Usando GQM para Gerenciar Riscos em Projetos de Software

Lisandra M. Fontoura M.Sc.[1,2,3], Roberto T. Price Eng., M.Sc., D.Phil[1,4]
[1]Instituto de Informática, Universidade Federal do Rio Grande do Sul, Av. Bento Gonçalves 9500, Campus do Vale
Porto Alegre RS, Brazil 91501-970, + 55 51 33166165
[2]Universidade Regional Integrada, Av. Batista Bonoto Sobrinho s/n
Santiago RS, Brazil 97700-000, + 55 55 2513151
[3]Centro Universitário Franciscano, Rua dos Andradas, 1614
Santa Maria RS, Brazil 97010-032, + 55 55 2201200
[4]TeleHumana Comunicações S.A., Av. Carlos Gomes, 141 Cj. 1102, Bela Vista
Porto Alegre RS, Brazil 90480-003, + 55 51 33280636
e-mail: [lisandra, tomprice]@inf.ufrgs.br

Resumo

Muitos projetos de desenvolvimento falham em entregar o software dentro do cronograma, orçamento e níveis de qualidade, por causa da avaliação inadequada dos fatores de risco. Este trabalho propõe uma abordagem sistemática para prevenir e monitorar riscos, baseado em adaptação do processo de desenvolvimento. Para prevenção de risco é sugerido um procedimento para listar, analisar e priorizar os riscos, visando selecionar ações preventivas para incluir no processo de desenvolvimento, minimizando a exposição ao risco. Utilizando o paradigma Goal/Question/Metric são definidas métricas do processo de software para serem usadas para acompanhar o progresso dos fatores de risco, possibilitando ao gerente de projeto tomar ações corretivas, quando necessário e no momento adequado.

Abstract

Many development projects still fail to deliver software within schedule, budget and quality levels because of inadequate assessment of the envolved risk factors. This paper proposes a systematic approach to risk prevention and monitoring based on development process adaptation. For risk prevention a procedure for risks enumeration, analysis and priorization is suggested to select preventive actions to include in the development process in order to minimize risk exposure. Using the Goal/Question/Metric paradigm, metrics are defined for each risk of a project, enabling project managers to follow the progress of the risk factors, enabling corrective actions when required at the adequate time.

1. Introdução

Gerência de riscos de software (GRS) é uma abordagem que busca organizar o enfoque dos riscos correlatos ao desenvolvimento bem-sucedido de sistemas de software em um conjunto de princípios e técnicas para analisar, preparar ações preventivas e controlar os riscos de projetos de software. GRS sugere medidas para prevenir que riscos afetem o projeto ou para reduzir seu impacto, e deve ser vista como um componente fundamental do processo de gerenciamento de projetos [1]. Gerenciamento de riscos deve focar nas questões que podem por em risco o alcance dos objetivos críticos de um projeto [18]. O dicionário Webster define risco como "a possibilidade de perda ou ferimento". Essa definição pode ser traduzida em um conceito fundamental de risco: exposição do risco,

algumas vezes chamada de "impacto do risco" ou "fator de risco" [3]. O gerenciamento de riscos envolve dois aspectos principais: a avaliação do risco e o controle do risco [4].

Avaliação de risco é o processo de identificação das fontes de riscos e avaliação de seus efeitos potenciais [9]. A avaliação do risco pode ser subdividida nas etapas: identificação do risco, análise do risco e priorização do risco [3]. A identificação dos riscos resulta em uma lista de possíveis riscos para o projeto. Análise do risco consiste em estimar a probabilidade e a conseqüência de cada risco identificado [9]. Na priorização, os riscos identificados são ordenados por importância [3].

Como forma de facilitar a identificação de riscos, na seção 2 deste trabalho é exibido um conjunto dos principais riscos que afetam os projetos de software, elaborado a partir de uma comparação entre os riscos identificados em trabalhos realizados pelos autores Keil et al. [12], Addison [1], Boehm [3] e os riscos sugeridos pelo modelo CMMI [18].

O controle de risco abrange as atividades que tratam da solução do risco [3]. Controle de risco é o processo de elaborar planos de resolução de riscos, monitorar o *status* do risco, implementar planos de resolução de riscos, e corrigir desvios do plano [9]. O controle do risco abrange as atividades de: planejamento de gerenciamento de riscos, resolução de risco e monitoramento de risco [3]. O planejamento ajuda a organização a se preparar para reduzir ou eliminar cada um dos riscos identificados, desenvolvendo estratégias e documentando no plano de risco. O monitoramento de riscos envolve acompanhar o *status* de cada risco e tomar ações corretivas quando algum limite for ultrapassado. Resolução de riscos consiste na eliminação ou resolução dos riscos, por meio da execução das ações descritas no plano de riscos [3].

Com o objetivo de controlar os riscos, na seção 3 são sugeridas ações preventivas para cada um dos riscos elencados na seção 2, de como o risco pode ser mitigado ou eliminado, por meio da adaptação do processo. As sugestões são obtidas a partir de ações propostas por modelos como o CMM [13], por processos como RUP [16], XP [10], Scrum [8], PMBok [14] ou por autores como Pressman [15], Boehm [4], entre outros.

Com base nas ações preventivas sugeridas, o processo de software pode ser adaptado, combinando características de métodos ágeis e planejados em um único processo de desenvolvimento (abordagem híbrida).

Segundo Boehm [4], empresas com uma grande base de clientes não necessitam apenas de rapidez de processo ou alta qualidade de produto - necessitam de ambos. Métodos ágeis ou planejados puros podem não atender essas necessidades. Por vezes, pode ser necessário misturar características de ambos tipos de processos, e Boehm propõe o uso do gerenciamento de riscos para definir o quanto de planejamento e de agilidade é necessário em cada projeto.

Utilizando-se como base o Processo Unificado da Rational (RUP) [16], na seção 4, são identificadas em quais etapas do ciclo de vida do software, as ações sugeridas devem ser inseridas. Por exemplo, a ação *"uso de técnica para elicitação de requisitos"* sugerida para o risco *"Instabilidade dos requisitos"*, deve ser inseridas na etapa (disciplina) de *Requisitos*.

Para acompanhar a efetividade das ações preventivas inseridas ao processo é necessário monitorar os fatores de risco, visando garantir que os riscos estão sob controle ou acionar o plano de resolução de riscos, caso algum risco se materialize.

