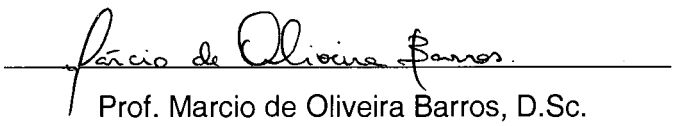
UMA ABORDAGEM ECONÔMICA BASEADA EM RISCOS PARA AVALIAÇÃO DE UMA CARTEIRA DE PROJETOS DE SOFTWARE

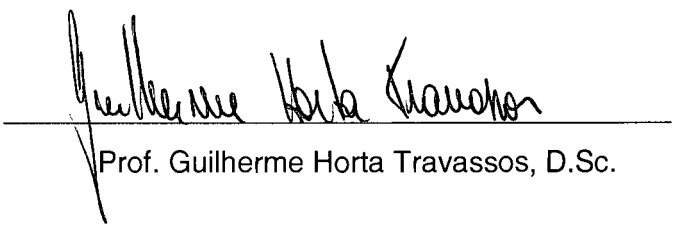
Hélio Rodrigues Costa

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Aprovada por:



Prof. Marcio de Oliveira Barros, D.Sc.



Prof. Guilherme Horta Travassos, D.Sc.

Prof. Ana Regina Cavalcanti da Rocha, D.Sc.

Prof. Manoel Gomes de Mendonça Neto, Ph.D.

RIO DE JANEIRO, RJ - BRASIL

ABRIL DE 2005

COSTA, HÉLIO RODRIGUES

Uma abordagem econômica baseada em riscos para avaliação de uma carteira de projetos de software [Rio de Janeiro] 2005

VIII, 102p. 29,7cm (COPPE/UFRJ, M.Sc, Engenharia de Sistemas e Computação, 2003)

Tese – Universidade Federal do Rio de Janeiro, COPPE

1. Economia aplicada à Engenharia de Software

2. Gerência de Riscos em Projetos de Software

3. Quantificação de Riscos

4. Estudos Experimentais em Engenharia de Software

I. COPPE/UFRJ II. Título (série)

Resumo de tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

UMA ABORDAGEM ECONÔMICA BASEADA EM RISCOS PARA AVALIAÇÃO DE UMA CARTEIRA DE PROJETOS DE SOFTWARE

Hélio Rodrigues Costa

Abril/2005

Orientadores: Marcio de Oliveira Barros
Guilherme Horta Travassos

Programa: Engenharia de Sistemas e Computação

Gerenciar riscos vem se tornando uma prática cada vez mais relevante para o sucesso de projetos de software, pois minimizar as incertezas existentes durante todo o ciclo de vida de sistemas tende a conduzir estes projetos a resultados mais próximos dos objetivos determinados por uma empresa. Uma das atividades mais críticas deste tipo de gerência é a quantificação dos riscos, pois não nos basta saber que existem riscos em um projeto, mas também, quão arriscados são os projetos e quais as suas probabilidades de fracasso. No entanto, uma organização não possui, normalmente, um único projeto em desenvolvimento, e sim uma carteira de projetos. Desta forma, os lucros provenientes de alguns projetos podem compensar fracassos derivados de outros.

Este trabalho apresenta uma técnica para quantificar os riscos de projetos de software de uma categoria específica de sistemas e faz uma analogia entre projetos de software e uma abordagem econômica para análise de riscos de crédito. Um estudo experimental foi realizado e uma ferramenta foi implementada para apoiar as propostas. Desta forma, uma organização tem como realizar análises de sua carteira de projetos e verificar a sua probabilidade de perdas e ganhos, bem como determinar estratégias a serem traçadas visando a maximização de seus lucros.

Abstract of Thesis presented to COPPE/UFRJ as a partial fullfilment of the requirements for the degree of Master of Science (M.Sc.)

A RISK-BASED ECONOMIC APPROACH FOR EVALUATION OF A SOFTWARE PROJECT PORTFOLIO

Hélio Rodrigues Costa

April/2005

Advisors: Marcio de Oliveira Barros
Guilherme Horta Travassos

Department: System and Computer Engineering

Risk management has become a relevant practice for the success of software projects, because minimizing uncertainties during the entire systems life-cycle tends to conduct these projects to results that are closer to the objectives established by an enterprise. One of the most critical activities of this type of management is risk quantification, because it is not only enough to know that there exist risks in a project, but also how risky is such project and what is its failure probability. Moreover, an organization does not usually has a single project under development, but a project portfolio. Therefore, the profits derived from successful projects can compensate losses from failing ones.

This work presents a technique to quantify software project risks of a specific system category and makes an analogy between software projects and an economic approach for credit risk analysis. In this way, an organization has means to perform analyses of its projects portfolio and verify its losses and earnings probabilities, as well as determining strategies to be developed aiming at maximizing the profits.

MAR PORTUGUÊS

(Fernando Pessoa)

Oh mar salgado, quanto do teu sal
São lágrimas de Portugal!
Por te cruzarmos, quantas mães choraram,
Quantos filhos em vão rezaram!
Quantas noivas ficaram por casar
Para que fosses nosso, oh mar!

Valeu a pena? Tudo vale a pena
Se a alma não é pequena.
Quem quer passar além do Bojador
Tem que passar além da dor.
Deus ao mar o perigo e o abismo deu,
Mas nele é que espelhou o céu.

Aos meus pais que me deram o maior presente
que uma pessoa pode receber: a vida

AGRADECIMENTOS

Poucas virtudes são tão importantes como a Gratidão. É através dela que reconhecemos, no fundo de nosso coração, a energia e o sentimento que brota em nossa alma, no momento em que percebemos a vontade de, simplesmente, dizer obrigado àquele que dedicou alguns instantes de sua vida à maravilhosa missão de ajudar o próximo.

Fico feliz de, ao final desta jornada, ter uma grade lista de pessoas a quem posso dedicar este sentimento.

Inicialmente, agradeço a Deus e a meus mentores pelo grandioso amor e eterna presença em minha vida, pois sem eles nada seria possível.

À minha mãe, por todo o sacrifício e dedicação despendidos para que um dia eu pudesse chegar onde estou.

Ao meu pai, pelo carinho, amor e todas as lições que pude aprender através dele.

À Fabiana, pelo amor, apoio, compreensão e incentivo ao longo desta jornada.

Aos Professores Marcio Barros, Guilherme Travassos, Ana Regina e Cláudia Werner por depositarem em mim um voto de confiança na seleção de candidatos para o curso. Espero não os ter decepcionado.

Em especial, ao Prof. Marcio Barros pela ajuda, orientação e conhecimentos adquiridos ao longo deste trabalho. Obrigado por me mostrar que sempre é possível melhorar, e que se quisermos chegar a algum objetivo, temos que, de alguma forma, aprender o caminho.

Em especial, ao Prof. Guilherme Travassos pelas valiosas críticas, por demonstrar o valor de Estudos Experimentais em Engenharia de Software e pela contribuição à minha formação.

Em especial, à Profª Ana Regina pela valiosa contribuição à minha formação, pelas críticas construtivas recebidas, pelo incentivo ao longo do curso e por fazer parte da banca examinadora da tese.

Ao Prof. Manoel Gomes de Mendonça Neto pela participação na banca examinadora da tese.

À minha chefia no CCA-RJ pelo apoio e compreensão durante o curso.

A todos os participantes do Estudo Experimental realizado, por aceitarem contribuir com sua experiência para a realização deste trabalho.

Às minhas amigas Lucia e Gladys pelo companheirismo ao longo do curso e pelos conselhos recebidos.

Ao Alexandre Teixeira pela ajuda e troca de conhecimentos.

A todos os amigos da COPPE que me ajudaram ao longo do curso.

ÍNDICE

1

INTRODUÇÃO

"There is always a certain risk in beeing alive.
If you are more alive, there is more risk".
Henrik Ibsen

1.1. Motivação para a tese

Desde que o fenômeno da crise do software foi instaurado no final dos anos sessenta, cada vez mais pesquisadores preocupam-se com a melhor forma de se desenvolver sistemas dentro do prazo, do orçamento e com a qualidade necessária.

Quase quarenta anos depois, questões levantadas à época continuam sem resposta e estudos das mais diversas ordens vêm buscando soluções ou explicações para amenizar os problemas que envolvem uma atividade tão complexa, instável e evolutiva como a Engenharia de Software.

Duas linhas de pensamento dividem as opiniões de especialistas como fatores responsáveis pelos insucessos industriais de desenvolvimento de software. A primeira linha associa as falhas a problemas tecnológicos, agravados pela crescente complexidade dos produtos desenvolvidos (AUGUSTINE, 1982). A segunda linha transfere à gerência a responsabilidade pelos fracassos de projetos (REEL, 1999).

A julgar pelo recente histórico da computação, dificilmente será possível deter o aumento da demanda por sistemas cada vez mais complexos e mais eficientes. No entanto, os defensores da linha de pensamento tecnológica acreditam que o desenvolvimento de novas tecnologias pode minimizar os problemas decorrentes do desenvolvimento destes sistemas.

Já em relação à gerência de projetos, pode-se dizer que há ainda um longo caminho a se percorrer para que projetos de software sejam gerenciados de maneira correta e eficaz. As técnicas de gerenciamento de projetos de software têm sua origem nas forças armadas americanas nas décadas de 50 e 60. Os conceitos básicos desse tipo de gerenciamento são semelhantes aos adotados por outras áreas de conhecimento, tais como a Engenharia e a Manufatura. No entanto, características

específicas da Engenharia de Software, tais como a variedade de domínios, a constante instabilidade dos requisitos, incertezas, ambigüidades, falta de padrões e dados históricos confiáveis, fazem com que algumas das técnicas de gerenciamento de projetos tradicionais sejam de difícil aplicação para projetos de software.

Dentre as diversas áreas de estudo da gerência de projetos, tais como o custo, o escopo e o cronograma, encontra-se a gerência de risco. A relevância dos riscos em projetos é tão grande que em 1994 um grupo de renomados gerentes de projetos foi contratado pelo Departamento de Defesa americano para aprimorar as técnicas de gerenciamento dos projetos de software contratados por esse departamento (BROWN, 1996). Dentre as várias sugestões elaboradas pelo grupo de trabalho, a que aparece em primeiro lugar é a adoção da gerência de risco nos projetos.

A importância dos riscos para projetos de software também é enfatizada em outros documentos tais como (ISO/IEC, 1999; ABNT, 2000; PMBOK, 2000). Portanto, entender a gerência de risco, os fatores que estão envolvidos nessa atividade, as diversas perspectivas e possibilidades dessa área de conhecimento é um grande passo para que gerentes sejam mais eficientes, os projetos possam ser conduzidos a seus objetivos de maneira mais controlada e possam ser entregues dentro do prazo, do orçamento e com a qualidade solicitada pelos clientes.

No entanto, para que um processo de gerência de risco seja eficiente são necessárias abordagens concretas para se quantificar os riscos de um projeto. Estimativas puramente subjetivas podem induzir os gerentes a erros, tornando o processo de gerência falho e não alinhado com os objetivos de uma organização.

Assim como em qualquer outro negócio, uma empresa de desenvolvimento de software visa, em última análise, aumentar seus lucros e gerar valor, o que faz com que a Engenharia de Software tenha que buscar auxílio em outras áreas de conhecimento para resolver problemas que não são exclusivamente técnicos, mas fazem parte do contexto do negócio como um todo.

Dentre estas áreas de conhecimento, uma das que mais tem apoiado engenheiros de software é a Economia, que visa, por essência, fornecer a empresários e investidores, meios eficientes de aumentar os lucros provenientes de um negócio. Para que isto possa acontecer, economistas buscam identificar e quantificar os riscos que estão envolvidos no negócio, pois estes são fundamentais para se avaliar a probabilidade de sucesso.

Assim, a união entre Economia e Engenharia de Software vem cada vez mais se estreitando, na busca de troca de informações e experiências, a fim de que tanto os

problemas técnicos como estratégicos envolvidos no contexto de um negócio, se tornem cada vez mais claros e passíveis de quantificação e análise a fim de apoiar os processos decisórios implícitos na gerência de projetos de desenvolvimento de software.

Vale ressaltar que decisões em projetos de software não são, em geral, tomadas apenas por aspectos de maximização econômica e técnica. Existem também questões políticas ou estratégicas para a manutenção da viabilidade do negócio, onde empresas, simplesmente não podem escolher se vão ou não realizar um projeto. As propostas apresentadas nesta tese tratam apenas de um destes aspectos, qual seja, a análise financeira global da carteira de projetos de uma empresa em relação a seus riscos.

1.2. Objetivos da tese

O primeiro objetivo desta tese é propor uma técnica para quantificar os riscos de projetos de software de acordo com uma visão econômica dos riscos envolvidos em um projeto de software. A proposta é fornecer a gerentes de projetos uma estimativa da probabilidade de falha do seu projeto. Para que este objetivo seja atingido, foi definida uma técnica de quantificação de riscos e um estudo experimental foi executado para apoiar a técnica proposta.

O segundo objetivo da tese é realizar a avaliação de uma carteira de projetos de uma empresa desenvolvedora de software, com o intuito de determinar a distribuição de probabilidade de perdas e ganhos da empresa em decorrência dos projetos componentes da carteira. Esta avaliação visa apoiar gerentes de projetos e empresários na identificação da combinação de projetos que leva a maximização da relação risco-retorno.

Para que os resultados gerados por esta tese possam ser efetivamente utilizados por gerentes de projetos, foi implementada a ferramenta *RISICARE* que apóia as abordagens propostas.

1.3. Organização do trabalho

Este trabalho está organizado em seis capítulos e dois anexos. O primeiro capítulo é composto por esta introdução. O segundo capítulo apresenta uma revisão bibliográfica sobre o estado da arte em gerência de risco em projetos de software. As fases do processo de gerência de risco são discutidas e atenção especial é dada ao levantamento dos riscos e aos métodos de avaliação de riscos.

No terceiro capítulo uma introdução à Economia aplicada à Engenharia de Software é apresentada. Diversos conceitos econômicos utilizados por engenheiros de

software e suas aplicações neste contexto são descritos, caracterizando o trabalho que vem sendo realizado por pesquisadores nesta área.

O quarto capítulo apresenta a técnica de avaliação de riscos em projetos de software proposta por esta tese, bem como o estudo experimental que foi planejado e executado para fornecer os dados necessários ao uso da técnica proposta. Resultados e considerações são apresentados para que repetições do estudo possam ser realizadas.

O quinto capítulo apresenta o conceito de risco de crédito e seus modelos de quantificação. Em seguida, é realizada uma analogia entre empréstimos financeiros e projetos de software e sugere-se uma forma de calcular as perdas e ganhos de uma carteira de projetos de software através de simulações. Ao final do capítulo, apresentamos a ferramenta *RISICARE* que foi concebida e implementada para apoiar a técnica de quantificação de risco descrita no capítulo 4 e o cálculo da distribuição de probabilidades de perdas e ganhos de uma carteira de projetos de software.

O sexto capítulo apresenta as considerações finais dessa tese, ressaltando suas contribuições, limitações e perspectivas futuras.

O Anexo 1 exibe o questionário de identificação de riscos elaborado durante a revisão bibliográfica. O Anexo 2 apresenta o formulário de caracterização do participante e do projeto utilizado no estudo experimental.

2

GERÊNCIA DE RISCOS EM PROJETOS DE SOFTWARE

"Se você não atacar os riscos de forma sistemática, eles irão atacá-lo"
Tom Gilb

2.1. Introdução

Dentre as diversas áreas reconhecidas como processos relevantes para a gerência de projetos descritas pelo PMI (*Project Management Institute)* encontra-se a gerência de riscos, que visa identificar potenciais problemas antes que eles ocorram, de forma que ações possam ser tomadas a fim de reduzir ou eliminar a probabilidade de ocorrência e o impacto desses problemas (IEEE Std 1540-2001, 2001). Normalmente, em um projeto, um risco é qualquer condição ou evento cuja ocorrência não é certa, mas que pode afetar negativamente o projeto caso ocorra.

PFLEEGER *et al.* (2001), por exemplo, sugerem três aspectos associados a um risco: (1) a perda associada ao evento; (2) a probabilidade de ocorrência do evento; (3) o grau com que se pode alterar as conseqüências do evento sendo definido como exposição ao risco, o produto da probabilidade de ocorrência pela perda associada ao evento.

Para que a exposição possa ser determinada, faz-se necessária uma abordagem quantitativa desses fatores. No entanto, de acordo com PFLEEGER *et al.* (2001), o cálculo da exposição ao risco representa mais uma arte que uma ciência pelo fato de não existirem técnicas eficientes para as estimativas quantitativas de impacto e probabilidade dos riscos (FARIAS, 2002).

HALL (1998) define a análise de riscos como um conjunto de procedimentos que, quando aplicados a um risco específico, fazem com que as conseqüências de sua concretização sejam aceitáveis em qualquer circunstância. GOLDMAN (1998) define a análise de riscos como um conjunto completo de normas e procedimentos que as organizações utilizam para gerenciar, monitorar e controlar sua exposição aos riscos.

No contexto de projetos de software, os riscos começaram a ser estudados no fim da década de 80, quando BOEHM (1989) elaborou um modelo para a gerência de riscos baseado no ciclo de vida espiral proposto em (BOEHM, 1988).

Na década de 90 diversos autores elaboraram estudos e propuseram formas diferentes de gerenciar o risco em projetos de software (CARR *et al.*, 1993; FAIRLEY, 1994; JONES, 1994; GREY, 1995; KAROLAK, 1996; KONTIO, 1997; MADACHY, 1997; HALL, 1998).

Apesar de todas as evidências de que a gerência de risco é importante para se alcançar o sucesso nos projetos de software, essa área ainda encontra obstáculos para ser institucionalizada pelas empresas (WIEGERS, 1999). Uma avaliação de melhoria de processos conduzida em 1999 na Austrália constatou que a gerência de risco é executada de forma planejada e com acompanhamento em somente 12% das empresas, 48% a executam de maneira informal e 40% não executam o processo (ROUT *et al.*, 2000). No Brasil, a pesquisa da Secretaria de Planejamento em Informática (SEPIN) sobre qualidade de software de 2001 demonstrou que a situação não é diferente, pois somente 11,43% das 446 pesquisadas executam gerência de risco (SEPIN, 2002).

A gerência de risco tem, na última década, tem ganho grande importância no contexto da Gerência de Projetos e nas normas e modelos de Engenharia de Software. O CMMI (*Capability Maturity Model Integration)* proposto pelo (SEI, 2002), por exemplo, estabelece a gerência de risco como requisito obrigatório para a PA (*Process Área*) de *Gerência de Software Integrada* no nível 3. Portanto, para que uma organização possa se enquadrar nesse nível de maturidade é necessário possuir e executar processos de gerência de riscos.

A emenda 01 da ISO/IEC 12207 (JTC1/SC7, 2001) também já inclui em seus processos a gerência de risco como um dos subprocessos da Gerência de Projetos. Versões mais antigas do CMM (*Capability Maturity Model*), da NBR ISO/IEC 12207 (Processos de Ciclo de Vida de Software) e da ISO 9000 (Gestão da Qualidade) não enfatizavam a gerência de risco ou distribuíam suas atividades em outros processos (HUMPHREY, 1987), (PAULK *et al.*, 1993), (ABNT, 1998). Pode-se observar o mesmo comportamento com o PMBOK (*Project Management Body of* Knowlege), onde, na versão atual (PMI, 2000), o processo de gerência de risco foi ampliado em relação à versão anterior (PMI, 1996).

Para se justificar a gerência de riscos nos projetos, basta que se observe os três aspectos básicos sobre o assunto, compilados pelo *Software Engineering Institute* (SEI, 2003).

- O Gerenciamento de Riscos provoca uma mudança cultural na equipe de projeto, pois esta deixa de ser reativa (Gerenciamento da Crise) para ser pró-ativa (Gerenciamento do Risco), evitando-se os problemas antes que eles surjam. A antecipação daquilo que pode dar errado passará a fazer parte do dia-a-dia da equipe.

- As conseqüências negativas de não se implantar uma Gerência de Riscos são os recursos gastos com problemas que poderiam ter sido evitados, surpresas desagradáveis que podem ocorrer, decisões tomadas sem uma real noção das conseqüências futuras e a redução na probabilidade de sucesso do projeto.

- A Implementação da Gerência de Riscos não garante o sucesso dos projetos. No entanto, ela melhora o processo decisório e ajuda a evitar surpresas, aumentando assim as suas chances de sucesso.

Este capítulo fornece uma visão geral sobre o estado da arte em gerência de riscos em projetos de software. Na seção 2.2 desse capítulo são tratadas as fases de um processo de gerência de risco genérico baseado na literatura de engenharia de software. Na seção 2.3 aspectos mais específicos da identificação de risco são discutidos. Na seção 2.4 são apresentados os principais questionários utilizados na identificação de riscos e publicados na literatura. Considerações finais são elaboradas na seção 2.5.

2.2. As fases da gerência de risco

A gerência de riscos deve estar presente nas cinco fases de um projeto citadas pelo PMBOK (2000) - Iniciação, Planejamento, Execução, Controle e Fechamento – visto que as incertezas percorrem todas as fases do projeto e só terminam com a sua conclusão.

Muitas abordagens são citadas na literatura para o gerenciamento de risco. BOEHM (1991), por exemplo, sugere um processo formado por duas etapas principais: avaliação e controle. A avaliação é constituída pela identificação, análise e priorização dos riscos, ao passo que o controle inclui o planejamento, a resolução e monitoração de riscos.

O SEI (*Software Engeneering Institute*) utiliza um paradigma de gerência de risco composto de seis atividades: identificar, analisar, planejar, monitorar, controlar e

comunicar. A figura 2.1, representada como um círculo, enfatiza o caráter contínuo do processo de gerência de risco enquanto que as setas mostram o fluxo lógico e temporal das informações. A atividade de comunicação está colocada no centro do paradigma, pelo fato de ser o conduíte para o fluxo das informações e por representar, freqüentemente, o maior obstáculo à gerência de risco (CARR *et. al.*, 1993).

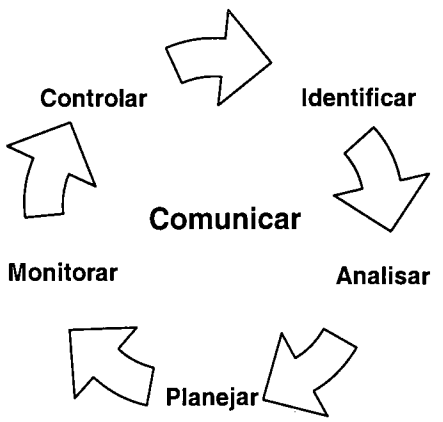


Figura 2.1 – Paradigma de gerenciamento de risco do SEI (CARR *et. al.*, 1993)

O padrão IEEE Std 1540-2001 de processos de gerência de riscos em projetos de software foi desenvolvido para ser adaptado às necessidades das organizações e situações específicas de projetos. O processo é composto das seguintes atividades: planejamento e implementação da gerência de risco, gerenciamento do perfil de risco do projeto, análise de risco, monitoração do risco, tratamento do risco e avaliação do próprio processo de gerência de risco.

De forma genérica, um processo de gerenciamento de risco pode ser dividido em quatro grandes etapas: identificação, análise, planejamento e controle, sendo que, dependendo da abordagem utilizada, essas etapas podem ser caracterizadas como uma atividade, um grupo de atividades, uma tarefa ou mesmo um processo isolado.

As próximas seções descrevem mais detalhadamente cada uma dessas quatro etapas da gerência de risco.

2.2.1. Identificação

Identificar riscos em um projeto de software significa levantar todas as situações que podem ameaçar o sucesso de um projeto e as oportunidades que devem ser buscadas para o aprimoramento da equipe, da organização ou do projeto como um todo. No contexto dessa tese, apenas os aspectos negativos do risco serão considerados.

O objetivo de identificar os riscos desde o início de um projeto é ter uma postura pró-ativa perante os problemas que podem afetar algum aspecto do projeto e tornar o gerente capaz de preparar planos de contenção ou contingência para controlar os riscos.

Segundo CARR *et al.* (1993), os riscos podem ser classificados como conhecidos, desconhecidos e não identificáveis. Os conhecidos são aqueles que um ou mais membros da equipe de projeto têm consciência. Os desconhecidos são os que podem se tornar conhecidos se as condições, oportunidades e informações necessárias forem dadas à equipe. Os não identificáveis são aqueles que, mesmo com o auxílio de métodos e técnicas de identificação, não são possíveis de serem previstos e, portanto, podem constituir sérios danos ao projeto.

A identificação de riscos deve ser uma atividade contínua e permanente pelo fato das incertezas estarem presentes durante todas as fases de realização de um projeto. Uma mentalidade de gerenciamento de risco e canais livres de comunicação para todos os interessados no projeto devem ser criados para que qualquer pessoa possa trazer à tona eventuais riscos que possa identificar durante a realização de suas atividades, mesmo não fazendo parte da equipe de gerência do projeto.

Após a identificação dos riscos de um projeto, esses devem ser documentados para que possam ser analisados futuramente. Normalmente essa documentação é feita através de formulários ou documentos padronizados que podem, inclusive, formar um repositório ou um banco de dados sobre riscos de projeto. WILLIANS *et al.* (1997), GALLAGHER *et al.* (1997) e HALL (1998) apresentam exemplos de formulários para identificação e acompanhamento de riscos em projetos de software.

Outra possibilidade para a documentação dos riscos é a criação de padrões, onde através de uma estrutura comum, as atividades de representação, leitura e disseminação de informações podem ser facilitadas para todos os membros de uma equipe. JONES (1994) apresenta uma estrutura padronizada para a documentação de riscos e fornece vários exemplos baseados nessa estrutura.

BARROS (2001) sugere a criação de arquétipos para a identificação e documentação de riscos de software. Um arquétipo de risco é uma descrição de um problema potencial, recorrente ao longo de diversos projetos, cuja ocorrência pode causar prejuízos a um produto ou processo de desenvolvimento de software. Ele inclui uma descrição do contexto em que o problema pode ocorrer e apresenta estratégias que podem ser aplicadas para solucionar o problema potencial antes ou depois de sua ocorrência.

2.2.2. Análise

Com base na lista de riscos gerada na atividade de identificação, uma avaliação dos potenciais riscos do projeto deve ser realizada. Essa atividade divide-se em três partes. A primeira parte tem como objetivo estimar a probabilidade de ocorrência de um determinado risco. A segunda parte visa determinar o impacto que a ocorrência do risco pode causar ao projeto e a terceira parte busca definir a prioridade com que os riscos devem ser tratados.

A probabilidade de ocorrência de um risco e o seu impacto podem ser descritos qualitativa ou quantitativamente. Estimativas qualitativas estão baseadas em uma observação subjetiva e geralmente assumem a forma de termos em linguagem natural. Estas estimativas podem variar desde medidas booleanas, onde apenas dois termos são válidos ("sim" e "não"), (JALOTE, 1999) até medidas mais refinadas, onde uma avaliação mais detalhada pode ser realizada ("moderado", "pouco provável", "muito provável", etc). O uso da linguagem natural facilita a expressão e a comunicação do conhecimento da equipe de desenvolvimento, visto que sua experiência não está expressa em números (PRESSMAN, 2000).

Essas informações qualitativas podem ser convertidas em dados quantitativos através da criação de escalas ou tabelas de conversões. Exemplos de técnicas de quantificação incluem o mapeamento de termos de linguagem natural em intervalos numéricos ou números *fuzzy* (ZADEH, 1988).

