Apoio Automatizado à Gerência de Riscos Cooperativa.

Conference Paper · January 2007

Source: DBLP

CITATIONS
2

READS
63

3 authors, including:

Victorio Albani Carvalho
Instituto Federal de Educação, Ciência e Tecnologia do Espírito Santo...
17 PUBLICATIONS **98** CITATIONS

SEE PROFILE

Ricardo de Almeida Falbo
Universidade Federal do Espírito Santo
172 PUBLICATIONS **1,661** CITATIONS

SEE PROFILE

Apoio Automatizado à Gerência de Riscos Cooperativa

Victorio Albani de Carvalho, Alexandre G. N. Coelho, Ricardo de Almeida Falbo

Departamento de Informática, Universidade Federal do Espírito Santo, Vitória - ES - Brasil
{victorioalbani, alexandrecoelho20}@yahoo.com.br,
falbo@inf.ufes.br

Abstract. Software process is inherently cooperative. During its activities, developers work together in order to develop a quality product. Especially, project managers are involved in decision make activities that can drive the project to success or failure. Those activities are better performed by experienced people, when developers work cooperatively, and when knowledge is shared. In this paper, we present a tool that supports cooperative development of Risk Plans.

1 Introdução

As atividades do processo de software são reconhecidamente cooperativas. Durante um projeto de software, diversos profissionais trabalham em conjunto com o objetivo comum de desenvolver um produto de software de qualidade, que atenda às necessidades do cliente, obedecendo a restrições de custos e prazos.

Em especial, gerentes de projetos são responsáveis pela tomada de decisões que podem levar um projeto ao sucesso ou ao fracasso. A experiência é unanimemente apontada como uma das principais características desejáveis de um bom gerente de projetos. Entretanto, cada projeto de software é único. Assim, por mais experientes que sejam os gerentes de projetos, muitos ganhos podem ser obtidos através da troca de experiências durante a realização das atividades de gerência.

Essa troca de experiências pode se dar de várias formas. Outros gerentes podem ser chamados a opinar informalmente ou a relatar suas experiências, profissionais podem realizar atividades efetivamente de modo cooperativo, elaborando conjuntamente certos artefatos, ou podem-se utilizar dados históricos de projetos anteriores similares. Neste último caso, contudo, para tomar uma decisão em um projeto, muitas vezes, é importante conhecer, também, as justificativas e razões existentes por trás de uma decisão tomada, além de conhecer a decisão em si e os produtos dela decorrentes. Assim, é útil registrar, não apenas os artefatos gerados em uma atividade, mas também as razões por trás das decisões tomadas, incluindo justificativas, alternativas consideradas e os argumentos que conduziram à decisão.

Esta situação geral se aplica à gerência de riscos. Todo projeto está sujeito a riscos que podem afetar o seu andamento e os seus custos. Gerentes de projeto devem determinar a probabilidade e o impacto da ocorrência desses eventos indesejáveis e fazer planos para evitar que eles ocorram ou, caso sejam inevitáveis, procurar minimizar as conseqüências negativas dos mesmos [1]. Tipicamente, todas essas

informações são registradas em um documento, o Plano de Riscos. Mas além de ter acesso ao Plano de Riscos em si, é bastante útil conhecer as razões que levaram às decisões que eles escondem.

O trabalho apresentado neste artigo tem por objetivo facilitar a troca de experiências na Gerência de Riscos, por meio da elaboração cooperativa de Planos de Risco e da captura do raciocínio seguido durante a realização dessa atividade e posterior disseminação desse conhecimento. Para tal, GeRis [2], a ferramenta de apoio à gerência de riscos do ambiente de desenvolvimento de software ODE [3], foi reformulada e estendida.

Este artigo está organizado da seguinte forma: a seção 2 discute brevemente a Gerência de Riscos e três potenciais instrumentos de apoio a seu complexo processo: automatização do processo, gerência de conhecimento e apoio ao trabalho cooperativo; a seção 3 apresenta sucintamente o ambiente ODE, sua ferramenta de apoio à gerência de riscos (GeRis) e sua infra-estrutura de gerência de conhecimento, procurando destacar recentes melhorias feitas; a seção 4 trata das novas funcionalidades providas no ambiente para a elaboração cooperativa de planos de riscos e como o conhecimento envolvido nessa tarefa é capturado; finalmente a seção 5 apresenta as considerações finais do artigo.

2 Apoio ao Processo da Gerência de Riscos

Segundo o padrão IEEE Std 1540-2001 [4], a Gerência de Riscos visa a identificar potenciais problemas antes que eles ocorram, de forma que ações possam ser tomadas a fim de reduzir ou eliminar a probabilidade e o impacto desses problemas.

De modo simplificado, podemos pensar um risco como uma probabilidade de alguma circunstância adversa ocorrer, ameaçando o projeto, o software que está sendo desenvolvido ou a organização. Os riscos podem surgir como decorrência de requisitos mal definidos, de dificuldades em estimar prazos, custos e recursos, da dependência de habilidades individuais e de mudanças nos requisitos, dentre outros. Um gerente de projeto deve prever riscos, compreender o impacto desses riscos no projeto, no produto e nos negócios e tomar providências para evitá-los. Planos de contingência podem ser traçados para que, se os riscos vierem realmente a ocorrer, seja possível uma ação imediata que vise à recuperação [5].