Utilizando-se a abordagem GQM, na seção 5, são elaboradas metas visando à eliminação ou a avaliação da existência e monitoramento dos fatores de riscos. As metas são desdobradas em questões quantificáveis e posteriormente em métricas a serem obtidas para monitorar o progresso dos fatores de risco do projeto, indicando ao gerente quando as ações corretivas devem ser tomadas. Essas métricas são sugestões, a organização pode definir suas próprias métricas, de acordo com suas necessidades.

Na seção 6 é proposto um enfoque sistemático para tratamento dos riscos, com base nos assuntos discutidos nas seções anteriores. Na seção 7, o trabalho é concluído e são propostos trabalhos futuros.

2. Comparação entre os Riscos Identificados por Vários Autores

Vários estudos foram realizados buscando identificar os principais riscos de projetos de software. Alguns desses estudos consideram apenas os riscos sobre os quais os gerentes de projeto têm controle, outros consideram todos os tipos de riscos.
Neste trabalho, os riscos de projeto identificados pelos autores Keil et al. [12], Boehm [3], Addison e Vallabh [1] e pelo modelo CMMI [18] são comparados, buscando identificar um conjunto significativo dos principais riscos de projetos de software. Nas seções 2.1, 2.2, 2.3 e 2.4 são descritos brevemente os trabalhos realizados pelos autores acima citados, e na seção 2.5 é apresentada a comparação realizada.

2.1. Keil, Cule, Lyytinen e Schmidt

Em [12], é descrito um estudo envolvendo a identificação dos principais fatores de risco e a importância relativa de cada um destes. Inicialmente, os autores do artigo montaram painéis com gerentes de projetos de software experientes em Hong Kong, Finlândia e USA – e solicitaram a estes para identificarem os principais riscos de software. A pesquisa se baseou na experiência desse grupo de gerentes de projeto. A técnica Delphi [8] foi utilizada para obter o consenso no grupo.

2.2. Addison e Vallabh

Addison e Vallabh, descrevem em [1], um conjunto de riscos identificados por diversos pesquisadores e que segundo eles ocorrem em projetos de software. Os riscos elencados são usados em um estudo empírico realizado com gerentes de projetos de software sobre a percepção destes quanto a estes riscos de projetos e os controles mais efetivos para reduzi-los, ou minimizar os seus efeitos. Os autores utilizaram questionários para coletar dados para a pesquisa. A pesquisa abrangeu 70 gerentes de projeto. A análise dos questionários retornados foi conduzida usando uma combinação de técnicas de regressão e de ordenação.

2.3. Dez Riscos de Boehm

Boehm [3] descreve os dez principais riscos de software com base em uma pesquisa realizada com vários gerentes de projeto experientes.

2.4. Riscos Típicos Segundo o CMMI

No *Capability Maturity Model Integration* (CMMI) é citado um conjunto de riscos externos e internos típicos em projetos de software, dentro da área-chave de Gerência de Riscos [18]. Nesse artigo é desconsiderado o risco "Capacidade do vendedor inadequada ou incerta", pois nesse estudo são considerados somente os riscos relacionados ao processo de desenvolvimento (cliente, requisitos, planejamento e execução).

2.5. Comparação entre os riscos

A Tabela 1 mostra a relação de riscos citados por estes autores, agrupados por similaridade. Na coluna 1 são listados todos os riscos. Para as colunas 2, 3, 4 e 5, utilizou-se a seguinte convenção: a letra 'S' na coluna correspondente ao nome do autor indica que o mesmo cita o risco em seu trabalho e a letra 'N' indica que o autor não cita o risco no seu trabalho.

Tabela 1 – Comparação entre os Principais Riscos de Software

Riscos	**Keil**	**Boehm**	**Addison**	**CMMI**
Falta de compromisso da gerência sênior com o projeto	S	N	S	N
Falha em obter compromisso dos usuários e falta de envolvimento adequado dos usuários	S	N	S	N
Não-entendimento dos requisitos, falha na gerência de expectativas dos usuários e instabilidade dos requisitos	S	S	S	S
Escopo/objetivos não-claros e diferentes expectativas sobre o sistema entre distintos usuários.	S	N	S	N
Falta de conhecimento/habilidade requerida do pessoal do projeto e equipe insuficiente ou inadequada	S	S	S	S
Introdução de novas tecnologias	S	N	S	S
Cronograma e orçamento não-realistas	N	S	S	S
Falta de uma metodologia de projeto efetiva	N	N	S	N
Acréscimo de funcionalidades idealizadas pela equipe sem o aval do cliente (*gold plating*)	N	S	S	N
Desenvolvimento errado das funções (implementação não atende à especificação) ou interface	N	S	S	N
Subcontratação (tarefas ou componentes desenvolvidos externamente)	N	S	S	S
Uso de recursos e desempenho do sistema inadequados	N	S	S	N
Projeto (desenho) inviável	N	N	N	S

Os riscos listados na Tabela 1 podem ser classificados utilizando um critério adaptado do *framework* de classificação de riscos proposto por Keil em [12]. Segundo essa classificação, os riscos podem ser:

Riscos de cliente: são classificados como risco de clientes, os que se originam no grau de envolvimento dos clientes com o sistema que está sendo desenvolvido [12].

Riscos de Requisitos: a definição dos requisitos de software é um dos pontos mais importantes do processo de desenvolvimento, pois se o software entregue ao cliente não atende aos requisitos definidos, não satisfazerá as necessidades do cliente [12]. Os riscos de requisitos descrevem preocupações quanto ao entendimento das necessidades do cliente, tais como: definição inadequada de escopo, definições ambíguas ou inexistentes dos requisitos funcionais e não funcionais, entre outras.

Riscos de Planejamento: são classificados como riscos de planejamento, os riscos que monitoram o planejamento inicial do projeto de software, tais como: recursos disponíveis, qualificação da equipe, estimativas de cronograma e custos, apoio da gerência sênior ao projeto, etc.

Riscos de Execução: os riscos de execução descrevem situações que podem ocorrer afetando a qualidade do software desenvolvido, incluindo: desempenho inadequado do sistema, desenvolvimento errado das funções de software, erros, etc.