Avaliações quantitativas são expressas por números, funções ou expressões matemáticas e são particularmente interessantes quando existem informações de projetos passados (BARROS, 2001). Através de procedimentos estatísticos, as medidas podem ser extraídas destas informações, dissociando-se a análise de riscos de opiniões pessoais (NBR ISO 10006, 2000).

As técnicas quantitativas são desejáveis por oferecerem medidas objetivas de impacto dos riscos, mas também possuem suas limitações. GEMMER (1997) observa que a conotação de um número ou de um termo qualitativo pode variar entre diferentes pessoas. A diferença de interpretação pode levar o analista por uma linha de raciocínio distinta da planejada, incorrendo em um possível erro. O autor ressalta que as evidências e o contexto do risco são mais relevantes que os números associados a suas características.

Embora sofram das limitações apresentadas por GEMMER (1997), considera-se que a análise de riscos deve prezar as técnicas quantitativas, devido a sua objetividade.

Entretanto, por sua abrangência, as técnicas qualitativas não devem ser descartadas, podendo auxiliar na identificação dos riscos (BARROS, 2001).

Diversos métodos de avaliação de risco podem ser encontrados na literatura. FAIRLEY (1994) também utiliza modelagem estatística em seu processo de análise de riscos. Cada fator de risco é modelado como uma função de distribuição de probabilidade e simulações de Monte Carlo são utilizadas para calcular a exposição agregada a risco de um projeto.

GREY (1995) apresenta um processo de avaliação baseado na estrutura de decomposição de um projeto – *work breakdown structure* (WBS) e nas relações de dependência entre suas tarefas, definidas através de redes de *PERT/CPM* – para modelar o risco financeiro e de cronograma de um projeto. Utilizando Simulações de Monte Carlo (VOSE, 1996), o custo e o tempo necessários para a realização de cada tarefa da WBS são modelados por funções de distribuição de probabilidade.

O *Just-in-Time* (KAROLAK, 1996) trabalha somente em relação à probabilidade, não focando o impacto. Através de respostas a um questionário onde a percepção de valores é dada pelo respondente, calcula-se a média aritmética de todas as respostas e, desta forma, a probabilidade é determinada. Esta probabilidade é definida em função de pesos arbitrados pelo respondente (alto médio ou baixo) desde que o somatório assuma sempre o valor 1.

KÄNSÄLÄ (1997) apresenta o processo de análise de riscos *RiskMethod*, onde cada risco é associado a uma probabilidade de ocorrência e uma magnitude de impacto. A probabilidade é representada por um número entre zero e um. A magnitude do impacto é representada por um número em uma escala definida para o risco. Quando diversos resultados distintos são possíveis para um mesmo risco, cada resultado é associado com uma probabilidade e uma medida de impacto. O valor de um risco é calculado pelo somatório dos produtos das probabilidades dos riscos por suas medidas de impacto.

A última parte da fase de avaliação é a priorização dos riscos. Aqui, o objetivo é determinar quais os riscos que devem ser tratados segundo um critério utilizado para essa priorização. A justificativa para essa atividade é o fato de que, em princípio, pode não haver recursos financeiros para que todos os riscos sejam tratados. Dessa forma, apenas os riscos considerados prioritários deverão ser levados para as próximas fases do processo de gerência de riscos.

Os critérios mais utilizados para a priorização de riscos são a exposição e a severidade. A severidade é representada pela exposição dividida pelo tempo estimado

para a ocorrência do evento. Assim, quanto maior for a exposição ao risco ou menor o tempo para que o evento ocorra, maiores serão os danos causados por um determinado risco e, portanto, precisam ser analisados mais detalhadamente. No entanto, esses critérios são discutíveis e pouco precisos pelos motivos explanados anteriormente (PFLEEGER *et al.*, 2001).

Pode-se encontrar na literatura algumas técnicas de priorização de riscos. Entre elas pode-se citar o ranqueamento, o ranqueamento parcial (GALLAGHER *et al.*, 1997), a diversificação e a multivotação (HALL, 1998).

BARROS (2001) considera que mais importante que o método de priorização são os mecanismos utilizados para a determinação da relevância de um risco. Os relacionamentos complexos que podem existir entre os componentes de um projeto de software podem esconder situações onde as causas e os efeitos estão distantes no tempo. Assim, um risco considerado irrelevante, dado seu pequeno impacto a curto prazo, pode provocar um grande efeito no projeto a longo prazo. Os mecanismos de avaliação de riscos devem preconizar uma análise holística do impacto dos riscos, focalizando detalhadamente seus efeitos sobre os projetos.

2.2.3. Planejamento

Após a identificação e a priorização dos riscos, a próxima etapa do processo é definir que procedimentos serão adotados diante dos riscos que foram considerados prioritários. O planejamento do risco tem como objetivo estabelecer ações para minimizar ou eliminar os efeitos causados por estes riscos. Esses procedimentos, razões e decisões tomadas devem ficar registrados em planos de riscos, para que desenvolvedores e clientes possam estar cientes de suas responsabilidades e atribuições quando do gerenciamento dos riscos.

Dois tipos de planos podem ser realizados: planos de contenção e planos de contingência. Os planos de contenção têm como objetivo evitar que um risco ocorra ou reduzir o dano que o risco provocaria, sendo executados antes da ocorrência do risco. Os planos de contingência são elaborados para minimizar os danos causados por um risco depois que este se realiza e, portanto, são executados após a ocorrência dos riscos. Durante a elaboração desses planos, limites devem ser determinados para que haja uma prévia definição de quando os planos devem ser colocados em ação.

A elaboração desses planos depende da quantidade de recursos disponíveis para o projeto, da estratégia utilizada para o seu gerenciamento, da política organizacional e do tipo de projeto que se está realizando.

Pode-se encontrar na literatura diversas estratégias adotadas para o tratamento dos riscos (HALL, 1998), (PFLEEGER *et. al.*, 2001). KERZNER (2001), por exemplo, divide essas estratégias em quatro ações possíveis:

- Assumir – é a consciência da existência do risco e a decisão de aceitá-lo, bem como suas conseqüências sem nenhuma medida preventiva;
- Evitar – consiste em eliminar a ameaça, normalmente eliminando sua causa, pois os níveis de riscos são inaceitáveis;
- Controlar – são as ações necessárias para minimizar ou mitigar a probabilidade de ocorrência ou o impacto que o risco terá no projeto;
- Transferir – é o ato de dividir o evento de risco com outros interessados no projeto, repassá-lo completamente ou colocá-lo de alguma forma sob seguro ou garantia.

A norma NBR ISO 10006 (2000) alerta que a definição de estratégias de tratamento dos riscos deve ser baseada em tecnologias conhecidas ou em experiências anteriores, evitando assim a introdução de novos riscos no projeto.

FARIAS (2002) define uma abordagem para o planejamento de riscos em projetos de software baseada na reutilização do conhecimento organizacional sobre riscos. A abordagem fundamenta-se nos conceitos de Gestão de Conhecimento e Ambientes de Desenvolvimento de Software Orientados à Organização.

Inicialmente foi definido um processo de gerência de riscos que captura e utiliza os conhecimentos organizacionais de riscos nas diversas atividades do processo. Visando identificar fatos causadores de risco, os riscos comumente encontrados em projetos de software e possíveis relações entre estes fatos e riscos, foi realizado um estudo experimental com gerentes de projetos, para que este conhecimento fosse utilizado no processo, que é apoiado por uma ferramenta.

A abordagem disponibiliza aos gerentes o conhecimento acumulado por vários gerentes de uma organização, considerando projetos anteriores similares. Estes conhecimentos são utilizados nas atividades de identificação, análise e planejamento da gerência de riscos, sendo que a estratégia para identificação de riscos baseia-se em três

pilares: conhecimentos de riscos de projetos similares, um *checklist* de fatos causadores de risco e a experiência pessoal do gerente de projeto.

2.2.4. Controle

O objetivo da fase de controle é o acompanhamento dos riscos dos projetos, bem como a verificação da necessidade de alteração dos planos de tratamento de riscos, pois a percepção que se tem dos riscos pode mudar ao longo do projeto e algumas alterações podem ser necessárias (JALOTE, 1999).

Esse controle deve ser realizado periodicamente e durante todo o projeto. JALOTE (1999), por exemplo, sugere que revisões dos planos de riscos ocorram nos marcos do projeto. Já MURCH (2000) sugere que sejam feitas reuniões semanais ou até mais freqüentes, caso seja necessário.

Durante o acompanhamento dos riscos, o objetivo mais importante é a verificação dos limites impostos para a aplicação dos planos de contenção e contingência elaborados na fase de planejamento. Caso esses limites sejam atingidos ou estejam prestes a serem atingidos, os planos devem ser executados a fim de mitigar os possíveis danos.

Quando um risco for detectado, um acompanhamento mais efetivo deverá ser realizado até que o risco seja eliminado ou reduzido a níveis aceitáveis. Em caso de falha dos planos de contenção ou contingência, deve-se avaliar a possibilidade de realocação de recursos de outras áreas do projeto para que a crise seja minimizada.

Todas as observações e discrepâncias em relação ao planejamento devem ser registradas para auxiliar planejamentos futuros. WILLIAMS *et al.* (1997) ainda ressaltam que quando um risco tem suas atividades encerradas todas as informações relacionadas devem ser documentadas para auxiliar novos planejamentos: justificativas de encerramento, ações resultantes em sucesso ou fracasso, suposições errôneas, custos da mitigação e retorno do investimento.

2.3. As técnicas de identificação de risco

CARR *et al.* (1993) observam que sem métodos efetivos e contínuos de identificação de riscos é impossível realizar uma gerência de risco eficiente.

Existe uma enorme variedade de técnicas para o levantamento de risco que, de forma geral, podem ser utilizadas dentro de diversos contextos de projeto de software:

- ***Checklists*:** baseado em informações históricas e conhecimentos acumulados de projetos anteriores cria-se uma lista de possíveis fontes de risco (PRITCHARD, 1997). Uma das vantagens de se usar os *checklists* é que a identificação de riscos é feita de maneira rápida e simples, mas corre-se o risco do *checklist* não ser completo e riscos importantes não serem identificados pelo fato de não constarem da lista utilizada;

- **Questionários:** são uma extensão dos *checklists*, pois além de possibilitarem a identificação de riscos através de perguntas, fornecem métodos para a discussão dos riscos levantados e são usados como guias nas entrevistas;

- **Comparação por Analogia:** esse método prevê a identificação de projetos similares, de modo que riscos identificados anteriormente possam servir de base para a elaboração ou mesmo a revisão dos riscos de projetos atuais. A identificação de projetos similares envolve a determinação de características comuns aos projetos, como o tipo de tecnologia, funcionalidade, estratégia de contrato e processo de desenvolvimento (PRITCHARD, 1997);

- **Diagrama de Causa e Efeito:** também conhecido como Diagrama de Ishikawa ou Espinha de Peixe, é útil para identificar as causas dos riscos. A proposta da análise de causa e efeito é que se um erro ocorreu, ele irá acontecer novamente, a menos que alguma providência seja tomada (HALL, 1995);

- **Técnica *Delphi*:** um grupo de especialistas liderados por um facilitador gera uma lista de riscos sobre um determinado assunto de maneira individual e anônima. O facilitador consolida a lista e a expõe para os especialistas, que têm a oportunidade de revisar ou complementar a lista de riscos;

- ***Brainstorming*:** técnica utilizada para um levantamento rápido, porém não-estruturado dos riscos. Os participantes da reunião, liderados por um facilitador e auxiliados por um secretário geram uma lista de riscos sem realizar nenhuma espécie de questionamento sobre a razão, causa ou efeito desses riscos. A regra a ser seguida é o não questionamento dos motivos que levaram à identificação

desses riscos. Futuramente essa lista de riscos deverá ser analisada e aperfeiçoada;

- **Geradores de Custos:** motivados pelos constantes erros nas estimativas de custo de projetos, alguns autores criaram técnicas de identificação de riscos a partir de geradores de custos. MADACHY (1997) apresenta uma técnica de identificação e avaliação de riscos baseada em padrões de respostas ao questionário de avaliação dos geradores de custos do método COCOMO2 (BOEHM *et al.*, 1995). O princípio desta técnica é que uma situação de risco pode ser descrita como uma combinação de valores extremos para um conjunto de geradores de custo, indicando que uma margem de esforço foi reservada para a resolução de problemas potenciais nesta área. O autor argumenta que as práticas de análise de riscos podem ser aprimoradas, explorando-se o conhecimento utilizado durante o processo de estimação de custos.

Outras técnicas tais como Entrevistas com especialistas e protótipos (MACHADO, 2000), *Slip de Crowford*, NGT (*Nominal Group Technique*) (SALLES, 2003), 6W (*who, why, what, whichway, wherewithal e when*) (CHAPMAN e WARD, 1970), também podem ser usadas dependendo do contexto e das necessidades do projeto.

Durante o processo de criação de listas de riscos deve-se atentar para algumas observações (SALLES, 2003):

- Ser o mais específico e claro possível;
- Não incluir perguntas nas descrições dos riscos;
- Não incluir itens de ação;
- Formular o texto do risco e o potencial impacto.

FAIRLEY (1994) comenta ainda que, deve-se distinguir os riscos dos efeitos gerados por eles. Por exemplo, um aumento nos custos pode ser o efeito de uma falha na definição de atividades e conseqüente falta de planejamento adequado. Nesse exemplo, o risco é a falha na definição das atividades e não o aumento do custo.

A criação de categorias e de taxonomias para fontes de risco é importante na etapa de identificação, pois divide um corpo de conhecimento e define as relações entre as partes (IEEE 1002, 1987). O particionamento faz com que a identificação das fontes de

risco seja facilitada, pois o responsável por identificar os riscos concentra seu esforço em um aspecto do projeto de cada vez, além de facilitar a inserção de novos riscos que não estejam contemplados, através da associação por similaridade de assunto. Outra razão importante para a categorização de riscos é o fato de que dependendo do grupo a que um risco pertence, a forma de atacá-lo será diferente.

Diversas são as categorias criadas pelos autores na área de Engenharia de Software (CARR *et al.*, 1993; BARKI *et al.*, 1993; SCHMIDT *et al.*, 1996; KAROLAK, 1996; HALL, 1998; WALLACE, 1999; PERSSMAN, 2000). Tabelas descritivas dessas categorias podem ser encontradas em (BARROS, 2001).

2.4. Questionários de risco

De acordo com McFARLAN e McKENNY (1996), falhas na avaliação dos riscos são uma das maiores fontes de problemas em desenvolvimento de sistemas de informação. Conseqüentemente, muitos estudos têm pretendido identificar esses fatores de risco (BARKI *et al.*, 1993; JONES, 1996; SCHMIDT *et al.*, 1996; KÄNSÄLÄ, 1997; MOYNIHAN, 1997; WALLACE, 1999; ROPPONEN & LYYTINEN, 2000 e JIANG *et al.*, 2000).

Os modelos de estimativas foram os primeiros a trabalhar com fatores de risco, tais como, COCOMO - Modelo de Custo Construtivo e a FPA - Análise de Ponto por Função. Para obter os fatores de ajustes do modelo COCOMO, foi avaliado um banco de dados de 63 projetos, onde foram encontrados mais de 100 fatores, os quais, após um processo de seleção, foram reduzidos a 15.

O primeiro estudo sobre fatores de risco fora do contexto dos modelos de estimativas foi conduzido por BOEHM (1991), quando foi produzida uma lista dos 10 fatores de risco mais significativos no desenvolvimento de software. BARKI *et al.* (1993) conduziram outro estudo, baseado no estudo de Boehm, em estudos de outros autores e em pesquisas de campo, onde foram encontrados 35 fatores de risco.

CARR *et al.*, (1993) através de uma pesquisa conduzida pelo SEI (*Software Engineering Institute*), disponibilizaram o TBQ (*Taxonomy Based Questionnaire*), composto de 194 questões e passível de utilização em vários domínios de aplicação.

MOYNIHAM (1997), realizando uma pesquisa com 14 gerentes de projetos, levantou 113 fontes de riscos que foram reduzidas posteriormente a 21. JONES (1994) através de pesquisa de campo e análise estatística listou 60 fatores de risco e criou seis

classes distintas de software onde esses fatores podem ser enquadrados (Sistemas de Informação, Softwares de Sistemas, Comerciais, Militares, Contratos e Usuários Finais).

WALLACE (1999) relacionou 23 fontes de riscos através de pesquisa na literatura e entrevistas com mais de 500 gerentes de projetos. Já KAROLAK (1996), separa as 81 fontes de risco que encontrou na literatura em 10 grandes áreas para auxiliar em seu processo de avaliação de risco.

Observa-se, contudo, que nenhuma das listas é completa e diferem em seu conteúdo e abrangência. Barki *et al.* (1993), por exemplo, estudou fatores relacionados ao ambiente organizacional, inovação tecnológica, expertise, tamanho e complexidade da aplicação. Wallace não contemplou a categoria de processo de desenvolvimento, aspecto fundamental na engenharia de software que Schmidt, SRE e Karolak contemplaram. Jones, por sua vez, volta seu foco para a área gerencial, sem, contudo, deixar de observar os aspectos técnicos.

O que se buscou nessa tese foi analisar e agrupar questionários já consolidados na literatura para se obter uma lista de riscos o mais completa possível, mas é sabido que outras fontes de risco existentes num contexto de desenvolvimento de software podem não estar incluídas no questionário final. MOYNIHAM (1997) afirma ser irreal a probabilidade de se construir uma taxonomia de riscos que contemple todos os aspectos relativos a riscos em desenvolvimento de software e que o ideal, talvez, seja construir uma taxonomia específica para cada contexto.

2.5. Conclusões

Algumas áreas de conhecimento como a Economia, a Engenharia e a Medicina já estudam os riscos presentes em seus projetos há muitos anos e possuem processos e técnicas bem estabelecidas para seu gerenciamento.

No entanto, pode-se dizer que a Engenharia de Software ainda está dando seus primeiros passos no intuito de estabelecer regras e padrões aceitáveis de gerenciamento riscos. Muitas são as razões para que isto esteja ocorrendo, mas talvez o principal motivo seja a forte influência do fator humano no processo de desenvolvimento de um software.

Para que um gerenciamento de risco possa ser realizado eficientemente, é necessário que uma atitude pró-ativa seja adotada e que um ambiente e uma mentalidade de risco seja cultivado, criando canais livres para que todos participem do processo. Assim como na garantia de qualidade, o gerenciamento do risco é de responsabilidade de todos os envolvidos no projeto e se assim não o for considerado, antes mesmo de se

iniciar o projeto, o primeiro risco de um projeto já está presente e tem-se grande chance de não alcançar os objetivos pretendidos.

Esse capítulo tratou da gerência de riscos em projetos de software, ressaltando os principais tópicos envolvidos nessa atividade, especialmente na etapa de identificação. Algumas das formas com que os autores categorizam os riscos também foram apresentadas. Técnicas de levantamento de risco foram citadas e um breve resumo dos principais questionários de risco encontrados na literatura foi apresentado.

3

ECONOMIA APLICADA A ENGENHARIA DE SOFTWARE

"Tomar decisão é descobrir o que fazer quando é incerto o que acontecerá".
Hacking

3.1. Introdução

Tradicionalmente, a Engenharia de Software é tratada como um desafio técnico e pouca atenção tem sido voltada para o seu contexto econômico (ERDOGMUS *et al.*, 2004). Esta preocupação seria justificada pelos números apresentados em pesquisas, onde se observa que o setor comercial de software fatura entre 200 e 240 bilhões de dólares por ano nos Estados Unidos. Acrescenta-se a isso a estimativa de crescimento econômico mundial da ordem de 1 trilhão de dólares entre 1995 e 2004 envolvendo o setor de Engenharia de Software (BOEHM E SULLIVAN, 1999).

No entanto, mesmo diante de números tão expressivos, a Engenharia de Software continua sendo um ramo complexo e problemático da tecnologia moderna. Um dos sintomas mais claros deste contexto é a alarmante quantidade de projetos que fracassam. De acordo com estudos do STANDISH GROUP (2004) cerca de 55 bilhões de dólares foram gastos nos Estados Unidos no ano de 2004 com projetos de software que falharam.

Portanto, não há como negar que a Engenharia de Software faz parte de um contexto financeiro e é uma atividade econômica como qualquer outra, onde, além dos benefícios introduzidos pelos sistemas, empresas buscam ampliar seus lucros, aumentar a expectativa de ganhos futuros ou minimizar prejuízos em um mercado dinâmico, cada vez mais competitivo e repleto de incertezas.

Neste cenário, encontramos engenheiros de software cujo principal objetivo é a tomada de decisões técnicas em prol do desenvolvimento de melhores produtos. Por outro lado, têm-se gerentes e empresários buscando agregar valor a seus investimentos. BOEHM E SULLIVAN (1999) comentam que, normalmente, os engenheiros de software não estão envolvidos ou mesmo não compreendem os aspectos gerenciais de criação de valor para a empresa. Entretanto, gerentes executivos comumente desconhecem os critérios de sucesso para o desenvolvimento de software ou como investimentos em áreas técnicas podem contribuir para a geração de valor.

Surge então, um ramo de estudo denominado Economia aplicada à Engenharia de Software, cujo campo de atuação situa-se na interseção entre ciências econômicas e a Engenharia de Software, tendo como objetivo o estudo de como alocar recursos (financeiros, materiais e pessoais) de maneira eficiente a projetos de software. Seu maior foco é aproximar conceitos e compreender relacionamentos entre questões técnicas de desenvolvimento de software e objetivos econômicos, visando a criação de valor agregado nos mais diversos níveis: projetos, programas e empresas.

Segundo (BOEHM e SULLIVAN, 1999) a Economia aplicada a Engenharia de Software é definida como o campo de estudo que visa fornecer melhorias à Engenharia de Software por meio de raciocínio e justificativas econômicas sobre questões relacionadas a produtos, processos, carteiras de projetos e políticas organizacionais.

A origem do estudo econômico relacionado a Software data dos anos 1960 e seus objetivos originais eram a propaganda e busca por melhores preços (STIGLER, 1962), investimentos em pesquisa e desenvolvimento (ARROW, 1962) e economia global de uma empresa (MACHLUP, 1962).

A primeira aplicação específica para assuntos de computação foi a *The Economics of Computers* (SHARPE, 1969) que tratava de assuntos tais como escolha entre compra, *leasing* ou aluguel de computadores, apreçamento de serviços de computação e economia de escala em sistemas computacionais. Possuía também uma pequena seção sobre custos de software, baseado na primeira grande abordagem nesse sentido, desenvolvida para a Força Aérea Americana pela *System Development Corporation* - SDC (NELSON, 1966). Esse estudo possibilitou a criação de uma série de modelos de custos de software, tais como o SLIM (PUTNAM, 1978), o PRICE (FREIMAN e PARK, 1979) e o COCOMO (BOEHM, 1981). No estudo, o autor sumariza os principais conceitos e técnicas de microeconomia e fornece exemplos de como aplicá-los ao processo de decisão em Engenharia de Software.

Muitas das dúvidas e problemas referentes ao sentido econômico da Engenharia de Software necessitam apoio de outras áreas de conhecimento, tais como a Teoria da Decisão (RAIFFA, 1968), Matemática (KEENEY, 1993) e Finanças Corporativas (BREALEY e MYERS, 1996), que fornecem auxílio quando se trata de tomada de decisões sob incerteza, aversão a riscos e busca de investimentos voltados a lucro. Com base em vários conceitos dessas teorias, engenheiros de software vêm, desde a última década, buscado auxílio para o desenvolvimento de pesquisas e introdução de

abordagens que, apesar de ainda necessitarem maiores estudos para sua comprovação, fornecem uma direção para futuros trabalhos.

O objetivo deste capítulo é fornecer um panorama de como alguns conceitos de Economia podem ser aplicados à Engenharia de Software e mostrar os benefícios que a união entre estas duas áreas de conhecimento pode trazer, tanto para a melhoria dos produtos de software quanto para a geração de valor econômico para uma empresa.

Está fora do escopo desta tese discutir a validade dos estudos, os dados, resultados e parâmetros utilizados pelos autores para suas abordagens. Vale ainda ressaltar que, todos os exemplos citados foram retirados dos artigos dos próprios autores e que representam suas opiniões e experiências em relação aos assuntos que serão tratados.

Este capítulo, após a introdução, descreve nas próximas seções várias aplicações de conceitos econômicos que vêm sendo utilizados por engenheiros de software. Na seção 3.2 discute-se o conceito de Valor Presente Líquido. Na seção 3.3 apresenta-se a Teoria do Portfolio. Na seção 3.4 é introduzido o conceito de Opções Reais. Finalmente, realizamos alguns comentários finais na seção 3.5.

3.2. Valor presente líquido (VPL)

Entre os diversos modelos utilizados para avaliação de negócios, o Fluxo de Caixa Descontado (FCD) é tido como aquele que melhor demonstra a efetiva capacidade de geração de valor na ausência de incerteza sobre os valores observados no futuro (MARTINS, 2001).

Um dos modelos mais utilizados de FCD é o Valor Presente Líquido (VPL), sendo este um dos principais conceitos econômicos utilizados por engenheiros de software em suas abordagens econômicas. O VPL de um projeto ou investimento é definido como sendo a diferença entre o valor de mercado deste investimento e seu custo na data de hoje (ROSS *et al.*, 1998). Em outras palavras, o VPL é uma medida de quanto valor é criado ou adicionado hoje, realizando-se um investimento. Caso a diferença entre o valor gerado no futuro e o custo de um projeto seja positiva, este será um bom investimento.

O objetivo é calcular quanto vale, no momento presente, a soma do valor de todos os fluxos de caixa, positivos e negativos e compará-la com os custos do projeto. Vale ressaltar que esta transposição de tempo em relação ao capital investido leva em consideração, principalmente, as taxas de juros e as correções do dinheiro ao longo do período, pois o valor do dinheiro é alterado ao longo do tempo e, portanto, é necessário

realizar a comparação como se todos os fluxos ocorressem em um único instante, a fim de se ter uma exata noção de seus valores. Assumindo que todos os custos de um projeto são incorridos em sua data de início, tem-se que o VPL é calculado como VP – I_0, sendo I_0 o investimento inicial no projeto e VP calculado segundo a fórmula abaixo, onde *i* é o número de intervalos de tempo do investimento, VF é o valor recebido no futuro (no tempo *i*) e *k* a taxa de juros.

$$VP = \sum_{i=1}^{n} \frac{VF}{(1+k)^i}$$

Caso o fluxo de caixa seja incorrido à medida que um projeto se desenvolve, ou seja, os custos e as receitas ocorram de forma parcelada ao longo do tempo, o VPL é calculado pela fórmula abaixo, onde *R* representa as receitas e *C* os custos.

$$VPL = \sum_{i=1}^{n} \frac{R_i}{(1+k)^i} - \sum_{i=1}^{n} \frac{C_i}{(1+k)^i}$$

Embora seja um instrumento extremamente valioso e comprovadamente útil em Economia, o cálculo do VPL, segundo FAVARO *et al.*, (1998) foi originalmente concebido para avaliar instrumentos financeiros tais como ações e bônus que possuem uma natureza passiva e pouco se pode fazer para mudar esta natureza. No entanto, em um contexto de projetos de software, gerentes possuem opções de escolha e, desta forma, podem intervir para alterar o curso dos acontecimentos. Assim, no que se refere a contribuição para a geração de valor a um negócio, tais como flexibilidade ou oportunidade estratégica, a VPL tem pouco a oferecer e, portanto modelos mais complexos são necessários.