Os riscos sempre envolvem duas características: incerteza (o evento que caracteriza um risco pode ou não acontecer) e perda (se um risco se tornar realidade, conseqüências indesejáveis vão ocorrer). Quando riscos são analisados, é importante quantificar o nível de incerteza e o grau de perdas associados a cada risco. Além disso, as atividades da gerência de riscos devem ser aplicadas iterativamente durante todo o acompanhamento do projeto de software [6]. Essas atividades incluem:

- Identificação de riscos: visa a identificar possíveis ameaças (riscos) para o projeto, antecipando o que pode dar errado;
- Análise de riscos: trata de analisar os riscos identificados, estimando sua probabilidade e impacto (grau de exposição ao risco);
- Avaliação de riscos: busca priorizar os riscos e estabelecer um ponto de corte, indicando quais riscos serão gerenciados e quais não serão;

- Planejamento de ações: trata do planejamento das ações a serem tomadas para evitar que um risco ocorra (ações de mitigação) ou para definir o que fazer quando um risco se tornar realidade (ações de contingência);
- Elaboração do Plano de Riscos: todos os aspectos envolvidos na gerência de riscos devem ser documentados em um plano de riscos, indicando os riscos que compõem o perfil de riscos do projeto, suas avaliações, a definição dos riscos a serem gerenciados e, para esses, as ações de mitigação e contingência;
- Monitoramento de riscos: à medida que o projeto progride, os riscos têm de ser monitorados para se verificar se os mesmos estão se tornando realidade ou não. Novos riscos podem ser identificados, o grau de exposição de um risco pode mudar e ações podem ter de ser tomadas. Essa atividade é realizada durante o acompanhamento do progresso do projeto.

A importância de se gerenciar riscos é atualmente inquestionável. Quase todas as normas e modelos de qualidade e padrões de gerência de projetos advogam que essa prática é essencial. O CMMI [7], por exemplo, possui uma área de processo totalmente dedicada à gerência de requisitos no nível 3 de maturidade (Gerência de Requisitos), além de possuir práticas específicas e sub-práticas nas áreas de processo Planejamento de Projeto e Controle e Monitoração de Projetos, ambas do nível 2.

Contudo, apesar de ter sua importância reconhecida, muitas organizações encontram dificuldades na realização dessa atividade. Segundo resultados da pesquisa em Qualidade e Produtividade no Setor de Software Brasileiro de 2001 [8], apenas 11,8% das organizações de software brasileiras realizavam esta prática. Assim, é muito importante procurar prover meios de apoiar a realização desta atividade. Dentre esses meios podemos citar o apoio automatizado, o compartilhamento de experiências e o apoio ao trabalho cooperativo.

As ferramentas CASE (*Computer-Aided Software Engineering*) têm sido um meio bastante utilizado para apoiar gerentes e profissionais de Engenharia de Software na realização de atividades do processo de software. Neste conjunto de ferramentas inclui-se também a gerência de riscos. Dentre as diversas ferramentas CASE de apoio à gerência de riscos podem ser citadas RiskRadar[1], ProRisk[2] e RiskyProject[3]. Para uma lista mais ampla, veja http://www.rspa.com/spi/project-risk.html.

Quando se trata da automatização do processo de software, uma questão importante é a integração de ferramentas. Ferramentas CASE geralmente atendem a uma, ou a poucas atividades, resolvendo problemas pontuais. Entretanto, na prática, esses problemas não estão dissociados, sobretudo quando se trata da gerência de projetos. Isso implica na utilização de várias ferramentas para que se possa atender às diversas atividades do processo. Assim, quanto mais ferramentas forem utilizadas, mais trabalho é necessário para integrar suas informações, já que resultados produzidos por uma ferramenta devem ser passíveis de processamento por outras. Assim, ao menos algum tipo de integração de dados é necessário [9].

A busca pela integração levou ao desenvolvimento de Ambientes de Desenvolvimento de Software (ADSs). ADSs podem ser descritos como coleções de ferramentas integradas que facilitam as atividades da engenharia de software, durante

[1] http://www.iceincusa.com/Products.aspx?p=Products_RiskRadar
[2] http://eng-sun3.murdoch.edu.au/~geoff/ProRisk/ProRiskBrowser.html
[3] http://www.intaver.com/riskyprojectprof.html

todo o ciclo de vida do software ou pelo menos em porções significativas dele [10]. Em escala bem menor que ferramentas CASE de apoio à gerência de riscos, há ADSs que possuem ferramentas dedicadas a essa atividade, tais como os ambientes Odyssey [11] e TABA [12].

No que tange ao compartilhamento de experiências, a gerência de conhecimento (GC) tem sido bastante utilizada no contexto da Engenharia de Software em geral [13] e na Gerência de Riscos mais especificamente [2]. A gerência de riscos é uma atividade complexa e de intenso conhecimento e, portanto, gerenciar o conhecimento organizacional nela envolvido é importante. Apoiados por sistemas de GC, gerentes de projeto podem utilizar conhecimento organizacional sobre riscos, juntamente com suas próprias experiências, para realizar as atividades desse complexo processo [2].

Mas para ser efetivo, um sistema de GC deve estar integrado ao processo de trabalho [14]. No caso do processo de software, quando apoiado de forma automatizada por um ADS, é natural integrar o sistema de GC a esse ADS [15]. Essa preocupação, por exemplo, tem sido foco do ADS TABA, incluindo apoio de GC ao processo da gerência de riscos [12].

Por fim, deve-se considerar que o processo de software é uma atividade essencialmente cooperativa, na qual diferentes profissionais atuam em conjunto para desenvolver um produto de software de qualidade. Equipes tendem a ser mais produtivas quando há cooperação e troca de idéias. Esse é precisamente o caso da gerência de projetos. Por exemplo, Pressman [6], Pfleeger [1] e Jørgesen [16] apontam a combinação de estimativas de diferentes especialistas utilizando diferentes abordagens como uma boa prática para a geração de estimativas confiáveis. Mas isso não se restringe às estimativas. Planos de riscos podem ser melhor desenvolvidos se diversos especialistas trabalharem em conjunto, apontando riscos, debatendo seus graus de exposição e definindo ações, até chegarem a uma proposta consensual.