A Tabela 2 mostra os riscos agrupados por tipo de risco. Neste trabalho, os riscos de gerência de projetos foram separados em riscos de planejamento e execução, para permitir separar os riscos que devem ser mitigados no início do projeto (planejamento) e os demais riscos de gerência (execução).

Tabela 2 – Riscos Classificados por Tipo de Risco

Cliente Falta de envolvimento dos usuários Falha em obter compromisso dos usuários Falha na gerência de expectativas dos usuários. Conflito entre os departamentos do usuário	**Requisitos** Não-entendimento dos requisitos Escopo/objetivos não claros Instabilidade de requisitos
Planejamento Falta de compromisso da gerência sênior com o projeto Falta de conhecimento/habilidade pela equipe Equipe insuficiente ou inadequada Cronograma e orçamento não-realistas Falta de uma metodologia de projeto	**Execução** Introdução de novas tecnologias Requisitos criados pelos desenvolvedores (*gold plating*) Desenvolvimento errado das funções ou interface Sub-contratação Uso de recursos e desempenho do sistema Projeto (desenho) inviável

3. Sugerindo práticas para Minimizar Riscos

Uma organização de software pode precisar adaptar seu processo de desenvolvimento de acordo com o projeto que está desenvolvendo, acompanhando as características do software e do cliente. O ideal é que o processo padrão dessas organizações possa ser adaptado de acordo com as características do projeto, por exemplo, para projetos pequenos com requisitos instáveis, o processo padrão da organização pode ser customizado para englobar características de métodos ágeis, e para projetos grandes, que exigem grandes equipes de programadores, o processo pode ser customizado para englobar mais formalismo e planejamento, aproximando-se de um método planejado [4].

Este trabalho propõe o uso do gerenciamento de riscos como auxílio para a customização do processo de software da organização (mais ágil ou mais planejado). O primeiro passo é identificar os riscos que podem afetar o projeto, usando o conjunto de riscos mostrados na Tabela 1. Nesta seção são sugeridas algumas ações que poderiam ser tomadas para minimizar cada um dos riscos do projeto. Estas ações podem resultar em mudanças no processo de modo a torná-lo mais ágil ou planejado. A sugestão das ações a serem tomadas foi compilada de diversos autores já citados, e está sujeita a acréscimos de outras fontes, e adaptações à organização em questão.

Risco 1. Falta de compromisso da gerência sênior com o projeto.

Para minimizar esse risco é necessário envolver a alta gerência durante todo o ciclo de vida do projeto [1]. Algumas ações que podem ser tomadas:

- ⇨ Participação do gerente sênior (GS) nas reuniões de acompanhamento nos marcos do projeto (fim das iterações) [16], ou em reuniões de planejamento como o *Planning Game* – XP [10] ou planejamento do Sprint – SCRUM [17];
- ⇨ O gerente sênior revisando os compromissos do projeto com o gerente de projeto [13] - CMM.

Risco 2. Falha em obter compromissos do usuário, falta de envolvimento adequado dos usuários.

O cliente envolvido e cooperando com o desenvolvimento do software é um dos fatores de sucesso dos métodos ágeis [4]. Projetos utilizando métodos ágeis nos quais o cliente não está envolvido com o desenvolvimento apresentam altos riscos. Em [11], é descrita uma implantação do método RUP, adaptado para se tornar ágil. O processo "RUP Agile" foi implementado em dois projetos, e no projeto no qual os clientes não estavam envolvidos, este fracassou.

A suavização ou eliminação desse risco exige um bom relacionamento com o cliente/usuário. Ações sugeridas:

- ⇨ Envolver os usuários durante todo o ciclo de vida do projeto, como exemplo pode-se citar a prática *Onsite customer* – XP [10].
- ⇨ Envolver o usuário por meio da validação de protótipos [15] e critérios de aceite de produtos [13].

Risco 3. Não entendimento dos requisitos, requisitos instáveis e falha na gerência de expectativas dos usuários. Ações sugeridas para minimizar esse risco:

- ⇨ Desenvolvimento iterativo e incremental [16]; ao final de cada iteração é entregue uma versão do software ao cliente. Os métodos ágeis focam em iterações bastante curtas (máximo um mês) [10], enquanto métodos dirigidos a planos aceitam iterações bastante longas, seis meses, por exemplo. Com base nos riscos de requisitos, é possível adotar iterações mais longas (requisitos estáveis) ou mais curtas (requisitos mudando rapidamente);
- ⇨ Uso de técnicas de elicitação de requisitos, tais como: *User Story* – XP [10];
- ⇨ Validação junto ao cliente dos requisitos definidos [13]. Essa prática é bastante importante nos casos em que o preço é fixo por contrato [13];
- ⇨ Elaboração de protótipos para validação junto ao usuário [15];
- ⇨ Gerenciar as alterações (mudanças) de requisitos utilizando um procedimento documentado [13,16].

Risco 4. Escopo / objetivos não claros e conflito entre os departamentos do usuário. Ações sugeridas para minimizar esse risco:

- ⇨ Pequenos incrementos de funcionalidades em versões (*releases) de software*, incluindo alterações sugeridas na próxima versão do software (desenvolvimento incremental) – prática comum em métodos ágeis [10];
- ⇨ Refatoração sem custos adicionais – XP [10];
- ⇨ Processo de controle de mudanças [16,13].

Risco 5. Pessoal do projeto com falta de conhecimento e/ou habilidade requerida e equipe insuficiente ou inadequada. Ações sugeridas para minimizar esse risco:

- ⇨ Atribuir papéis e responsabilidades para os membros da equipe [14];
- ⇨ Assegurar que as pessoas estão aptas para desempenhar seu papel, seja porque a pessoa já possui a habilidade ou porque foi treinada [13] para tal.

Risco 6. Introdução de Novas Tecnologias. Ações sugeridas para minimizar esse risco:

- ⇨ Uso de práticas de métodos ágeis, que facilitem o re-trabalho controlado, como a refatoração em XP [10];
- ⇨ Validar a arquitetura da solução já nas primeiras iterações [16];
- ⇨ Garantir que o fornecedor está apto para dar suporte a tecnologia [14].

Risco 7. Cronograma/orçamento não-realista.