3.2.1. VPL associado a métricas

Uma segunda abordagem que utiliza o conceito de VPL é a proposta por ERDOGMUS (1999). O autor destaca o fato de que apesar do VPL ser um valioso instrumento de avaliação econômica de projetos, ele não é suficiente para o contexto de software, pois existe a necessidade de melhor compreender os fatores que afetam o VPL em um ambiente de múltiplas alternativas incertas.

O autor propõe, então, uma abordagem onde um projeto de software é considerado como uma atividade de alocação de capital orientada por métricas. A partir de duas possíveis alternativas de estratégias de desenvolvimento, o VPL do projeto é expresso em função de seis variáveis de alto nível. A escolha da estratégia afeta as

estimativas destas variáveis e, assim, decisões podem ser tomadas de maneira mais segura.

As variáveis identificadas pelo autor são: o tempo de desenvolvimento (T), o custo de desenvolvimento no tempo zero (I), futuros valores gerados (C), custos operacionais em um tempo T qualquer (M), a contribuição, sob incerteza, para o valor total do projeto da estratégia analisada (Ω) e os riscos do produto ou a taxa de desconto (d). Desta forma, o VPL de um projeto é expresso por: $VPL = (C - M) / (1 + d)^T - I + \Omega$.

O autor não explica como se chegou a esta fórmula. No entanto, a interpretação é a seguinte: o valor presente de um projeto é dado, inicialmente, pela diferença entre o valor futuro (C) e os custos operacionais (M) em um determinado tempo T. Esta diferença é trazida ao tempo presente através da taxa de desconto (d). Em seguida deduz-se do resultado o valor do custo de desenvolvimento (I) no tempo zero. Finalmente adiciona-se a contribuição sob incerteza, que na realidade representa a flexibilidade na utilização da estratégia.

Como forma de exemplificar a abordagem o autor considera a existência de duas estratégias de resolução para um projeto, calculando-se o VPL do projeto para cada estratégia, pode-se avaliar aquela que possui o maior valor agregado e, portanto, deve receber a preferência.

A grande vantagem desta pesquisa é agregar em uma única fórmula, diversas variáveis que podem afetar o VPL de um projeto ou uma estratégia. No entanto, o autor não explica como estas variáveis foram encontradas e, principalmente, como quantificar as variáveis d (riscos do produto) e Ω (contribuição). Vale ressaltar ainda, que a taxa de desconto ao longo do tempo de desenvolvimento não foi incluída na fórmula e, portanto, futuros estudos são necessários para adaptá-la ao cenário onde os custos e futuros valores gerados podem ocorrer de forma parcelada ao longo do processo de desenvolvimento.

3.2.2. VPL e extreme programming

MÜLLER e PADBERG (2002) utilizam o conceito de Valor Presente Líquido (VPL) para avaliar as condições em que o uso do paradigma de *Extreme Programming* (BECK, 2000) pode ser uma vantagem em relação a um processo convencional de desenvolvimento. Os autores analisam os custos e benefícios de um projeto de software fictício, onde por meio da variação sistemática de parâmetros do projeto que influenciam o uso da *Extreme Programming* (XP), diversas informações podem ser obtidas.

Do ponto de vista econômico, as mais importantes práticas da XP são a programação em pares e o conceito de *test-first.* Como o número de desenvolvedores aumenta para a realização da mesma tarefa, os custos de pessoal são praticamente dobrados. No entanto, defensores da XP argumentam que estes custos elevados são compensados por três fatores: (*i*) um par de programadores desenvolve programas mais rapidamente que um único programador, (*ii*) a contínua verificação do código através de casos de teste melhora a qualidade do código e reduz o esforço de testes ao final do projeto, (*iii*) o código produzido por um par de programadores possui uma menor taxa de defeitos.

No estudo realizado pelos autores, o tempo de chegada de produto ao mercado foi considerado como fator decisivo para a avaliação dos resultados, pois uma falha no cumprimento dos prazos leva à perda de *market share* e, conseqüentemente, reduz o valor agregado pelo projeto. No entanto, para o processo de análise, foram considerados também os custos de desenvolvimento, a quantidade de defeitos encontrados após a conclusão do código e o lucro obtido pelo projeto, caso seja concluído no tempo previsto.

Seguindo estes parâmetros, foram calculados os VPL para projetos considerados convencionais e usando XP em dois diferentes cenários. Assumindo que o número de desenvolvedores seja fixado em oito, no caso da XP o projeto seria executado por quatro pares de programadores, ao passo que o processo convencional utilizaria os mesmos oito desenvolvedores. No segundo cenário, um maior número de desenvolvedores está disponível para o projeto e, portanto, podem ser alocados oito pares de desenvolvedores para a XP enquanto que o processo convencional continua mantendo os oito desenvolvedores. A fórmula de VPL utilizada pelos autores foi:

$$\text{VPL} = \frac{L}{(1+i)^{\frac{t}{12}}} - C$$

onde L representa o lucro obtido pela conclusão do projeto e foi considerado fixo para os dois cenários. A variável i é a taxa de juros utilizada para corrigir o valor do dinheiro ao longo do tempo de desenvolvimento. O tempo de desenvolvimento (t) foi estabelecido em relação ao tamanho do produto (fixo em 16.800 linhas de código), considerando a quantidade média de linhas de código que um programador consegue produzir por mês (300 a 350) (SOMMERVILLE, 1996) e o tempo necessário para remover os defeitos no código. O custo de desenvolvimento (C) foi considerado apenas como o salário dos desenvolvedores e do gerente do projeto.

Os autores comentam que, segundo (HUMPHREY, 1997), um programador insere, tipicamente, 100 defeitos (D) por cada 1000 linhas de código, sendo que um bom processo convencional de desenvolvimento consegue eliminar até 70% destes defeitos. Portanto, utilizando um desenvolvimento convencional tem-se:

$$D = TP \times \frac{100}{1000} \times 30\%, \text{ onde TP é o tamanho do produto.}$$

Segundo estudos experimentais realizados por (COCKBURN e WILLIAMS, 2000), a XP leva a uma revisão simultânea ao tempo de desenvolvimento e cria uma vantagem em relação ao desenvolvimento convencional na ordem de 15%. Portanto, dado um número de defeitos encontrado por um par de programadores utilizando XP, a diferença de defeitos (DD) para um desenvolvimento convencional pode ser expressa por:

$$DD = D \times 15\%$$

Finalmente chega-se ao tempo de remoção de defeito (RD), expresso pela fórmula abaixo, onde T é o tempo para se remover um defeito, N o número de desenvolvedores e considerando-se que estes trabalham 135 horas por mês (MÜLLER e PADBERG, 2002).

$$RD = DD \times \frac{T}{135h} \times \frac{1}{N}$$

De acordo com os resultados obtidos pelos autores, no primeiro cenário (08 desenvolvedores para cada abordagem) o desenvolvimento convencional obteve um maior VPL em relação à XP. Isto se explica pelo fato de apenas quatro atividades serem realizadas pela XP em relação a oito tarefas do desenvolvimento convencional. Esta desvantagem não conseguiu ser compensada pela menor quantidade de defeitos produzidos e pelo menor tempo de desenvolvimento da XP.

No segundo cenário (08 pares de desenvolvedores para a abordagem XP e 08 desenvolvedores para a abordagem convencional), o VPL na XP foi consideravelmente maior que o do desenvolvimento convencional, sendo este fato explicado pelo menor tempo de desenvolvimento e pelo menor número de defeitos, sendo que o maior custo de desenvolvimento devido ao maior número de desenvolvedores foi compensado pelos lucros obtidos e pelo aumento de *market share*. Desta forma, os autores destacam o fato de que decisões de projeto, tais como a alocação de recursos, podem ser fortemente baseadas em conceitos econômicos ao invés de opiniões subjetivas ou de maneira *ad hoc*.

Apesar da simplicidade sugerida pelos autores, alguns tópicos da abordagem proposta podem ser considerados discutíveis em relação à realidade de um projeto de

software. Primeiramente, não foi considerado o fato de que apesar da XP diminuir o tempo estimado de desenvolvimento, o aumento do número de pessoas envolvidas no projeto aumenta o volume de comunicação entre elas, o que geralmente aumenta o tempo necessário para se realizar uma tarefa.

Finalmente, os custos de desenvolvimento de um projeto, conforme os autores, foram apenas de pessoal, e isto pode não ser a realidade da maioria das empresas, mais uma vez podendo incorrer em erros no resultado final da avaliação.

3.3. Teoria do portfolio

O objetivo da Teoria do Portfolio, proposta por MARKOWITZ (1952), é conseguir maximizar a riqueza do investidor minimizando o que ele próprio considera como indesejável: a variância ou o risco. Utilizando o conceito de retorno médio esperado (média dos retornos passados), da variância como medida de risco e através de gráficos, equações e cálculos algébricos, mostra-se que a variância (risco) deve ser considerada tão importante quanto o retorno na hora de se decidir sobre como e onde investir.

Markowitz mostra que, utilizando a diversificação de investimentos, ou seja, aplicação de recursos em ativos com diferentes fontes de risco, consegue-se um retorno que representa a média ponderada dos diferentes ativos, embora a variância seja menor que a média das variâncias isoladamente. A diferença entre a média das variâncias e a variância real da carteira (portfolio) depende diretamente da correlação histórica entre os diferentes ativos.

Existem três possíveis resultados na análise das correlações entre ativos. Primeiramente, a correlação entre eles pode ser positiva, ou seja, o crescimento de um influencia no crescimento do outro. Em seguida, a correlação entre eles pode ser nula, ou seja, não há nenhuma relação implícita entre o crescimento de um ativo em relação ao crescimento ou decréscimo de outro ativo. Finalmente, dois ativos podem ser relacionados negativamente, o que significa que o crescimento de um implica no decréscimo de outro. Desta forma, quando dois ou mais ativos pouco correlacionados compõem uma carteira de investimentos consegue-se um risco menor que a média ponderada dos riscos individuais, conseguindo, algumas vezes, um risco menor que o do ativo de menor risco com um retorno maior que o deste ativo.

A maior crítica em relação à Teoria do Portfolio refere-se a definição das correlações, pois mesmo em um ambiente econômico, onde existe uma base histórica dos ativos, são necessários muitos cálculos, na maioria das vezes complexos, para que

se chegue aos resultados desejados. Além disso, ERRUNZA *et al.* (1996) discutem ainda o fato de Markowitz ter adotado a curva normal como uma representação da distribuição dos retornos dos ativos. No entanto, estas críticas se aplicam às premissas utilizadas por Markowitz para justificar a aplicação de sua teoria num cenário de Economia e, portanto, está fora do escopo desta tese. Contudo, vale ressaltar que a Teoria do Portfolio quantificou o que antes era apenas sentimento, o risco de uma carteira.

3.3.1. Teoria do portfolio para projetos de software

BUTLER *et al.* (1999) utilizam os conceitos apresentados pela Teoria do Portfolio para a alocação de recursos em projetos de software. Os autores comentam que o desenvolvimento de sistemas complexos envolve decisões sobre como alocar recursos orçamentários a um conjunto de alternativas que possuem benefícios futuros incertos. Sendo assim, sugerem que os gerentes analisem os problemas relacionados à alocação de recursos como uma oportunidade de investimento, os benefícios sejam vistos como os retornos decorrentes deste investimento e o processo decisório como sendo a seleção de uma carteira ótima de atividades, onde cada atividade necessita uma parte dos recursos e o objetivo é determinar qual a melhor combinação possível para que o resultado global seja maximizado.

Os autores destacam o fato de que, de modo geral, decisões sobre alocação de recursos são tomadas através de julgamento subjetivo, por comparação, intuitivamente ou mesmo por tradição organizacional. No entanto, poucas abordagens sistemáticas estão disponíveis para engenheiros de software.

A abordagem é exemplificada através da seleção de Técnicas de Validação de Software e o objetivo é escolher quais técnicas devem ser utilizadas e qual proporção de recursos deve ser alocada a cada uma, de forma a maximizar os objetivos que são definidos, como a minimização do conjunto de defeitos no sistema.

Cada uma das técnicas envolvidas (ativos) no processo possui um conjunto de atributos necessários para sua implantação e utilização (riscos), o grau de confiabilidade em relação à detecção de defeitos (retornos) e os recursos técnicos necessários para a aplicação da técnica (custos). Estas características são representadas numericamente através de relações de custo-benefício, calculadas através de uma função de utilidade, que pode diferir de projeto para projeto. Assim, a melhor combinação de atividades poderia ser encontrada.

Desta forma, tem-se um Orçamento (*O*), que se deseja distribuir entre diversas (*n*) técnicas de validação, Para cada técnica, necessita-se, inicialmente, quantificar a relação de custo-benefício de sua implantação, ou seja, o benefício por unidade de recurso investida. Este benefício é dado por uma representação numérica do nível de confiabilidade de sua utilização, em relação à remoção de defeitos, e pode ser modelada como uma variável aleatória e deve ser calculada através da utilização da função de utilidade definida para o sistema. O Custo (*C*) é determinado pela necessidade de alguma habilidade especial para a utilização da técnica, pelo tempo de sua implantação ou alguma restrição na implementação do código. A tabela 3.1 ilustra algumas técnicas de validação e seus atributos, conforme exemplo dos autores.

Tabela 3.1 Técnicas de validação e atributos. BUTLER *et al.* (1999)

Técnica	Habilidades especiais	Tempo de implantação	Grau de confiabilidade	Restrição na implementação do código
Random Testing	Mínima	Alta	Baixa	Nenhuma
Boundary Testing	Baixa	Alta	Baixa	Nenhuma
Model Checking	Muito alta	Alta	Muito alta	Nenhuma
Static Analysis	Alta	Indefinido	Muito alta	Algumas
Eraser	Dependente das restrições de programação	Baixa	Alta	Duas restrições

Com base nas informações contidas na tabela e nas premissas assumidas, os autores afirmam ser possível mapear o contexto para um problema de otimização de ativos com o objetivo de maximizar os resultados, conforme proposto pela Teoria do Portfolio, por meio da resolução da seguinte equação de otimização global:

$$\max_{\pi} E[u(O_{\pi}{}^{T}\delta],$$

onde π representa o vetor de ativos envolvido no portfolio, que neste caso são as técnicas de verificação, *E* representa a utilidade esperada, *u* é a função de utilidade utilizada, *O* é o orçamento disponível, π^{T} o vetor transposto de π e δ o vetor dos retornos de cada retorno relativo a cada ativo, mais uma vez representado pelas técnicas em análise.

Na Teoria do Portfolio, um componente fundamental para a composição da melhor carteira de investimentos é a correlação entre os ativos. Em economia, esta correlação é calculada com base no histórico dos lucros provenientes do investimento nos ativos em questão. Este mapeamento para projetos de software não é uma tarefa fácil, pois a correlação entre as atividades, ou outro elemento qualquer que esteja envolvido no cálculo, não está claramente definida e, dificilmente, uma organização possui dados

históricos suficientes para criar uma correlação confiável, devido à natureza diversificada de projetos de software. Isto torna a definição de parâmetros um grave problema a ser considerado.

Na abordagem proposta, os autores não incluem a correlação entre as atividades como uma variável para o cálculo e sim uma relação de custo-benefício, que é transformada em números por meio de funções de utilidade. Os próprios autores admitem que estas funções não podem ser definidas facilmente e envolvem muitos fatores subjetivos, o que pode tornar o resultado uma medida com uma ampla margem de erro. Além disto, no artigo não é explicado como é feita a conversão dos atributos em custo, visto que estes são atributos qualitativos.

3.4. Opções reais

A *Análise de Opções Reais* (LUEHRMAN, 1998) é primariamente utilizada na avaliação de investimentos estratégicos. Sua aplicabilidade depende da presença de duas propriedades: (*i*) uma ou mais fontes de incerteza, e (*ii*) decisões futuras irreversíveis provenientes de fatos incertos. Este tipo de análise é apropriada em casos onde as técnicas clássicas de avaliação financeiras tais como o Valor Presente Líquido (ROSS et al., 1998) não se aplicam, devido a aspectos dinâmicos do processo de tomada de decisão sob incerteza.

Uma *opção* é definida como uma ação discricionária cuja execução é dependente de certas realizações futuras (ERDOGMUS e BARRE, 2000). Dado que uma opção representa uma ação discricionária, ou seja, uma oportunidade sem a obrigação, ela só deve ser exercida quando o retorno é positivo.

Quando administradores começam a pensar em termos de opções, eles vêem que o investimento é um sistema atualizável, devendo ser considerado como uma série de investimentos feitos em estágios. As técnicas comumente utilizadas na análise de investimentos, como o VPL, assumem que um plano previamente determinado será seguido, sem considerar que o desdobramento dos eventos pode exigir a realização de mudanças no plano. A Teoria das Opções oferece justamente esta possibilidade de análise, onde diante de situações passíveis de mudanças, gerentes podem reavaliar suas alternativas, a fim de tomar a melhor decisão possível.

3.4.1. Opções associadas a riscos

Baseado no conceito de *Opções Reais*, ERDOGMUS e BARRE (2000) descrevem uma técnica para capturar as possibilidades de lucro através da estimativa de riscos de

um cenário de desenvolvimento de software. A idéia dos autores é comparar possíveis decisões em desenvolvimento de software com decisões em mercados financeiros utilizando o conceito de *opções*. A proposta inspira-se no fato de que a incerteza pode gerar valor a uma empresa quando gerenciada apropriadamente, pois, de forma geral, quanto maior o nível de incerteza, maior a possibilidade de lucros.

A abordagem é exemplificada pelos autores através de um projeto realizado por uma empresa que pretende desenvolver uma nova arquitetura de software baseada em tecnologia Java e que servirá de base para uma futura linha de produtos. Desenvolvida a arquitetura, a empresa será capaz de produzir novos sistemas mais rapidamente. Neste caso, a empresa está incorrendo em um risco de investir em uma nova plataforma e, possivelmente instável. Por outro lado, o retorno do investimento do projeto dependerá do mercado para os sistemas que ela pretende desenvolver em um tempo futuro.

O custo da nova plataforma é estimado em $0,5M e seu tempo de desenvolvimento está previsto para um ano. Ao final do primeiro ano do projeto, a empresa fará uma avaliação em relação ao mercado de *e-business* com base em plataforma Java. Se o mercado for favorável, uma segunda fase do projeto terá início, consistindo na extensão do componente de repositório e o desenvolvimento da linha de produto. Esta segunda fase do projeto está orçada em $5M e levará mais um ano para ser implementada. Caso o mercado não seja favorável, o investimento inicial será perdido e o projeto cancelado.

Como se pode observar, a empresa tem uma ação discricionária a realizar, ou seja, a possibilidade de executar a segunda parte do projeto ou não, de acordo com os resultados e a análise do mercado ao término da primeira fase do projeto.

Baseado nas fórmulas de avaliação de *opções* (BLACK e SCHOLES, 1973) observa-se a diferença entre considerar o projeto como um único investimento e adotar a estratégia de um desenvolvimento parcelado, onde a possibilidade de prosseguir com o projeto é definida pelo seu VPL no ponto de decisão. Através das fórmulas do modelo, um limite mínimo para prosseguimento do projeto é estabelecido e dados relativos à viabilidade do projeto são fornecidos aos gerentes para auxiliar o processo decisório.

Esta abordagem, ao utilizar o conceito de *opções*, que possibilita uma decisão mais acurada em relação a investimentos futuros sob incertezas, introduz várias dificuldades para a aplicação de suas fórmulas, que, originalmente, utilizam parâmetros disponíveis no mercado financeiro, mas que não podem ser encontrados com a mesma facilidade num contexto de desenvolvimento de software.

As entradas básicas para as equações são: o valor presente do projeto completo, o preço da opção a ser exercida (no caso $5M), o tempo da execução da primeira fase do projeto (data de maturidade), a taxa livre de risco (assumida como 6% ao ano) e a volatilidade do investimento (assumida como 47% ao ano). A volatilidade é uma medida da variância do valor presente no projeto, uma vez que seus fluxos futuros podem ser afetados por diversos riscos, tanto de flutuações nos ativos de mercado quanto relacionados ao processo de desenvolvimento de software. Tanto a taxa livre de risco como a volatilidade foram calculadas com base nos dados fornecidos por um conjunto de 18 empresas que possuem produtos ou serviços baseados em Java.

Está fora do escopo desta tese analisar as fórmulas que são utilizadas para quantificar *opções.* No entanto, pela exposição das variáveis envolvidas, pode-se visualizar a dificuldade que a maioria das empresas teria para obter os dados necessários para a utilização das fórmulas, tais como volatilidade do investimento. Ressalta-se também a dificuldade em se estabelecer o tempo onde a *opção* terá que ser exercida ou rejeitada, principalmente em função dos problemas que podem ocorrer em relação ao cronograma de projeto.

AMRAM e KULATILAKA (1999) acreditam que a Teoria das Opções tem sido lenta em se difundir no mundo dos negócios porque muito da discussão tem se centrado nas equações e modelos. A complexidade das ferramentas tem obscurecido o poder da idéia subjacente. O real valor das opções reais está não nos resultados das fórmulas de BLACK e SCHOLES (1973), mas na reformulação do pensamento dos executivos a respeito de investimento estratégico. Por prover uma introspecção objetiva da incerteza presente em todos os mercados, as opções reais capacitam os executivos a pensar mais clara e realista acerca da complexidade e risco das decisões estratégicas.

3.4.2. Opções e teoria do portfolio

ASUNDI E KAZMAN (2001) mesclam os conceitos de *Opções* e a *Teoria do Portfolio* e propõem uma abordagem estruturada para apoiar engenheiros de software na escolha das melhores alternativas de projetos arquiteturais. Os autores desenvolveram um método para realizar análises econômicas de software e sistemas centradas em suas arquiteturas. O método foi denominado CBAM (*Cost Benefit Analysis Method*) (KAZMAN *et al.*, 2000a) que, por sua vez, é baseado no ATAM (*Architecture Tradeoff Analysis Method*) (KAZMAN *et al.,* 2000b) que provê aos analistas uma melhor compreensão dos custos e benefícios de decisões arquiteturais em projetos de software.

Dado o fato de que existe uma grande quantidade de incertezas – tanto técnicas quanto econômicas – durante a fase de projeto arquitetural de sistemas, analistas estão sempre enfrentando riscos que podem impedir que seus objetivos sejam atingidos. Os autores comentam que uma das vantagens de sua abordagem reside no fato de que os interessados no projeto são obrigados a quantificar explicitamente os riscos, os custos e os atributos de qualidade referentes ao sistema, bem como as incertezas em relação a estas estimativas.

O problema em questão está em escolher dentre uma série de alternativas arquiteturais, o melhor conjunto possível para que os objetivos do projeto sejam maximizados. Ao passo que no ATAM os autores geram uma lista de atributos de qualidades importantes para o projeto, no método CBAM os interessados ordenam individualmente cada atributo de forma que a soma dos valores seja 100, assinalando para cada atributo de qualidade um valor denominado QA*Score.*

Em seguida, os interessados precisam estimar o impacto resultante da aplicação de cada atributo de qualidade, chamado de *Contribution.* Multiplicando-se o QA*Score* pela *Contribution,* obtém-se o benefício de cada atributo. Em princípio, a solução lógica seria escolher os atributos com melhor relação de custo-benefício. No entanto, os autores ressaltam o fato de que, assim como qualquer outra atividade econômica, o investimento em projetos de software possui incertezas e algumas decisões são praticamente irreversíveis. Portanto, uma melhor análise necessita ser realizada.

Baseados na *Análise de Opções Reais,* onde é possível retardar a tomada de decisão até o ponto onde se tenham mais informações sobre os benefícios de se implementar uma solução, e na *Teoria do Portfolio,* onde a melhor combinação possível de alternativas pode ser escolhida de forma a minimizar o risco global e aumentar os benefícios, os autores determinam quais opções de arquitetura devem ser adotadas para o projeto com o auxílio da correlação existente entre estas alternativas.

Vale ressaltar que as correlações necessárias para os cálculos são determinadas através da quantidade de componentes afetados pela adoção de uma determinada estratégia e o que esta estratégia realiza na arquitetura do sistema. Desta forma, assume-se uma correlação zero (0) para estratégias com nenhuma dependência óbvia. Estratégias que são subconjunto de outras estratégias assumem correlação um (1). Caso duas estratégias impliquem na adoção de tecnologias excludentes, assume-se uma correlação perfeitamente negativa (-1). Valores intermediários entre –1 e 1 são assinalados subjetivamente pelo gerente.

Esta abordagem agrega duas teorias complexas e que, dependendo do contexto aplicado, pode levar a decisões equivocadas em relação a projetos de software. Mesmo para o contexto financeiro, algumas premissas são assumidas para que as teorias sejam aplicáveis.

A Teoria do Portfolio, por exemplo, assume que um investidor tem condições de identificar e quantificar os riscos envolvidos no investimento de ativos da carteira e a correlação existente entre estes ativos. Ao se transpor estes problemas para a realidade de software, algumas técnicas de identificação e quantificação de riscos podem ser encontradas na literatura (BARROS, 2001). No entanto, pouco se sabe sobre a correlação existente entre fatores de risco de software, componentes, técnicas e paradigmas de desenvolvimento e estudos experimentais são necessários para que estas correlações possam ser estabelecidas. Portanto, a forma como os autores determinaram as correlações em sua abordagem, que são fundamentais para a aplicação da Teoria do Portfolio pode não ser muito confiável, levando a erros de interpretação e resultados não confiáveis.

3.4.3. Opções e reuso de software

FAVARO *et al.* (1998) apresentam um método para avaliação de investimentos na reutilização de software que se baseia na junção de diversos conceitos econômicos. Segundo os autores, os benefícios do reuso de software podem ser divididos em duas categorias: *operacionais* e *estratégicos.* Os benefícios operacionais são aqueles que tratam de melhoria da qualidade, aumento da produtividade e redução dos custos de manutenção. Os benefícios estratégicos referem-se à oportunidade de ingresso em novos mercados e à habilidade de responder eficientemente a forças competitivas e mudanças nas condições de mercado.

De acordo com os autores, um grande progresso já foi alcançado pela comunidade de métricas no sentido de quantificar os benefícios operacionais do reuso de software. No entanto, poucas abordagens sistemáticas para mensurar os benefícios estratégicos têm sido desenvolvidas. Baseados nesta necessidade, os autores apresentam um *framework* denominado *Value Based Reuse Investment* (VBRI) – Investimento em Reuso Baseado em Valor - para avaliação de estratégias de negócio relacionadas a reuso de software. Para criar o *framework* VRBI, diversas teorias econômicas tais como o VPL, o CAPM (*Capital Asset Pricing Model*) (ROSS *et al.*, 1998) e a as Opções Reais foram analisadas

e integradas com conceitos estratégicos estabelecidos pelo *Value Based Management* (VBM) (McTAGGART *et al.*, 1994) gerando quatro princípios básicos:

- A maximização do valor econômico deve direcionar o investimento estratégico em reuso;
- A estratégia do negócio deve direcionar a seleção de investimento em reuso;
- Investimentos devem ser ativamente estruturados para que possam maximizar as opções estratégicas;
- Tanto as técnicas de Fluxo de Caixa Descontado quanto às de Opções Reais devem ser utilizadas para capturar o valor dos investimentos em reuso.