Ao analisar esses três diferentes potenciais mecanismos de apoio à gerência de riscos – automatização do processo, gerência de conhecimento e apoio ao trabalho cooperativo – identificamos que há uma forte sinergia entre eles e, portanto, maiores benefícios podem advir se eles forem combinados. Este é o foco deste trabalho: explorar a sinergia entre automatização do processo, apoio ao trabalho cooperativo e gerência de conhecimento no apoio à gerência de riscos. Seu desenvolvimento se deu no contexto do ambiente ODE, apresentado a seguir.

3 O Ambiente ODE

O ambiente ODE (*Ontology-based Software Development Environment*) [3] é um Ambiente de Desenvolvimento de Software (ADS) centrado em processo e baseado em ontologias. A premissa do Projeto ODE é a seguinte: se as ferramentas do ADS são construídas baseadas em ontologias, a integração das mesmas é facilitada, pois os conceitos envolvidos estão bem definidos e compartilhados.

Tal ambiente vem sendo desenvolvido no Laboratório de Engenharia de Software da Universidade Federal do Espírito Santo (LabES-UFES) desde 1999 e a partir do final de 2004 foi implantado em caráter experimental em uma organização de

desenvolvimento de software, atualmente CMMI nível 2, visando obter um feedback de sua utilidade em situações reais.

Formando a base ontológica de ODE, há diversas ontologias, dentre elas, ontologias de Processo de Software, de Riscos de Software, de Qualidade de Software e de Organizações de Software. Essas ontologias fornecem uma conceituação sólida para a construção de diversas ferramentas do ambiente, incluindo a sua infra-estrutura de Gerência de Conhecimento [17]. Em especial, merece destaque no contexto deste trabalho a ontologia de requisitos de software [2], usada como base para a construção da ferramenta GeRis [2], de apoio à Gerência de Riscos.

Ontologias são usadas em ODE para derivar as classes que implementam o ambiente ou algumas de suas ferramentas. Essas classes são conceitualmente organizadas em três níveis, como mostra a Figura 1 [18]. O nível ontológico (pacote *Ontologia*) é responsável pela descrição das ontologias no ambiente e suas classes correspondem à meta-ontologia assumida no ambiente. O nível de conhecimento (pacote *Conhecimento*) abriga as classes que descrevem o conhecimento em relação a um domínio de aplicação. Suas classes são derivadas das ontologias, sendo que seus objetos atuam no papel de conhecimento sobre os objetos do nível base. Por fim, o nível base (pacote *Controle*) define as classes que implementam o ambiente e suas ferramentas. As ontologias exercem um papel fundamental em ODE, na medida em que são responsáveis por organizar e estabelecer as conexões entre itens distribuídos pelos três níveis [18].

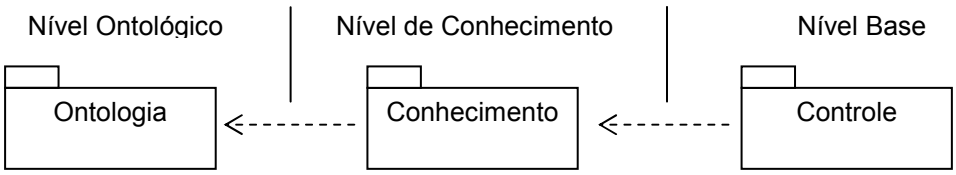

Fig. 1. Arquitetura Conceitual de ODE.

O ambiente conta atualmente com diversas ferramentas, dentre elas ferramentas de apoio à definição de processos de software, de apoio à engenharia de requisitos, de apoio à realização de estimativas e de apoio à gerência de riscos. Além disso, o ambiente possui uma infra-estrutura de gerência de conhecimento [17] e uma infra-estrutura de caracterização de itens de software [19] que é utilizada pela primeira para apoiar a busca de itens de conhecimento úteis, com base em similaridade de projetos.

3.1 A Infra-estrutura de Gerência de Conhecimento de ODE

A infra-estrutura de Gerência de Conhecimento (GC) de ODE [17] é composta por uma memória organizacional, estruturada em repositórios de conhecimento, e cinco tipos básicos de serviços para permitir a captura e criação, recuperação e acesso, disseminação pró-ativa, uso e manutenção de itens de conhecimento.

Como mostra a Figura 2, os repositórios de conhecimento da memória organizacional de ODE contêm itens de conhecimento. Esses itens podem ser formais ou informais.