Algumas práticas para minimizar esse risco:

- ⇨ Estabelecer um procedimento para estimar prazo e custo dos projetos [14];
- ⇨ Manter uma base histórica de medidas de projetos anteriores, para ser usada nas estimativas [14].

Risco 8. Falta de uma metodologia de projeto efetiva. Ações sugeridas:

- ⇨ Definir uma metodologia de projeto [13];
- ⇨ Avaliar e melhorar a metodologia ao final de cada iteração [16];
- ⇨ Realizar revisões internas ou inspeções de qualidade, para garantir que a metodologia está sendo usada [13].

Risco 9. Criação de requisitos pela equipe de desenvolvimento (Gold Plating)

Freqüentemente desenvolvedores e analistas promovem alterações e elaboram capacidades adicionais nos sistemas, as quais eles imaginam tornariam o sistema melhor ou mais atrativo (daí a expressão em inglês *gold plating*). Estes desvios podem resultar em usuários insatisfeitos e custos desnecessários [1]. Ações sugeridas:

- ⇨ Garantir que a implementação dos requisitos tenha como base os requisitos definidos pelo cliente [15].
- ⇨ Iterações curtas com entregas freqüentes [16].

Risco 10. Desenvolvimento errado das funções ou interface

Esse risco ocorre quando a equipe de desenvolvimento não é comunicada sobre mudanças ou melhorias projetadas ou incluídas no software pelo cliente. A definição das alterações está correta, mas falha na comunicação destas a equipe. Ações sugeridas:

- ⇨ Definir um procedimento para mudanças e para gerência de configuração [13];
- ⇨ Garantir que a comunicação entre a equipe está fluído [14].

Risco 11. Subcontratação. Ações sugeridas:

- ⇨ Definir um processo de gerência de subcontratação – definir marcos de acompanhamento das atividades ao longo do projeto para garantir que os prazos e os custos estejam sendo cumpridos, e verificar a qualidade dos produtos desenvolvidos por terceiros. O CMM, apresenta no nível 2, uma área chave de processo que trata especificamente de gerência de subcontratação [13] ;

Risco 12. Uso de recursos e desempenho do sistema

Esse risco está associado à verificação de que os requisitos de desempenho são viáveis com a solução técnica proposta. Ações sugeridas:

- ⇨ Validar os requisitos de desempenho nas iterações iniciais do projeto [16].

Risco 13. Projeto (design) Inviável

A validação da arquitetura é dos fatores que afeta a viabilidade do desenho do sistema. Por isso, são sugeridas as seguintes ações:

- ⇨ Validar a arquitetura do sistema nas iterações iniciais do projeto. No RUP [16], um dos objetivos primários da fase de elaboração é a validação da arquitetura.
- ⇨ XP foca nas práticas de refatoração e refinamento da arquitetura sem custos, como forma de manter o projeto simples [10].

4. Mapeamento de Riscos para o RUP

Para cada ação preventiva sugerida na seção 3 deste trabalho, é necessário identificar a etapa de desenvolvimento onde esta ação deve ser inserida. As etapas de desenvolvimento variam conforme o modelo e o ciclo de vida escolhidos.

Neste trabalho, optou-se por utilizar as etapas descritas no Processo Unificado da Rational (RUP) [16], por este ser um dos mais utilizados processos de software e ser bastante completo [6], e os autores desse trabalho já terem familiaridade com este processo. As análises aqui realizadas podem ser adaptadas a outros processos.

O Processo Unificado Rational (RUP) é um processo de negócio genérico para engenharia de software orientada a objetos. Seu objetivo é assegurar a produção de produtos de alta qualidade que satisfaçam as necessidades de seus usuários finais, dentro de cronogramas e orçamentos previsíveis [16]. O Processo Unificado descreve um conjunto de atividades necessárias para transformar os requisitos do usuário em um sistema de software. RUP utiliza a Linguagem de Modelagem Unificada (UML) para descrever os modelos e diagramas do sistema [16].

As atividades em RUP são organizadas em áreas de concentração ou disciplinas, que são divididas em seis disciplinas de engenharia: modelagem de negócio, requisitos, análise e projeto, implementação, teste e implantação; e três disciplinas de suporte: gerência de configuração e alteração, gerência de projeto e ambiente.

4.1. Organização das ações

As ações sugeridas na seção 3 serão organizadas em tabelas identificando para cada risco, quais as ações que podem ser inseridas e em qual etapa do desenvolvimento. A Tabela 3 mostra as ações sugeridas para suavizar e eliminar os riscos de Cliente, Requisitos e Planejamento. Na Tabela 4 são exibidas as ações sugeridas para eliminar os riscos de execução. As tabelas foram organizadas por tipo de risco para facilitar a visualização. As etapas requisitos e modelagem de negócio; e análise e projeto e implementação serão tratadas em conjunto, para evitar a exibição duplicada das ações sugeridas nas duas etapas.

Tabela 3 – Ações sugeridas para os riscos de cliente, requisitos e planejamento

Etapas	**MN e R**		**AP**	**GC**	**GP**					**AM**		**TE**	
Ações Sugeridas / **Riscos de Cliente, Requisitos e Planejamento**	Validar protótipos e critérios de aceite com o usuário	Uso de técnicas para elicitação de requisitos	Refatoração sem custos adicionais	Gerenciar as alterações nos requisitos	GS deve participar de reuniões Acompanhamento	Revisar compromissos do projeto com o GP	Definir cargos e responsabilidades para equipe	Assegurar que a equipe está habilitada para executar atividades	Definir estimativas de prazo e de custo usando dados históricos	Definir um processo de desenvol-vimento e melhorá-lo	Realizar revisões para verificar o uso do processo	Envolver os usuários durante todo o ciclo de vida	Utilizar um desenvolvimento iterativo e incremental
Falta de compromisso da gerência sênior com o projeto					X	X							
Falta de envolvimento adequado dos usuários e falha em obter compromisso dos usuários	X											X	
Não-entendimento dos requisitos, falha na gerência de expectativas dos usuários e requisitos instáveis	X	X		X									X
Escopo / objetivos não claros e conflito entre os departamentos do usuário			X	X									X
Falta de conhecimento/ habilidade requerida do pessoal do projeto e equipe insuficiente ou inadequada							X	X					
Cronograma e orçamento não-realistas									X				
Falta de uma metodologia de projeto efetiva										X	X		

Legenda:
MN e R: Modelagem de Negócio e Requisitos
AP: Análise e Projeto
GC: Gerência de Configuração
GP: Gerência de Projeto
AM: Ambiente
TE: Todas as Etapas

5. Monitorando os Riscos de Software

Para manter os fatores de risco sob controle, é necessário constantemente medi-los e observar os indicadores do projeto para identificar tendências, planejar as ações futuras e os índices de probabilidade do risco se materializar [9]. Neste trabalho, seguindo uma sugestão de W. Mello [19], formulada em troca de e-mails com os autores, utilizou-se a abordagem *Goal/Question/Metric* [2] para elaborar um conjunto de métricas que possibilitem o acompanhamento de riscos. A medição, por si só, não é o objetivo deste

trabalho, o objetivo é utilizar a medição para monitorar riscos, provendo realimentação para a análise e a melhoria dos processos da organização.