Segundo os autores, o planejamento tradicional de investimentos em reuso é feito de forma linear, onde inicialmente são planejadas as estratégias de investimento, em seguida os sistemas são construídos e, finalmente, análises são feitas num processo *top-down*. A partir deste raciocínio, projetos e tecnologias são aceitos ou rejeitados. Os autores comentam que este processo passivo de aceitação de execução de projetos pode explicar, parcialmente, porque as atividades econômicas de reuso são mais voltadas para o contexto operacional do que o estratégico. Para tentar remodelar este processo, o VRBI propõe um ciclo diferente de planejamento de investimentos.

Desta forma, o VRBI sugere um ciclo contínuo e cíclico de identificação, formulação e avaliação de opções estratégicas realizadas por planejadores, analistas e gerentes. Em um cenário típico de VRBI, técnicos apresentam a estrutura de um projeto a analistas financeiros, que por sua vez sugerem modificações com o objetivo de agregar mais flexibilidade às opções de projeto. Em seguida, o projeto reestruturado é apresentado a planejadores estratégicos, que recomendam modificações para agregar opções estratégicas e em seguida o ciclo é reiniciado. Diversos ciclos podem ser realizados na estrutura dos projetos, que podem, em sua versão final, ser totalmente diferente da sua versão original, mas consideravelmente mais ajustadas aos objetivos globais de uma organização.

Com base neste tipo de análise dentro da estrutura do VRBI que os autores sugerem a avaliação de estratégias de negócio relacionadas a reuso de software e relatam diversos outros trabalhos correlatos utilizando abordagens semelhantes.

Além das abordagens apresentadas nesta seção, outras aplicações da Teoria de Opções podem ser encontradas na literatura de Engenharia de Software. Por exemplo,

(WITHEY, 1996) descreve a utilização de opções reais para a análise de investimentos em Linhas de Produtos de software. BALDWIN e CLARK (1993) e SULLIVAN *et al.* (1997) utilizam opções reais para analisar os custos e benefícios proporcionados por diferentes técnicas aplicáveis na atividade de projeto de software.

3.5. Conclusões

Neste capítulo, introduzimos os conceitos de Economia aplicada à Engenharia de Software, suas origens, evolução e aplicações. Em seguida apresentamos um resumo de vários conceitos e teorias econômicas que já foram adaptados para a Engenharia de Software, para que seja possível ter uma visão abrangente dos trabalhos realizados por pesquisadores e praticantes de Engenharia de Software, a fim de aproximar estas áreas de conhecimento e apoiar o processo decisório de gerentes e investidores.

Assim como qualquer outra atividade econômica, a Engenharia de Software visa lucros. A decisão de investir ou não em um projeto ou uma tecnologia específica requer uma análise cuidadosa e eficiente, pois em um mercado competitivo e com pouca margem de erros não há como, simplesmente, arriscar empreitadas intempestivas e inconseqüentes.

Analisar a viabilidade de projetos e tecnologias de software não é uma tarefa simples, pois envolve muitas variáveis e principalmente incertezas inerentes a esta atividade. Não há como negar a contribuição e o potencial que a união entre Economia e Engenharia de Software pode trazer para a melhoria da qualidade dos produtos de software e para o crescimento de uma organização. No entanto, muitas pesquisas ainda precisam ser realizadas neste domínio de conhecimento e principalmente embasar estas pesquisas em estudos experimentais para que resultados mais confiáveis possam ser obtidos e se tenha uma clara noção de onde, quando e como estes conceitos podem ser aplicados.

Observamos inicialmente que as técnicas de fluxo de caixa descontado, tais como o VPL, conseguem fornecer informações sobre o valor do dinheiro ao longo do tempo e auxiliar investidores a tomar decisões. No entanto, ao se considerar fatores mais complexos como a flexibilidade e dinamismo dos projetos de software, essas técnicas têm pouco a oferecer e, portanto, abordagens mais complexas são necessárias.

Em seguida, a análise e quantificação dos riscos foi discutida em função de uma carteira de projetos, por meio de conceitos da Teoria dos Portfolio, onde também encontramos algumas dificuldades de aplicações práticas, devido à complexidade dos

cálculos a serem realizados e a uma série de premissas que devem ser assumidos para a adaptação a Engenharia de Software.

Finalmente, o conceito de opões foi introduzido para agregar flexibilidade e possibilidade de retardar a tomada de decisões no que se refere a investimento em projetos de software. No entanto, nenhuma das teorias apresentadas neste capítulo consegue nos apoiar para que possamos atingir o objetivo deste trabalho, qual seja, analisar uma carteira de projetos de software baseado nos riscos envolvidos em cada projeto a fim de determinar a probabilidade de perdas e ganhos da empresa em decorrência do desenvolvimento dessa carteira de projetos. Para tanto, faz-se necessário, identificar e, principalmente, encontrar uma forma de quantificar estes riscos de um projeto de software para, em seguida, associar estes valores a teorias econômicas que possam ser mapeadas para o contexto de Engenharia de Software. Estes assuntos serão tratados nos capítulos 4 e 5 respectivamente.

4

AVALIAÇÃO DO NÍVEL DE RISCO DE PROJETOS DE SOFTWARE

"Se a natureza humana não caísse na tentação de enfrentar riscos... talvez pouco se investisse, como resultado da fria avaliação".
John Maynard Keynes

4.1. Introdução

Falhas na avaliação dos riscos são uma das maiores fontes de problemas no desenvolvimento de sistemas de informação (MCFARLAN & MCKENNY, 1996). Logo, muitos estudos têm sido realizados na busca das principais fontes de riscos em projetos de software e na determinação da melhor maneira de quantificá-las e tratá-las.

Das quatro fases do processo genérico de gerenciamento de risco (Identificação, Análise, Planejamento e Controle), esta tese refere-se apenas às técnicas de execução das duas primeiras fases.

Baseado em um questionário de levantamento de riscos de projetos foi criada uma técnica para a quantificação dos riscos relacionados com projetos de software. A técnica proposta requer o conhecimento da importância relativa entre os fatores de risco que afetam os projetos de software. A fim de determinar esta informação, um estudo experimental foi realizado e seus resultados nos permitiram determinar a importância dos fatores de risco de acordo com três tamanhos distintos de projetos de software.

Além da introdução apresentada, esse capítulo está organizado da seguinte maneira: a seção 4.2 descreve a abordagem adotada nesta tese para a avaliação de riscos. Na seção 4.3 apresentamos a técnica proposta para a avaliação do nível de riscos em projetos de software. A seção 4.4 descreve o planejamento e os resultados obtidos no estudo experimental realizado durante esta tese. Considerações finais são elaboradas na seção 4.5.

4.2. Avaliação de riscos baseada em conceitos econômicos

Riscos de projetos de software podem ser classificados em dois grupos: riscos sistêmicos e riscos específicos. Os riscos sistêmicos são aqueles que afetam todos os projetos de software de uma determinada categoria. Como exemplos de categorias de projetos de software, podemos citar os sistemas de informação, os sistemas militares, os

sistemas comerciais, os componentes *off-the-shelf*, entre outros. Qualquer projeto em uma categoria está sujeito, até certo nível, aos riscos sistêmicos da categoria. Riscos específicos, por sua vez, são os fatores de risco associados às características específicas de um projeto.

Esta separação entre riscos sistêmicos e específicos tem sua origem na Economia. Como exemplo desta separação nos mercados, pode-se citar uma bolsa de valores, onde cada ação negociada possui riscos associados ao mercado como um todo e riscos vinculados a aspectos específicos da empresa que subscreveu a ação. Os riscos sistêmicos (mercado financeiro) são oriundos das condições econômicas do país onde a empresa está localizada, de políticas governamentais, do cenário econômico internacional e de outros fatores globais que afetam todas as empresas que atuam em um mesmo mercado. Riscos específicos (empresa) são relacionados a uma organização específica, condições locais de mercado, administração, reputação da empresa e diversos outros fatores. Portanto, enquanto riscos sistêmicos avaliam o cenário geral onde a empresa está localizada, os riscos específicos focalizam os fatores internos que afetam o desempenho da empresa.

Neste capítulo apresentamos uma abordagem para avaliar o nível de risco de um projeto de software baseado em seus riscos sistêmicos e específicos. Esta abordagem se baseia em um questionário construído através da agregação de taxonomias de identificação de riscos apresentadas na literatura de engenharia de software. Cada uma das questões busca levantar características particulares de um projeto de software (riscos específicos). As questões estão agrupadas em fatores de riscos, que permitem ponderar as respostas ao questionário de acordo com a importância sistêmica de cada fator de risco. Através desta ponderação, calcula-se o nível de risco de um projeto de software.

O nível de risco e as respostas ponderadas são úteis no processo decisório de projetos de software. Primeiramente, o nível de risco explicita, em um único número, quão arriscado é um projeto, auxiliando um gerente a comparar dois ou mais projetos de acordo com suas relações de risco e retorno[1]. Além disto, as médias ponderadas permitem a diversificação de projetos. Por exemplo, considerando que uma organização está desenvolvendo um projeto com alto risco relacionado à codificação (pela falta de conhecimento da linguagem a ser utilizada pela equipe do projeto), um gerente pode

[1] A razão de risco e retorno é normalmente usada no lugar do ROI (retorno do investimento). No entanto, diferentemente do ROI que considera apenas quanto uma empresa terá de lucro em um projeto, a razão de risco e retorno mede o quanto uma empresa vai se arriscar, na tentativa de obter os benefícios de um projeto.

decidir somente desenvolver outro projeto se seu nível de risco na atividade de codificação não for tão elevado, a fim de balancear o nível de risco global que uma organização está disposta a aceitar em sua carteira de projetos. Pode-se também observar quais são os pontos mais críticos de um projeto a fim de melhor aplicar os recursos disponíveis para ele.

4.3. Técnica de avaliação de riscos

Para se determinar o nível de risco de um projeto, o primeiro passo consiste em responder a um questionário de identificação de risco. Este questionário foi construído com base na taxonomia de riscos apresentada por CARR et al. (1993), que foi complementada com outros questionários e taxonomias apresentadas na literatura (BOEHM, 1991; BARKI, 1993; JONES, 1994; KAROLAK, 1996; MOYNIHAM, 1997 e WALLACE, 1999).

Após a análise dos questionários, as fontes de risco encontradas foram reagrupadas por semelhança. Algumas questões foram eliminadas pelo fato de estarem presentes em mais de um questionário. Algumas questões necessitaram ser reescritas para que pudessem ser enquadradas no modelo proposto para essa tese. Assim, todas as perguntas foram expressas de forma que quanto mais presente em um projeto, mais esta fonte de risco contribui para o seu insucesso. Vale ressaltar que o sentido original das perguntas não foi alterado quando tiveram que ser reescritas.

O questionário possui 211 questões que são classificadas em dez grupos, denominados fatores de risco. Cada questão pertence a um único fator, tendo como objetivo descrever de forma detalhada o conceito abstrato representado pelo fator de risco, relacionando-o com elementos mais práticos que podem ser avaliados pelo gerente, de acordo com as características do projeto em análise. Este questionário encontra-se no Anexo 1. A tabela 4.1 apresenta os fatores de risco e o número de questões em cada fator.

Tabela 4.1 - Distribuição das questões nos fatores de risco

Fator de Risco	*# Questões*
Análise	28
Projeto	17
Codificação	11
Teste	25
Planejamento	36
Controle	17
Equipe	32
Política e Estrutura	08
Contratos	21
Clientes	16

No fator "Análise" estão descritos problemas relacionados com o levantamento dos requisitos, sua estabilidade, nível de dificuldade de implementação, validação e complexidade do sistema.

No fator "Projeto" estão cita-se problemas relacionados à correta concepção da arquitetura, interfaces, algoritmos e mecanismos que facilitem a implementação do sistema.

No fator "Codificação" encontram-se problemas relacionados à complexidade de implementação dos algoritmos, inadequação de linguagem ou hardware e reutilização de código.

No fator "Teste" descreve-se problemas relacionados ao planejamento e execução, condições de realização, tipos e abrangência dos testes do sistema.

No fator "Planejamento" são listados problemas relacionados à experiência dos gerentes, capacidade de elaboração de planejamento e estimativas do projeto, bem como aspectos de definição, utilização e adequação do processo de desenvolvimento de sistemas.

No fator "Controle" estão descritos problemas relacionados à condução do projeto, aprovação de artefatos, resolução de conflitos e apoio à equipe de desenvolvimento, bem como as atividades de acompanhamento e replanejamento ao longo do projeto e a avaliação do processo de desenvolvimento.

No fator "Equipe" lista-se problemas relacionados à capacidade, estabilidade, treinamento, maturidade e forma de trabalhar da equipe, bem como o ambiente de desenvolvimento, e o grau com que a equipe segue os planejamentos e os processos.

No fator "Política e Estrutura" encontram-se problemas relacionados à Política e à Estrutura Organizacional, apoio da Alta Gerência ao projeto, metas e conflitos de interesses.

No fator "Contratos" descreve-se problemas relacionados aos contratos, subcontratos, fornecedores e dependências externas do projeto.

No fator "Clientes" estão citados problemas relacionados ao envolvimento do cliente no projeto, número de usuários e nível de mudanças que serão provocadas pelo sistema.

Existem seis respostas possíveis para cada questão, expressando a opinião do gerente em relação a sua importância em um projeto de software. O valor 0 (zero) indica que o problema tratado pela questão não representa um risco para o projeto. Os demais valores (1 a 5) são codificados de uma escala de ordenação subjetiva, que varia de Risco Muito Baixo (1) a Risco Muito Alto (5). Esta operação de codificação é baseada nos estudos de LIKERT (1932), onde números podem ser utilizados para expressar valores de uma escala ordinal. O gerente também possui a opção de assinalar a uma questão como não relevante para o projeto (NR).

Em relação à seleção da quantidade de valores de uma escala de ordenação, GUILFORD (1954) sugere algumas considerações: se poucos valores forem considerados, a escala torna-se muito simplista e algumas informações podem ser perdidas, pois a escala não captura o poder discriminatório que os respondentes são capazes de executar. Por outro lado, utilizar muitos valores torna a escala tão detalhista que pode ficar além do poder de discriminação dos respondentes. MILLER (1956) argumenta que um indivíduo, em média, pode processar simultaneamente sete mais ou menos duas informações. Portanto, as sete opções possíveis na técnica podem ser consideradas compatíveis com o processo discriminatório da maioria das pessoas.

No segundo passo da técnica de quantificação de risco, o gerente calcula a mediana dos valores atribuídos às respostas de cada fator de risco. Questões marcadas como não relevantes (NR) são tratadas como valores ausentes e não são consideradas no cálculo das medianas. Calcula-se a mediana devido ao fato das respostas às questões serem expressas em uma escala ordinal, mesmo sendo representadas por números (codificação de Likert). Caso os valores das respostas estivessem em uma escala intervalar, o melhor valor representativo seria expresso pela média aritmética.

No terceiro passo, divide-se a mediana calculada de cada fator por cinco (5) e multiplica-se o resultado pelo valor de ajuste do fator. Dado que o valor máximo de cada mediana é cinco (5), a divisão visa normalizar estes valores, transformando-os em um número compreendido no intervalo entre 0 e 1. A multiplicação pelo valor de ajuste tem o objetivo de ponderar este fator em relação a sua importância sistêmica em relação aos

outros fatores. Finalmente, no quarto passo, os números resultantes referentes a cada fator são somados, combinando em um único número os riscos sistêmicos e específicos do projeto, representando seu nível de risco.

O valor de ajuste captura a relevância sistêmica de um fator de risco para uma categoria específica de sistemas e o processo para a determinação desses valores será descrito na seção 4.4. No entanto, algumas propriedades dos valores de ajuste devem ser ressaltadas:

- A soma de todos os valores de ajuste deverá ser sempre 100%;
- Quanto maior a relevância de um fator, maior será o seu valor de ajuste.

A primeira propriedade garante que o nível de risco de um projeto seja representado como um valor entre 0 e 100%. Isto é assegurado através dos procedimentos realizados no terceiro passo da técnica, onde as medianas das respostas às questões são normalizadas. A segunda propriedade ajusta a avaliação específica dos riscos (respostas das questões) aos riscos sistêmicos, de acordo com a importância de cada fator de risco dentro da categoria de sistemas sob avaliação.

Os valores de ajuste de um fator de risco são um ponto delicado na técnica proposta. Pode-se argumentar, por exemplo, que esses valores serão diferentes para cada projeto sendo desenvolvido, dificultando assim qualquer tentativa de agregação das respostas ao questionário. Observando este fato, KAROLAK (1996) sugere que o gerente seja responsável pela determinação do peso de cada um dos fatores onde são agregadas as perguntas de seu questionário de avaliação de riscos. Esta abordagem, embora genérica, deixa muita responsabilidade para o gerente de projeto, o que não é desejável quando este busca na medida de risco mais subsídios para apoiar decisões relacionadas ao projeto.

A tabela 4.2 apresenta um exemplo do cálculo do nível de risco de um projeto. Neste exemplo, apenas dois fatores de risco são considerados. O primeiro possui três (3) questões e um valor de ajuste de 70%. O segundo fator possui duas (2) questões e um valor de ajuste de 30%.

Tabela 4.2 – Calculando o nível de risco de um projeto

	Fator 1			Fator 2	
Responda o questionário	Q1	Q2	Q3	Q1	Q2
	2	4	3	2	4
Calcule a mediana das questões	2, 4 e 3 = 3			2 e 4 = 3	
Ajuste os valores das medianas	$\frac{3}{5} * 70\% = 42\%$			$\frac{3}{5} * 30\% = 18\%$	
Some os valores ajustados	Nível de risco = 60%				

Nesta tese, utilizamos o conceito de risco sistêmico para mapear as diferenças de importância entre os fatores de risco. Reconhecemos que podem existir diferenças entre estes fatores em diversos projetos de software, mas utilizamos a premissa de que, dentro de uma mesma categoria de sistemas, esta diferença pode ser explicada pelos riscos específicos, permanecendo fixas as relações entre os fatores de risco sistêmicos.

Os riscos específicos são capturados pelas respostas às questões da taxonomia de identificação de riscos proposta, sendo refletidos nas medianas submetidas aos valores de ajuste de cada fator de risco (sistêmicos). De acordo com esse raciocínio, um alto valor de ajuste de um fator que não é relevante para um projeto terá seu valor reduzido pela ausência de riscos ou a atribuição de baixos pesos para cada questão.

4.4. Estudo experimental para levantamento de valores de ajuste

4.4.1. Objetivo do estudo

O agrupamento das categorias para o cálculo do nível de risco do projeto exige que seja atribuído um peso para cada fator (valor de ajuste), permitindo assim que os resultados obtidos nas respostas a suas perguntas possam ser agregados com os resultados obtidos nas respostas às perguntas que compõem as outras categorias. No entanto, esses valores não são conhecidos, surgindo, então, a necessidade de se realizar um estudo experimental para identificá-los.

Devido aos diferentes tipos de projetos de software que podem ser desenvolvidos para um número cada vez maior de domínios, é suposto que os valores de ajuste,

requeridos pela técnica de avaliação de risco apresentada na seção anterior, podem variar em função das diferentes categorias de sistema.

Com o objetivo de reduzir o escopo desta pesquisa e aumentar sua precisão, limitamos o estudo experimental à avaliação dos valores de ajuste para Sistemas de Informação.

A estratégia adotada para essa tese foi a Pesquisa de Opinião, pois o objetivo era conseguir uma generalização dos resultados através da opinião de especialistas. A metodologia utilizada para planejar e executar este estudo foi baseada na proposta de (WOHLIN *et al.,* 2000) para processos de experimentação.

A ponderação dos fatores de risco foi realizada no sentido de determinar o grau de criticalidade no contexto de um projeto de software. Entende-se por criticalidade o grau de importância com que um fator contribui para o **insucesso** de um projeto, ou seja, quanto mais crítico um fator, maior será a probabilidade de **insucesso** em um projeto, caso ocorra algum problema nessa área.

Definição:

Analisar um conjunto de fatores de risco
Com o propósito de caracterizar
Em relação ao grau de criticalidade e o peso dos fatores
Do ponto de vista de engenheiros de software e gerentes de projetos
No contexto de gerência de riscos de projetos de software

4.4.2. Planejamento do Experimento

- **Contexto**: O estudo foi realizado com a colaboração de professores, alunos de pós-graduação e profissionais de empresas com experiência em desenvolvimento de software e gerência de riscos em projetos. Um conjunto de fatores de risco em projetos de software foi apresentado e um instrumento para realizar a ponderação desses fatores foi entregue aos participantes do estudo que tiveram o seu tempo e ambiente para preencher o formulário, colaborando com o estudo.

- **A Formulação de Hipóteses:** Não havia, nesse estudo experimental, hipóteses a serem confirmadas ou rejeitadas, visto que o objetivo era coletar a opinião dos participantes sobre a prioridade dos fatores de risco.

- **Seleção das Variáveis:** As variáveis independentes eram compostas pelos dez (10) fatores de risco apresentados na pesquisa, ao passo que as variáveis dependentes eram representadas pelo peso (valores de ajuste) de cada fator de risco.

- **Seleção dos Participantes:** O método adotado para a escolha dos participantes foi a *Amostragem* de *Quota*, onde foram selecionados vários elementos da população alvo, que são os gerentes de projeto, estudantes, professores e profissionais de empresas desenvolvedoras de software que forneceram a opinião para os pesos dos fatores. Para cada um desses elementos a escolha foi feita com *Amostragem por Conveniência.* Desta forma, os participantes selecionados representam uma amostra de todos os elementos da população, mas não foram escolhidos de forma aleatória.

- **Projeto do Experimento:** Dentre os modelos de projeto de estudos experimentais, aquele que mais se aplicava a esse estudo era o Agrupamento (*Blocking).* Utiliza-se esse princípio quando existe algum fator que pode influenciar a resposta do estudo, gerando efeitos indesejáveis. No caso específico da Pesquisa de Opinião realizada neste estudo experimental, tinha-se clara a noção de que a experiência de cada participante possuía influência direta sobre os resultados da pesquisa.

Para que esses efeitos fossem minimizados, uma caracterização dos participantes foi realizada segundo quatro perspectivas: a formação acadêmica, o número de projetos em que haviam participado, o tempo de experiência na área e o grau de conhecimento em gerência de risco. A caracterização foi realizada com o formulário expresso no Anexo 2.

Através dessas informações, foi possível determinar um peso específico para cada participante, de acordo com suas características individuais. Em seguida, os participantes foram organizados em três diferentes grupos, a fim de que um único peso pudesse ser aplicado a todos os participantes de um mesmo grupo, tendo-se assim, um resultado mais apurado e próximo da realidade.

Pelo fato de se tratar de uma Pesquisa de Opinião, esse estudo possuía um único fator (que no caso era a criticalidade dos fatores de risco) e um tratamento. Todos os

participantes tiveram que analisar o grau de criticalidade de um projeto em relação a cada um dos fatores, para que a ponderação final pudesse ser determinada.

- **Instrumentação:** Retornando ao objetivo desse estudo experimental, tinha-se 10 fatores de risco que se desejava determinar o peso de cada um deles em relação aos outros fatores, com base na criticalidade de um projeto de software, de acordo com o tamanho do projeto que o participante está desenvolvendo. Portanto, era necessário definir um instrumento que pudesse ordenar a opinião dos participantes e, ao final, obter uma ponderação para ser utilizada na técnica de avaliação do risco de um projeto.

MEISTER e RABIDEU (1965) propuseram três grupos de métodos de classificação de dados: o Observacional, a Base de Dados e o Subjetivo. No caso específico dessa tese, o que se buscava era a opinião de especialistas para a ponderação dos fatores. Portanto, os métodos subjetivos são os mais adequados para o estudo.

Dentre os métodos disponíveis no grupo subjetivo, o escolhido para esse estudo foi a Comparação aos Pares. Esse método é aquele onde dois objetos de estudo são apresentados ao participante, que determina qual deles está presente em maior quantidade ou é mais representativo em relação a um critério ou uma característica específica.

O objetivo de se comparar os objetos aos pares é reduzir a complexidade da avaliação e aumentar a precisão do resultado. Uma das justificativas para a adoção desse método é que a mente humana consegue mais facilmente estabelecer diferenças do que estimar valores absolutos. A segunda vantagem é que a comparação aos pares produz valores redundantes, os quais compensam erros e inconsistências ocorridas no processo de julgamento. Finalmente, através da comparação de cada objeto com todos os demais em questão, o participante é forçado a explicitar uma decisão sobre uma relação entre duas entidades (MIRANDA, 1999).

A maior crítica em relação a esse método é a quantidade de comparações que têm que ser efetuadas no caso de muitos objetos, pois esse valor é determinado pela equação $n(n-1)/2$, onde n é o número de objetos.

A ferramenta escolhida para a realização desse estudo foi o software MACBETH desenvolvido por COSTA e VASNICK (1995). Esta ferramenta foi desenvolvida especialmente para permitir a ordenação de fatores em um processo decisório baseado

em critérios pré-definidos, onde através da comparação aos pares de cada fator envolvido e o julgamento de valor dessa comparação chega-se à ordenação desejada.

Apesar da relativa complexidade do uso da ferramenta, as principais vantagens apresentadas são a verificação de inconsistências no processo de ordenação e a geração automática da escala de ordenação dos fatores. Esta ferramenta permitia que os participantes formalizassem suas preferências de forma semântica.

Estas preferências eram transformadas, através de programação linear, pela ferramenta, em uma escala intervalar que tinham seus valores expressos em percentil. A ferramenta MACBETH também permitia verificar as inconsistências nos votos dos participantes e auxiliava na resolução de conflitos, colaborando para a consistência dos resultados das comparações e reforçando a coerência dos julgamentos, sem contudo influenciar ou restringir a liberdade de julgamento dos participantes.

Durante o estudo foi utilizada uma versão *demo* da ferramenta. No entanto, apesar desta limitação, MACBETH possui, nesta versão, todas as funcionalidades necessárias para a realização do estudo e coleta dos dados. Um exemplo da tela da ferramenta MACBETH utilizada pelos participantes do estudo pode ser observada na figura 4.1.

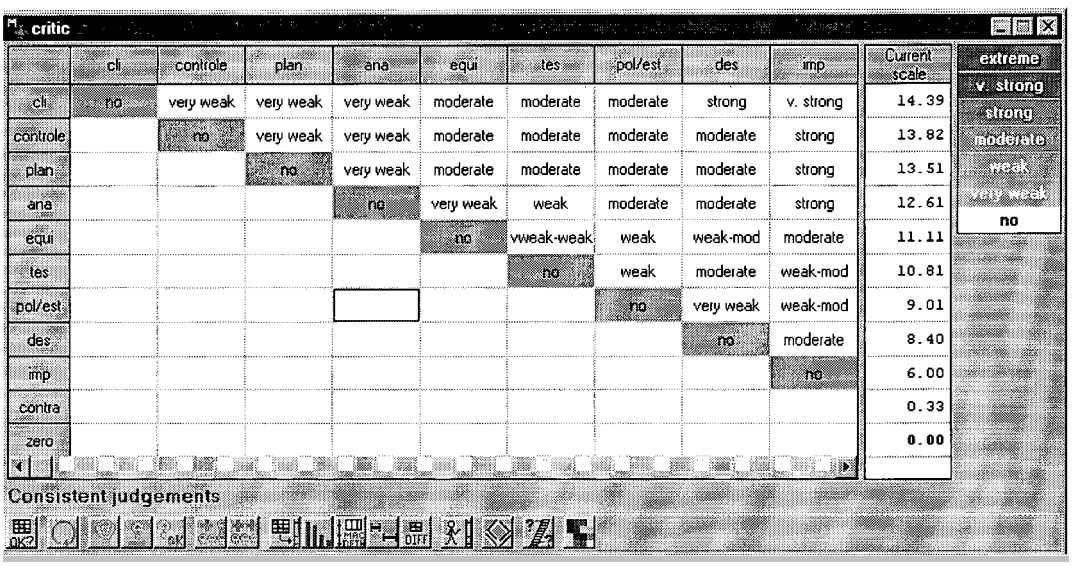


Figura 4.1 – Tela da ferramenta MACBETH

- **Caracterização dos Projetos:** Antes de analisarem os fatores de risco, foi solicitado aos participantes que mantivessem em mente um tamanho (complexidade) específico de projeto. Foi solicitado também, que escolhessem como parâmetro de sua avaliação o tamanho de projeto que mais representava

sua experiência. Os projetos escolhidos foram caracterizados pelo esforço requerido para seu desenvolvimento (medidos em homem/mês) e registrados no formulário de caracterização. Os tamanhos de projetos foram usados na análise dos dados para agrupar a opinião dos participantes, de forma que valores de ajuste diferentes pudessem ser determinados para tamanhos distintos de projetos.

