

O Gerenciamento de Requisitos no Ambiente COCAR

André Di Thommazo¹

Marcos Danilo Martins

Sandra Fabbri

UFSCar – Universidade Federal de São Carlos

{andredt, marcosdanilo}@gmail.com

sfabbri@dc.ufscar.br

Abstract

Background. This research group has been working towards building an environment which helps software development, providing support to some activities mainly based on use case model. The reading technique TUCCA and a template to specify requirements were previously defined in another work of this group and are the basis of this environment. **Aim.** The aim of this paper is to give an overview of COCAR environment, emphasizing the requirement management activities supported by it. **Method.** Based on literature and several proposals of other authors, we introduced in the environment some functionalities that support requirement management and provide traceability between the requirements and the use case model. **Results.** The environment was used in a simple example of a real company and the results motivated the improvement of the environment and the continuity of this research. **Conclusions.** The information provided by the requirement functionalities give relevant support to maintain the requirement exchange controls, consistence and the development planning, supporting and facilitating some activities related to requirement management. **Keywords.** requirements management, automated environment, use case, traceability.

1. Introdução

O uso de software no cotidiano das pessoas vem aumentando a cada dia e pode-se dizer que já faz parte de praticamente qualquer atividade humana. Por outro lado, a indústria de software enfrenta desafios de gerar produtos que atinjam às

expectativas dos clientes, obedecendo prazos, custos e critérios de qualidade.

Pesquisas do instituto Standish Group [1] mostram que no ano de 2004, a quantidade de projetos que foi finalizada com sucesso, ou seja, respeitando os prazos estabelecidos, o orçamento e, sobretudo, atendendo às expectativas dos clientes em todas as funcionalidades e características do software, é de apenas 29%.

Outra pesquisa realizada pelo mesmo instituto buscou determinar quais fatores eram os mais importantes para definir o sucesso ou não de um projeto de software. O resultado foi que, os três primeiros fatores que provocaram insucesso aos projetos somando, aproximadamente, 37% eram: falta de especificação do usuário, requisitos incompletos e mudança nos requisitos. Tais fatores estão diretamente relacionados com o gerenciamento de requisitos. Ainda em 2006 essa situação continua, de acordo com Salem [2], que afirma que a maioria dos erros dos software é fruto de erros na fase de levantamento de requisitos e durante o acompanhamento de sua evolução ao longo do processo de desenvolvimento.

Dada a importância da fase de Engenharia de Requisitos, este grupo de pesquisa tem explorado e desenvolvido alguns trabalhos com o intuito de prover contribuições para a área. Assim, considerando que no documento de requisitos são especificados os requisitos solicitados pelo cliente e que, para modelar esses requisitos uma das técnicas mais utilizadas é a de casos de uso, em um dos trabalhos do grupo foi definida a técnica TUCCA (*Technique for Use Case Construction and Construction-Based Requirements Analysis*). Essa técnica tem por objetivo sistematizar a construção do Modelo de Casos de Uso sendo que, simultaneamente, é feita uma inspeção do documento

¹ Professor do Grupo Anhanguera Educacional

de requisitos [3][4][5][6][7]. Além disso, foi definido também um *template* para descrição de requisitos que considera informações relevantes que devem ser registradas quando do levantamento dos mesmos [8]. A intenção ao realizarem-se esses trabalhos era desenvolver um ambiente que desse suporte a várias atividades do ciclo de desenvolvimento de software, baseadas nos requisitos e no modelo de casos de uso. Exemplos dessas atividades são o gerenciamento de requisitos, o planejamento do software e a geração de casos de teste. Assim, deu-se início ao desenvolvimento do ambiente **COCAR**, sendo que neste artigo o objetivo é abordar as atividades de gerenciamento de requisitos que estão disponíveis nesse ambiente.

O artigo está organizado da seguinte forma: na Seção 2 abordam-se as atividades de gerenciamento de requisitos situando-se essa etapa no contexto da engenharia de requisitos. O ambiente **COCAR** é apresentado na Seção 3, focando-se especialmente as atividades de gerenciamento contempladas no ambiente. Na Seção 4 exemplifica-se o uso do ambiente **COCAR** em relação às atividades de gerenciamento de requisitos. Uma comparação entre o ambiente **COCAR** e outras ferramentas é feita na Seção 5. As conclusões e os trabalhos futuros são mostrados na Seção 6.

2. Gerenciamento de Requisitos

De acordo com Sommerville [12], a Engenharia de Requisitos é dividida em quatro fases: i) Levantamento dos Requisitos – na qual se busca determinar “o quê” o sistema deve fazer; ii) Especificação dos Requisitos – que compreende a documentação dos requisitos levantados na fase anterior, tendo como produto do trabalho o Documento de Requisitos; iii) Validação dos Requisitos – que busca eliminar as possíveis inconsistências e falhas do Documento de Requisitos, visando à geração de um documento descrito da forma apropriada e completa, buscando eliminar problemas de ambigüidade e inconsistência; e iv) Gerenciamento de Requisitos – que se estende durante todo o processo de desenvolvimento de software.