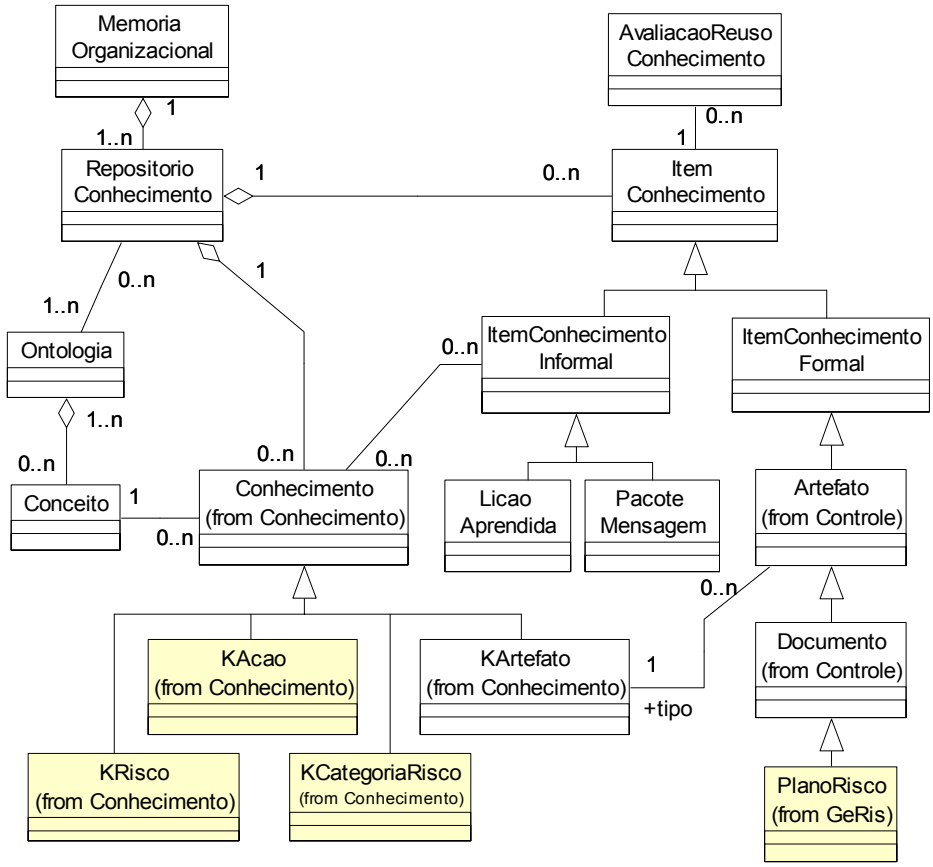

Fig. 2. Diagrama de classes parcial da Infra-estrutura de Gerência de Conhecimento de ODE.

Os itens formais são os diversos artefatos produzidos pelas ferramentas do ambiente, dentre eles os planos de riscos. Já os itens informais incluem lições aprendidas e pacotes de mensagens, sendo estes últimos formados por mensagens trocadas nas ferramentas de comunicação de ODE (correio eletrônico e fórum), que são empacotadas usando um serviço de empacotamento de mensagens [17]. Todos os itens são indexados por objetos do nível de Conhecimento, sub-classes da classe *Conhecimento* (sempre precedidas pelo prefixo *K*) que descrevem o conhecimento organizacional sobre um domínio descrito por uma ontologia de ODE. Por exemplo, a ontologia de riscos de ODE possui os conceitos *categoria de risco*, *risco* e *ação*, que foram mapeados, respectivamente, nas classes *KCategoriaRisco*, *KRisco* e *KAcao*, cujos objetos, por sua vez, descrevem o conhecimento organizacional sobre categorias de risco consideradas pela organização e potenciais riscos e ações de contingência e mitigação. Já artefatos são anotados com o tipo específico do artefato (*KArtefato*). Por exemplo, planos de risco dos diversos projetos são sempre associados ao mesmo objeto de conhecimento que representa o tipo de artefato "Plano de Risco". Assim, as ontologias são utilizadas para estruturar a memória organizacional de ODE, usando

um mecanismo de anotação conceitual. Essas anotações são usadas posteriormente por serviços de busca e disseminação.

No que se refere aos serviços, cinco tipos básicos são providos pela infra-estrutura:

- captura e criação: artefatos produzidos pelas ferramentas do ambiente são disponibilizados como itens formais de conhecimento. Assim, planos de risco produzidos na ferramenta GeRis, quando finalizados, passam a ser tratados como itens de conhecimento. Para tratar lições aprendidas, há um serviço específico que permite seu registro, aprovação e publicação [15]. Por fim, conforme citado anteriormente, há um serviço de empacotamento de mensagens trocadas por membros da organização usando as ferramentas de comunicação de ODE [17];
- recuperação e acesso: é o serviço geral de busca de itens de conhecimento, com iniciativa por parte dos usuários do ambiente. Esse serviço se apóia fortemente no esquema de anotação ontológica descrito anteriormente;
- disseminação pró-ativa: serviço análogo ao anterior, mas iniciado pelo sistema, fornecendo itens relevantes para uma determinada atividade que está sendo executada. Esse serviço é provido por agentes especializados, sendo que ele não é geral, precisando ser tratado pelas ferramentas que o implementam. Na ferramenta de gerência de riscos, há um sistema multi-agente que provê esse serviço [20];
- feedback da relevância: durante o uso de um item de conhecimento, o desenvolvedor pode registrar uma avaliação da relevância de um item para o seu trabalho (classe *AvaliacaoReusoConhecimento* na Figura 2) [15]; e
- manutenção: com base no feedback provido pelos desenvolvedores, o serviço de manutenção permite a exclusão de itens da memória organizacional [15].

3.2 GeRis: A Ferramenta de Apoio à Gerência de Riscos de ODE

GeRis é a ferramenta de apoio à gerência de riscos de ODE. Conforme brevemente comentado anteriormente, o ambiente e algumas de suas ferramentas são estruturados e construídos refletindo as conceituações definidas pelas ontologias de ODE. Dado que ODE é desenvolvido usando a tecnologia de objetos, é realizado um mapeamento de ontologias para modelos de objetos, aplicando-se parcialmente a abordagem sistemática de derivação de infra-estruturas de objetos a partir de ontologias, descrita em [21]. Nessa abordagem conceitos, relações, propriedades e restrições definidas como axiomas de uma ontologia são mapeados para classes, associações, atributos e métodos de um modelo de objetos que é, então, usado como modelo base para uma parte do ambiente ou ferramenta. Além disso, segue-se o esquema de anotação conceitual descrito anteriormente, no qual as classes derivadas das ontologias são anotadas com os conceitos da ontologia.

GeRis segue essa abordagem, tomando por base uma ontologia de riscos [2]. Os conceitos e relações da ontologia são descritos como instâncias das classes do pacote *Ontologia*. Para a derivação das classes dos outros dois níveis, levaram-se em conta quais conceitos tratam de conhecimento organizacional (derivados como classes do pacote *Conhecimento*) e quais são elementos específicos de projetos (derivados como

classes do nível base) [2]. A Figura 3 mostra um modelo de classes simplificado de GeRis, com atualizações recentes feitas em relação a [2].