Devido à grande variedade de tipos de projetos existentes e a falta de uma caracterização aceita pela comunidade de software que pudesse ser utilizada nessa tese, adotou-se uma adaptação do modelo apresentado por VILLELA (2004) para a caracterização dos projetos em três grupos distintos em relação ao seu tamanho, sendo eles, pequenos, médios e grandes.

Para a determinação do tamanho de um projeto foi apresentado aos participantes um questionário que buscava identificar o tipo de projeto escolhido, em função da sua duração e do número médio de pessoas envolvidas no projeto. Dessa forma, a fórmula usada para calcular o esforço de um projeto e caracterizá-lo como pequeno, médio ou grande, foi o produto de sua duração, expressa em meses, pelo número médio de pessoas envolvidas em sua equipe de desenvolvimento. A caracterização dos projetos foi realizada com o formulário expresso no Anexo 2.

- **Caracterização dos Participantes:** Era necessário diferenciar as respostas de cada participante, para que o resultado final levasse em consideração suas experiências. A fórmula utilizada para definir o peso de um participante foi adaptada do modelo proposto por (FARIAS, 2002):

$$\text{PT(i)} = \frac{TA(i)}{MedianaTA} + \frac{QP(i)}{MedianaQP} + f(i) + g(i), \text{ onde:}$$

- PT*(i)* é o peso total de um participante *i*;
- *TA(i)* é o tempo de atuação na área de um participante *i*;

- *MedianaTA* é a mediana[2] do tempo de atuação, considerando o tempo de atuação de cada participante do estudo;
- *QP(i)* é a quantidade de projetos que um participante *i* já esteve envolvido;
- *MedianaQP* é a mediana da quantidade de projetos que um participante *i* já esteve envolvido, considerando o número de projetos de cada participante do estudo;
- $f(i)$ é a formação acadêmica na área de Ciência da Computação, sendo considerados os seguintes valores:

  $f(i) = 0$, se o participante for graduando;

  $f(i) = 1$, se o participante possuir graduação completa;

  $f(i) = 2$, se o participante possuir especialização;

  $f(i) = 3$, se o participante possuir mestrado;

  $f(i) = 4$, se o participante possuir doutorado;
- $g(i)$ é o conhecimento do participante em relação à gerência de riscos, sendo considerados os seguintes valores:

  $g(i) = 0$, se for nenhum;

  $g(i) = 1$, se for baixo;

  $g(i) = 2$, se for médio;

  $g(i) = 3$, se for alto;

  $g(i) = 4$, se for excelente;

Dentre as respostas consideradas válidas para o estudo, suspeitava-se que seria obtida uma distribuição heterogênea dos participantes, fato este que deveria ser avaliado pela caracterização dos especialistas e a atribuição do peso de cada um.

Para diminuir essa heterogeneidade, foi realizada a divisão dos participantes em três grupos distintos de acordo com seus respectivos pesos. Esta divisão visava refletir o nível de experiência de cada participante, diminuir a quantidade de pesos que seriam atribuídos a cada voto na pesquisa de opinião e reduzir o desvio padrão dos pesos. Os grupos definidos neste estudo caracterizaram os participantes com Pouca, Média e Grande experiência.

---

[2] Utilizou-se a mediana, e não a média, para evitar que valores extremos distorcessem o valor representativo do Tempo de Atuação e da Quantidade de Projetos.

O procedimento para o cálculo do peso atribuído a cada grupo foi executado da seguinte forma: (*i*) cálculo da média dos pesos totais dos participantes de cada grupo; (*ii*) ao grupo com menor experiência foi atribuído o peso 1; e (*iii*) os grupos subseqüentes tiveram seus pesos calculados pela divisão da média de seus pesos totais pela média do grupo de menor experiência.

- **Grupamento de Participantes e Projetos:** Para que os grupos de experiência dos participantes e de tamanhos de projetos fossem determinados foi necessário distribuir os dados em gráficos. Inicialmente um ponto de corte visual foi selecionado e, a partir deste ponto, várias simulações foram efetuadas com valores próximos aos escolhidos, a fim de verificar quais valores de corte faziam com que os grupos tivessem os menores valores de desvio padrão em relação aos pesos de seus participantes. Desta forma, com os menores valores obtidos, determinou-se o ponto em que os grupos se tornaram mais homogêneos e, portanto, o mais aconselhável para efetuar os cortes.

- **Eliminação de *Outliers*:** Após a ponderação dos votos dos participantes, os dados foram analisados e dados discrepantes foram eliminados a fim de não distorcer os resultados. O critério de eliminação foi determinado pelo cálculo da média e do desvio padrão em relação aos votos dos 10 fatores de risco. Desta forma, todos os valores que se encontravam a uma distância igual ou superior a dois (02) desvios padrão em relação à média dos valores foram eliminados.

- **Piloto do Estudo Experimental:** A fim de se avaliar o instrumento do estudo, foi realizado um teste piloto com um participante para que fosse possível capturar uma opinião sobre os procedimentos adotados e, eventualmente, corrigir erros antes que a execução real do estudo. O participante do piloto, após a realização do teste, respondeu as seguintes perguntas:

  - As instruções para o preenchimento do instrumento são claras? Caso negativo, justifique.
  - As descrições dos fatores são claras e fornecem o nível de informação necessário para diferenciar um fator do outro? Caso negativo, justifique.
  - Relacione os fatores que não tenha compreendido o significado.

- Qual o nível de dificuldade encontrado na utilização da ferramenta?
- Quantas inconsistências foram encontradas durante o preenchimento da planilha?
- Qual o tempo total necessário para realizar o estudo? O tempo deverá ser contado desde o início da leitura das instruções até o término do preenchimento da planilha.
- Há alguma sugestão para a melhoria do instrumento?

As lições aprendidas com esse teste piloto foram a comprovação da necessidade de acompanhamento do entrevistador para a utilização da ferramenta e um melhor detalhamento sobre o que representa cada um dos fatores de risco. Observou-se também, a grande utilidade da verificação automática de inconsistência entre os votos, que é oferecida pela ferramenta MACBETH.

No início do uso da ferramenta, o participante tendia a consultar diversas vezes as instruções sobre o conteúdo de cada fator, o que aumentou o tempo dedicado ao estudo. Contudo, foi observado que esta verificação foi diminuindo e, ao final, praticamente não havia mais necessidade de sua utilização para o uso ideal da ferramenta. Os resultados observados nesta análise estão descritos na seção 4.4.3.

- **Análise Estatística:** Os dados considerados válidos para o estudo foram submetidos a um Teste-T com nível de significância de 0,05 (5%) para verificar se existia diferença estatística entre os valores obtidos e se os resultados podiam ou não ser generalizados. Foram comparadas as médias dos valores de ajuste dos diferentes fatores de risco em relação aos tamanhos dos projetos e a influência da experiência dos participantes em relação aos votos.

- **Validade Interna:** Os participantes foram selecionados por Amostragem de Quota e Conveniência com base nos seus conhecimentos em projetos de software e foram agrupados em três grupos diferentes. O número de participantes que votou para cada tamanho de projeto foi aleatório e nenhuma mortalidade foi observada. Assumiu-se que eles eram representativos da população de desenvolvedores de software. Desta forma, os relacionamentos entre os fatores de risco e a ponderação tiveram como base unicamente a experiência individual de cada participante.

- **Validade Externa:** Considera-se que os participantes eram representativos da população, uma vez que apenas pessoas que desenvolvem Sistemas de Informação foram escolhidas para participar do estudo e dentre elas existiam representantes dos mais diversos tipos de desenvolvedores (alunos, professores e profissionais da indústria).

Cada participante teve a oportunidade de operar a ferramenta para preencher a matriz de comparação no tempo que julgou adequado e no ambiente que melhor lhe convinha. O entrevistador estava presente no momento do preenchimento da matriz apenas para auxiliar no uso da ferramenta, mas não podia influenciar em nenhum julgamento do participante, a fim de que o estudo não fosse tendencioso.

A performance da ferramenta MACBETH foi considerada positiva pelos participantes do estudo através de análise de um questionário qualitativo preenchido após o uso da ferramenta.

- **Validade de Construção:** As questões que compunham os fatores de risco foram todas retiradas de questionários de levantamento de risco de autores reconhecidos na literatura e que realizaram outros estudos experimentais para a sua validação. Portanto, podem ser consideradas representativas para o problema em questão.

Um piloto foi executado antes da execução real do estudo, a fim de que o plano fosse testado, observações fossem feitas em relação à ferramenta MACBETH e melhorias pudessem ser realizadas.

Os participantes foram informados em relação à diferença dos fatores e as características das questões que compunham cada fator, a fim de melhor efetuarem os julgamentos.

Um risco que poderia afetar o resultado final seria o participante tentar adivinhar uma ponderação final dos fatores. Esse risco foi minimizado através da Comparação aos Pares, onde cada par de fatores foi analisado separadamente, sem a influência dos outros fatores, gerando um resultado mais confiável. As inconsistências geradas no julgamento dos participantes foram automaticamente apontadas pela ferramenta e resolvidas pelos próprios participantes, o que permitiu a obtenção de resultados confiáveis. Os valores

percentis de peso dos fatores de risco foram automaticamente calculados pela ferramenta, sem a influência dos participantes.

- **Validade de Conclusão:** Foram realizados Teste-T com nível de significância de 0,05 (5%), que levaram a conclusões confiáveis sobre os valores de ajuste obtidos. O mesmo instrumento foi apresentado aos participantes e, portanto, o estudo pode ser considerado confiável. O risco de heterogeneidade dos participantes foi minimizado quando do agrupamento dos participantes em relação à sua experiência, bem como a eliminação de *outliers* reduziu a possibilidade de mal-interpretação dos dados. Vale ressaltar também, que os resultados só foram buscados em relação à Sistemas de Informação, o que torna a pesquisa mais restritiva, mas com resultados mais confiáveis.

4.4.3. Análise dos dados obtidos no estudo experimental

Nesta seção serão apresentados os resultados obtidos no estudo experimental como forma de determinar os valores de ajuste (pesos) dos dez fatores de risco em questão:

- **Caracterização dos Projetos:** O gráfico da figura 4.1 representa os participantes do estudo e o esforço necessário para o desenvolvimento dos projetos (em homem/mês). Desta forma foi possível realizar a divisão dos projetos e caracterizá-los em Pequenos, Médios e Grandes.

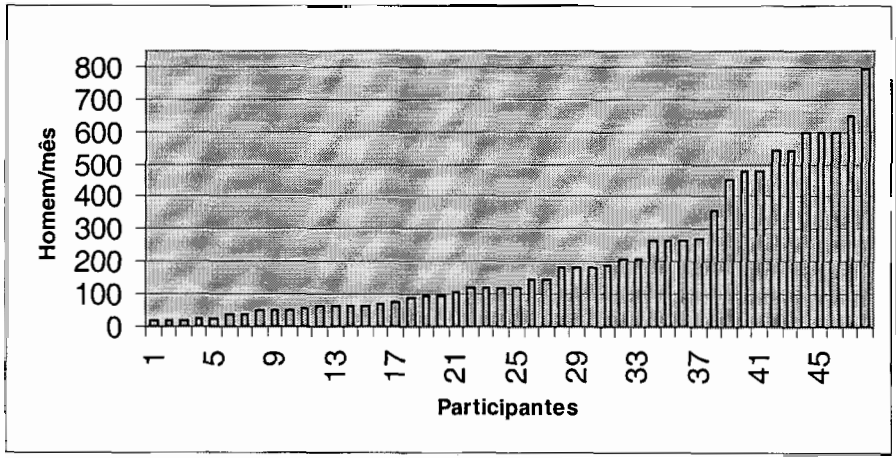


Figura 4.1 – Caracterização dos projetos

Conforme o critério de caracterização dos projetos, os que possuíam menos que 80 homem/mês foram considerados pequenos; os que possuíam entre 80 e 250 homem/mês foram considerados médios e os projetos com mais de 250 homem/mês foram classificados como grandes. A tabela 4.3 apresenta o número de projetos e a média do esforço em cada grupo.

Tabela 4.3 – Projetos e esforço médio por grupo

	Pequenos	*Médios*	*Grandes*
# Projetos	17	18	15
Esforço Médio	45.71	151.78	507.27

- **Caracterização dos Participantes:** Durante a execução do estudo, 50 especialistas foram entrevistados e seus perfis foram computados para uma melhor caracterização e análise dos dados. Todos os participantes concordaram em participar voluntariamente no estudo e assinaram um formulário de consentimento desta participação. Dos 50 participantes, sete (7) possuíam o grau de Doutor em Ciência da Computação, quinze (15) eram Mestres, dezenove (19) possuíam algum tipo de especialização e nove (9) eram apenas graduados.

Dentre os participantes, onze (11) eram pesquisadores acadêmicos ou professores, enquanto que trinta e nove (39) eram profissionais de empresas de desenvolvimento de software da cidade do Rio de Janeiro. Durante o estudo, vinte e seis (26) diferentes instituições foram visitadas.

Trinta e quatro (34) participantes atuavam, no momento da entrevista, como gerentes de projeto, ao passo que dezesseis (16) eram analistas sênior. A média de experiência em desenvolvimento de software de cada participante foi de doze (12) anos, enquanto que, em média, participaram de quatorze (14) projetos.

A figura 4.2 apresenta a distribuição dos participantes em relação a seus pesos, calculados segundo o processo descrito na seção 4.4.2.

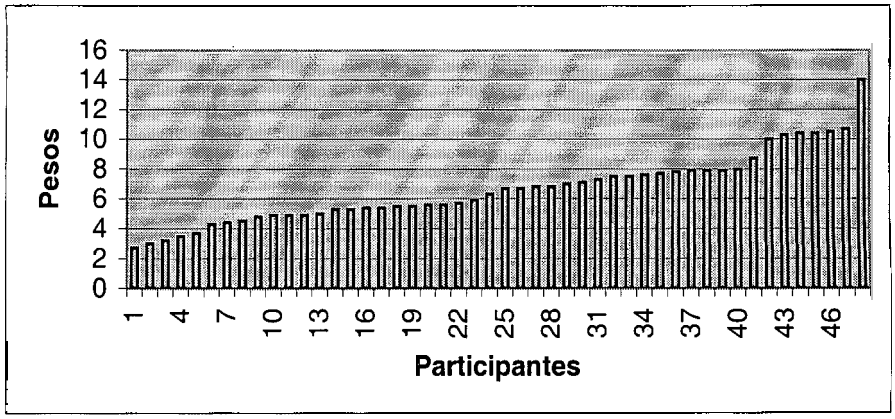


Figura 4.2 – Caracterização dos participantes

Através do grupamento dos participantes, foi possível determinar o peso comum para todos os participantes de um mesmo grupo. Determinou-se que os participantes do grupo com pouca experiência teriam peso 1 (um). Os pesos dos demais grupos foram determinados através da divisão de suas médias pelo valor da média do grupo de pouca experiência. Este processo resultou nos valores expressos na tabela 4.5, que expressa o peso de cada grupo. Estes valores foram usados para ponderar o voto de cada participante e obter os resultados finais do estudo.

Tabela *4.5 – Pesos dos participantes de um grupo*

	Pouca Exp.	*Média Exp.*	*Grande Exp.*
Pesos Comuns	1	1.61	2.63

- **Eliminação de *Outliers*:** Dos 500 dados brutos obtidos durante o estudo, apenas 14 foram rejeitados pelo critério de eliminação de *outliers*. Um valor foi eliminado dos fatores *Codificação*, *Contratos* e *Controle* e dois valores foram eliminados dos fatores *Análise*, *Projeto*, *Equipe* e *Clientes*. Três valores foram eliminados do fator *Planejamento*.

- **Valores de Ajuste dos Fatores de Risco:** Após a ponderação dos votos de todos os participantes, eliminação de *outliers* e separação por tamanho de projeto, as médias dos resultados foram normalizadas a fim de respeitar a regra de que a soma de todos os pesos dos fatores para um projeto de um dado tamanho deve somar 100%.

Os resultados para os valores de ajuste para cada fator e em seus diferentes tamanhos de projeto estão expressos na tabela 4.6 e representam o grau de importância em um projeto de desenvolvimento de software para Sistemas de Informação.

Tabela 4.6 – Valores de ajuste

	Pequeno (%)	*Médio (%)*	*Grande (%)*
Análise	12,36	12,57	10,78
Projeto	8,59	7,52	6,23
Codificação	4,84	4,13	4,00
Teste	7,36	6,17	5,82
Planejamento	15,26	13,04	13,85
Controle	10,39	11,64	12,19
Equipe	11,28	11,53	12,63
Contratos	3,40	5,37	4,60
Política/Estrutura	11,41	12,47	14,11
Clientes	15,11	15,57	15,79

Pode-se observar, pela tabela acima, que os fatores considerados mais importantes para os especialistas são o envolvimento dos clientes durante a realização do projeto, o planejamento do projeto e a análise de requisitos. Por outro lado, fatores como codificação e contratos, de uma maneira geral, parecem ter menos importância e afetam menos os resultados finais de um projeto de software.

- **Análise Estatística:** Os resultados obtidos foram submetidos a um Teste-T em um pacote estatístico para a observação de semelhanças e diferenças significativas entre as médias dos valores de ajuste de cada fator. O objetivo era verificar se existia diferença entre os pesos dos fatores para os diferentes tamanhos de projetos (pequenos, médios e grandes).

As comparações realizadas demonstraram não haver nenhuma diferença significativa para o tamanho de projetos dos fatores *Análise*, *Projeto*, *Codificação*, *Teste*, *Planejamento* e *Contratos*. Os demais fatores – *Equipe*, *Controle*, *Política*/Estrutura e *Clientes* – apresentaram uma pequena, mas significante, diferença entre os valores obtidos para projetos pequenos e grandes, mas nenhuma diferença entre projetos pequenos e médios ou médios e grandes.

Foi observado também que, independente da experiência do entrevistado, não parece haver diferença significativa na opinião destes em relação à importância dos fatores de risco para os diferentes tamanhos de projetos, pois foram comparados os votos

dos participantes com pouca, média e grande experiência para todos os fatores de risco e nenhum resultado significativo foi observado.

Independente da grande semelhança dos resultados estatísticos obtidos decidiu-se manter, para utilização na técnica de avaliação de riscos de projetos proposta, todos os valores calculados neste estudo, pelo fato destes resultados representarem apenas uma primeira abordagem para este tipo de informação em projetos de software.

4.5. Conclusões

Neste capítulo apresentamos uma técnica de avaliação de riscos para projetos de software baseada em riscos sistêmicos e específicos. Esta técnica baseia-se em conceitos econômicos e está estruturada em um questionário elaborado pela agregação de diversas taxonomias presentes na literatura.

No entanto, para que esta técnica seja viável, fez-se necessário planejar e executar um estudo experimental a fim de levantar, junto a especialistas, a importância relativa de fatores de risco em projetos de software. Diversas conclusões e dados puderam ser extraídos da análise dos resultados obtidos neste estudo. Para que os resultados tivessem maior confiabilidade e validade, este estudo restringiu-se a uma categoria específica de sistemas (Sistemas de Informação).

Com base nos riscos quantificados de um projeto, pode-se visualizar uma série de possíveis aplicações para um cenário de Gerência de Projetos de Software. Apresentaremos, no próximo capítulo, uma abordagem que reúne conceitos de gerência de risco em projetos de software e economia, tornando possível um estreitamento cada vez maior entre a Economia e a Engenharia de Software, no sentido de prover uma gerência de projetos mais eficiente e realista.

5

ANÁLISE DE UMA CARTEIRA DE PROJETOS BASEADA EM RISCOS

"Pessoas preferem assumir riscos com bases em probabilidades conhecidas".
Daniel Ellsberg

5.1. Introdução

A gerência de riscos em projetos de software tem se tornado uma preocupação constante, não só para gerentes, mas também para executivos envolvidos com objetivos estratégicos de organizações que desenvolvem ou dependem de projetos de software. Portanto, diversos estudos vêm sendo realizados para melhor compreender e controlar os efeitos negativos causados por incertezas em projetos que podem levar a seu fracasso ou a prejuízos para uma organização.

No capítulo 3 foi observado que a Engenharia de Software é uma atividade econômica como qualquer outra e, como tal, deve ser avaliada não só em relação aos seus problemas técnicos, mas também quanto à possibilidade de gerar valor para a organização desenvolvedora. Para tanto, deve haver uma justificativa para a utilização de recursos nos projetos, que, normalmente, é feita em função de seus custos, retornos e riscos envolvidos na escolha das possíveis opções. Em desenvolvimento de software, os custos compõem-se dos gastos com a equipe, hardware, software e infra-estrutura necessária durante o projeto. Por sua vez, os retornos são representados pelos lucros esperados ou pelos benefícios proporcionados pelo sistema.

No capítulo 4, uma técnica para calcular o nível de risco em projetos de software foi apresentada, visando obter informação sobre a possibilidade de insucesso de um projeto, onde este insucesso é representado pela não conclusão do projeto e conseqüente falha em alcançar os objetivos propostos. Gerentes e executivos podem utilizar o nível de risco de projetos em diversas atividades relacionadas ao processo de decisão. Neste capítulo apresentamos uma proposta para calcular quanto capital uma organização desenvolvedora de software pode ganhar ou perder em função de uma carteira de projetos, assumindo que alguns destes projetos podem falhar devido aos riscos incorridos durante o processo de desenvolvimento. Assim como na técnica de

cálculo do nível de risco, esta proposta se baseia na aplicação de conceitos de Economia à Engenharia de Software, mais especificamente, da análise de risco de crédito.

Além desta introdução, este capítulo é composto por quatro seções. A seção 5.2 descreve uma comparação entre os conceitos de risco de crédito utilizados em finanças para a avaliação de uma carteira de empréstimos e as características de uma carteira de projetos de desenvolvimento de software. A seção 5.3 descreve os conceitos básicos sob modelagem de Risco de Crédito. Na seção 5.4 a ferramenta *RISICARE* é apresentada e suas funcionalidades são explicadas. Finalmente, algumas conclusões são feitas na seção 5.5.

5.2. Riscos de crédito e projetos de software

Buscando aproximar os domínios da Engenharia de Software e da Economia, faremos uma analogia entre o desenvolvimento de sistemas e o mercado financeiro. Nesta analogia, um conjunto de projetos de software é analisado como uma carteira de empréstimos a longo prazo.

Instituições financeiras fornecedoras de empréstimos possuem um especial interesse em determinar a probabilidade de perder ou ganhar uma determinada quantia em dinheiro em função da composição de sua carteira de empréstimos. Esta informação é utilizada para limitar as possíveis perdas financeiras em virtude de eventos de *default.* Um evento de *default* ocorre quando um devedor não consegue cumprir suas obrigações para com a instituição financeira que lhe concedeu um empréstimo.

A análise de riscos de crédito é um conjunto de procedimentos, modelos e técnicas que tem como objetivo determinar, com um determinado nível de certeza, a perda potencial esperada pela manutenção de uma carteira de empréstimos. Ela se baseia na atribuição de uma probabilidade de *default* (ou seja, a probabilidade de ocorrer um evento de *default*) para cada devedor. A análise também requer o retorno esperado em cada empréstimo, uma vez que lucros provenientes de empréstimos bem-sucedidos podem compensar perdas geradas por débitos não saudados.

Projetos de software, assim como empréstimos, possuem probabilidades incertas de sucesso, representadas pelo seu nível de risco. Um evento de *default* para um projeto de software pode ser associado ao seu cancelamento, independente das causas que levaram a este fato. Projetos de software também possuem um custo, que é o capital empenhado em seu desenvolvimento, como o montante financeiro concedido em um empréstimo. Em certas ocasiões, este capital será perdido caso o projeto falhe, assim

como lucros serão obtidos se os projetos alcançarem seus objetivos. Para completar a analogia, projetos bem sucedidos geram lucros para a organização desenvolvedora e podem compensar, até certo nível, perdas causadas pela falha de outros projetos.

Portanto, organizações desenvolvedoras de software podem analisar sua carteira de projetos para determinar a probabilidade de perdas e ganhos financeiros, a fim de manter o negócio economicamente viável, mesmo após um (limitado) número de falhas de projetos. Baseado nesta analogia, apresentamos, na próxima seção, uma aplicação dos conceitos de risco de crédito adaptada para o contexto de avaliação de projetos de software.

5.3. Modelagem de risco de crédito

Riscos de crédito podem ser definidos como a probabilidade de um devedor falhar em cumprir sua obrigação em relação a termos previamente acordados. Sua medida permite que uma empresa mantenha sua exposição ao risco, para cada tipo de devedor, em um nível aceitável. As principais abordagens para análise de risco de crédito presentes na literatura financeira são os modelos *Credit Metrics* (J.P MORGAN, 1997), *Credit Risk+* (CREDIT SUISSE FIRST BOSTON, 1997) e o KMV (K.M.V. CORPORATION, 2003). Cada um desses modelos requer um conjunto distinto de parâmetros e são baseados em premissas particulares.

O modelo KMV analisa a probabilidade de inadimplência de uma empresa com intenção de obter crédito em função do valor de mercado de suas ações, dependendo da existência de um mercado líquido[3] para estas ações, cujo valor seria determinado pela negociação entre investidores. No contexto de projetos de software, o valor envolvido na negociação é determinado pelos clientes e pela organização responsável pelo desenvolvimento. Por outro lado, os riscos de projeto são endógenos, ocorrendo por conta de aspectos incertos do processo de desenvolvimento ou do produto. Assim, estes riscos não podem ser determinados por ativos negociados em um mercado aberto[4]. Dadas estas restrições, o modelo KMV não é diretamente aplicável ao problema proposto.

O modelo *Credit Metrics* exige um mercado para negociar empréstimos e ativos de proteção para garantia desses empréstimos, tais como *swaps de crédito* e *opções de default* (BARBANSON, 2004). Mais uma vez, como os riscos de um projeto de software são (em sua maioria) endógenos, não existem ativos negociáveis para proteger os

[3] Um mercado líquido é aquele onde as ações de uma ou mais empresas estão suficientemente distribuídas entre investidores de tal forma que nenhum investidor em particular pode influenciar no valor das ações.
[4] Um mercado aberto é aquele onde qualquer pessoa ou empresa pode comprar ou vender ativos.

projetos de falhas em relação a aspectos relacionados ao seu desenvolvimento. Assim, o modelo *Credit Metrics* também não é diretamente aplicável à nossa analogia.

Por sua vez, o modelo *Credit Risk+* não necessita de um mercado para negociação de ativos de proteção e a única informação necessária em relação a uma contraparte é a sua probabilidade de *default* para cada empréstimo. Portanto, sua aplicação para análise de carteira de projetos de software é mais adequada.