A atividade de gerenciamento de requisitos é extremamente dispendiosa e trabalhosa, tornando necessário o uso de ferramentas que auxiliem a equipe responsável pelo gerenciamento e desenvolvimento do software em todo esse processo. De acordo com o Standish Group [9], apenas 5% dos software são desenvolvidos com uso de ferramentas

de gerenciamento de requisitos, o que pode explicar, em parte, os grandes problemas das companhias em constituir uma gerência efetiva de requisitos e manter a rastreabilidade. Nesse sentido, Marcus *et al.* [10], Egyed e Grübacher [11], Sommerville [12] e Alexander [13] enfatizam a necessidade de suporte de ferramentas para a viabilização do gerenciamento de requisitos e, sobretudo, da rastreabilidade dos requisitos do software. Zisman e Spanoudakis [14] acreditam que esse seja o único caminho para o sucesso nessa tarefa. Munson e Nguyen [15] afirmam que as práticas de rastreabilidade irão melhorar apenas quando forem apoiadas por ferramentas que reduzam o esforço requerido para mantê-las.

A rastreabilidade de requisitos refere-se à habilidade de descrever e acompanhar a vida de um requisito [16]. Esse controle deve abranger toda a sua existência, desde a fonte de origem - quando o requisito foi elicitado, especificado e validado - passando pela fase de projeto, implementação e terminando na fase de testes do produto, sendo portanto, parte do processo de desenvolvimento do software. Trata-se de uma técnica que permite a visualização do relacionamento de dependência entre os requisitos levantados e entre os requisitos e os demais artefatos gerados ao longo do desenvolvimento do software.

A rastreabilidade dos requisitos é um fator crítico de sucesso para a agilidade no processo de tratamento das mudanças nos requisitos, as quais certamente acontecem durante todo o desenvolvimento do software. Considerando o processo de Engenharia de Requisitos, a rastreabilidade faz parte do Gerenciamento de Requisitos que, segundo Zisman e Spanoudakis [14], embora as definições dessa etapa sejam divergentes, ela tem como principais objetivos: 1) organizar e armazenar os requisitos e 2) gerenciar mudanças dos requisitos.

Sempre que os requisitos identificados inicialmente com os clientes forem alterados, os planos de software, os artefatos e as atividades afetadas devem sofrer ajustes para continuarem consistentes. O fato é que os requisitos são ativos e estão em uso durante todo o ciclo de vida. Simplesmente congelar os requisitos após a etapa de validação é utopia [17], uma vez que as regras de negócio e a dinâmica das organizações não são estáveis.

3. O ambiente COCAR e seu suporte ao Gerenciamento de Requisitos

Conforme comentado anteriormente, o ambiente COCAR visa abranger diversas etapas do processo de desenvolvimento de software, dando suporte à realização de algumas atividades, tendo como ponto em comum o modelo de casos de uso. Resumidamente, inserem-se os requisitos do sistema para o qual se desejam realizar as atividades do processo de desenvolvimento e, com base neles, o ambiente permite construir o modelo de casos de uso, os pontos de casos de uso, controles de gerenciamento de requisitos, casos de teste baseados em casos de uso, etc. Na versão atual estão contempladas as seguintes funcionalidades:

- i) Inserção dos requisitos de um sistema: nessa funcionalidade, o ambiente dá suporte à inserção dos requisitos obedecendo o *template* definido no trabalho de Kawai [8], o qual engloba identificação, descrição, dados de entrada, saídas, processamento, restrições e atores, além dos atributos relacionados com a rastreabilidade propostos por Wieggers [18], que são o gerente e o solicitante do requisito;
- ii) Construção do modelo de casos de uso: nessa funcionalidade, o ambiente dá suporte à elaboração do Modelo de Casos de Uso a partir dos requisitos inseridos previamente. Essa construção é feita por meio da aplicação da técnica TUCCA¹ – *Technique for Use Case Construction and Construction-based Requirements Analysis*, proposta por Belgamo [3]. Essa técnica contribui para a construção de Modelos de Casos de Uso mais padronizados, de tal forma que a experiência e a subjetividade do projetista não tenham tanta interferência nessa construção. Além disso, ao construir o modelo, faz-se, simultaneamente, uma inspeção do documento de requisitos [4][5][6][19];
- iii) Geração Automática de Pontos de Casos de Uso: nessa funcionalidade, o ambiente dá suporte à geração dos Pontos de Casos de Uso com base nas especificações dos Casos de Uso elaboradas com a aplicação da TUCCA. Seguindo o *template* proposto em [3], o qual é usado para descrever as especificações, as informações necessárias para a geração dessa métrica são organizadas de maneira a facilitar a sua geração. O valor dos Pontos de Casos de Uso também pode ser convertido em Pontos por Função; e

¹ Essa técnica era referenciada anteriormente por GUCCRA - *Guidelines for Use Case Construction and Requirements Analysis*

iv) Suporte ao Gerenciamento de Requisitos: nessa funcionalidade, o ambiente dá suporte ao acompanhamento dos requisitos ao longo do processo de desenvolvimento, tendo como base principal aspectos de rastreabilidade.