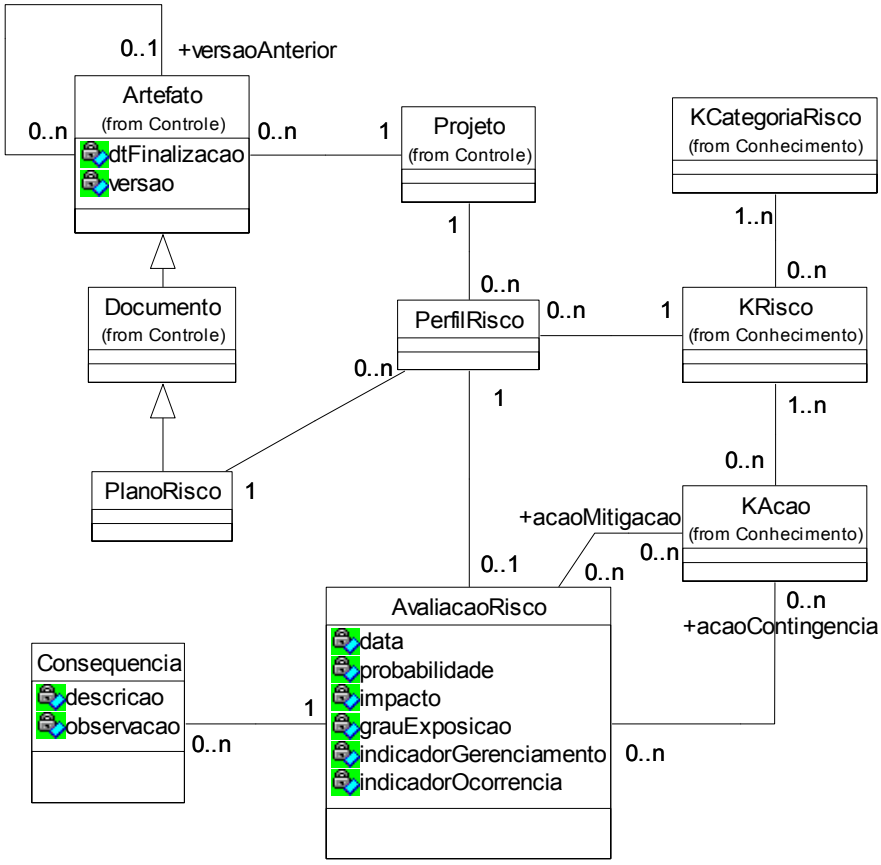

Fig. 3. Diagrama de classes parcial da versão atual de GeRis.

Os objetos das classes de conhecimento (*KCategoriaRisco*, *KRisco* e *KAcao*) são criados por meio de uma funcionalidade do ambiente para cadastrar conhecimento, disponibilizada apenas para usuários com papel de Gerente de Conhecimento. Os objetos das demais classes são criados pelo uso da ferramenta GeRis.

Desde sua primeira versão, GeRis oferece as seguintes funcionalidades [2]: (i) na identificação de riscos, o conhecimento organizacional sobre potenciais riscos para um projeto (objetos da classe *KRisco*) é apresentado para que o gerente de projeto selecione quais riscos compõem o perfil de risco do projeto em questão; (ii) uma vez identificados os riscos, podem-se avaliar a probabilidade e o impacto de cada um deles, definindo o grau de exposição de cada risco no projeto; (iii) os riscos são ordenados, então, pelo grau de exposição e apresentados para que o gerente de projeto selecione quais serão gerenciados; (v) finalmente, para os riscos a serem gerenciados, são definidas ações de contingência e mitigação.

Os serviços gerais da gerência de conhecimento de ODE podem ser usados, por exemplo, para buscar e reutilizar um plano de risco de um projeto similar [2]. Além disso, foi desenvolvido um sistema multi-agente para disseminação pró-ativa de potenciais riscos, graus de exposição e ações [20].

A primeira versão de GeRis foi disponibilizada juntamente com o ambiente para uso em caráter experimental por uma organização de software no final de 2004. Desse uso, várias melhorias foram sugeridas, dentre elas a necessidade de aperfeiçoamento na usabilidade da ferramenta e a possibilidade de se ter várias versões concorrentes de um mesmo plano de risco.

No que se refere à melhoria da usabilidade, ainda que as funcionalidades acima citadas claramente sigam uma ordem, dada pelo processo da Gerência de Riscos, a ferramenta, por usar um sistema de menus convencional, não mostrava explicitamente essa ordem, dificultando o seu uso. Para tratar esse problema, que não é específico da ferramenta de gerência de riscos, mas de qualquer ferramenta que tenha de automatizar um processo bem definido com atividades tendo de ser realizadas segundo uma determinada ordem, um novo componente foi desenvolvido. Esse componente permite dar clareza ao processo, provendo as funcionalidades na ordem definida por esse, como ilustra a Figura 4.

