por informações, por exemplo. O correto é fazer com que o software obtenha o desempenho mais equilibrado possível a fim de não comprometer a experiência de quem está à frente do software.

- **Segurança do *Front-end* -** A segurança no front-end é o cuidado que deve ser tomado ao levar informações inseridas pelo usuário através dos dispositivos de entrada até o *back-end*, onde estas informações são armazenadas e processadas. Levando em conta estas definições, é preciso que haja filtragens e verificações antes que essas informações sejam encaminhadas, evitando complicações posteriores.

- **Segurança do *Back-end* -** A segurança do *back-end* compartilha das mesmas definições de segurança já mencionadas, porém estas são aplicadas na proteção das informações que o usuário confiou ao software, evitando que estas informações corram risco de exposição e vulnerabilidade. E não menos importante, a segurança destas informações também implica na garantia de que estas serão processadas corretamente.

- **Hardware -** Considerado como um dos fatores ocasionadores de risco de desempenho, são os componentes físicos presentes nas máquinas onde os respectivos softwares estão sendo executados. Exemplos desses componentes são o processador, a placa gráfica, a memória primária, secundária, e etc. Softwares executados em hardwares que não são projetados para tais execuções, podem afetar bastante na experiência de utilização, como lentidão, travamentos ou inexecução de operações.

- **Recursos multimídia -** São os recursos midiáticos presentes nos softwares. Imagens, ícones, vídeos e áudios. Esses recursos são fundamentais para complementar a experiência do usuário, seja visual ou auditiva. Os riscos presentes nestes recursos ao que diz respeito ao *front-end* se dão por conta da falta de disponibilidade dos mesmos aos usuários, ou quando são mostrados de forma incorreta, sem a semântica ideal, ou quando simplesmente não são mostrados por incompatibilidade entre as tecnologias aplicadas no software.

- **Software Desatualizado -** No mundo do desenvolvimento, é muito comum que um software seja projetado visando no futuro a implantação de novas funcionalidades, correção de falhas, ou para manutenção do que já foi implementado. E para obter controle dessas mudanças, os softwares são mantidos em diferentes versões, onde versões mais antigas não possuem as novas implementações das versões atuais. Software desatualizado é considerado um risco de software por não conter os reparos necessários que garantirão a integridade do mesmo onde ele está sendo implementado. Isso pode impedir a conversação do software com tecnologias

atuais, ou até mesmo com tecnologias antigas, mas que o software obteve o reparo para que a compatibilidade fosse possível.

- **Vulnerabilidade das Informações -** É quando o software não teve o planejamento correto sobre o armazenamento e proteção do o mesmo será capaz de armazenar, fazendo com que estas informações estejam sujeitas a ataques de hackers, a vazamentos, ou até a exclusão inesperada por não estarem sendo armazenadas de forma segura. Para isso, existem mecanismos que garantem maior segurança desses dados, como armazenamento na nuvem, backup, e etc.

- **Processamento de Dados -** O Processamento de Dados é a execução das operações disponíveis no software com as informações fornecidas, ou com informações que são solicitadas. Essas operações podem ser exemplificadas como operações aritméticas, operações de checagem, de pesquisas, e as informações podem estar presentes tanto no front-end como no banco de dados, ou requisitadas através de APIs. O risco no processamento de dados se dá por falhas nas execuções das operações implementadas, causadas por diversos fatores como má implementação, hardware incapaz, ou incompatibilidade de tecnologias, por exemplo.

- **Sobrecarga de Dados -** Enquanto o processamento se refere às operações, a sobrecarga se refere a como o software está lidando no que diz respeito ao armazenamento dos dados fornecidos. Para haver o controle, é fundamental que o gerenciamento desses dados estejam em perfeita conversação com o hardware e com as tecnologias inseridas no software, tendo em vista que eles podem ser armazenados tanto em memória física, como em memória virtual, ou até em ambas, isso depende da finalidade de cada dado. A sobrecarga afeta principalmente o desempenho e por isso é considerada um risco de software.