Nas subseções a seguir detalham-se as funcionalidades específicas do gerenciamento de requisitos que estão contempladas no ambiente.

3.1 – Geração da Matriz de Rastreabilidade de Requisitos

A matriz de rastreabilidade indica o relacionamento existente entre os requisitos do software. Todos os requisitos são listados na primeira linha e na primeira coluna. A intersecção linha/coluna representa a existência ou não de relacionamento entre eles. Diversos autores comentam a importância e a necessidade desse tipo de relacionamento para o processo de desenvolvimento de software [20], [12], [2], [21], [22]. Essa matriz permite a previsão do impacto de uma mudança ou inserção de um requisito no sistema. Sommerville [12] enfatiza a dificuldade de se obter e manter esse tipo de matriz e, além disso, propõe que a indicação da dependência entre os requisitos determine, além da existência ou não de relação entre eles, uma forma de registrar, subjetivamente, se essa relação é forte ou fraca.

O ambiente COCAR gera a matriz de rastreabilidade de requisitos de forma automática, assim como sua manutenção. Essa relação é dada por meio do percentual de dependência que existe entre os requisitos, o qual corresponde ao percentual de dados de entrada coincidentes que os requisitos possuem. Isso é possível porque a forma de armazenamento dos requisitos no ambiente COCAR separa, atômica, os dados de entrada de cada requisito, conforme descrito no trabalho de Kawai [8]. A ferramenta oferece suporte à criação dos dados de entrada dos requisitos, evitando inconsistências, duplicidades, etc.

3.2 – Criação do Indicador de Estabilidade

O Indicador de Estabilidade, proposto por Hazan e Leite [21], permite a obtenção de métricas durante o processo de gerenciamento de requisitos. A necessidade de medidas durante esse processo também é enfatizada e defendida pelo SWEBOK [23] e pelos trabalhos do Serpro [20]. Com o apoio das métricas é possível encontrar estratégias mais eficientes para a gerência de requisitos, identificando,

por exemplo, as maiores fontes de mudanças, relacionando o número de requisitos alterados, inseridos ou excluídos com a ocorrência de erros ou atrasos no desenvolvimento do software. No ambiente, além do Indicador de Estabilidade [21], esses números são mantidos em um histórico, o que torna possível a determinação dos seguintes indicadores propostos:

- Percentual de novos requisitos no período, obtido pela divisão do número de requisitos novos pelo número de requisitos que já haviam sido elicitados
- Percentual de requisitos alterados no período, obtido pela divisão do número de requisitos alterados pelo número de requisitos que já haviam sido elicitados.
- Percentual de requisitos excluídos no período, obtido pela divisão do número de requisitos excluídos pelo número de requisitos que já haviam sido elicitados.

Para que a análise seja feita, o usuário deve determinar o período desejado. Isso permite ainda identificar quais as fases do processo de desenvolvimento de software em que houve mais solicitações de exclusão, alteração ou inserção de requisitos no sistema.

3.3 – Rastreabilidade entre o Documento de Requisitos e o Modelo de Casos de Uso

A rastreabilidade entre os artefatos criados ao longo do processo de desenvolvimento de software é fonte de estudo de diversos autores. Alexander [13] apresentou uma proposta para a rastreabilidade entre os Requisitos e o Modelo de Casos de Uso, baseada na ferramenta comercial DOORS. Marcus *et al.*[10] exploraram a visualização dos links de rastreabilidade entre artefatos, definindo inclusive atributos para os mesmos, no intuito de agregar conhecimento nessa ligação. Munson e Nguyen [15] mencionaram a importância da rastreabilidade abranger um número cada vez maior de artefatos, sendo que esse processo deveria ser apoiado por ferramentas que, segundo os autores, reduziriam o esforço requerido para manter a rastreabilidade. Kelleher [24] propôs um framework para representar a rastreabilidade, englobando também a dependência entre os artefatos. Wiegers [25] destacou que rastrear requisitos com outros componentes do sistema iria ajudar a garantir que os artefatos estivessem consistentes ao longo do processo de desenvolvimento de software. Outros pesquisadores abordaram o mesmo tema e também consideraram

muito importante a existência dessa informação [14] [26].

No ambiente *COCAR* foi possível prover a rastreabilidade entre os requisitos funcionais e o Modelo de Casos de Uso de forma automática por dois motivos:

- As especificações dos requisitos funcionais são armazenadas individualmente, de acordo com as diretrizes propostas por Kawai [8], em um repositório de requisitos, o que permite que eles sejam tratados separadamente.
- O Modelo de Casos de Uso é obtido a partir da especificação dos Requisitos, por meio da aplicação da técnica TUCCA [3] e pelos próprios passos da técnica, não é necessário que o usuário indique, manualmente, qual Caso de Uso está relacionado com qual Requisito Funcional.