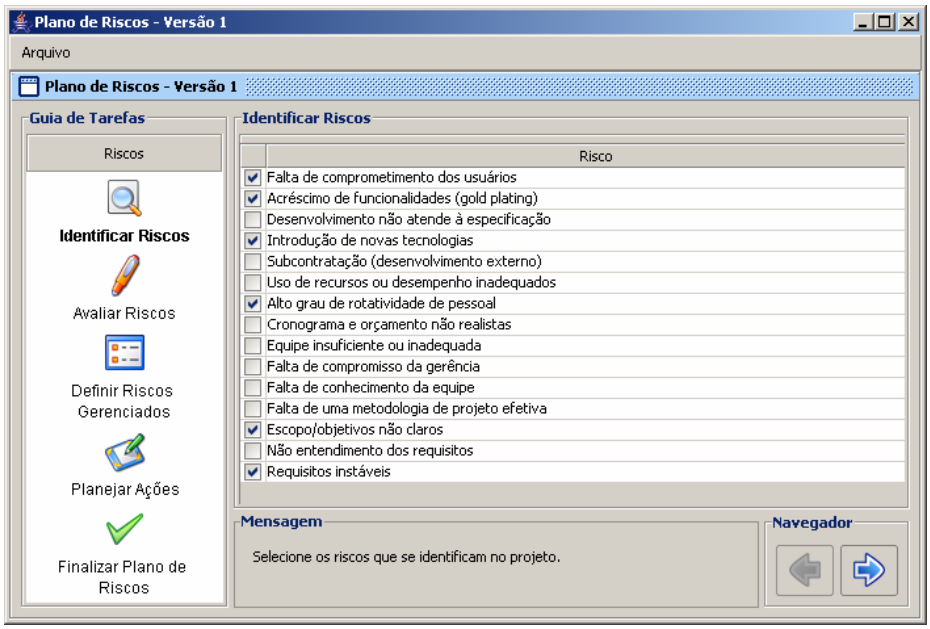

Fig. 4. Identificação de Riscos na Nova Versão de GeRis.

Em relação à oportunidade de melhoria que apontava a possibilidade de se ter várias versões diferentes do mesmo plano de risco, possivelmente considerando aspectos diferentes, conduziu ao desenvolvimento de funcionalidades de apoio ao trabalho cooperativo em GeRis e de captura e disseminação do conhecimento envolvido nesta tarefa, discutidas a seguir.

4 Apoio ao Trabalho Cooperativo em GeRis

A abordagem para permitir a elaboração cooperativa de planos de risco foi desenvolvida a partir de uma adaptação da técnica Delphi [1], usualmente aplicada na realização de estimativas. Segundo essa técnica, por meio de uma reunião com várias rodadas, mediada por um moderador, as estimativas de diversos especialistas são combinadas visando à geração de uma estimativa consensual. Cada participante é livre para utilizar a abordagem que lhe for mais conveniente para a elaboração de sua proposta e a cada rodada da reunião, os especialistas têm a oportunidade de rever suas estimativas com base na troca de idéias entre os participantes, com argumentações e justificativas.

A adaptação proposta para apoiar a elaboração cooperativa de planos de riscos prevê a realização de uma reunião para construção cooperativa desse artefato, em um fluxo de trabalho iterativo composto de sete etapas, a saber:

1. Identificação da necessidade de realizar a Gerência de Riscos de forma cooperativa: o gerente de projeto identifica que a gerência de riscos deve ser realizada de forma cooperativa e propõe uma reunião para a realização da mesma.
2. Planejamento Inicial da Reunião: o gerente de projeto define o moderador, os participantes e o tipo da reunião, e prepara materiais e recomendações que julga serem importantes para os participantes.
3. Planejamento de Rodada: o moderador estabelece as datas de início e término de uma rodada e notifica os participantes da mesma.
4. Elaboração de Propostas: os participantes elaboram suas propostas de planos de riscos, devendo explicar os motivos que os levaram a tais propostas, e as submetem para o moderador.
5. Elaboração de Proposta de Consenso: o moderador, de posse das propostas dos participantes, procura elaborar uma proposta consensual que contemple as principais considerações de cada participante. Essa proposta é submetida para apreciação de todos. Caso considere as propostas dos participantes divergentes a ponto de não permitirem a elaboração de uma proposta de consenso, o moderador pode apresentar novas informações e recomendações aos participantes e iniciar uma nova rodada de discussões, retomando o processo a partir do passo 3.
6. Discussão sobre a Proposta de Consenso: por meio de um fórum, os participantes emitem opiniões sobre a proposta de consenso, justificando seus pontos de vista.
7. Avaliação da Proposta de Consenso: tomando por base as opiniões dos participantes, o moderador avalia se a proposta de consenso pode ser aprovada, se precisa de pequenas alterações mas pode ser aprovada, ou se uma nova rodada de discussões é necessária, retomando o processo a partir do passo 3. Caso acredite ser inviável chegar a um consenso entre os participantes, o moderador pode, ainda, finalizar a reunião sem que uma proposta de consenso seja aprovada.

Uma reunião pode ser de dois tipos: (i) aberta, na qual, a cada rodada, os participantes têm acesso, de forma anônima, às propostas apresentadas por todos os outros participantes na rodada anterior e a um relatório sumário dessas propostas, ou

(ii) fechada, na qual cada participante só tem acesso à sua própria proposta da rodada anterior, ao relatório sumário das propostas e à proposta consensual apresentadas na rodada anterior.

Tomando por base essa abordagem para apoiar a elaboração cooperativa de planos de riscos, novas funcionalidades foram desenvolvidas, algumas delas diretamente no ambiente ODE (por potencialmente se aplicarem a outras situações) e outras estendendo a ferramenta GeRis. A Figura 5 mostra um diagrama de casos de uso provendo uma visão geral dessas funcionalidades. As funcionalidades Agendar Reunião e Controlar Reunião são providas no ambiente, enquanto a elaboração de planos de risco é feita usando GeRis.

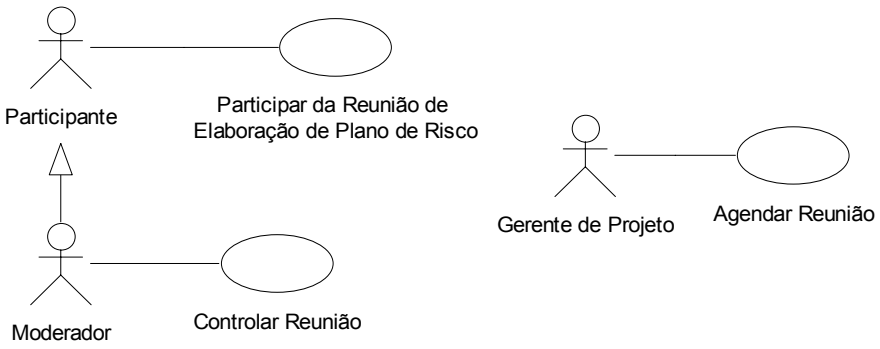

Fig. 5. Visão geral das funcionalidades de apoio à Gerência de Riscos Cooperativa.