- **Incompatibilidade *Framework* -** *Frameworks* são criados para simplificar o desenvolvimento onde a implementação de códigos comuns já é prevista. Assim, como existem diferentes tipos de softwares, também existem diferentes tipos de *frameworks* para atender as mais diversas necessidades, seja no back-end ou no front-end. O Erro na Escolha do *Framework* é um nó filho de Recursos Multimídia pela probabilidade dos riscos desses recursos se darem pela escolha de um *framework* inadequado para tal ocasião, onde nas pesquisas, vimos que há vários bugs relacionados a *frameworks* porque os desenvolvedores não buscam por um que realmente facilite ou incremente positivamente no objetivo esperado. O que resulta em problemas de exibição de imagens, de redimensionamento de tela, entre outros riscos.

- **Falha no Acesso ao Banco de Dados -** Assim como existem diferentes *frameworks* para facilitar a implementação de códigos comuns, há também diversos tipos de Banco de Dados para atender os mais diversos tipos de armazenamento, sejam eles estruturados ou não. O Erro na Escolha do Banco de Dados é um risco consecutivo da sobrecarga de dados pela probabilidade da sobrecarga ocorrer devido o banco de dados não ser o ideal para os dados que o software precisa armazenar. Há bancos mais propícios para sistemas tradicionais armazenados em máquinas físicas, bancos especializados para sistemas web, bancos que armazenam na nuvem, além das diversas estruturas como tabelas, grafos e árvores. Para lidar com os diversos tipos de dados deve haver cautela e análise, para assim evitar riscos posteriores, tomando por exemplo o de sobrecarga (como já mencionado), ou o de migração futura para outro banco.

5. Experimento

Uma avaliação da Rede Bayesiana proposta foi realizada com três equipes de desenvolvimento de software. Ao todo, quatorze profissionais de TI de diferentes níveis participaram da validação da abordagem.

Em cada projeto, os nós intermediários foram instanciados de acordo com o entendimento do time. Por simplicidade, optou-se por representar esses nós como booleanos. Ou seja, se o time informar que um determinado fator de risco faz parte do projeto, esse nó é instanciado como *TRUE*, caso contrário como *FALSE*. Com relação às tabelas de probabilidade optou-se pela distribuição uniforme de probabilidades.

Após instanciá-las, a rede é executada e o risco arquitetural do projeto é calculado. Para essa atividade, usamos o software AgenaRisk[1] para modelar e executar a Rede Bayesiana.

Os valores de probabilidade calculados nos três projetos foram, respectivamente: 90%, 65% e 72%. Este valor corresponde à probabilidade do nó raiz da Rede Bayesiana. Após acompanharem o processo, os profissionais do time foram convidados a responder duas perguntas:

QP1: A Rede Bayesiana calcula corretamente o risco arquitetural do projeto?

QP2: Você acredita que a Rede Bayesiana é útil para o gerenciamento de riscos arquiteturais?

Como resposta, os profissionais responderam as perguntas com opções em uma escala Likert de cinco níveis. Também foram coletadas sugestões e críticas à abordagem proposta.

Como resultado, para 12 dos 14 desenvolvedores o resultado foi condizente com a realidade do projeto (i.e., 85,71%). Os participantes apontaram melhorias e acreditam que a ferramenta pode ser útil para o dia a dia dos projetos de software, mas identificaram que outros nós precisam ser instanciados para melhor representação do risco arquitetural dos projetos de software.

A pesquisa encontra-se em andamento, e o próximo passo será realizar uma validação com mais equipes em diferentes empresas de desenvolvimento de software.