Tabela 4 – Ações sugeridas para riscos de execução

Etapas	**AP e I**			**GC**	**GP**		**AM**	**TE**
Ações Sugeridas / **Riscos de Execução**	Retrabalho e refatoração controlados	Validar a arquitetura da solução	Desenvolvimento baseado nos requisitos definidos pelo cliente	Definir procedimento GC e mudança	Comunicação equipe satisfatória	Acompanhar atividades de subcontratação	Obter garantia que o fornecedor está habilidade a dar suporte a tecnologia	Utilizar desenvolvimento iterativo e incremental
Introdução de novas tecnologias		X	X				X	
Requisitos criados pelos desenvolvedores (*Gold Plating*)			X					X
Desenvolvimento errado das funções ou interface				X	X			
Subcontratação (tarefas ou componentes desenvolvidos externamente)						X		
Uso de recursos e desempenho do sistema (ou desempenho em tempo real)		X						X
Projeto (desenho) inviável	X	X						X

Legenda:
AP: Análise e Projeto
I: Implementação
GC: Gerência de Configuração
GP: Gerência de Projeto
AM: Ambiente
TE: Todas as Etapas

5.1. Abordagem *Goal/Question/Metric*

O paradigma *Goal/Question/Metric* (GQM) [2] é uma abordagem orientada a metas e utilizada em engenharia de software para a medição de produtos e processos de software. GQM é baseado no requisito de que toda a coleta dos dados deve ser baseada num fundamento lógico, em um objetivo ou meta, que é documentado explicitamente. O primeiro passo nessa abordagem é definir metas a serem alcançadas no programa de medição. Após a identificação das metas, um plano GQM é elaborado para cada meta selecionada. O plano consiste, para cada meta, em um conjunto de questões quantificáveis que especificam as medidas adequadas para sua avaliação [2]. As questões identificam a informação necessária para atingir a meta e as medidas definem operacionalmente os dados a serem coletados para responder as perguntas.

As metas GQM devem ser formuladas da seguinte forma: "Analisar o <*objeto de estudo*> com a finalidade de <*objetivo*> com respeito ao <*enfoque*> do ponto de vista de <*ponto de vista*> no seguinte contexto <*contexto*>".

Sendo que os atributos, em itálico na sentença acima, podem ser definidos como [21]:

- Objeto de estudo: identifica o que será analisado. Exemplo: processo de software, projeto, documento, sistema, etc.
- Objetivo: porque o objeto será analisado. Exemplo: avaliar, melhorar, monitorar, controlar, etc.
- Enfoque: identifica o atributo que será analisado. Exemplo: confiabilidade, custos, correção, etc.

- ⇨ Ponto de vista: identifica quem utilizará as métricas coletadas. Exemplo: equipe de desenvolvimento, gerente de projeto, etc.
- ⇨ Contexto: identifica o ambiente onde o programa de medição está localizado. Exemplo: projeto A, departamento X.

5.2. Usando GQM para Monitorar Riscos

Neste trabalho, GQM é utilizado para definir métricas a serem coletadas com o objetivo de monitorar os riscos.
A Tabela 5 mostra algumas metas GQM, formuladas a partir dos riscos de projeto exibidos na Tabela 1; questões e métricas derivadas a partir da meta, apenas como sugestão. Cada projeto deverá elencar as métricas mais adequadas.
Neste trabalho, omitiu-se o contexto na definição da meta, pois este vai depender do ambiente em que a sistemática será aplicada.

Tabela 5 – Metas e Questões GQM por Risco

Risco 1	
Meta 1: Analisar o projeto com a finalidade de monitorar com respeito ao comprometimento da gerência do ponto de vista da equipe de desenvolvimento.	
Questão 1.1: A gerência sênior está participando das reuniões de acompanhamento do projeto?	*Métrica 1.1:* Percentual de participação do gerente sênior PPGS = (número de reuniões que o gerente sênior participou / número de reuniões realizadas) * 100
Risco 2	
Meta 2: Analisar o projeto com a finalidade de monitorar com respeito ao envolvimento dos usuários do ponto de vista da equipe de desenvolvimento.	
Questão 2.1: Os usuários estão participando das reuniões do projeto?	*Métrica 2.1a:* Percentual de participação do usuário PPU = (número de reuniões que representantes dos usuários participaram / número de reuniões realizadas) * 100 *Métrica 2.1b:* Interações do usuário (comparar histórico) NIU = número de interações (telefone, e-mail, pessoalmente) entre o usuário e a equipe
Questão 2.2: Os usuários estão cumprindo os prazos acordados com o gerente de projeto para validação de protótipos, critérios de aceite de produtos, documentos?	*Métricas 2.2a:* Percentual de cumprimento de prazos pelo usuário PCPU = (número de vezes que os prazos foram cumpridos / número de solicitações de validação) * 100 *Métricas 2.2b:* Cumprimento de tarefas pelo usuário (comparar histórico) TMCT = Tempo médio de cumprimento de tarefas pelo usuário por tipo de tarefa
Risco 3 e Risco 9	
Meta 3: Analisar o projeto com a finalidade de monitorar com respeito à definição dos requisitos do ponto de vista da equipe de desenvolvimento.	
Questão 3.1: Os usuários validaram os documentos de requisitos do projeto?	*Métrica 3.1a:* Percentual de validação de requisitos pelo Cliente PVRC = (número de documentos de requisitos validados/ número total de documentos de requisitos) * 100 *Métrica 3.1b:* Percentual de requisitos elaborados pelo Cliente PRE = (número de requisitos elaborados pelos usuários/requisitos elaborados pelo pessoal de sistema) * 100
Questão 3.2: Os requisitos estão mudando durante a	*Métrica 3.2:* Percentual de Alteração dos requisitos