O caso de uso "Agendar Reunião" permite que o gerente do projeto agende uma reunião para a elaboração cooperativa de planos de riscos, definindo, dentre outras coisas, quem será o moderador da mesma (que pode, inclusive, ser ele próprio). Um fórum para discussão é criado e, a partir de então, o moderador controla a reunião.

O caso de uso "Controlar Reunião" envolve diversas funcionalidades, todas relacionadas ao controle de rodadas da reunião, permitindo, dentre outros, iniciar uma nova rodada, encerrar uma rodada e encerrar a reunião. A funcionalidade relativa a iniciar uma nova rodada é sempre usada no início do processo ou quando o moderador julgar que o consenso não foi atingido e que uma nova rodada deve ser realizada para avançar na realização da atividade. No caso do encerramento da reunião, três são as possibilidades: (i) aceitar a proposta de consenso sem alterações, elegendo-a como proposta consensual da reunião; (ii) com base nas opiniões dos participantes, o moderador pode optar por efetuar pequenas alterações na proposta de consenso e definir a proposta alterada como sendo o resultado final da reunião; ou (iii) encerrar a reunião sem um resultado, dado que não foi possível atingir uma proposta satisfatória.

Durante as rodadas de uma reunião, os participantes devem gerar suas propostas e, ao final da rodada, o moderador pode elaborar uma proposta de consenso. Para que os participantes e o moderador possam elaborar propostas bem fundamentadas, é necessário que tenham acesso ao maior número possível de informações sobre o projeto em questão. Assim, o caso de uso "Participar de Reunião de Elaboração de Plano de Riscos" envolve diversas funcionalidades, dentre elas:

- Consultar Recomendações: permite que os participantes tenham acesso às recomendações feitas pelo gerente do projeto e pelo moderador da reunião de modo a apoiá-los na elaboração de suas propostas.
- Consultar Caracterização do Projeto: em ODE, projetos podem ter associadas diversas características, tais como domínio da aplicação, paradigma, tipo de equipe etc. Essas informações são muito importantes para a gerência de riscos e, portanto, têm de estar acessíveis aos participantes da reunião.
- Editar Proposta: permite que os participantes elaborem suas propostas. Corresponde, de fato, ao uso da ferramenta GeRis com todas as suas funcionalidades. Vale destacar que uma proposta anteriormente elaborada pode ser como ser usada como ponto de partida.
- Analisar Impacto das Características do Projeto sobre a Proposta: permite que o participante indique o nível de impacto (um valor de 0 a 10) que cada característica do projeto exerceu sobre sua proposta, justificando.
- Submeter Proposta: permite que os participantes submetam suas propostas ao moderador e que o moderador submeta uma proposta de consenso para avaliação dos participantes.
- Consultar Proposta de Consenso: permite que o moderador e os participantes consultem a proposta de consenso corrente.
- Consultar Propostas Anteriores: permite que o moderador e os participantes consultem propostas submetidas em rodadas anteriores.
- Participar de Discussões: permite que o moderador e os participantes leiam as mensagens já postadas no fórum da reunião, postem novas questões nesse fórum e respondam a questões postadas no mesmo.

A elaboração cooperativa de planos de risco propicia aprendizado, tendo em vista que os participantes argumentam, justificam e discutem as propostas, buscando chegar a um consenso. Deixar esse conhecimento em nível de indivíduo em um ambiente que possui uma infra-estrutura de gerência de conhecimento (GC) seria um contra-senso. Assim, com o objetivo de levar esse aprendizado para o nível organizacional, um novo item de conhecimento informal foi adicionado à infra-estrutura de GC de ODE: as histórias de reunião.

Conforme apontado por Buckingham [22], o processo de software tem sido altamente orientado a artefatos, ou seja, focado em gerar e manter os artefatos durante o ciclo de vida, culminando no sistema final. Apesar desse processo atingir seu objetivo, as razões por trás das decisões que levaram aos artefatos permanecem implícitas e são, conseqüentemente, difíceis de recuperar e reutilizar. Assim, é importante, também, capturar e reusar o conhecimento envolvido na elaboração desses artefatos. Com a abordagem utilizada para a elaboração de planos de riscos de forma cooperativa isso é possível, já que pelo menos parte do raciocínio seguido pelos participantes (incluindo o moderador) foi capturada.

Assim, as histórias de reunião apresentam, em um relatório gerado dinamicamente, informações acerca de uma reunião, que são divididas em duas partes: dados gerais e dados das rodadas. Dentre os dados gerais, constam objetivo da reunião, número e papéis dos participantes, número de rodadas e de propostas geradas e resultado final. De cada rodada, são apresentadas as seguintes informações: propostas, justificativas, características consideradas, mensagens trocadas no fórum, resultado da rodada e dados sumários, tais como riscos mais apontados e probabilidades e impactos médios.

O formato do relatório das histórias de reunião foi elaborado com base no formato definido no trabalho de Torres [23], que aplica Histórias de Aprendizagem na captura e disseminação do conhecimento adquirido em projetos de Tecnologia de Informação.

5 Considerações Finais

Este trabalho apresentou a evolução da ferramenta de Gerência de Riscos de ODE (GeRis) para apoiar a elaboração cooperativa de planos de riscos e a captura do conhecimento envolvido.

Conforme citado na seção 2, há diversas ferramentas e ambientes de desenvolvimento de software que apóiam essa atividade, alguns inclusive com apoio de gerência de conhecimento, como é o caso da Estação TABA. Contudo, não encontramos em nenhuma das ferramentas pesquisadas funcionalidades como as propostas neste trabalho para a elaboração cooperativa de planos de riscos, com a captura do conhecimento envolvido nas decisões tomadas.