6. Considerações Finais

As Redes Bayesianas descritas na literatura utilizam diferentes fatores de riscos para modelar as variáveis que influenciam os riscos dos projetos. Os fatores de riscos mais utilizados estão relacionados aos requisitos do projeto, equipe e processo de desenvolvimento [Drew Procaccino et al. 2002, Ancveire et al. 2015, Nguyen et al. 2018]. Porém, a literatura se mostra escassa para apresentar os fatores de riscos relacionados às características arquiteturais dos projetos. Dada a dificuldade de reprodução ou a pouca utilidade dos modelos presentes no estado da arte [Fenton et al. 2008, Kumar and Yadav 2017, Chin et al. 2009], neste trabalho foi criado um modelo próprio para o gerenciamento de riscos. O foco da abordagem está em modelar os fatores de riscos relacionados às características tecnológicas dos projetos de software, com foco em aspectos arquiteturais.

Neste trabalho, foi apresentado a versão inicial de uma Rede Bayesiana para cálculo de riscos arquiteturais de projetos de software. Espera-se que a abordagem proposta seja útil para auxiliar o gerenciamento de projetos de software. Como trabalhos futuros, destaca-se a necessidade de ajustar a Rede Bayesiana com opinião de outros especialistas da área. Com isso, espera-se que a Rede Bayesiana possa refletir os riscos arquiteturais em projetos de software para o maior número possível de contextos.

[1] https://www.agenarisk.com/

Referências

1) Ancveire, I., Gailite, I., Gailite, M., and Grabis, J. (2015). Software delivery risk management: Application of bayesian networks in agile software development. *Information Technology and Management Science*, 18(1):62–69.

2) Arumugam, C., Kameswaran, S., and Kaliamourthy, B. (2017). Global software development: A design *framework* to measure the risk of the global practitioners. In *Proceedings of the 7th International Conference on Computer and Communication Technology*, ICCCT-2017, page 1–8, New York, NY, USA. Association for Computing Machinery.

3) Ben-Gal, I. (2008). Bayesian networks. *Encyclopedia of statistics in quality and reliability*, 1(2).

4) Boehm, B. (1989). Software risk management. In Ghezzi, C. and McDermid, J. A., editors, ESEC '89, pages 1–19, Berlin, Heidelberg. Springer Berlin Heidelberg.

5) Chadli, S. Y. and Idri, A. (2017). Identifying and mitigating risks of software project management in global software development. In *Proceedings of the 27th International Workshop on Software Measurement and 12th International Conference on Software Process and Product Measurement*, IWSM Mensura '17, page 12–22, New York, NY, USA. Association for Computing Machinery.

6) Chin, K.-S., Tang, D.-W., Yang, J.-B., Wong, S. Y., and Wang, H. (2009). Assessing new product development project risk by bayesian network with a systematic probability generation methodology. *Expert Systems with Applications*, 36(6):9879–9890.

7) Drew Procaccino, J., Verner, J. M., Overmyer, S. P., and Darter, M. E. (2002). Case study: factors for early prediction of software development success. *Information and Software Technology*, 44(1):53–62.

8) Emilia Mendes, Practitioner's Knowledge Representation: a pathway to improve software effort estimation. 4 Edição, 2014, Springer.

9) Emam, K. E. and Koru, A. G. (2008). A replicated survey of it software project failures. *IEEE Software*, 25(5):84–90.

10) Fan, C.-F. and Yu, Y.-C. (2004). Bbn-based software project risk management. *Journal of Systems and Software*, 73(2):193–203. Applications of statistics in software engineering.

11) Fenton, N., Neil, M., Marsh, W., Hearty, P., Radlinski, U., and Krause, P. (2008). On the ´
effectiveness of early life cycle defect prediction with bayesian nets. *Empirical Softw. Engg.*, 13(5):499–537.

12) Gupta, D. and Sadiq, M. (2008). Software risk assessment and estimation model. In 2008 *International Conference on Computer Science and Information Technology*, pages 963–967.