Por essas razões, a determinação da rastreabilidade entre os Requisitos Funcionais e os Casos de Uso é obtida de forma automática, de acordo com o que propõem Zisman e Spanoudakis [14]. Assim, o risco dessa rastreabilidade ser obtida de maneira incorreta é muito baixo, uma vez que ele não é gerado com base na análise textual, conforme criticado por Alexander [13]. No caso da TUCCA, os próprios passos da técnica encarregam-se de garantir a consistência dessa dependência entre os requisitos funcionais e os Casos de Uso gerados [3]. Além disso, o uso de técnicas formais, como a TUCCA, é defendido por Salem [2] como um fator que favorece a determinação correta da rastreabilidade.

É interessante notar que, quando a rastreabilidade é apresentada no ambiente *COCAR*, tanto para os Requisitos Funcionais como para os Casos de Uso, é possível o acesso aos detalhes de ambos por meio de *hyperlinks*. O uso de *hyperlinks* e hiperdocumentos para as ferramentas de rastreabilidade de requisitos é sugerido por Zisman e Spanoudakis [14] e Munson & Nguyen [15].

3.4 – Consulta sobre os requisitos

Uma vez que os requisitos são armazenados no repositório, é possível a realização de consultas aos mesmos. Assim, dois atributos sugeridos no trabalho de Wiegers [25], que fornecem importantes informações para a rastreabilidade, foram incorporados às diretrizes de Kawai [8]:

- Solicitante do Requisito: refere-se ao *stakeholder* que levantou esse requisito como necessidade para o sistema. Caso alguma alteração seja proposta nesse requisito ao longo do desenvolvimento, essa pessoa deve ser

questionada sobre a mudança. Esse tipo de rastreabilidade, acompanhando o requisito desde sua origem, é denominado pré-rastreamento [14] [27].

- Gerente do Requisito: refere-se à pessoa responsável por acompanhar o ciclo de vida de um requisito ao longo do processo de desenvolvimento de software. Dessa forma, é possível a realização de pesquisa para verificar quais são os requisitos sob responsabilidade de determinado gerente.

4. Exemplos de atividades de Gerenciamento de Requisitos no ambiente COCAR

Nesta seção são apresentados alguns exemplos de atividades relacionadas ao gerenciamento de requisitos disponíveis no ambiente *COCAR*. Os exemplos foram coletados da aplicação do ambiente em uma empresa do seguimento de CTI (*Computer Telephone Integration*). O sistema desenvolvido com o uso do ambiente *COCAR* é o UserPrePaid. Trata-se de um sistema simples e com poucas funcionalidades, mas capaz de ilustrar o funcionamento do ambiente *COCAR*, abrangendo todas as funcionalidades referidas anteriormente. O objetivo desse sistema é dar suporte aos usuários de cartões pré-pagos para realização da compra de créditos e verificação do histórico de atividades, exigindo para isso a autenticação do usuário.

4.1 – Exemplo de Geração da Matriz de Rastreabilidade de Requisitos

No ambiente *COCAR* buscou-se uma solução para determinação automática dessa matriz, indicando ainda qual o percentual de dependência existente entre os requisitos.

Para a determinação da dependência entre os requisitos, o ambiente faz a comparação entre os dados de entrada, ou seja, os dados que são manipulados e processados por cada um dos requisitos. Caso dois requisitos manipulem os mesmos dados de entrada, a dependência entre eles tende a ser alta. Se eles estiverem manipulando parcialmente os mesmos dados, a dependência entre esses requisitos deve ser menor. Caso dois requisitos de um mesmo sistema não possuam nenhum dado de manipulação em comum, não se indica nenhuma

dependência entre eles na matriz. Na Figura 1 segue um exemplo da matriz de rastreabilidade de requisitos, considerando o sistema UserPrePaid.

Outro aspecto observado na visualização da Matriz de Rastreabilidade de Requisitos é o uso de *hyperlinks* na representação dos requisitos, o que, de acordo com Zisman e Spanoudakis [14] e Munson e Nguyen [15], favorece a rastreabilidade, à medida que permitem o acesso rápido às informações envolvidas e rastreadas. Sendo assim, o usuário pode clicar sobre os requisitos apresentados na matriz e serão exibidos os detalhes desse requisito.

4.2 – Exemplo da Criação do Indicador de Estabilidade

O indicador de estabilidade tem como objetivo oferecer uma medida sobre o processo de gerenciamento de requisitos. O usuário da ferramenta pode, a qualquer momento do processo de desenvolvimento, consultar o percentual de requisitos inseridos, alterados ou excluídos dentro do período de tempo que ele determinar. Para isso, basta informar a data inicial e final do período sobre o qual se deseja realizar a pesquisa dos indicadores.

Isso é possível, pois cada inserção ou alteração do requisito tem a data de ocorrência registrada. Com relação à exclusão de requisitos, ela é feita logicamente, mantendo-se o requisito armazenado no banco de dados, porém registrado como excluído, salvando-se a data em que essa operação foi feita.