Como continuação deste trabalho, está-se estendendo a abordagem proposta para outras ferramentas de ODE, tal como a ferramenta de apoio a estimativas [19]. Além disso, está prevista a implantação de uma nova versão do ambiente ODE, contendo a nova versão de GeRis, nas organizações parceiras em abril de 2007, a partir de quando se pretende avaliar na prática a eficácia da abordagem proposta.

Agradecimentos

Este trabalho foi realizado com o apoio do CNPq e da CAPES, entidades do Governo Brasileiro dedicadas ao desenvolvimento científico e tecnológico, da FAPES, Fundação de Apoio à Ciência e Tecnologia do Espírito Santo, e das empresas VixTeam e Projeta, parceiras que têm financiado o projeto e dado feedback de sua aplicação a casos reais.

Referências

1. Pfleeger, S.L. (2001) *Software Engineering: Theory and Practice*, 2nd Edition, New Jersey: Prentice Hall.
2. Falbo, R.A., Ruy, F.B., Bertollo, G., Togneri, D.F. (2004) "Learning How to Manage Risks Using Organizational Knowledge". Proceedings of the 6th International Workshop on Advances in Learning Software Organizations, LSO'2004, pp. 7-18, Banff, Canada.
3. Falbo, R.A.; Natali, A.C.C.; Mian, P.G.; Bertollo, G.; Ruy, F.B. (2003) "ODE: Ontology-based software Development Environment", IX Congreso Argentino de Ciencias de la Computación, p. 1124-1135, La Plata, Argentina.
4. IEEE Std 1540-2001, "IEEE Standard for Software Life Cycle Processes – Risk Management".
5. Sommerville, I. (2003) *Engenharia de Software*, São Paulo: Addison-Wesley, 6ª edição.
6. Pressman, R. S. (2002) *Engenharia de Software*, McGraw-Hill, 5ª edição.

7. Chrissis, M.B., Konrad, M., Shrum, S. (2003) *CMMI: Guidelines for Process Integration and Product Improvement*, Addison Wesley.
8. Subcomitê Setorial da Qualidade e Produtividade em Software, Programa Brasileiro da Qualidade e Produtividade (2002) Qualidade e Produtividade no Setor de Software Brasileiro – Pesquisa 2001.
9. Gruhn, V. (2002) "Process-Centered Software Engineering Environments: A Brief History and Future Challenges", In: Annals of Software Engineering 14, p. 363-382.
10. Harrison, W., Ossher, H., Tarr, P. (2000) "Software Engineering Tools and Environments: A Roadmap". Proceedings of The Future of Software Engineering (ICSE'2000). Limerick, Ireland, pp. 263 – 277.
11. Barros, M.O., Werner, C.M.L., Travassos G.H. (1999) "Risk Analysis: a key success factor for complex system development". Proceedings of the 12th International Conference Software & Systems Engineering and their Applications, Paris, France.
12. Farias, L.L., Travassos, G.H., Rocha, A.R.C. (2003) "Managing Organizational Risk Knowledge", Journal of Universal Computer Science, vol. 9, no. 7.
13. Rus, I.; Lindvall, M. (2002) "Knowledge Management in Software Engineering". IEEE Software 19(3) May/Jun. pp 26-38.
14. Fischer, G., Ostwald, J. (2001) "Knowledge Management: Problems, Promises, and Challenges". *IEEE Intelligent Systems,* v. 16, n. 1, pp. 60-72.
15. Natali, A.C.C., Falbo, R.A. (2002) "Knowledge Management in Software Engineering Environments", Proc. of the 16th Brazilian Symposium on Software Engineering, Gramado, Brazil.
16. Jørgesen, M. (2004), "A Review of Studies on Expert Estimation of Software Development Effort", Journal of Systems and Software, vol. 70, issue 1-2, pp. 37-60.
17. Falbo, R.A., Arantes, D.O., Natali, A.C.C. (2004) "Integrating Knowledge Management and Groupware in a Software Development Environment". Proceedings of the 5th International Conference on Practical Aspects of Knowledge Management, Karagiannis, D., Reimer, U. (Eds.): LNAI 3336, Springer-Verlag Berlin Heidelberg, Austria, pp. 94-105.
18. Ruy, F.B., Falbo, R.A. (2006) "Tratamento Semântico de Conhecimento Organizacional em um Ambiente de Desenvolvimento de Software", First Workshop on Ontologies and Metamodels for Software and Data Engineering, Florianópolis, Brazil.
19. Carvalho, V.A., Arantes, L.O., Falbo, R.A. (2006) "EstimaODE: Apoio a Estimativas de Tamanho e Esforço no Ambiente de Desenvolvimento de Software ODE", Anais do V Simpósio Brasileiro de Qualidade de Software (SBQS´2006), Vila Velha, Brasil.
20. Falbo, R.A., Pezzin, J., Schwambach, M.M. (2005) "A Multi-Agent System for Knowledge Delivery in a Software Engineering Environment", 17th International Conference on Software Engineering and Knowledge Engineering, Taipei, China, pp. 253 – 258.
21. Falbo, R.A., Guizzardi, G., Duarte, K.C. (2002) "An Ontological Approach to Domain Engineering". Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering, SEKE'2002, Ischia, Italy, p. 351- 358.
22. Buckingham, S.S. (1996), "Design Argumentation as Design Rationale", The Encyclopedia of Computer Science and Technology (Marcel Dekker Inc: NY), Vol. 35 Supp. 20, 95-128.
23. Torres, A.H.S. (2006) Captura e Disseminação do Conhecimento em Projetos de Software, Dissertação de Mestrado, Universidade Católica de Brasília, Brasília - DF, Brasil.