O modelo *Credit Risk+* é composto por um conjunto de três modelos distintos, cada um construído sobre o modelo precedente e, portanto, adicionando detalhes e complexidade ao seu processo de solução. O primeiro modelo é baseado em uma fórmula analítica que calcula a distribuição de probabilidade de perdas e ganhos de uma carteira de empréstimos, onde cada empréstimo é caracterizado pela quantidade de dinheiro envolvida, o retorno esperado e o risco inerente desta negociação.

O modelo assume que todo o dinheiro emprestado é perdido em caso de um evento de *default.* Além disto, os devedores são normalmente agrupados de acordo com a sua habilidade em cumprir com suas obrigações, sendo que para cada grupo é atribuída uma probabilidade de *default.* Esta classificação em grupos é possível no mercado financeiro devido à grande quantidade de devedores e dados históricos existentes. Tal agrupamento no contexto de projetos de software torna-se praticamente inviável, pelo fato de uma empresa, normalmente, não possuir tantos projetos sendo desenvolvidos simultaneamente em categorias distintas, a fim de atribuir uma probabilidade de falha para todo o grupo, baseada em dados estatísticos.

O segundo e o terceiro modelos do *Credit Risk+* também levam em consideração a variabilidade da probabilidade de *default* e a distribuição de devedores entre setores econômicos. Como a técnica de cálculo de nível de risco proposta no capitulo 4 não prevê a variabilidade do valor encontrado e a impossibilidade de se estabelecer uma distribuição de projetos por categorias dentro do contexto desta tese, estes dois últimos modelos não foram considerados.

A proposta para projetos de software apresentada por esta tese é baseada no primeiro modelo do *Credit Risk+*, que lida com probabilidades fixas de *default*, ou seja, não leva em consideração a variabilidade destas probabilidades. Mapeando este modelo para o contexto de projetos de software, a abordagem proposta assume que cada projeto tem uma probabilidade fixa de ser mal-sucedido, representada pelo nível de risco do projeto, que pode ser calculado pela técnica apresentada no capítulo 4.

Com o objetivo de estabelecer o capital necessário para manter uma carteira de empréstimos, o *Credit Risk+* utiliza a *Distribuição de Poisson* (JOHNSON e BHATTACHARYYA, 1977) para descrever o número de eventos de *default* que podem ocorrer em uma carteira de empréstimos, dado o número de devedores de cada grupo. De acordo com o número de *defaults* em cada grupo, os lucros esperados e a quantidade de dinheiro investida em cada empréstimo, o modelo estima o retorno esperado para a carteira e sua variabilidade em relação a cada grupo.

Ao mapear conceitos relacionados a riscos de crédito para carteiras de projetos de software, observamos que não é usual uma organização possuir um número elevado de projetos em sua carteira, impossibilitando, portanto, a formação de grupos de risco, conforme realizado no mercado financeiro. Portanto, ao invés de determinar a probabilidade de *default* para cada grupo (utilizando a *Distribuição de Poisson*), calculamos a probabilidade de *default* para cada projeto utilizando a *Distribuição de Bernoulli* (JOHNSON e BHATTACHARYYA, 1977) para descrever este comportamento.

A *Distribuição de Bernoulli* é a mais simples função de distribuição de probabilidade, recebendo como entrada um único parâmetro (uma probabilidade *p*) e retornando dois possíveis resultados: positivo ou negativo, sendo o primeiro tão freqüente quanto a probabilidade atribuída a *p*.

Ao substituir a função de distribuição de probabilidade, a fórmula analítica do modelo *Credit Risk+*, baseada na *Distribuição de Poisson*, não é mais aplicável. Portanto, propomos o uso da Simulação de Monte Carlo (VOSE, 1996) para estimar a distribuição de probabilidade de perdas e ganhos de uma carteira de projetos de software, baseado na proposta de alteração do modelo *Credit Risk+* elaborada por (ARAGÃO *et al.*, 2003).

A substituição das funções de distribuição é justificável pelo fato de que a *Distribuição de Poisson* converge para a de *Bernoulli* quando os grupos são formados por um único elemento. Este fato ocorre porque a *Distribuição de Poisson* mede a probabilidade de ocorrência de N eventos em M elementos, sendo $M \geq N$. Se tivermos apenas um elemento (M=1), duas possibilidades podem ocorrer: positivo (um evento) ou negativo (nenhum evento). Considerando que a probabilidade de um evento é *p*, a *Distribuição de Bernoulli* descreve exatamente a situação de projetos de software, onde não existem grupos com probabilidades de *default,* mas projetos isolados que podem falhar ou ser bem-sucedidos.

O modelo *Credit Risk+* assume algumas premissas que precisam ser consideradas para estabelecer os limites de nossa abordagem. Por exemplo, o modelo presume que

todos os custos são incorridos quando o empréstimo é fornecido e os retornos são recebidos quando os empréstimos são pagos. Esta realidade pode não ser verdadeira para todos os projetos de software, pois alguns possuem previsões de fluxo de caixa (positivos e negativos) que ocorrem ao longo do ciclo de vida do projeto. A ampliação do modelo proposto está fora do escopo desta tese e constitui uma perspectiva futura de ampliação deste trabalho.

A tabela 5.1 apresenta uma comparação entre as premissas assumidas para o contexto do mercado financeiro e de projetos de software.

Tabela 5.1 – Premissas financeiras e de software

Mercado Financeiro	Projetos de Software
Cada empréstimo possui seu próprio risco	Cada projeto possui seu próprio risco
Estados do devedor: inadimplente ou não	Estados do projeto: cancelado ou não
O valor envolvido na operação de empréstimo é acordado antes da operação ser realizada	Os custos de projeto são definidos antes de sua realização
O pagamento do empréstimo é realizado ao fim do período acordado	O pagamento do projeto é feito após sua conclusão
Em caso de evento *default* as perdas possuem um valor fixo (o valor acordado para o empréstimo), desconsiderando-se a taxa de retorno prevista	Em caso de falha do projeto, a perda possui um valor fixo (os custos do projeto), independente do retorno esperado
Não existe correlação entre os eventos *default,* exceto por fatores externos que podem afetar os devedores de maneira similar	Não existe correlação entre as falhas dos projetos, exceto por riscos externos (sistêmicos) que podem afetar os projetos de maneira similar

Assumindo estas premissas e por meio das simulações, é possível estabelecer a distribuição de probabilidades de perdas e ganhos para qualquer número de projetos de uma carteira de uma organização, baseado apenas na probabilidade de falha dos projetos (nível de risco), nos seus custos de desenvolvimento e nos retornos esperados. O cálculo para a distribuição de probabilidade ocorre da seguinte forma:

- O simulador gera um valor aleatório para cada projeto envolvido na simulação segundo a Distribuição de *Bernoulli.* Estes números representam a possibilidade de sucesso ou fracasso do projeto da simulação corrente;
- Caso o projeto seja definido como uma falha (nível de risco atingido), o resultado da operação é a perda dos custos determinados para este projeto;

- Se o projeto for bem-sucedido, obterá um lucro e o resultado da operação é calculado pela diferença entre os retornos esperados e os custos do projeto;
- Ao final de cada simulação, os resultados das avaliações dos projetos (custos ou lucros) são somados. Este resultado representa a quantidade de dinheiro perdida ou ganha pela empresa no desenvolvimento dos referidos projetos;
- A simulação é repetida diversas vezes a fim de fornecer uma melhor estimativa dos retornos esperados para a carteira de projetos. Para garantir a convergência do processo de cálculo, sugere-se que as simulações sejam repetidas por, pelo menos, 10.000 ciclos;
- Ao final de todas as simulações, as perdas ou ganhos são agrupados e mapeados para o domínio da freqüência. Nesta tese adotou-se pela divisão do espaço entre a pior perda e o melhor ganho em 20 intervalos fixos, onde as freqüências de ocorrência em cada intervalo são representadas por barras. O mapeamento das freqüências é realizado pela contagem do número de perdas e ganhos gerados pela simulação em cada intervalo;
- Finalmente, a percentagem de perdas e ganhos em cada grupo é calculada em relação ao número total de simulações realizadas e estes valores são plotados em um gráfico de barras de maneira cumulativa.

A fim de ilustrar a abordagem proposta, tomaremos como exemplo uma empresa fictícia com uma carteira de projetos descrita pela tabela 5.2, onde cada projeto possui o seu nível de risco, calculado pela técnica proposta no capítulo 4, os custos planejados para seu desenvolvimento e o retorno esperado.

Tabela 5.2 – Projetos com níveis de risco, custos e retornos

Projeto	Nível de Risco	Custo	Retorno
#1	10%	$ 10 K	$ 25 K
#2	20%	$ 20 K	$ 35 K
#3	15%	$ 30 K	$ 45 K
#4	25%	$ 40 K	$ 50 K
#5	20%	$ 50 K	$ 75 K

Após aplicar a simulação de Monte Carlo 10.000 vezes com os parâmetros fornecidos no exemplo, a distribuição de probabilidade cumulativa de perdas e ganhos que podem ser obtidas pela empresa está representada na figura 5.1.

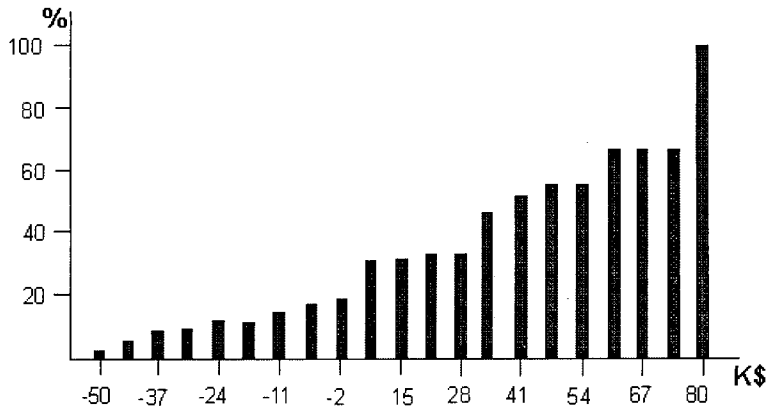


Figura 5.1 - Distribuição de probabilidade

Os resultados expressos na figura 5.1 são apresentados de forma cumulativa, ou seja, as barras verticais representam a soma da probabilidade da quantia anterior mais a probabilidade do valor sob observação. Desta forma, os resultados do processo de simulação podem ser interpretados de duas maneiras: (*i*) esta empresa tem aproximadamente 70% de chances de ganhar cerca de $70.000,00 ou menos; ou (*ii*) 30% (100 – 70) de chances de ganhar mais de $70.000,00.

Estes resultados podem não ser aceitáveis para a organização, pois, supondo que a empresa necessite de uma quantia mínima de $70.000,00 para se manter, ela gostaria de ter mais do que 30% de chances de ganhar esta quantia. Portanto, um gerente pode desejar alterar os parâmetros dos projetos para realizar outras simulações e verificar se existem opções mais compatíveis com as metas ou as possibilidades da empresa. Várias possibilidades podem ser analisadas para a alteração dos parâmetros:

- Diminuir o nível de risco e diminuir o custo;
- Tentar aumentar o retorno ou diminuir o nível de risco;
- Apenas diminuir os custos;
- Diminuir o nível de risco e aumentar o custo, aplicando um plano de contingência;
- Apenas tentar aumentar o retorno;
- Eliminar um ou mais projetos da carteira.

Como exemplo de variação de parâmetros, temos a situação onde, após uma análise das possibilidades a serem feitas na carteira de projetos, gerentes ou executivos da organização resolvem alterar parâmetros conforme pode ser observado na tabela 5.3.

Tabela 5.3 – Projetos com níveis de risco, custos e retornos alterados

Projeto	Nível de Risco	Custo	Retorno
#1	10%	$ 10 K	$ 30 K
#2	18%	$ 22 K	$ 35 K
#3	15%	$ 25 K	$ 45 K
#4	20%	$ 45 K	$ 60 K
#5	15%	$ 55 K	$ 80 K

Realizando mais uma vez a simulação, podemos obter uma distribuição semelhante à apresentada na figura 5.2.

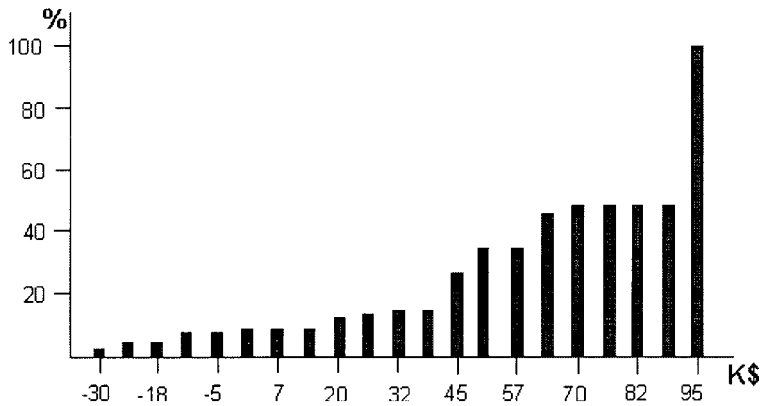


Figura 5.2 - Distribuição de probabilidade alterada

Pode-se observar que a probabilidade da empresa obter ganhos superiores a $70.000,00 passou de 30% para aproximadamente 50%, ou seja, um cenário mais favorável do que a situação anterior, tendo, portanto, mais chances de ganhar a quantia necessária para se manter.

Vale ressaltar que esta abordagem deve ser utilizada, preferencialmente, por empresas que possuam diversos projetos sendo desenvolvidos simultaneamente, pois desta forma, pode-se obter uma maior possibilidade de variação de parâmetros e uma análise mais proveitosa em relação ao que a abordagem proporciona.

Também é possível dizer que, caso uma empresa de grande porte tenha um número considerável de projetos de forma que seja viável formar grupos com números suficientes de projetos para se criar uma distribuição de probabilidades, a fórmula original do modelo *Credit Risk+* e a *Distribuição de Poisson* poderiam ser utilizadas para chegar a resultados semelhantes, conforme preconiza o modelo.

5.4. A ferramenta *RISICARE*

Em virtude da grande quantidade de questões que precisam ser respondidas, das operações necessárias para o cálculo do nível de risco dos projetos e das simulações que geram a distribuição de probabilidade de perdas e ganhos de uma carteira de projetos, torna-se difícil para um gerente realizar estas atividades manualmente. Portanto, buscando apoiar estas atividades, a ferramenta *RISICARE* foi concebida e implementada.

Como o objetivo da criação da ferramenta era, simplesmente, demonstrar a viabilidade das abordagens propostas nesta tese, decidiu-se adotar arquiteturas, interfaces, linguagens e dispositivos de persistências os mais simples possíveis. A arquitetura básica da ferramenta é do tipo monousuário tendo como sua concepção a criação de interfaces gráficas que acessam uma base de dados única utilizada por todo o sistema por meio de um *Módulo de Dados*. A figura 5.3 ilustra a arquitetura básica da ferramenta.

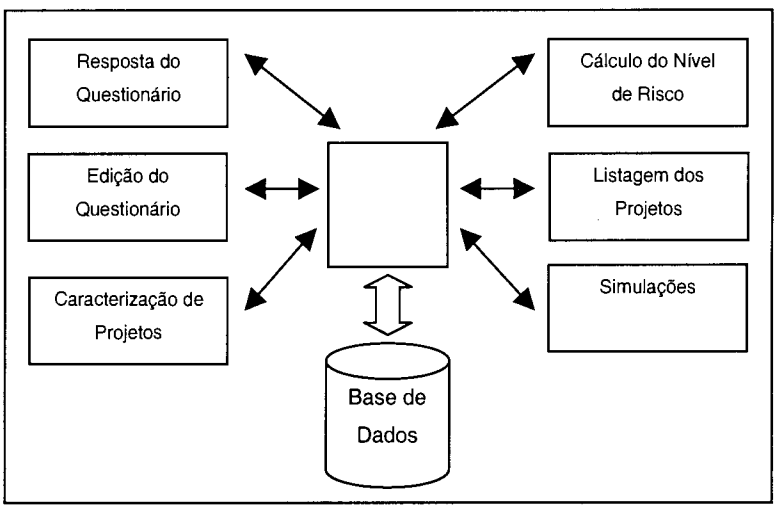


Figura 5.3 – Arquitetura da ferramenta *RISICARE*

A ferramenta foi construída utilizando o paradigma da orientação a objetos e a linguagem de programação empregada foi a Object Pascal disponibilizada pela ferramenta Delphi. Todas as funcionalidades implementadas utilizaram componentes já existentes nesta ferramenta, bem como a sua base de dados nativa (Paradox). A opção pela escolha desta base de dados justifica-se pela simplicidade da ferramenta e pela facilidade de integração entre as interfaces e a base de dados, proporcionada pelos próprios componentes do Delphi. As classes básicas do sistema foram agrupadas em três pacotes a fim de reunir funcionalidades semelhantes: *Persistência, Riscos* e *Simulações.*

O pacote de persistência é responsável por agrupar classes do sistema que geram dados que necessitam ser armazenados na base de dados. Este pacote, por sua vez, é dividido em dois subpacotes. O primeiro é responsável por reunir as classes que persistem dados relativos aos projetos e as respostas das questões vinculadas aos projetos. O segundo subpacote reúne classes que armazenam dados relativos às simulações. As figuras 5.4 e 5.5 ilustram, respectivamente, os diagramas de classes dos subpacotes de persistências de projetos e simulações.

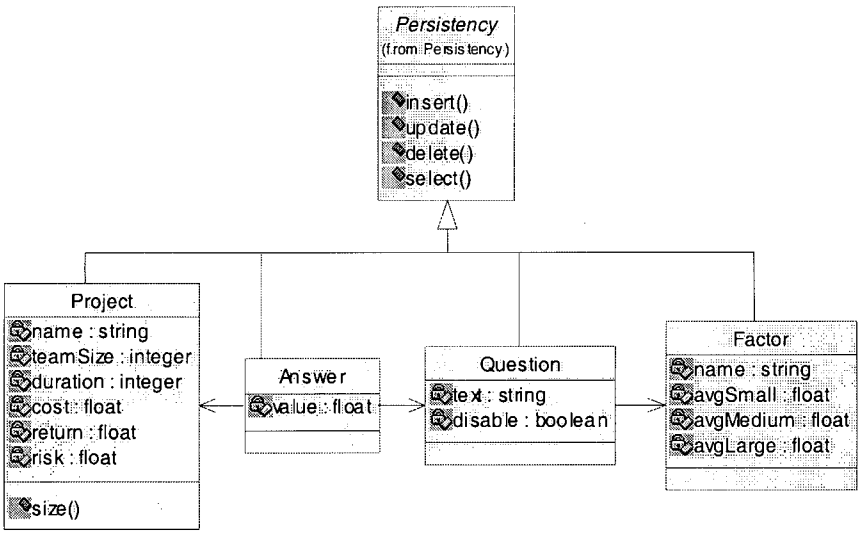


Figura 5.4 – Diagrama de classes de persistência de projetos

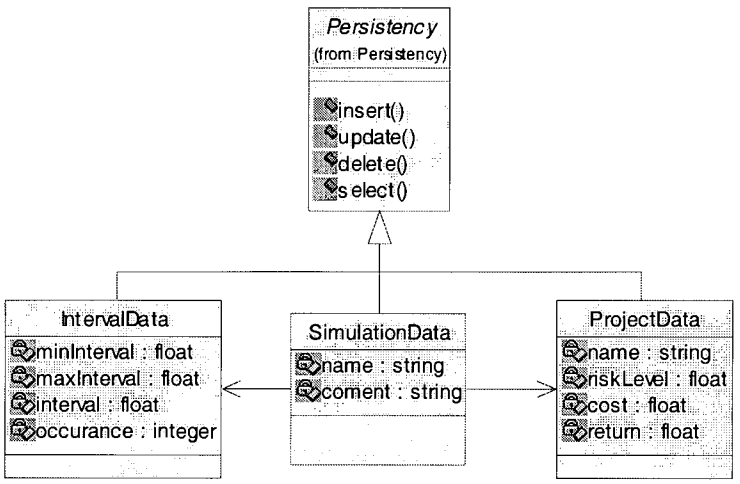


Figura 5.5 – Diagrama de classes de persistência de simulações

O Pacote de riscos agrupa classes que permitem que o nível de risco dos projetos sejam calculados e está representado pela figura 5.6.

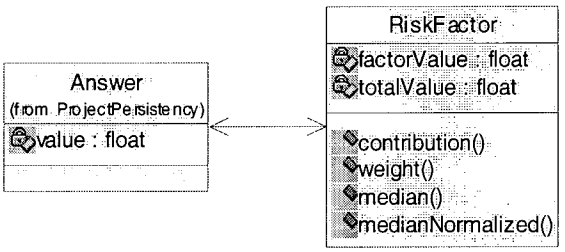


Figura 5.6 – Diagrama de classes do pacote de riscos

O Pacote de simulações reúne classes que permitem que as simulações sejam realizadas e o gráfico de distribuição de probabilidades de perdas e ganhos de uma carteira de projetos seja elaborado. Este pacote está representado pela figura 5.7.

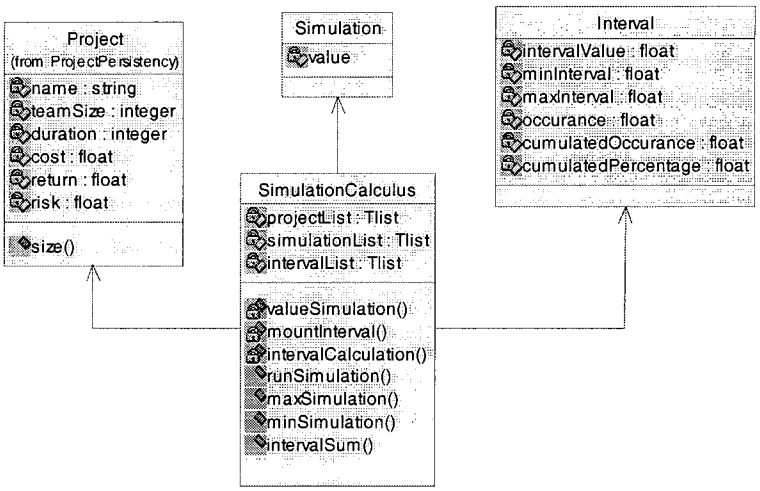


Figura 5.7 – Diagrama de classes do pacote de simulações

5.4.1. Caracterização de um projeto

A caracterização de um projeto de desenvolvimento de software é o primeiro passo para que o processo de cálculo seja realizado. Para a caracterização de um projeto é necessário fornecer dados tais como: Nome, Tamanho da Equipe, Duração, Custo e Retorno. A figura 5.8 ilustra a tela de caracterização de projetos.

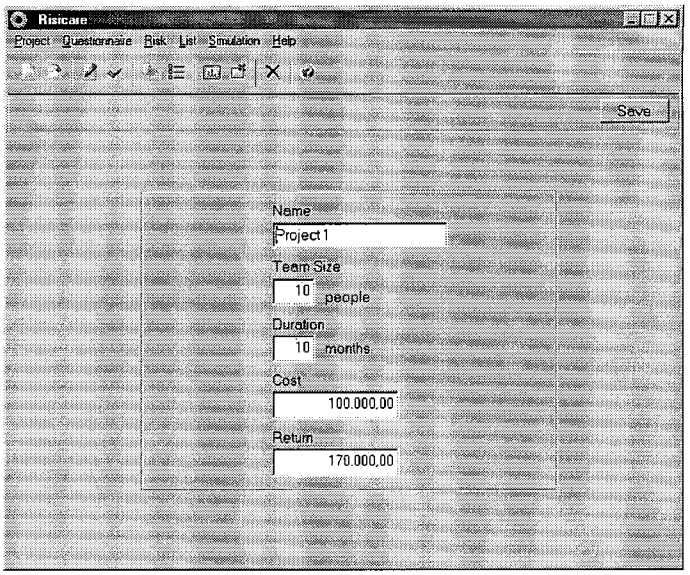


Figura 5.8 – Tela de caracterização de projetos

5.4.2. Questionário de identificação de riscos

Após a caracterização de seus projetos, o usuário tem acesso a duas novas funcionalidades: a edição e o preenchimento dos questionários de identificação de riscos. Editar o questionário de identificação de riscos significa inserir questões na lista de perguntas, editar as perguntas já existentes e salvar estas informações no banco de questões. Na tela de edição de questionário existem dez palhetas contendo os fatores de risco identificados no estudo experimental.

Ao inserir uma nova questão, uma caixa de edição de texto é disponibilizada para que o usuário possa digitar o enunciado de uma nova questão referente ao fator selecionado. Todas as questões existentes no banco de questões serão exibidas para o usuário na tela de preenchimento do questionário. Como o banco de questões pode ser muito grande e conter riscos que dificilmente farão parte do contexto de uma empresa, durante a edição do questionário, o usuário poderá desabilitar determinadas questões. Desta forma, questões desabilitadas não estarão disponíveis para preenchimento e não serão consideradas no cálculo do nível de risco dos projetos. A figura 5.9 exibe uma tela de edição de questão.

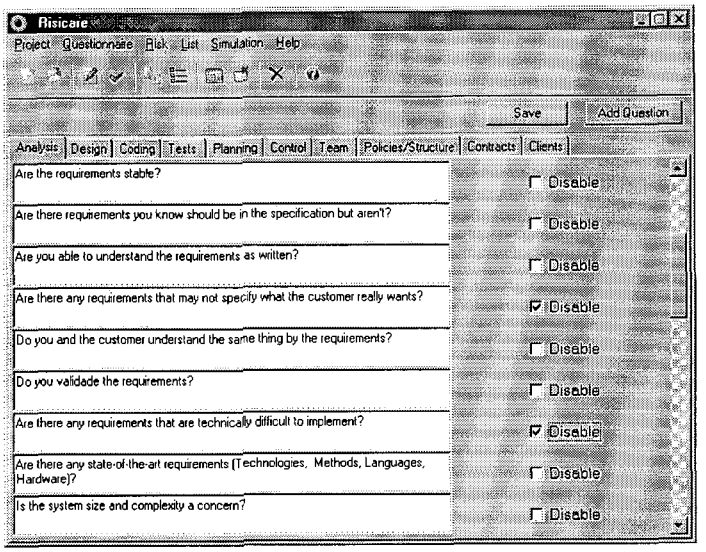


Figura 5.9 – Tela de edição de questões

Durante o preenchimento do questionário existem seis possíveis respostas para cada questão, indicando o grau de risco que ela representa para o projeto: nenhum, muito baixo, baixo, médio, alto, muito alto e não relevante. A opção "Nenhum Risco" indica que, apesar da questão ser relevante para o projeto que está sendo analisado, ela não representa nenhum risco para o sucesso do projeto. A opção "Não Relevante" indica que o problema representado pela questão não tem nenhuma relevância para o projeto, devendo ser desconsiderada. As questões marcadas como não relevantes não são computadas quando do cálculo do nível de risco. A figura 5.10 ilustra a tela de preenchimento de questões.