fase de execução do projeto?	PAR = (número de solicitações de mudança de requisitos / número total de requisitos) * 100
Questão 3.3: Os requisitos implementados satisfizeram as necessidades dos clientes?	*Métrica 3.3a:* Requisitos aceitos RA = (número de requisitos aceitos / número total de requisitos) * 100 *Métrica 3.3b:* Requisitos rejeitados RR = (número de requisitos rejeitados / número total de requisitos) * 100
Questão 3.4: O tempo para incorporar as mudanças está adequado?	*Métrica 3.4:* Percentual de mudanças realizadas no prazo Classificar as mudanças em simples, média e complexa. Estabelecer uma política de prazo máximo para que a mudança seja incorporada no sistema. PMRP = (número de mudanças realizadas no prazo / número de mudanças solicitadas) *100
Risco 4	
Meta 4: Analisar o projeto com a finalidade de controlar com respeito às solicitações de mudança do ponto de vista da equipe de desenvolvimento.	
Questão 4.1: Qual o número de solicitações de alteração de escopo/objetivos do projeto?	*Métrica 4.1:* Número de solicitações de mudança (comparar com dados históricos) NSM = Número de solicitações de mudança de escopo/objetivos
Risco 5	
Meta 5: Analisar o projeto com a finalidade de avaliar com respeito à habilidade das pessoas da equipe do ponto de vista do gerente de projeto.	
Questão 5.1: Qual o percentual de pessoas habilitadas para desempenhar seu papel na equipe?	*Métrica 5.1:* Percentual de pessoas habilitadas PPH = (número de pessoas habilitadas / número de pessoas da equipe) * 100
Questão 5.2: Existem recursos humanos suficientes para o projeto?	*Métrica 5.2:* Percentual de Recursos Humanos PRH = (número de RH / número de RH estimados pelo gerente de projeto[1])*100
Risco 6	
Meta 6: Analisar o projeto com a finalidade de avaliar com respeito à tecnologia empregada no projeto do ponto de vista do gerente de projeto.	
Questão 6.1: Qual o índice de retrabalho por problemas de tecnologia?	*Métrica 6.1:* Percentual de Retrabalho por tecnologia PRT = (quantidade de horas de retrabalho por defeitos ou falhas devido à tecnologia / quantidade total de horas de retrabalho) * 100
Questão 6.2: Qual o percentual de defeitos por tecnologia?	*Métrica 6.2:* Percentual de defeitos por tecnologia PDT = (quantidade de defeitos por tecnologia / quantidade total de defeitos)*100
Questão 6.3: Qual o percentual de pessoas na equipe que dominam a tecnologia?	*Métrica 6.3:* Percentual de pessoas que dominam a tecnologia PPDT = (número de pessoas que dominam a tecnologia / número total de pessoas)*100
Risco 7	
Meta 7: Analisar o processo de software com a finalidade de avaliar com respeito as estimativas do ponto de vista do gerente de projeto.	
Questão 7.1: Qual a precisão das estimativas de prazos?	*Métrica 7.1a:* Precisão das estimativas de prazo AEP = (duração real da iteração/ duração estimada da iteração) *100 *Métrica 7.1b:* Percentual de casos de uso desenvolvidos iteração

[1] O número de recursos estimados é baseado no esforço necessário para desenvolver o sistema. O esforço pode ser estimado com base no tamanho do software, usando-se técnicas como: Análise por Pontos de Função ou Análise por Casos de Uso.

	CUDI = (numero de casos de uso assinalados no inicio da iteração/casos de uso concluídos na interação) * 100
Questão 7.2: Qual a precisão das estimativas de esforço?	*Métrica 7.2:* Precisão das estimativas de esforço AEP = (esforço real da iteração/ esforço estimado da iteração)*100
Risco 8	
Meta 8: Analisar o projeto com a finalidade de monitorar com respeito ao uso da metodologia do ponto de vista da equipe de desenvolvimento.	
Questão 8.1: Os gerentes estão utilizando a metodologia de projeto?	*Métrica 8.1:* Número de ocorrência geradas em revisões (comparar dados históricos) NOR = Número de ocorrência geradas nas revisões de qualidade
Questão 8.2: A equipe está produzindo todos os documentos previstos na metodologia?	*Métrica 8.2:* Percentual de tipos de documentos produzidos PDP = (tipos de documentos produzidos/tipos de documentos previstos na metodologia) * 100
Risco 10	
Meta 10: Analisar o projeto com a finalidade de avaliar com respeito à comunicação do ponto de vista da equipe de desenvolvimento.	
Questão 10.1: As solicitações de mudanças encaminhadas ao gerente de projeto são comunicadas a equipe?	*Métrica 10.1:* Percentual de solicitações de mudanças encaminhadas PSME = (Quantidade de solicitações de mudanças encaminhadas à equipe / Quantidade de solicitações de mudanças do cliente)*100
Risco 11	
Meta 11: Analisar o projeto com a finalidade de monitorar com respeito à subcontratação do ponto de vista do gerente de projeto.	
Questão 11.1: Os prazos acordados com a subcontratada estão sendo cumpridos?	*Métrica 11.1a:* Cumprimento de prazos pela subcontratada CPS = duração real da atividade / duração estimada da atividade *Métrica 11.1b:* Percentual de casos de uso entregues PCUE = (número de casos de uso entregues/número de casos de uso acordados) * 100
Questão 11.2: Os produtos entregues pela subcontratada satisfazem os critérios de aceite?	*Métrica 10.2:* Percentual de requisitos aceitos da subcontratada (inclui testes) PRAS = (número de requisitos aceitos / número de requisitos entregues)*100
Risco 12	
Meta 12: Analisar o sistema com a finalidade de avaliar com respeito ao desempenho do ponto de vista do usuário.	
Questão 12.1: O desempenho atual do sistema satisfaz os índices desejados pelo usuário?	*Métrica 12.1a:* Desempenho do sistema DS = (desempenho atual do sistema / desempenho desejado do sistema)*100
Risco 13	
Meta 13: Analisar o projeto com a finalidade de analisar com respeito ao projeto (desenho) do ponto de vista da equipe de desenvolvimento.	
Questão 13.1: A arquitetura definida é viável?	*Métrica 13.1a:* Viabilidade da arquitetura EA = (número de atividades de desenho concluídas / número de atividades de desenho total)
Observação: o desempenho do sistema envolve os requisitos não-funcionais da aplicação. Para cada projeto devem ser definidas métricas de desempenho para avaliar itens específicos estabelecidos pelo usuário, tais como: tamanho da base de dados, crescimento da base de dados, necessidades de reorganização da base de dados, tempo de resposta, processos concorrentes, etc.	