13) Hearty, P., Fenton, N., Neil, M., and Cates, P. (2005). Automated population of causal models for improved software risk assessment. In *Proceedings of the 20th IEEE/ACM International Conference on Automated Software Engineering*, ASE '05, page 433–434, New York, NY, USA. Association for Computing Machinery

14) International Standards Organization (2018). Iso 31000: Risk management—principles and guidelines. Disponível em: `https://www.iso.org/obp/ui/#iso:std: iso:31000:en`. Acesso em: 15 abril 2021.

15) Islam, S. (2009). Software development risk management model: a goal driven approach. In *Proceedings of the doctoral symposium for ESEC/FSE on Doctoral symposium*, pages 5–8.

16) Kajko-Mattsson, M. and Nyfjord, J. (2008). State of software risk management practice. IAENG *international journal of Computer Science*, 35(4).

17) Kerzner, H. (2017). *Project management: a systems approach to planning, scheduling, and controlling*. John Wiley & Sons, 11a edition. ISBN 978-1118022276.

18) Khodakarami, V. and Abdi, A. (2014). Project cost risk analysis: A bayesian networks approach for modeling dependencies between cost items. *International Journal of Project Management*, 32(7):1233–1245.

19) Kojima, T., Hasegawa, T., Misumi, M., and Nakamura, T. (2008). Risk analysis of software process measurements. *Software Quality Journal*, 16(3):361–376.

20) Kumar, C. and Yadav, D. K. (2017). Software defects estimation using metrics of early phases of software development life cycle. *International Journal of System Assurance Engineering and Management*, 8(4):2109–2117.

21) Kumar, R., Tayal, A., and Kapil, S. (2018). *Analyzing the Role of Risk Mitigation and Monitoring in Software Development*. IGI Global, 1a edition. ISBN 978-1522560296.

22) Mendes, E., Rodriguez, P., Freitas, V., Baker, S., and Atoui, M. A. (2018). Towards improving decision making and estimating the value of decisions in value-based software engineering: The value framework. *Software Quality Journal*, 26(2):607–656.

23) Min, W., Jun, Z., and Wei, Z. (2017). The application of a fuzzy comprehensive evaluation method in the software project risk assessment. In Proceedings of the 2017 *International Conference on Management Engineering, Software Engineering and Service Sciences*, ICMSS '17, page 76–79, New York, NY, USA. Association for Computing Machinery.

24) Neapolitan, R. E. et al. (2004). *Learning bayesian networks*, volume 38. Pearson Prentice Hall Upper Saddle River, NJ.

25) Nguyen, N.-T., Huynh, Q.-T., and Vu, T.-H.-G. (2018). A bayesian critical path method for managing common risks in software project scheduling. In *Proceedings of the Ninth International Symposium on Information and Communication Technology*, SoICT 2018, page 382–388, New York, NY, USA. Association for Computing Machinery.

26) Norvig, P. R. (2009). *Artificial Intelligence - A modern approach*. Prentice Hall, 3a edition. ISBN 978-0136042594.

27) Odzaly, E. E. and Des Greer, D. S. (2014). Lightweight risk management in agile projects. *26th Software Engineering and Knowledge Engineering (SEKE)*, pages 321–326.

28) PMI (2019). A guide to the project management body of knowledge (pmbok guide). 6.

29) (PMI), P. M. I. (2019). A *Guide to the Project Management Body of Knowledge (PMBOK Guide)*, volume 6. EUA.

30) Smith, P. G. and Merritt, G. M. (2002). Proactive risk management: *Controlling uncertainty in product development*. Productivity Press, 1a edition. ISBN 978-1563272653.

31) van der Gaag, L. C. (1996). Bayesian Belief Networks: Odds and Ends. *The Computer Journal*, 39(2):97–113.

32) Van Scoy, R. L. (1992). Software development risk: opportunity, not problem. Technical report, Univ. Pittsburgh - Software Engineering Inst.
33) Xu, Z., Khoshgoftaar, T. M., and Allen, E. B. (2003). Application of fuzzy expert systems in assessing operational risk of software. *Information and Software Technology*, 45(7):373–388. Software Engineering with Computational Intelligence S.I.