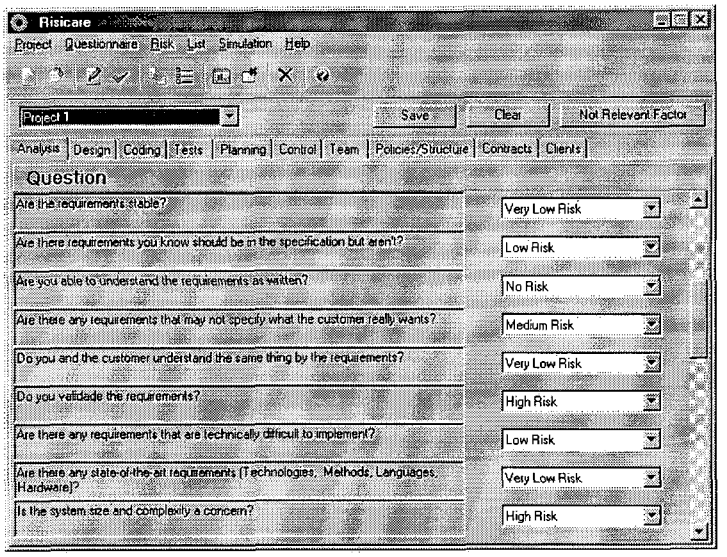


Figura 5.10 – Tela de preenchimento de questões

5.4.3. Nível de risco

Após responder ao questionário de identificação de riscos, o usuário pode calcular o nível de risco de um projeto específico e visualizar a contribuição de cada fator para este nível de risco, conforme a técnica apresentada no capítulo 4. Caso se deseje calcular o nível de risco de vários projetos simultaneamente e não visualizar a contribuição dos fatores, o usuário pode realizar esta operação diretamente na tela de Listagem de Projetos que será apresentada na próxima seção.

Antes que o nível de risco seja calculado, *RISICARE* verifica se todas as questões disponíveis no questionário foram respondidas pelo usuário. Caso isto não seja verdadeiro, o processo não é concluído. O procedimento de verificação das respostas foi implementado para garantir que todos os projetos tenham seu nível de risco calculado com base no mesmo questionário. Em caso de respostas incompletas, o usuário deverá retornar ao preenchimento de questionário para responder às questões ausentes e salvá-las no sistema.

Outras duas verificações são efetuadas antes do cálculo do nível de risco: (*i*) se existe algum fator que esteja completamente assinalado como Não Relevante e (*ii*) se existe algum fator onde todas as respostas sejam Nenhum Risco. Fatores com estas características não são considerados para o cálculo.

As respostas fornecidas pelo usuário dos fatores considerados válidos para o cálculo são convertidas para números, através do processo de codificação descrito na seção 4.3. Após esta conversão, o processo de cálculo de nível de risco é iniciado conforme a técnica descrita na seção 4.3: inicialmente são calculadas as medianas de todos os fatores de risco com base nas respostas ao questionário. Em seguida, cada mediana é dividida por 5 (cinco) e estes valores são multiplicados pelos respectivos valores de ajuste de cada fator. Finalmente a contribuição de cada fator é somada para que se obtenha o nível de risco do projeto.

Vale ressaltar que os valores de ajuste utilizados são os descritos na tabela 4.1, obtidos no estudo experimental, caso todos os fatores de risco sejam relevantes para o projeto. Em situações onde pelo menos um dos fatores não seja considerado relevante para o projeto, *RISCARE* recalcula estes valores dinamicamente, levando sempre em consideração que o somatório dos pesos deve ser igual a 100%. Após o cálculo, o usuário tem a opção de levar esta informação para a Listagem de Projetos. A figura 5.11 ilustra a tela de nível de risco preenchida.

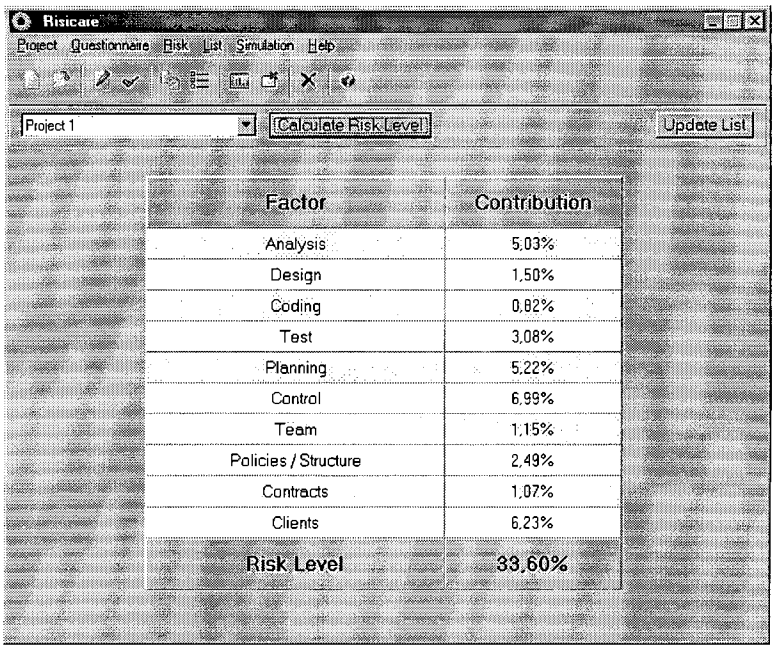


Figura 5.11 – Tela de nível de risco

5.4.4. Listagem de projetos

A Listagem dos projetos permite que o usuário tenha acesso a todos os projetos cadastrados no sistema, apresentando informações sobre o seu *status* em relação ao preenchimento do questionário, o custo de desenvolvimento, o retorno previsto e o nível de risco, caso tenha sido calculado. Estas informações são exibidas, por serem os dados necessários para que a simulação e cálculo da distribuição de perdas e ganhos da carteira sejam realizados.

Três possíveis operações podem ser realizadas nesta tela: (*i*) excluir os projetos, (*ii*) calcular o nível de risco dos projetos e (*iii*) transportar os projetos para a área de simulação. Deve-se atentar para os fatos de que, apenas os projetos que estiverem com o status do questionário na condição de preenchido poderão ter seu nível de risco calculado, e apenas os projetos com o nível de risco calculado podem ser transportados para a área de simulação. A figura 5.12 exibe a tela de listagem de projetos.

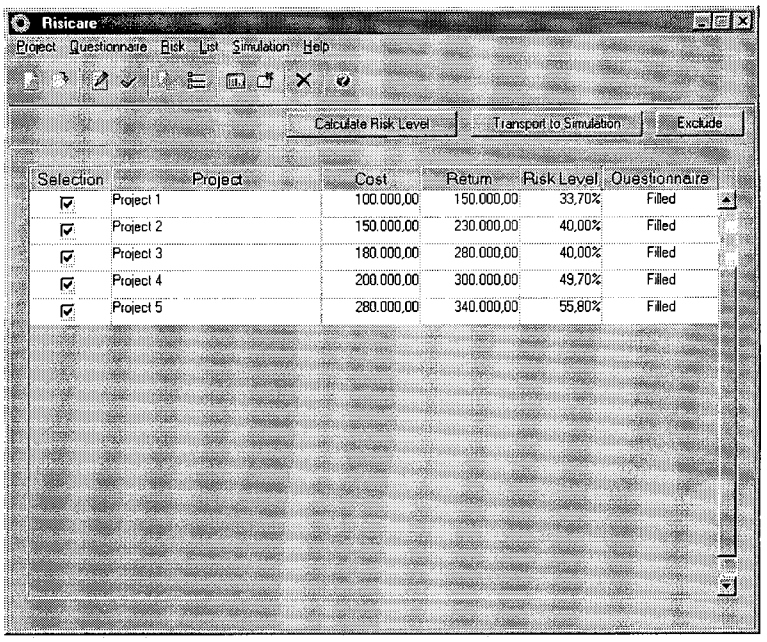


Figura 5.12 – Tela de listagem de projetos

5.4.5. Simulação

Após os projetos serem transportados para a área de simulação, o usuário pode realizar as simulações para que a distribuição de perdas e ganhos dos projetos seja calculada. A fim de aumentar a confiabilidade dos resultados, um número mínimo de 10.000 rodadas de simulação foi pré-estabelecido. Por outro lado, para que o tempo de processamento não seja demasiado elevado, o número máximo de rodadas foi limitado a 1.000.000.

Após o término da simulação é exibido um gráfico com a distribuição de probabilidade de perdas e ganhos da carteira de projetos. Caso o usuário deseje realizar algum comentário sobre a simulação, poderá utilizar a área de texto localizada na parte inferior da tela. Ao salvar a simulação, serão armazenados os parâmetros exibidos na tela, o gráfico e as observações feitas pelo usuário. Posteriormente, o usuário poderá abrir a simulação para visualização ou excluir simulações previamente armazenadas. Também é possível imprimir os resultados relativos a uma simulação. A figura 5.13 exibe uma tela de simulação com o gráfico de distribuição de probabilidades de perdas e ganhos.

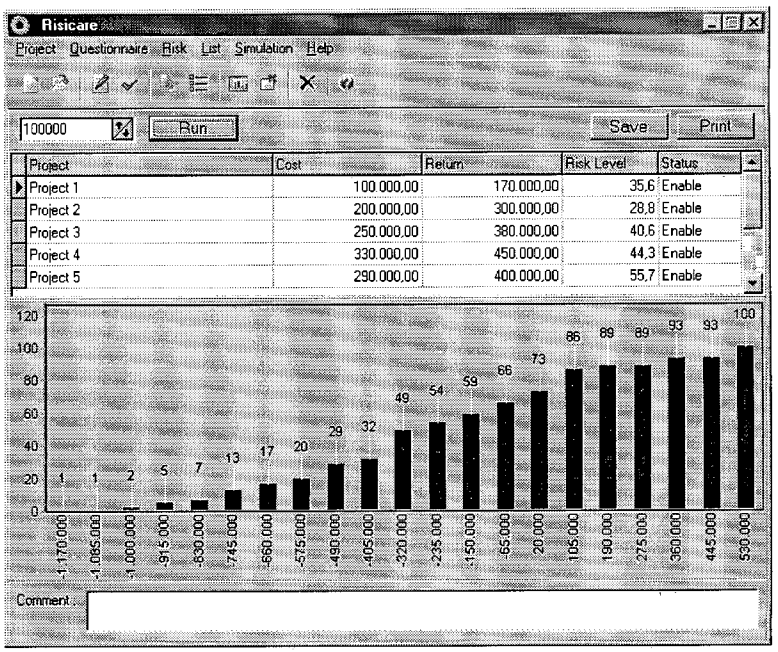


Figura 5.13 – Tela de simulação

Os parâmetros relativos aos projetos (Custo, Retorno e Nível de Risco) são editáveis e podem ser alterados, a critério do usuário, a fim de realizar novas simulações e verificar qual combinação de parâmetros leva a uma distribuição de probabilidades mais adequada para a carteira de projetos.

A coluna de *Status* possui duas opções: (*i*) habilitado, que permite que o projeto faça parte da simulação e (*ii*) desabilitado, que retira um projeto da simulação. Esta opção foi implementada para permitir ao usuário a possibilidade de observar como a sua carteira se comportaria se este projeto não fosse desenvolvido.

5.5. Utilização das abordagens

As abordagens propostas em associação com a ferramenta *RISICARE* podem ser utilizadas em diversos cenários de uma empresa:

- Na avaliação inicial de um projeto, pode-se realizar um levantamento do nível de risco de um projeto para se ter uma noção da sua viabilidade ou até mesmo quanto deverá ser cobrado para sua realização, em virtude do seu nível de risco;
- Durante o desenvolvimento de um projeto, pode-se realizar uma reavaliação do nível de risco de um projeto ou da contribuição dos fatores, para que um replanejamento possa ser realizado, em virtude de mudanças de cenário;
- Inserir um novo projeto numa carteira já existente e avaliar o impacto de sua inclusão no contexto geral da empresa;

- Analisar um conjunto de projetos e verificar quais as possibilidades de alteração do cenário existente podem levar a uma melhor combinação de resultados; e
- Reavaliar toda uma carteira de projetos após terem sido iniciado os projetos, para se visualizar atuais cenários da empresa.

5.6. Conclusões

Neste capítulo foi apresentada uma abordagem para o cálculo da distribuição de probabilidades de perdas e ganhos de uma carteira de projetos. Esta abordagem permite que gerentes e executivos obtenham informações importantes, tanto para a condução dos projetos, quanto para a realização dos objetivos estratégicos da organização.

Esta proposta foi baseada em uma analogia criada entre o conceito de Risco de Crédito e projetos de software. Para tanto, foram descritos os principais Modelos de Risco de Crédito utilizados no mercado financeiro e analisado o que mais se adequava ao contexto de software. A abordagem é apoiada pela técnica de levantamento de riscos descrita no capítulo 4. Finalmente, foi descrita a ferramenta *RISICARE*, concebida a fim de apoiar as propostas desta tese. Algumas limitações e possibilidades foram descritas.

6

CONSIDERAÇÕES FINAIS

"Software development is done in a business world where risk is inherent"

Alex Down

6.1. Conclusões

O contra-almirante Bill Carlson da marinha americana, que gerenciou durante muitos anos o desenvolvimento de complexos sistemas de combate, ficou conhecido por sua filosofia de que "se você não está gerenciando riscos, você está gerenciando a coisa errada". Apesar desta constatação e da crescente importância que vem sendo dada à gerência de riscos em projetos, não só de software, mas também de outras áreas de conhecimento, muitas empresas ainda têm dificuldades para realizar esta atividade de forma sistemática e eficiente.

Este problema pode ser justificado por duas razões: (i) a recente introdução dos conceitos relacionados a risco no contexto do desenvolvimento de software, os quais foram formalmente apresentados por BOEHM (1989), e (ii) a própria natureza incerta das causas e efeitos das fontes de risco, particularmente, em projetos de software.

No entanto, os riscos de um projeto devem ser encarados como um dos principais focos de atenção por parte dos gerentes, pois se algum evento não planejado ocorrer durante o desenvolvimento de um projeto levando a resultados inesperados, isto evidencia que havia um risco que deveria ter sido identificado e tratado de forma a não provocar tais prejuízos.

Desta forma, faz-se necessário analisar os riscos de projetos não só de maneira pontual, mas também se avaliar quão arriscados são os projetos e como eles podem influenciar para que as metas traçadas pela organização possam ser atingidas. Para isto, buscamos aproximar conceitos econômicos ao domínio da Engenharia de Software, visto que a pesquisa em gerenciamento de riscos nas Ciências Econômicas é mais tradicional, principalmente no que se refere à maximização de lucro e otimização de alocação de recursos escassos em um ambiente sujeito a incertezas.

6.2. Contribuições

Apoiados na união de conceitos das duas áreas de conhecimento apresentamos uma técnica para quantificar os riscos de projetos de software e propusemos a aplicação de modelos de análise de risco de crédito para o contexto de projetos de software. Desta forma, gerentes de projetos e investidores podem avaliar a probabilidade de obter perdas e ganhos em decorrência de manter uma carteira de projetos de desenvolvimento de software. As principais contribuições que identificamos neste trabalho são:

- Elaboração de um questionário para a identificação de riscos em projetos de software a partir da consolidação de diversas abordagens para identificação de riscos desta natureza apresentadas na literatura. O questionário é parte integrante da técnica de quantificação de riscos de um projeto de desenvolvimento software;
- Criação de uma técnica para a quantificação de riscos em projetos de software. A aplicação da técnica proposta informa ao gerente de projetos, a partir das respostas ao questionário citado no item anterior, a probabilidade de sucesso de seu projeto de software. Nesta técnica, conceitos econômicos de classificação de riscos (sistêmicos ou específicos) foram incorporados ao contexto de software;
- Planejamento e execução de um estudo experimental para determinação da importância relativa entre fatores de risco que afetam projetos de desenvolvimento de Sistemas de Informação. Com o auxílio dos resultados obtidos neste estudo, foi possível reduzir a subjetividade da técnica de quantificação dos riscos de um projeto de software mencionada no item anterior. O plano do estudo pode ser utilizado para sua repetição, no sentido de realizar pesquisas semelhantes e ampliar a abrangência, bem como a confiabilidade dos resultados obtidos;
- Determinação da importância relativa entre os fatores de risco que afetam projetos de Sistemas de Informação de diferentes tamanhos. Estes valores foram determinados através da análise dos dados coletados durante a execução do estudo experimental mencionado no item anterior, sendo utilizados na técnica de quantificação de riscos;
- Definição de uma abordagem para a avaliação da distribuição de probabilidade das perdas e ganhos financeiros proporcionados por uma carteira de projetos de software. Esta abordagem se baseia em conceitos financeiros de análise de risco de crédito;
- Especificação e implementação da ferramenta *RISICARE* que oferece apoio automatizado às abordagens propostas neste trabalho;

- Alguns artigos foram escritos e/ou submetidos durante a realização deste trabalho. No momento de sua finalização já haviam sido publicados ou aprovados para publicações:
    - Software Project Risk Evaluation Based on Specific and Systemic Risks – 260 SEKE, Jun 2004
    - Evaluating Risk Factors in Software Projects - 10 ESELAW, Out 2004.
    - A Risk Based Economical Approach for Evaluating Software Project Portfolios – EDSER 7, In Procedings of 270 ICSE, May 2005.

    Estavam ainda em processo de aprovação, os seguintes artigos:

    - Profits in a Software Projects Portfolio: a risk-based estimation approach – Information and Software Technology Journal
    - Uma abordagem econômica baseada em riscos para avaliação de uma carteira de projetos de software – SBES 2005.

Desta forma, buscou-se aproximar conceitos econômicos e de Engenharia de Software, no sentido de corroborar a idéia de que o desenvolvimento de sistemas é uma atividade econômica. Portanto, a união entre estes dois domínios de conhecimento pode trazer benefícios, tanto para projetos isoladamente, quanto para empresas desenvolvedoras de software.

A validade das abordagens propostas poderá ser avaliada através da utilização da ferramenta RISICARE em projetos reais. No entanto, esta avaliação requer seu uso em diversos projetos e contextos, para que informações possam ser coletadas e melhorias possam ser realizadas. Este procedimento está fora do escopo desta Tese de Mestrado. Contudo, a ferramenta está disponível para utilização em projetos e pode trazer benefícios para uma organização desenvolvedora de software no que se refere à avaliação de sua carteira de projetos.

6.3. Perspectivas futuras

Buscando aprimorar e ampliar as abordagens propostas neste trabalho, algumas perspectivas futuras podem ser destacadas. Inicialmente, o estudo experimental poderia ser repetido no intuito de ampliar o número de participantes e verificar se os resultados obtidos possuem alguma semelhança em outro cenário de pesquisa. O estudo realizado

no contexto deste trabalho contou apenas com 50 participantes de instituições de ensino e empresas na cidade do Rio de Janeiro. Não temos como avaliar se os resultados seriam semelhantes caso o estudo fosse realizado em um outro cenário.

Uma outra variação do estudo experimental seria realizar a pesquisa para outras categorias de sistemas, a fim de verificar se existem semelhanças entre estas categorias e se a importância relativa dos fatores de risco de projetos pode ser generalizada.

É possível realizar uma série de análises dos valores de ajuste encontrados no estudo experimental, para se obter maiores informações a cerca do comportamento dos fatores de risco durante o desenvolvimento de um projeto.

Também seria interessante, baseado nos valores já identificados, realizar estudos no sentido de verificar se existem correlações entre os fatores de risco que permitam comparar projetos entre si, a fim de utilizar conceitos da Teoria do Portfolio apresentados no capítulo 3 e determinar o risco global de uma carteira de projetos.

No que se refere à adaptação dos conceitos de risco de crédito para projetos de software reconhecemos as limitações impostas pelo modelo *Credit Risk+*, onde a dimensão do tempo e a possibilidade de analisar o fluxo de caixa ao longo da execução de um projeto não são consideradas. Portanto, seria válido estender os conceitos do modelo *Credit Risk+* para que esses limites fossem incorporados à abordagem.

Como possíveis evoluções da ferramenta RISICARE, seria interessante agregar conhecimento e inteligência à ferramenta, no sentido de auxiliar gerentes de projeto na análise dos resultados obtidos nas simulações, bem como permitir que os gerentes possam armazenar os conhecimentos adquiridos para consulta em novas avaliações.

Poderia também ser implementada uma forma de permitir que o questionário de identificação de risco fosse preenchido por mais de uma pessoa, visto que o processo de gerência de risco não é uma atividade realizada isoladamente. A edição cooperativa permitiria a discussão e a busca do consenso sobre os riscos de um projeto e quanto eles realmente contribuem para o sua falha ou sucesso.

Finalmente, seria interessante a utilização da ferramenta *RISICARE* em empresas de desenvolvimento do software, a fim de verificar sua real aplicabilidade e coletar maiores dados para seu aperfeiçoamento.

REFERÊNCIAS BIBLIOGRÁFICAS

ABNT - ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS, 1988, NBR *ISO/IEC 12207 - Tecnologia de informação - Processos de ciclo de vida de software.* Rio de Janeiro: ABNT.

ABNT, 2000, "NBR ISO 10006: Gestão da Qualidade – Diretrizes para a Qualidade no Gerenciamento de Projetos", Associação Brasileira de Normas Técnicas, Rio de Janeiro, Brasil.

AMRAM, M., KULATILAKA, N., 1999, "Disciplined Decisions : Aligning Strategy with the Financial Markets", *Harvard Business Review* , Jan/Feb, p. 95-105.

ARAGÃO C.S.L., CARVALHO L.E.Z.L., BARROS M.O., 2003, "Análise do Risco de uma Carteira de Crédito por meio de Simulação de Monte Carlo". *Resenha BM&F* nº 152.

ARROW K.J, 1962, "Economic Welfare and the Allocation of Resources for Invention", In: *The Rate and Direction of Inventive Activity*: Economic and Social Factors, NBER, Princeton University Press, pp. 609-626.

ASUNDI J., KAZMAN R., 2001, "A Foundation for the Economic Analysis of Software Architectures", In: *Proceedings of the Third Workshop on Economics-Driven Software Engineering Research.*

AUGUSTINE, N.R., 1982, Augustine's Laws, Nova York, NY: American Institute Of Aeronautics and Astronautics

BALDWIN, C., CLARK, K., 1993 , "Modularity and Real Options" Working Paper, *Harvard Business School,* Cambridge, MA.

BARBANSON R., 2004, An Introductin to Credit Risk with a Link to Insurance, AFIR, Working Grioup.

BARKI H.; RIVARD S.; TALBOT, J., 1993, "Toward an Assessment of Software Development Risk". *Journal of Management Information Systems*, v. 10, n. 2, 1993, p. 203-225.

BARROS, M.O., 2001, *Gerenciamento de Projetos Baseado em Cenários: uma Abordagem de Modelagem Dinâmica e Simulação*, Tese de Doutorado, COPPE/UFRJ, Rio de Janeiro, RJ, Brasil.

BASILI, V.R.; SELBY, R.W.; HUTCHENS, 1987, "Experimentation in software Engineering". *IEEE Transactions on Software Engineering*, 12 (7), pp.1728-1298.

BECK K., 2000, *Extreme Programming Explained*: Embrace change Addison-Wesley.

BERNSTEIN P. L., 1997, *Desafio aos Deuses: A Fascinante História do Risco*. Editora Campus, 2ª Edição.

BLACK F., SCHOLES M., "The Pricing of Options and Corporate Liabilities", *Jornal of Political Economy* 81, May/June pp. 673-659.

BOEHM B.W., 1981, *Software Engineering Economics*, Upper Saddle River, New Jersey: Prentice Hall.

BOEHM, B.W, 1988, "A Spiral Model of Software Development and Enhancement". *IEEE Computer*, v. 21, n. 5, pp. 61-72.

BOEHM, B.W., 1989, "Software Risk Management, Los Alamitos", CA: *IEEE Computer Society Press*.

BOEHM, B.W., 1991, "Software Risk Management: Principles and Practices", *IEEE Software*, vol. 8, n. 1 (January), pp. 32-41.

BOEHM, B.W. et al., 1995, "Cost Models for Future Software Life Cycle Processes: COCOMO 2.0", In: *Annals of Softwa re Engineering, Special Volume on Software Process and Product Measurement*, Amsterdam, Netherlands: J.D. Arthurs and S.M. Henry Editions, J.C. Baltzer Science Publishers

BOEHM B.W., SULLIVAN K., 1999, "Software Economics: Status and Prospects", *Information and Software Technology* 41, pp. 937–946.

BREALEY R.A. e. MYERS S.C, 1996, *Principles of Corporate Finance*, 5th edition, McGraw Hill.

BROWN, N., 1996, "Industrial-strength management strategies", IEEE Software, July, pp. 94-103.

BUSINESS WEEK, 1999, "Software Hell", December 6, pp. 104-118.

BUTLER S., CHALASANI P., JHA S., RAZ O., SHAW M., 1999, "The Potential of Portfolio Analysis in Guiding Software Decisions", In: *Proceedings of the First Workshop on Economics-Driven Software Engineering Research.*

CARR, M. J., KONDA, S.L, MONARCH, I., ULRICH, F.C., WALKER, C.F., 1993,"Taxonomy-Based Risk Identification", *Technical Report* CMU/SEI–93-TR-6, Software Engineering Institute, Carnegie Mellon University, EUA, July.

CHAPMAN, C., WARD, S., 1997. *Project Risk Management: Processes, Techniques, and Insights.* Chichester: John Wiley & Sons.

CLEMONS E.K., 1991, "Evaluation of Software Investments in Information Technology", *Communications of ACM* 34, 1, 22-36.

COCKBURN A., WILLIAMS L., 2000, "The Costs and Benefits of Pair Programming", In: eXtreme Programming and Flexible Process in Software Engineering XP2000, Cagliari, Italy, June.

COSTA, C.A B. 1995, VANSNICK, J.C. "A Theoretical Framework for Measuring Attractiveness by a Categorical Based Evaluation Technique (MACBETH)". In: *Clímaco, J., Multicriteria Analysis.* Berlin: Springer.

CREDIT SUISSE FIRST BOSTON, 1997, *CreditRisk+: A Credit Risk Management Framework.*

DYBA, T., 2000, "An Instrument for Measuring Key Factors of Success in Software Process Improvement". *Kluwer Academic Publishers*, Boston.

ELTON E.J., GRUBER M.J., 1995, *Modern Portfolio Theory and Investment Analysis.* Editora John Wiley & Sons, 5ª Edição, NY.

ERDOGMUS H., 1999, "Comparative Evaluation of Software Development Strategies Based on Net Present Value", In: *Proceedings of the First Workshop on Economics-Driven Software Engineering Research.*

ERDOGMUS H., BARRE, M., 2000, "Value of Commercial software Development under Technology Risk". *The Financier*, Vol. 7, NRC Publications nº 44890.

ERDOGMUS, H., CUSUMANO M.A., KONTIO J., RAFFO D., 2004, "The Sixth International Workshop on Economics-Driven Software Engineering Research". In: *Proceedings of the 26th International Conference on Software Engineering.*

ERRUNZA V., HOGAN J.K., MAZUMDAR S., 1996, "Behavior of International Stock Return Distributions: A Simple Test of Functional Form", *International Review of Economics and Finance*, Vol 5, p51-61.

FAIRLEY, R. 1994, "Risk Management for Software Projects", *IEEE Software*, Vol. 13, No. 3 (Maio), pp. 57 – 67.

FARIAS, L.L., 2002, *Planejamento de Riscos em Ambientes de Desenvolvimento de Software Orientados à Organização*, Dissertação de Mestrado, COPPE/UFRJ, Rio de Janeiro, RJ, Brasil.

FAVARO J.M, FAVARO K.R., FAVARO P.F., 1998, "Value Based Software Reuse Investment", In: *Annals of Software Engineering* 5, pp. 5-52.