Na Figura 2 é apresentado o Indicador de Estabilidade. O período é determinado pelo usuário e, posteriormente, os indicadores são disponibilizados. Como exemplo, considere os cinco requisitos apresentados na Figura 3. Como dois desses cinco requisitos foram alterados durante o desenvolvimento do produto, o percentual de requisitos alterados é de 40%. Caso mais um desses cinco requisitos sofra alterações, esse percentual passará a ser 60%. Com relação aos requisitos inseridos no sistema, durante o período pesquisado nenhum requisito foi acrescentado, uma vez que o percentual de requisitos inseridos é 0%. No que diz respeito aos requisitos excluídos, é possível saber que, anteriormente, existiam seis requisitos e um foi excluído, pois o percentual de requisitos excluídos indica que 20% de cinco requisitos, ou seja, um requisito foi excluído durante o período pesquisado.

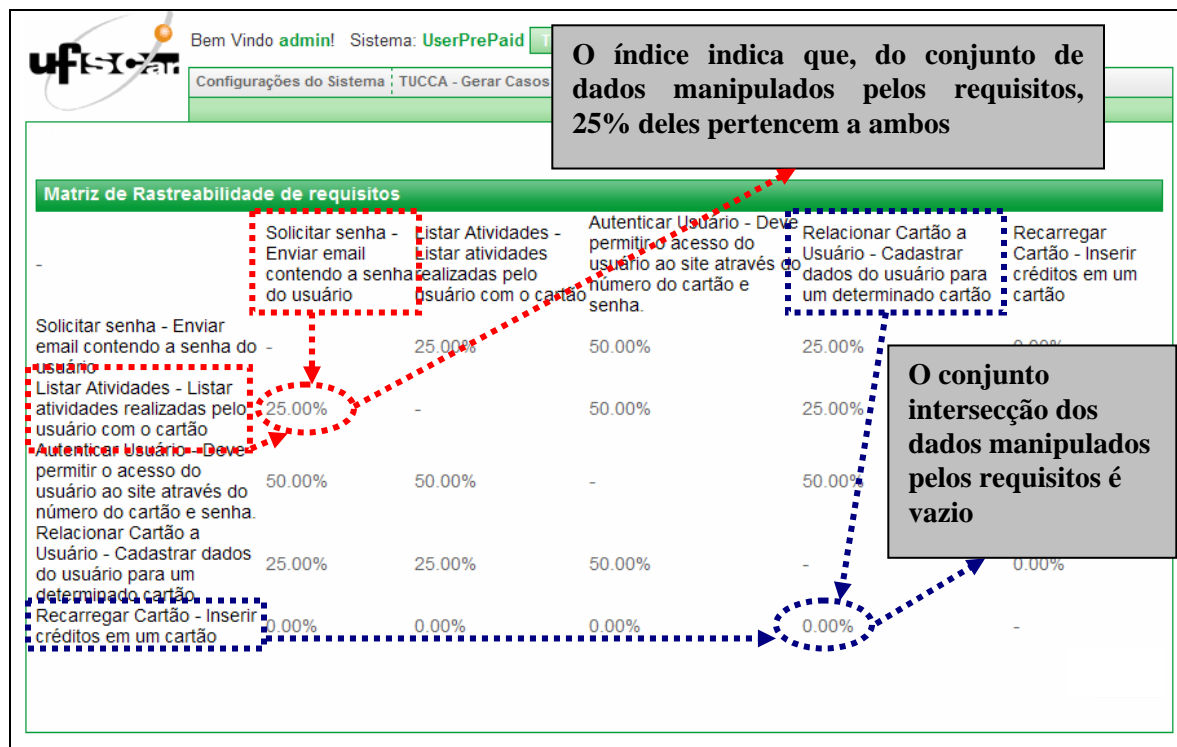


Figura 1 – Exemplo de Matriz de Rastreabilidade

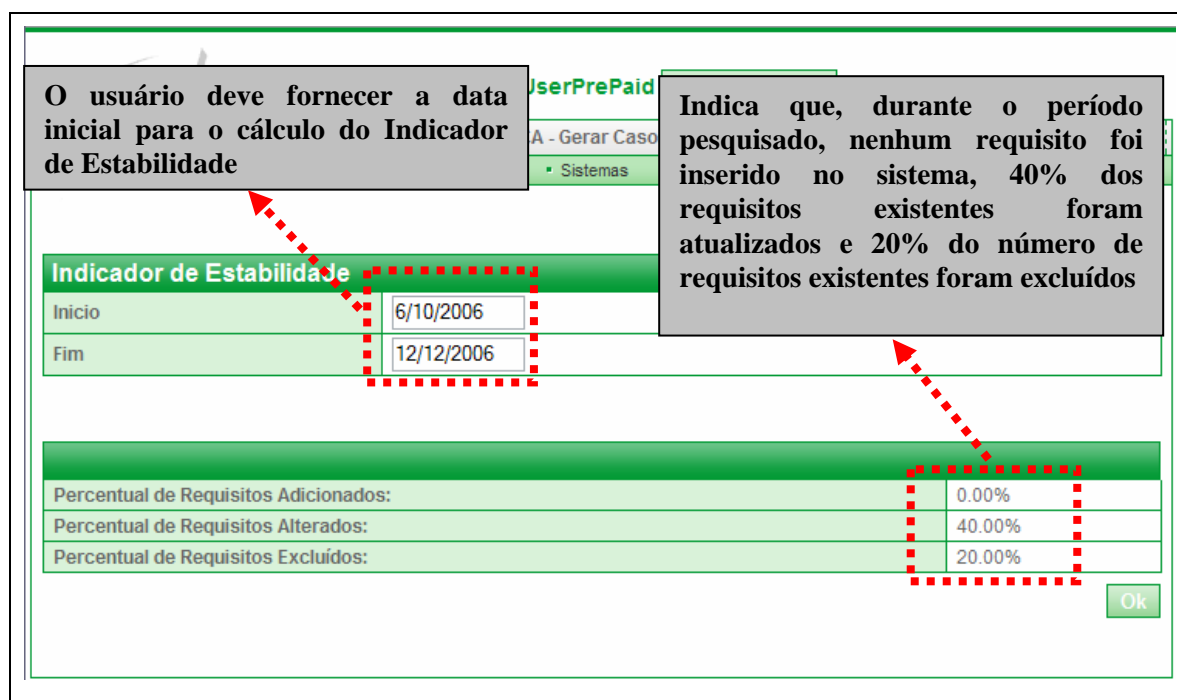


Figura 2 - Exemplo do Indicador de Estabilidade

4.3 – Exemplo de Rastreabilidade entre o Documento de Requisitos e o Modelo de Casos de Uso

A ligação entre os requisitos e os Casos de Uso é obtida em decorrência da aplicação da técnica TUCCA, de forma automática, conforme detalhado anteriormente. A visualização dessa dependência é exemplificada na Figura 3.

Para o acompanhamento da evolução dos requisitos e dos Casos de Uso, é feito o controle baseado no rótulo de tempo das ligações e das próprias versões dos requisitos e Casos de Uso, conforme proposto por Munson e Nguyen [15]. Sendo assim, se a versão do requisito que se relaciona com o Caso de Uso for mais recente que o Caso de Uso, isso indica que o mesmo deve ser revisto, pois, como o requisito que o originou foi

alterado, é possível que o Caso de Uso também necessite de atualização. Nesses casos, a dependência fica registrada no sistema com a cor vermelha para alertar sobre a necessidade de revisão. Na Figura 3, por exemplo, os Casos de Uso “Relacionar Cartão a Usuário” e “Listar Atividades” - que surgiram da aplicação da TUCCA – ficam marcados em vermelho para indicar que devem ser revisados para possível atualização, pois o requisito “Relacionar Cartão a Usuário - Cadastrar dados do usuário para um determinado cartão”, passou por uma alteração e seu rótulo de tempo foi atualizado e é mais recente que o dos Casos de Uso.

Vale notar que, assim como é feito com os requisitos, os Casos de Uso são disponibilizados também com *hyperlinks*, sendo possível a exibição dos detalhes associados a eles.