As métricas, descritas neste trabalho, são apenas sugestões, e precisam ser avaliadas de acordo com as características de cada projeto.

6. Um Enfoque Sistemático ao Tratamento de Riscos

O enfoque sistemático para tratamento de riscos, proposto por este trabalho, pode ser resumido nas atividades:

- ⇨ Elaborar lista de riscos, analisar e priorizar os riscos;
- ⇨ Selecionar e elaborar ações preventivas para cada risco priorizado;
- ⇨ Identificar um conjunto de métricas e monitorar os riscos;
- ⇨ Desenvolver mecanismos de capturar métricas associadas ao processo;
- ⇨ Monitorar as métricas continuamente, para tomar ações corretivas caso o progresso desvie do esperado.

Essas atividades serão detalhadas nas seções 6.1, 6.2 e 6.3.

6.1. Elaborar Lista de Riscos, Analisar e Priorizar os Riscos

O gerente de projeto pode identificar os riscos do projeto de software utilizando o conjunto de riscos definido na Tabela 1, como uma lista de verificação (*checklist*), para definir os riscos que podem impactar o projeto.
Após a identificação dos riscos, é necessário analisá-los e priorizá-los, como sugestão para estas atividades, utilizar a técnica de exposição do risco (ER).
A exposição do risco, algumas vezes chamada de impacto do risco ou fator de risco, é definida como o produto entre a probabilidade de um resultado não satisfatório ocorrer e a perda associada a esse resultado não-satisfatório [4].

ER = Probabilidade (Resultado NS) * Perda (Resultado NS)

Em [7], são sugeridas guias simples para definir a probabilidade e a perda, que são:

Probabilidade: 0 representa uma probabilidade de ocorrência de menos de 5%, 5 representa uma probabilidade de aproximadamente 50% e 10 representa uma probabilidade de mais que 95%.

Perda: 0 representa nenhum aumento no custo ou no tempo do projeto, 10 representa um aumento significativo no custo ou no tempo (em torno de 100%).

Após a quantificação de cada risco, estes podem ser ordenados pela exposição do risco. Cabe ao gerente de projeto definir a quantidade de riscos que serão priorizados, isto é, se serão definidas ações corretivas para os riscos com ER maior que 50%, 60%, etc.

6.2. Selecionar e Elaborar Ações Preventivas para cada Risco Priorizado

A proposta deste trabalho é inserir ao processo, práticas para minimizar a probabilidade dos riscos ocorrerem. Então, para cada risco identificado, definir as práticas que precisam ser inseridas no processo específico para o projeto. A idéia de customização proposta por este trabalho é semelhante à proposta pelo Processo Unificado da Rational, onde a customização se refere à eliminação de atividades (ou artefatos) e inclusão de atividades (ou artefatos).
Nas Tabelas 3 e 4 são exibidas algumas práticas sugeridas por risco. Essas práticas podem ser inseridas no processo específico para o projeto de acordo com os riscos deste, visando suavizar estes riscos. A equipe de desenvolvimento pode elaborar suas próprias ações preventivas, pois o conjunto de ações descrito nesse trabalho é apenas uma sugestão, citada como exemplo. Por exemplo, para o risco *"instabilidade dos requisitos"*, as práticas sugeridas são: *"Validar protótipos e critérios de aceite com o usuário", "Uso de técnicas para elicitação de requisitos", "Gerenciar as alterações nos requisitos"* e *"Utilizar um*

desenvolvimento iterativo e incremental". Essas sugestões visam diminuir a probabilidade do risco ocorrer.
As ações preventivas podem ser vistas como padrões organizacionais, como sugerido por Júlio Hartmann [20], isto é, serão procedimentos que podem ser inseridos no processo específico do projeto.

6.3. Identificar um Conjunto de Métricas e Monitorar os Riscos

O próximo passo é definir as métricas que serão utilizadas para monitorar os riscos priorizados. Na Tabela 5 são sugeridas algumas métricas que podem ser usadas para monitorar os riscos. Essas métricas foram extraídas utilizando-se a abordagem GQM. A equipe pode elaborar suas próprias métricas, usando como exemplo as sugestões deste trabalho.
Com base no conjunto de métricas identificado, criar um plano de medição para o projeto, detalhando como as medidas devem ser coletadas, e em qual periodicidade. O ideal é que as medidas sejam inseridas em uma base histórica, para serem usadas em estimativas futuras e no acompanhamento do projeto. As métricas auxiliam o gerente de projeto no monitoramento dos riscos, dando visibilidade quanto ao progresso destes, cabe ao gerente de projeto identificar o momento em que as ações corretivas devem ser tomadas porque o risco está próximo a se materializar.
Como exemplo para o risco "*instabilidade dos requisitos*", as métricas sugeridas são: "*Métrica 3.1a: Percentual de validação de requisitos pelo Cliente*", "*Métrica 3.1b: Percentual de requisitos elaborados pelo Cliente*", "*Métrica 3.2: Percentual de Alteração dos requisitos*", *"Métrica 3.3a: Requisitos aceitos"*, *"Métrica 3.3b: Requisitos rejeitados"* e *"Métrica 3.4: Percentual de mudanças realizadas no prazo"*.
Com o uso das métricas, o gerente de projeto obterá indicativos que poderão auxiliá-lo na identificação de quando um risco está próximo a se materializar.