FREIMAN F.R. e PARK R.E., 1979, "PRICE Software Model" - Version 3: An Overview, Proceedings, *IEEE/PINY Workshop on Quantitative Software Models*, October pp. 32-44

GALLAGHER, B. P.; ALBERTS, C.J.; BARBOUR, R.E., 1997, "Software Acquisition Risk Management: Key Process Area (KPA) – A Guidebook", *Relatório Técnico* SEI/CMU-97-HB- 002, Pittsburgh, PA.

GITMAN L.J., 1987, *Princípios de Administração Financeira*, 3ª Edição. Editora Harbra ltda, São Paulo.

GOLDMAN, SACHS & Co, SWISS BANK CORPORATION, 1998, *The Practice of Risk Management: Implementing Processes for Managing Firmwide Market Risk* , London, UK: Euromoney Publications PLC.

GREY, S., 1995, *Practical Risk Assessment for Project Management*, In: Wiley Series in Software Engineering Practice, Nova York, NY: John Wiley & Sons Inc.

GUILFORD, J. P., *Psychometric Methods*, 2nd edition, New York. McGraw-Hill, 1954.

HALL, E.M., 1995, *Proactive Risk Management Methods for Software Engineering Excellence.* Tese de Doutorado - Florida Institute of Technology, Melbourne.

HALL, E.M., 1998, "Managing Risk: Methods for Software Systems Development", In: *SEI series in Software Engineering, Reading*, MA: Addison Wesley Longman Inc.

HUMPHREY, W.,1987, "Characterizing the Software Process: a Maturity Framework", version 1.0.: *Software Engineering Institute* - Carnegie Mellon University, Technical report CMU/SEI-87-TR-11 Pittsburgh.

HUNTER, J.E.; SCHMIDT, F.L., 1990, *Methods of Meta-Analysis: Correcting Erros and Bias in Research Findings*. Sage Publications. Newbury Park, California.

IEEE 1002, 1987, *Standard Taxonomy for Software Engineering Standards*. Piscataway

IEEE Std 1540-2001, 2001, *IEEE Standard for Software Life Cycle Processes* – Risk Management.

ISO/IEC DTR 16326, 1999, Software Engineering – Guide for the application of ISO/IEC 12207 to project management.

JALOTE P., 1999, *CMM in Practice: Processes For Executing Software Projects At Infosys*, Addison-Wesley Publishing Company.

JIANG, J., KLEIN, G., MEANS, T.L., 2000, "The Risk Impact on Software Development Team Performance", *Project Management Journal*, v. 31, n. 4, Dec, pp. 19-26.

JONES, C., 1994, *Assessment and Control of Software Risks*, Yourdon Press.

JONES, C., 1994, *Assessment and Control of Software Risks*, Englewood Cliffs, NJ: Prentice-Hall, Inc.

JOHNSON R. e BHATTACHARYYA G., 1977, *Statistical Concepts and Methods*, John Wiley & Sons. New York.

J.P MORGAN, 1997, *CreditMetrics: The Benchmark for Understanding Credit Risk*, Technical Document, New York, NY.

JTC1/SC7, 2001, 12207/FDAM 1 - Software Engineering - Life Cycle Processes. Software & System Engineering Secretariat, Montreal, Canada.

KÄNSÄLÄ, K., 1997, "Integrating Risk Assessment with Cost Estimation", *IEEE Software*, v. 14, n. 3, p. 61-67, May/June.

KAROLAK, D.W., 1996, *Software Engineering Risk Management*, Los Alamitos, CA: IEEE, Computer Society Press.

KAZMAN R., ASUNDI J., KLEIN M., 2000, "Quantifying the Costs and Benefits of Arquitectural Decisions", In: *Proceedings of the 23rd International Conference on Software Engineering*, Toronto, Canada.

KAZMAN R., KLEIN M., CLEMENTS P., 2000, ATAM: "A Method for Architecture Evaluation", *CMU/SEI-2000*-TR-004. SEI, Carnegie Mellon University.

KEENEY R.L., 1993, *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*, Cambridge England: Cambridge University Press.

KEMERER C.F., 1993, "Reliability of Function Points Measurement: A Field Experiment", *Comunication of the ACM* 36, Feb. pp. 85-97.

KERZNER, H., *Project Management*. John Wiley & Sons, 7ª edição Cap.17.

KITCHENHAM, B.; PICKARD, L.; PFLEEGER, S.L., 1995, "Case Studies for Method and Tool Evaluation", *IEEE Software*, pp. 52-62, July.

K.M.V. CORPORATION, 2003, *Modelling Default Risk*, disponível na URL http://www.moodyskmv.com/research/whitepaper/ModelingDefaultRisk.pdf.

KONTIO, J., 1997, "The RiskIt Method for Software Risk Management", version 1.00, Relatório Técnico CS-TR-3782, UMIACS-TR-97-38, *Instituto para Estudos Avançados em Computação, Departamento de Ciência da Computação*, Universidade de Maryland, College Park, MD.

LIKERT, R., 1932, *A Technique for the Measurement of Attitudes*. Arquives of Psycology.

LUEHRMAN T.A., 1998, "Investments Opportunities as a Portfolio of Real Options", *Harvard Business Review*, July-August.

MACEDO M.A.S., 1999, "Avaliação de Projetos: uma visão da utilização da teoria de opções", XIX *ENEGEP*, Novembro, Rio de Janeiro.

MACHADO, C.A.F., 2000, *A-Risk : Um Método para Identificar e Quantificar Risco de Prazo em Projetos de Desenvolvimento de Software*. Tese de Mestrado. Pontifícia Universidade Católica do Paraná – PUCPR. Curitba

MACHLUP F., 1962, *The Production and Distribution of Knowledge*, Princeton University Press.McFARLAN, W.; McKENNY, J. L. 1996, "Corporate Information Systems Management - The Issue Facing Senior Management". (S.I.): *Public Richard Irwin.*

MADACHY, R.J., 1997, "Heuristic Risk Assessment Using Cost Factors", *IEEE Software*, Vol. 14, No. 3 (Maio/Junho), pp. 51 – 60, Computer Society Press.

MARKOWITZ, H., 1952, "Portfolio Selection", *Journal of Finance*, 7, p 77-91.

MARTINS, E., 2001, *Avaliação de Empresas: Da Mensuração Contábil à Econômica*, Editora Atlas, 1ª Edição.

McTAGGART J.M., KONTES P.W., MANKINS M.C., 1994, *The Value Imperative*, The Free Press, New York.

MEISTER, D.; RABIDEAU, G.F., 1965, *Human Factors Evaluation in System Development*. John Wiley & Sons, New York.

MILLER, G.A., 1956, "The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information". *Psychological Review*, 63, 81-97.

MIRANDA, E., 1999, *Establishing Software Size Using the Paired Comparisons Method*. Ericsson Research, Canada.

MONTGOMERY, D.C., 1997, *Design and Analysis of Experiments*, 4th edition, John Wiley & Sons.

MOYNIHAN, T., 1997, "How Experienced Project Managers Assess Risk", *IEEE Software*, v. 14, n. 3, p. 35-41, May/June.

MÜLLER M.M., TICHY W.F., 2001, Extreme Programing in a University Enviroment, In": 23rd *International Conference on Software Engineering*, Toronto, Canada.

MÜLLER M.M., PADBERG F., 2002, "Extreme Programming from an Engineering Economics Viewpoint", In: *International Workshop on Economics-Driven Software Engineering Research,* Orlando, USA.

MURCH, R., 2000, *Project Management – Best Practices for IT Professionals*, Prentice Hall.

NELSON E.A., 1966, "Management Handbook for the Estimation of Computer Programming Costs". AD A-648750, *Systems Development Corp.*, October.

NBR ISO 10006, 2000, *Gestão da Qualidade – Diretrizes para a Qualidade no Gerenciamento de Projetos*, Associação Brasileira de Normas Técnicas, Rio de Janeiro, RJ, Brasil.

PAULK, M., CURTIS, B., CHRISSIS, B. M., WEBER, C. V., 1993, "Capability Maturity Model for Software", version 1.1. *Software Engineering Institute* - Carnegie Mellon University. Technical Report CMU/SEI-93-TR-24, Pittsburgh.

PAULK, M. C., CURTIS, B., CHRISSIS, B. M., WEBER, C. V., 1995, *The Capability Maturity Model: Guidelines for Improving the Software Process*, Addison-Wesley.

PFLEEGER, S. L., HATON, L., HOWELL, C.C., 2001, *Solid Software*, Prentice Hall.

PMBOK, 1996, "A Guide to the Project Management Body of Knowledge – PMBOK Guide", *Project Management Institute (PMI) Standards Comitee*, Pennsylvania, USA.

PMBOK, 2000, "A Guide to the Project Management Body of Knowledge – PMBOK Guide", *Project Management Institute (PMI) Standards Comitee*, Pennsylvania, USA.

PRESSMAN, R., 2000, *Software Engineering: A Practitioner's Approach*, 5 a. Edição, IN: McGraw-Hill. Higher Education International Editions, London, UK: McGraw-Hill Co.

PRITCHARD, C. L., 1997, *Risk Management: Concepts and Guidance*. Arlington: ESI International.

PUTNAM L.H., 1978, "A General Empirical Solution to the Macro Software Sizing and Estimating Problem", *IEEE Transactions on Software Engineering*, July pp.345-361.

RAIFFA H., 1968, *Decision Analysis: Introductory Lectures on Choices Under Uncertainty*, McGraw-Hill, Inc., New York.

REEL, J.S., 1999, "Critical Success Factors in Software Projects", IEEE Software, Vol. 16, No. 3 (Maio/Junho), pp. 18 – 23

ROBSON, C., 1993, *Real World Research for Social Scientists and Practicioners* - Researchers, Blackwell.

ROPPONEN, J., LYYTINEN, K., 2000, "Components of Software Development Risk: how to address them? a project manager survey". *IEEE Transactions on Software Engineering*, v. 26, n. 2, Feb.

ROSS S., WESTERFIELD R.W., JORDAN B.D., 1998, *Princípios de Administração Financeira*, Ed. Atlas, São Paulo.

ROUT, T., TUFFLEY, A., CAHILL, B., HODGEN, B., 2000, "The Rapid Assessment of Software Process Capability". In: *Spice International Conference on Software Process Improvement and Capability Determination*, Ireland, Jun. pp. 47-56.

SALLES, C.A., 2003, *Gerência de Riscos em Projetos*. EPGE, Fundação Getulio Vargas. Rio de Janeiro.

SCHMIDT, R., KEIL, M.; CULE, P.; LYYTINEM, K., 1996, "A Framework for Identifying Software Project Risks", disponível na URL http://webpage.pace.edu/cm7596ow/dcs.

SEI, 2002, *Capability Maturity Model Integration*, Carnegie Mellon University, Pittsburg. disponível na URL http://www.sei.cmu.edu/cmmi.

SEI, 2003, "Risk Management", disponível na URL http://www.sei.cmu.edu.br.

SEPIN (Secretaria de Política de Informática), 2002, *Qualidade e Produtividade no Setor de Software Brasileiro*, Ministério da Ciência e Tecnologia, Brasília, disponível na URL http://www.mct.org.br

SOMMERVILLE I., 1996, *Software Engineering*, Addison-Wesley.

STIGLER G.J., 1962, "The Economics of Information", *Journal of Political Economy*, vol.69, pp. 213-225.

STANDISH GROUP, 2004, *Chaos Demographics – 2004 Third Quarter Research Report*, The Standish Group International Inc.

STRAUB, D.W., 1989, "Validating Instruments in MIS Research", *MIS Quartely*, 3(2), pp. 147-169.

SULLIVAN, K., J., CHALASANI, P., JHA, S., 1997, "Software Design Decisions as Real Options". Technical Report 97-14, *Department of Computer Science*, University of Virginia, Charllotesvile, VA.

TICHY, W.F., 1998, "Should Computer Scientists Experiment More?" *IEEE Computer*, 31 (5), pp.32-39.

VILLELA, K.C.L., 2004, *Sistema de Desenvolvimento de Software Orientados à Organização*, Tese de Doutorado, COPPE/UFRJ, Rio de Janeiro, RJ, Brasil.

VOSE, D., 1996, *Quantitative Risk Analysis: A Guide to Monte Carlo Simulation Modelling*, Nova York, NY: John Wiley & Sons Inc.

WALLACE, L., 1999, *The Development of an Instrument to Measure Software Project Risk.* Tese de Doutorado - College of Business Administration, Georgia State University, Georgia.

WIEGERS, K. E., 1999, "Read my lips: new models". *IEEE Software*, v. 15, n. 5, Sep./Oct, pp. 10-13.

WILLIANS, R.C., WALKER, J.A., DOROFEE, A.J. 1997, "Putting Risk Management into Practice", *IEEE Software*, Vol. 14, No. 3 (Maio/Junho), pp. 75 – 82.

WITHEY, J., 1996, "Investment Analysis of Software Assets for Product Lines", Technical Report CMU/SEI-96-TR-010, *Software Engineering Institute*, Pittsburgh, PA.

WOHLIN, C., RUNESON, P., HÖST, M., OHLSSON, M., REGNELL, B., WESSLÉN, A., 2000, *Experimentation in Software Engineering – An Introduction*, Kluwer Academic Publishers.

ZADEH, L.A., 1988, "Fuzzy Logic", *IEEE Computer*, Vol. 21, No. 4 (Abril), pp. 83 - 91

ZELKOWITZ, M.V e WALLACE, D.R., 1998, "Experimental Models for Validating Technology", *IEEE Computer*, 31 (5), pp. 23-31.

ANEXO 1 – QUESTIONÁRIO DE IDENTIFICAÇÃO DE RISCOS

Neste anexo, apresentamos o questionário de identificação de risco elaborado durante a revisão bibliográfica que serve de apoio para quantificar o risco de projetos na ferramenta *RISICARE.* O questionário teve como base outras taxonomias de risco presentes na literatura de Engenharia de Software caracterizados na tabela A.1.1.

Os questionários utilizados possuem diferenças quanto ao grau de formalidade com que foram realizados, pela metodologia utilizada para a elaboração do questionário pelo domínio de aplicação, pelo público alvo para qual a pesquisa foi direcionada e pela quantidade de fontes de risco encontradas.

Tabela A.1.1 – Taxonomias de riscos

Autor	Metodologia	Domínio	Público Alvo	Fontes de Risco
Boehm (1981)	• Pesquisa de Opinião	Não Especificado	Gerentes de Projeto	10
Barki *et al* (1993)	• Pesquisa de Opinião • Pesquisa na Literatura • Análise das Fontes de Risco	Sistemas de Informação	Gerentes e Usuários de 120 Projeto de 75 Organizações	35
Carr *et al* (1993)	• Pesquisa realizada pelo SEI	Diversos	Não Especificado	194
Jones (1994)	• Pesquisa de Opinião • Análise Estatística	Sistemas Comerciais, Militares, Contratos e Usuários	Coleta de dados desde 1985	60
Karolak (1996)	• Pesquisa na Literatura	Diversos	04 Fontes	81
Moyniham (1997)	• Pesquisa de Opinião • Análise de Regressão	Sistemas de Informação	14 Gerentes de Projeto	21
Wallace (1999)	• Pesquisa de Opinião • Pesquisa na Literatura • Análise Estatística	Diversos	Gerentes de Projeto	53

Por ser o mais completo de todos os questionários, a Taxonomia de Riscos do SEI foi utilizada como base para a elaboração do questionário final. A partir das perguntas iniciais do questionário do SEI foi realizada uma comparação com todas as outras perguntas existentes nos outros questionários a fim de eliminar redundâncias.

Após o término das perguntas do SEI, verificou-se que ainda existiam tópicos nos outros questionários e assuntos específicos que não eram tratados pelo SEI. Foi realizada, então, uma nova fase de busca de semelhanças entre os demais questionários, tomando como base o questionário de Karolak. Esse procedimento foi sendo repetido até que todas as perguntas dos sete questionários tivessem sido esgotadas. As fontes de riscos encontradas foram sendo acrescentadas ao questionário final de maneira a torná-lo o mais completo possível e estão expressas na tabela A.2.

Tabela A.1.2 – Questionário de identificação de riscos

#	FACTOR ANALYSIS
1	Are the requirements stable?
2	Are there requirements you know should be in the specification but aren't?
3	Are you able to understand the requirements as written?
4	Are there any requirements that may not specify what the customer really wants?
5	Do you and the customer understand the same thing by the requirements?
6	Do you validate the requirements?
7	Are there any requirements that are technically difficult to implement?
8	Are there any state-of-the-art requirements (Technologies, Methods, Languages, Hardware)?
9	Is the system size and complexity a concern?
10	Does the size require a larger organization than usual for your company?
11	Are reliability requirements allocated to the software?
12	Are availability requirements allocated to the software?
13	Are safety requirements allocated to the software?
14	Will it be difficult to verify satisfaction of safety requirements?
15	Are there unprecedented or state-of-the-art security requirements?
16	Have you implemented this level of security before?
17	Are there Human Factors requirements? Do you see any difficulty in meeting them?
18	Is the software requirements specification adequate to design the system?
19	Are there any problems with performance (Real-time response, Response time, Database response, Contention, or Access)?
20	Are the external interfaces changing?
21	Are the external interfaces completely defined?

22	Is there a requirements traceability mechanism that tracks requirements from the source specification through test cases?
23	Is the traceability mechanism used in evaluating requirement change impact analyses?
24	Is there a formal change control process?
25	Are changes at any level mapped up to the system level and down through the test level?
26	Is there adequate analysis when new requirements are added to the system?
27	Are automated tools used for modeling?
28	Are automated tools used for requirements traceability?
#	**FACTOR DESIGN**
	Are there any specified algorithms that may not satisfy the requirements?
2	Are there mechanisms for determining the feasibility of algorithms and designs (Prototyping, Modeling, Analysis, Simulation)?
3	Does any of the design depend on unrealistic or optimistic assumptions?
4	Are there any requirements or functions that are difficult to design?
5	Are the internal interfaces well defined?
6	Is there a process for defining internal interfaces?
7	Is hardware being developed in parallel with software?
8	Has a performance analysis been done?
9	Does the design include features to aid testing?
10	Does the hardware limit your ability to meet any requirements?
11	Are you reusing or re-engineering software not developed on the project?
12	Are there any problems with using COTS (commercial off-the-shelf) software?
13	Do you foresee any problem with integrating COTS software updates or revisions?
14	Are there design standards?
15	Is it possible to reuse models and standards?
16	Is it possible to reuse interfaces?
17	Are automated tools used for designing?
#	**FACTOR CODING**
1	Are any parts of the product implementation not completely defined by the design specification?
2	Are the selected algorithms and designs easy to implement?
3	Are the design specifications in sufficient detail to write the code?
4	Is the design changing while coding is being done?
5	Are there system constraints that make the code difficult to write?
6	Is the language suitable for producing the software on this project?
7	Are there multiple languages used on the project?
8	Is the development computer the same as the target computer?
9	Are the hardware specifications adequate to code the software?

10	Are the hardware specifications changing while the code is being written?
11	Are the codes reusable?
#	**FACTOR TEST**
1	Is there a formal test plan?
2	Are the test specifications adequate to fully test the system?
3	Will compromises be made regarding testing if there are schedule problems?
4	Have acceptance criteria been agreed to for all requirements?
5	Are there any requirements that will be difficult to test?
6	Do the testers get involved in analyzing requirements?
7	Are the test plans and procedures updated as part of the change process?
8	Do you begin unit testing before you verify code with respect to the design?
9	Has sufficient unit testing been specified?
10	Is there sufficient time to perform all the unit tests you think should be done?
11	Has sufficient product integration been specified?
12	Has adequate time been allocated for product integration and test?
13	Has sufficient system integration been specified?
14	Has adequate time been allocated for system integration and test?
15	Will the product be integrated into an existing system?
16	Will there be sufficient hardware to do adequate integration and testing?
17	Is there any problem with developing realistic scenarios and test data to demonstrate any requirements?
18	Are you able to verify performance in your facility?
19	Does hardware and software instrumentation facilitate testing?
20	Will the target hardware be available when needed?
21	Are all contractors part of the integration team?
22	Does the hardware limit your ability to meet any requirements?
23	Is regression test performed?
24	Are there automatic tools for regression tests?
25	Are the tests performed by different people from the development team?
#	**FACTOR PLANNING**
1	Is there a formal management methodology?
2	Is the project managed according to the plan?
3	Is re-planning done when disruptions occur?
4	Are the necessary plans being done?
5	Is there a detailed WBS for all activities?
6	Are there defined metrics for the project?
7	Are there tools for collecting and analyzing metrics from the project?

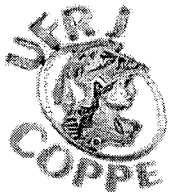
8	Are the project milestones well established?
9	Are there tools for project management?
10	Is there an effective risk management process?
11	Are there contingency plans for known risks?
12	Does the project have experienced managers?
13	Is the software quality assurance function adequately staffed on this project?
14	Do you have defined mechanisms for assuring quality?
15	Does schedule get in the way of quality?
16	Do you have an adequate configuration management system?
17	Is the configuration management function adequately staffed?
18	Has the schedule been stable?
19	Is the schedule realistic?
20	Is the estimation method based on historical data?
21	Has the method worked well in the past?
22	Is there anything for which adequate schedule was not planned?
23	Is the budget stable?
24	Is the budget based on a realistic estimate?
25	Is the estimation method based on historical data?
26	Has the method worked well in the past?
27	Is there anything for which adequate budget was not allocated?
28	Is there a development process well defined?
29	Can you measure whether the development process is meeting your productivity and quality goals?
30	Is there more than one development model being used?
31	Are there formal, controlled plans for all development activities?
32	Is the development process adequate for this product?
33	Is the development process supported by a compatible set of procedures, methods, and tools?
34	Does the development system support all aspects of the project?
35	Is there good documentation of the development system?
36	Is the development system considered reliable?
#	**FACTOR CONTROL**
1	Are there periodic structured status reports?
2	Does appropriate information get reported to the right organizational levels?
3	Do you track progress versus plan?
4	Are project members at all levels aware of their status versus plan?
5	Have features or functions been deleted as part of a design-to-cost effort?

6	Do budget changes accompany requirement changes?
7	Do schedule changes accompany requirement changes?
8	Are there external dependencies, which are likely to impact the schedule?
9	Are there dependencies on external products or services that may affect the product, budget, or schedule?
10	Does management communicate problems up and down the line?
11	Are conflicts with the customer documented and resolved in a timely manner?
12	Does management involve appropriate project members in meetings with the customer?
13	Does management work to ensure that all customer factions are represented in decisions regarding functionality and operation?
14	Is it good politics to present an optimistic picture to the customer or senior management?
15	Do you have a way to track interfaces?
16	Are the deviations to original plans corrected?
17	Are defect data collected during software development?
#	**FACTOR TEAM**
1	Do people get trained in skills required for this project?
2	Do people get assigned to the project who do not match the experience profile for your work area?
3	Is there any problem keeping the people you need?
4	Are there any areas in which the required technical skills are lacking?
5	Is the staffing stable?
6	Do you have access to the right people when you need them?
7	Have the project members implemented systems of this type?
8	Is the project reliant on a few key people?
9	Is there sufficient capacity for overlapping phases, such as coding, integration and test?
10	Are the people trained in use of the development tools?
11	Do people work cooperatively across functional boundaries?
12	Do people work effectively toward common goals?
13	Is it easy for project members to get management action?
14	Do people feel it's important to keep to the plan?
15	Does management consult with people before making decisions that affect their work?
16	Is management intervention sometimes required to get people working together?
17	Is there good communication among the members of the project?
18	Are the managers receptive to communication from project staff?
19	Does the team feel free to ask managers' help?
20	Do the project members get timely notification of events that may affect their work?
21	Is the morale on the project high?

22	Are the development facilities adequate?
23	Are there enough workstations and processing capacity for all staff?
24	Are all staff levels oriented toward quality procedures?
25	Is risk management mentality part of the team culture?
26	Do people understand their own and others' roles in the project?
27	Do people know who has authority for what?
28	Are developers familiar with the plans?
29	Does everyone follow the development process?
30	Are people comfortable with the development process?
31	Is the team size a risk for the project?
32	Do the team know software engineering?
#	**FACTOR CONTRACTS**
1	Does the type of contract represent a risk for the project?
2	Is the contract burdensome in any aspect of the project?
3	Is the required documentation burdensome?
4	Are there problems with data rights?
5	Is the customer approval cycle timely (Documentation, Project reviews, Formal reviews)?
6	Are the external interfaces changing without adequate notification, coordination, or formal change procedures?
7	Is there an adequate transition plan in case of associate contractors?
8	Are the contracts supported by all contractors and site personnel?
9	Is there any problem with getting schedules or interface data from associate contractors?
10	Are there any ambiguities in subcontractor task definitions?
11	Is the subcontractor reporting and monitoring procedure different from the project's reporting requirements?
12	Is subcontractor administration and technical management done by a separate organization?
13	Are you highly dependent on subcontractor expertise in any areas?
14	Is subcontractor knowledge being transferred to the company?
15	Is there any problem with getting schedules or interface data from subcontractors?
16	Are your task definitions from the prime contractor ambiguous? (If project is a subcontract)
17	Is there any problem with getting schedules or interface data from the prime contractor?
18	Do you interface with two separate prime organizations for administration and technical management?
19	Are you highly dependent on the prime contractor for expertise in any areas?
20	Are you relying on vendors for deliveries of critical components (Compilers, Hardware, COTS)?
21	Is the amount of external vendors a problem for the project?

#	FACTOR POLICIES / ORGANIZATIONAL STRUCTURE
1	Are politics affecting the project?
2	Are politics affecting technical decisions?
3	Does project management communicate problems to senior management?
4	Does corporate management give you timely support in solving your problems?
5	Is the organizational structure stable?
6	Is the top management involved with the project?
7	Are there internal conflicts or divergences that affect the project?
8	Do the goals of the project are well defined?
#	**FACTOR CLIENTS**
1	Are clients and users involved with the project?
2	Are users resistant to changes?
3	The amount of changes in users' life caused by the system is a concern?
4	Is the amount of users affected by the system a concern?
5	Does the system will change the organizational structure of the client?
6	Does the team know the client's organizational culture?
7	Does the team know the client's business?
8	Do you ever proceed before receiving customer approval?
9	Does the customer understand the technical aspects of the system?
10	Does the customer understand software?
11	Does the customer interfere with process or people?
12	Does management work with the customer to reach mutually agreeable decisions in a timely manner?
13	How effective are your mechanisms for reaching agreements with the customer?
14	Are all customer factions involved in reaching agreements?
15	Is it a formally defined process?
16	Does management present a realistic or optimistic picture to the customer?

ANEXO 2 – CARACTERIZAÇÃO DO PARTICIPANTE E DO PROJETO



Participante

1) Identificador: _____

2) Nome: _____

3) Função que ocupa no projeto: _____

4) E-mail: _____

5) Empresa: _____

6) Tempo de atuação na área de desenvolvimento de software: _____ anos(s)

7) Quantidade de projetos que já participou: _____

8) Formação acadêmica na área de Ciência da Computação:

a) Graduando b) Graduação c) Especialização d) Mestrado e) Doutorado

9) Conhecimento em relação à gerência de riscos:

a) Nenhum b) Baixo c) Médio d) Alto e) Excelente

Projeto

1) Duração do projeto: _____ meses

2) Número médio de pessoas envolvidas no projeto: _____

Autorizo o uso das informações por mim fornecidas no contexto dessa pesquisa, desde que a minha identificação e a da minha empresa sejam mantidas em sigilo.

_____

Participante