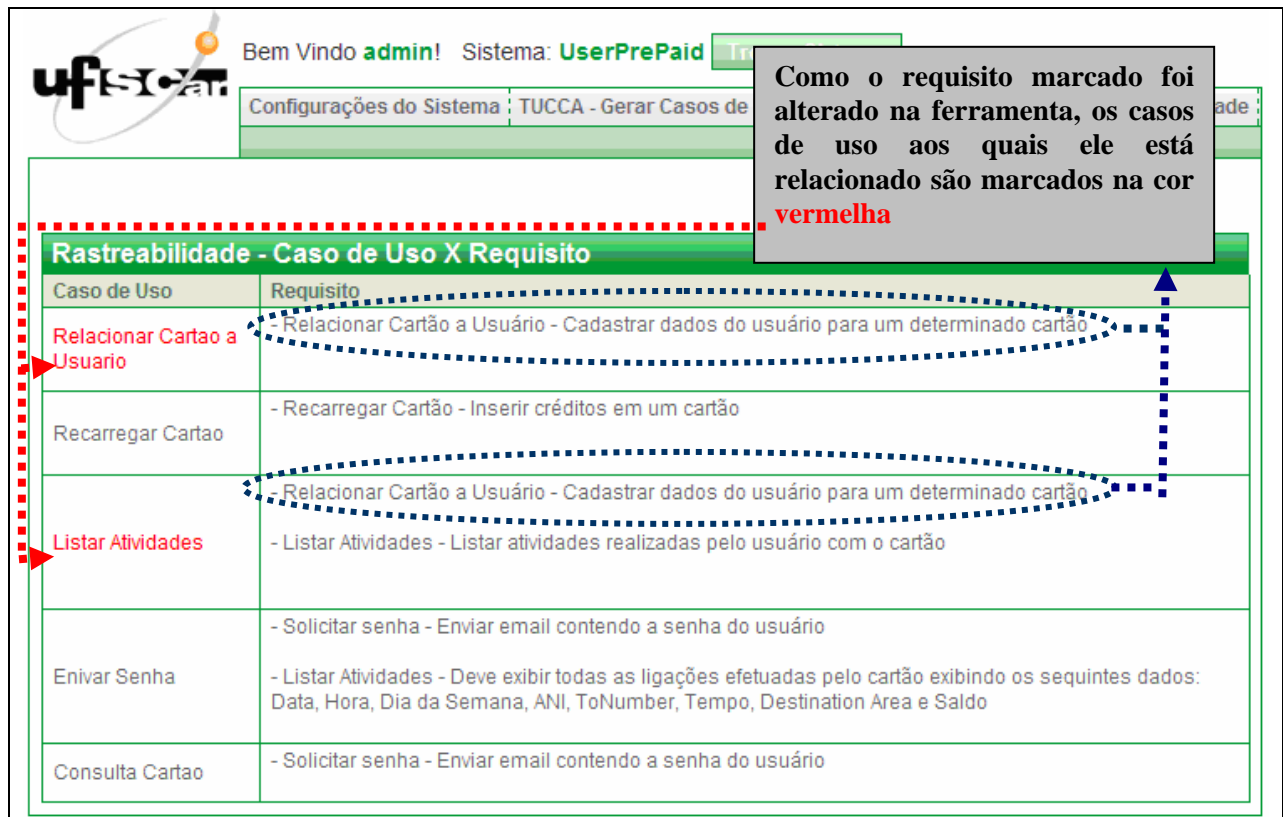


Figura 3 – Exemplo de indicação de alerta sobre alterações nos requisitos

4.4 – Exemplo de Consulta sobre os Requisitos

Essa funcionalidade do ambiente *COCAR* refere-se à possibilidade de efetuar consultas sobre os requisitos, com o intuito de levantarem-se os requisitos que são de responsabilidade de determinado gerente ou que foram solicitados por determinado *stakeholder*. A Figura 4 ilustra as possibilidades de pesquisa que podem ser feitas, com os seguintes filtros de sistema: gerente de requisito e solicitante de requisito.

Nesta seção foram apresentados exemplos práticos do uso do ambiente *COCAR*, possibilitando a visualização de suas funcionalidades específicas para o contexto do gerenciamento de requisitos.

Como pôde ser observado pelos exemplos dados, as informações relativas ao gerenciamento de requisitos são, certamente, de grande valia para o gerenciamento do projeto, uma vez que com elas o gerente pode, por exemplo, ter subsídios para estimar custo e tempo associados ao desenvolvimento do produto, por meio do impacto das mudanças, favorecendo a mudança racional nos requisitos.

O usuário pode realizar pesquisa compondo filtros por meio da seleção dos sistemas, gerentes ou solicitantes cadastrados

Os requisitos que atendem às restrições anteriormente são exibidos

| | Descrição | Data | Gerente |
|--------------------------|---|------------|-----------------|
| <input type="checkbox"/> | Autenticar Usuário - Deve permitir o acesso do usuário ao site através do número... | 2006-12-12 | Inezita Barroso |
| <input type="checkbox"/> | Solicitar senha - Enviar email contendo a senha do usuário | 2006-12-12 | Inezita Barroso |
| <input type="checkbox"/> | Recarregar Cartão - Inserir créditos em um cartão | 2006-12-12 | Zeca Baleiro |
| <input type="checkbox"/> | Listar Atividades - Listar atividades realizadas pelo usuário com o cartão | 2006-12-12 | Inezita Barroso |
| <input type="checkbox"/> | Relacionar Cartão a Usuário - Cadastrar dados do usuário para um determinado ca... | 2006-12-12 | Gilberto Gil |

Figura 4 – Possibilidades de consultas sobre os requisitos

5. Comparação com outras ferramentas

Uma vez que foram apresentadas as características e as funcionalidades do ambiente *COCAR* para o gerenciamento de requisitos, nesta seção apresenta-se uma comparação do mesmo com algumas ferramentas comerciais de gerenciamento de requisitos. Conforme já mencionado, apesar de apenas 5% dos software serem desenvolvidos com uso de ferramentas de gerenciamento de requisitos a não-adoção de uma ferramenta pelas empresas não

está relacionada à ausência de opções. A Incose [29] apresenta um conjunto de mais de 30 ferramentas disponíveis para a gerência de requisitos. Dentre as opções apresentadas, destaca-se a CaliberRM [30] da Borland. Essa ferramenta possui grande vantagem por estar integrada com os compiladores da fabricante. Dessa forma, a rastreabilidade entre o código-fonte e os requisitos pode ser estabelecida diretamente na ferramenta. O mesmo vale para a RequisitePro [31], da IBM. Apoiada no conjunto de soluções para o processo de desenvolvimento de software, o gerenciamento de requisitos pode ser

integrado com a modelagem (Rational Rose), gerenciamento de versão (ClearCase) e casos de teste (TestManager). Outra ferramenta de destaque é a DOORS [32], da Telelogic. A solução possui boa aceitação no mercado, contando com uma comunidade com mais de cem mil usuários e oferecendo bom suporte ao usuário.