7. Conclusão e Trabalhos Futuros

Este trabalho propõe uma sistemática para prevenção de riscos baseada na adaptação do processo de software, de acordo com os riscos definidos, de forma a torná-lo mais ágil ou planejado.
O tratamento sistemático dos riscos envolve elaborar uma lista com os principais riscos do projeto, analisando-os e priorizando-os, definir as etapas do processo de desenvolvimento onde o risco tem origem, visando inserir práticas nessas etapas que venham a minimizar ou eliminar os fatores de risco; e também monitorar constantemente os fatores de risco, como forma de identificar se a probabilidade dos riscos se materializarem está aumentando, pois neste caso o gerente de projeto deve executar as ações descritas no plano de resolução de riscos.
Para facilitar o uso da sistemática descrita acima, neste trabalho são sugeridas ações preventivas que podem ser inseridas no processo de software, com o objetivo de reduzir a probabilidade do erro ocorrer, e também são sugeridas algumas métricas elaboradas usando a abordagem *Goal/Question/Metric* para monitorar os riscos. Salienta-se que as ações e as métricas são algumas sugestões que devem ser complementadas de acordo com a organização de software.
A sistemática proposta e o conjunto de ações e métricas sugeridas estão sendo experimentados em projetos reais.

Os autores estão trabalhando no projeto de uma ferramenta para automatizar a sistemática de gerência de riscos descrito neste trabalho e auxiliar o gerente no monitoramento dos fatores de risco e na tomada de decisão.

A ferramenta monitorará os riscos, com base nos dados coletados de forma automática, caso possível, ou manual. Esses dados podem ser utilizados para avaliar a estabilidade e a capacidade do processo, predizer custos e desempenho futuros, prover *benchmarks*, plotar tendências de evolução dos riscos, avaliar e melhorar as práticas sugeridas para minimizar os riscos.

Segundo Fenton e Neil [5], o futuro para métricas de software é prover informação para suportar a tomada de decisão gerencial durante o ciclo de vida, e que o bom suporte é aquele que avalia e reduz o risco. Enquanto Fenton e Neil propõem a modelagem causal como forma de avaliar e reduzir riscos, neste artigo é proposta a abordagem GQM.

Este trabalho está inserindo em um grupo de pesquisa, sob a orientação do Prof. Roberto Tom Price. Júlio Hartmann, que também faz parte do grupo, está trabalhando em uma abordagem para a instanciação de metodologias de desenvolvimento de software utilizando padrões organizacionais e critérios de risco. Essa abordagem, chamada *Pattern-based Methodology Tailoring* (PMT), tem como objetivo facilitar a adaptação de uma metodologia ao contexto específico de um projeto e de uma organização, por meio da seleção dos padrões mais adequados aos requisitos metodológicos. As ações preventivas descritas neste trabalho poderiam ser vistas como os padrões organizacionais, sugeridos por Júlio.

Referências

1. Addison, T., Vallabh, S. Controlling Software Project Risks – An Empirical Study of Methods used by Experienced Project Managers. Proceedings of South African Institute of Computer Scientists & Information Technologists, Port Elizabeth, South Africa, September 2002, p.128-140.
2. Basili, V. R, Rombach, D. The TAME Project: Towards Improvement-Oriented Software Environments. IEEE Transactions on Software Engineering, v.14, n.6, p. 758-773, June 1988.
3. Boehm, B. Software Risk Management: Principles and Practices. IEEE Software, v.8, n.1, p. 32-41, January 1991.
4. Boehm, B. Get Ready for Agile Methods, with care. IEEE Computer, v.35, n.1, p. 64-69, January 2002.
5. Fenton, N., Neil, M. Software Metrics: A Roadmap. Proceedings of 22th Conference on The Future of Software Engineering, Limerick, Ireland, 2000, p. 357-370.
6. Manzoni, L. V., Price, R. T. Identifying Extensions Required by RUP (Rational Unified Process) to Comply with CMM (Capability Maturity Model) Levels 2 and 3. IEEE Transaction on Software Engineering, v. 29, n. 2, p. 181-192, February 2003.
7. Coppendale, J. Managing Risk in Product and Process Development and Avoid Unpleasant Surprises. Engineering Management Journal, v.5, n.1, p. 35-38, February, 1995.
8. Schmidt, R. C. Managing Delphi Surveys using Nonparametric Statistical Techniques. Decision Sciences Journal, v. 28, n.3, p. 763-774, 1997.
9. Hall, E. Managing Risk: Methods for Software Systems Development. New York: Addison-Wesley, 1998.
10. Hightower, R. Lesiecki, N. Java Tools for Extreme Programming. New York: John Wiley & Sons, 2002.
11. Hirsch, M. Making RUP Agile. Proceedings of Conference on Object Oriented Programming Systems Languages and Applications, Seattle, Washington, 2002, p.1–ff.

12. Keil, M. et al. A Framework for Identifying Software Project Risks. Communication of the ACM, v. 41, n.11, p. 76-83, November 1998.
13. Paulk, M. C. et. al. Key Practices of the Capability Maturity Model, Version 1.1. Technical Report CMU/SEI-93-TR-025, 1993.
14. Project Management Institute. A Guide to the Project Management Body of Knowledge. Pennsylvania, 2000.
15. Pressman, R. Engenharia de Software, 5 ed. Makron Books, 2002.
16. Rational Software Corporation. Rational Unified Process, v. 2001. Cupertino, 2001.
17. Schwaber, K., Beedle, M. Agile Software Development with Scrum. Upper Saddle River: Prentice Hall, 2001.
18. Software Engineering Institute. Capability Maturity Model Integration (CMMI), Version 1.1. Pittsburgh: Software Engineering Institute, Carnegie Mellon University, 2002. (CMU/SEI-2002-TR-012).
19. Melo, W. L. Polymorphism Measures for Early Risk Prediction. Proceedings of the 21th International Conference on Software Engineering, Los Angeles, CA, 1999, p.334-344.
20. Hartmann J., Price, R. T. *Pattern-based Methodology Tailoring.* Proposta de Dissertação de Mestrado, UFRGS, Porto Alegre, 2004.
21. Gresse, C. B., Hoisl, J. W. A Process Model for GQM- Based Measurement. Technical Report STTI-95-04-E, Software Technology Transfer Initiative, University of Kaiserslautern, Department of Computer Science, Kaiserslautern, Germany, 1995.