Em geral, todas as ferramentas de gerenciamento de requisitos oferecem as funcionalidades básicas também apresentadas pelo ambiente *COCAR*, tais como o cadastro e a manutenção de um conjunto de requisitos, além da realização de consultas sobre os requisitos cadastrados. Com relação à determinação da rastreabilidade entre os requisitos de um mesmo sistema, as ferramentas disponíveis, em geral, apresentam uma forma para mapear essa rastreabilidade de forma manual, criando-se assim a Matriz de Rastreabilidade de Requisitos. O ambiente *COCAR* determina essa dependência entre os requisitos de forma automática, baseando-se no conjunto de dados manipulados pelos requisitos. Diferentemente das outras ferramentas, apresenta o uso de indicadores como forma de avaliar o processo de engenharia de requisitos, contemplando o indicador de Estabilidade proposto na literatura.

Com relação à rastreabilidade entre o Documento de Requisitos e o Modelo de Casos de Uso, algumas ferramentas oferecem suporte a essa funcionalidade, como é o caso da CaliberRM. Entretanto, a rastreabilidade gerada pelo ambiente *COCAR* é feita de forma automática por meio do processo formal estabelecido pela técnica de leitura TUCCA, que padroniza a geração dos casos de uso. Além disso, com o término da implementação de outras funcionalidades, que já está em andamento, o ambiente dará suporte também, por exemplo, para manter a rastreabilidade entre requisitos, casos de uso e casos de teste.

6. Conclusões e Trabalhos Futuros

Este artigo apresentou com uma maior ênfase as atividades de gerenciamento de requisitos disponíveis no ambiente *COCAR*. Esse ambiente foi desenvolvido para dar suporte ao processo de desenvolvimento de software e, na sua versão atual contempla as seguintes funcionalidades: o registro dos requisitos de um sistema, segundo diretrizes estabelecidas no trabalho de Kawai [8]; a elaboração do Modelo de Casos de Uso, a partir desses requisitos, de acordo com a TUCCA [3], a geração de

Pontos de Casos de Uso [28] e o suporte ao gerenciamento de requisitos.

Com relação aos aspectos de gerenciamento de requisitos, as funcionalidades foram exemplificadas com o uso do próprio ambiente *COCAR*. A primeira funcionalidade específica que tratou o gerenciamento de requisitos foi a Geração da Matriz de Rastreabilidade de Requisitos, estabelecida de forma automática, baseada na análise dos dados de entrada que os requisitos manipulam. Além da determinação da dependência ou não entre os requisitos, definiu-se o percentual de dependência, também baseado no conjunto intersecção dos dados de entrada dos requisitos.

Outra funcionalidade implementada foi o Indicador de Estabilidade, proposto por Hazan e Leite [21], com o objetivo de fornecer dados para que os responsáveis pelo desenvolvimento tenham subsídios para as tomadas de decisão. O uso de indicadores integrado com a ferramenta visa proporcionar uma maneira prática de avaliar a evolução dos requisitos, sinalizando o percentual de requisitos alterados, excluídos ou inseridos no sistema em um determinado período.

A rastreabilidade entre o documento de requisitos e o Modelo de Casos de Uso também foi disponibilizado no ambiente. A determinação dos *links* de rastreabilidade é feito de forma automática, valendo-se do formalismo da TUCCA e das diretrizes propostas por Kawai [8]. Esse formalismo é apontado como necessário por Salem [2] para o processo de rastreabilidade. Com relação à determinação automática da rastreabilidade – tanto para a relação entre o documento de requisitos e o Modelo de Casos de Uso e para a Matriz de Rastreabilidade – Zisman e Spanoudakis [14] afirmam ser esse o melhor caminho para as ferramentas que tratam da rastreabilidade. Apesar de Alexander [13] defender o uso de métodos semi-automáticos, no caso do ambiente *COCAR*, como a associação entre um caso de uso e os requisitos que deram origem a ele é feita automaticamente, devido ao formalismo da TUCCA, o risco de ter-se o *link* gerado equivocadamente é minimizado.

Completando as funcionalidades do ambiente proposto, estão as consultas que podem ser realizadas sobre os requisitos, permitindo a pré-rastreabilidade. Para que essa funcionalidade pudesse ser inserida no sistema, as diretrizes propostas por Kawai [8] foram adaptadas, agregando-se os seguintes elementos: solicitante e gerente do requisito.

Todas essas funcionalidades referidas acima foram aplicadas em um ambiente real de

desenvolvimento de software com o uso do ambiente **COCAR**. Apesar de ter sido uma única aplicação real e do sistema utilizado não ser complexo, pôde-se obter uma idéia da aplicabilidade da ferramenta num ambiente corporativo. O *feedback* dos funcionários que utilizaram o ambiente motivam a continuidade deste trabalho e o seu aprimoramento.

As principais contribuições incorporadas no ambiente **COCAR** estão relacionadas com a detecção automática da dependência entre os requisitos de software, permitindo a criação da matriz de rastreabilidade de forma automática. Isso favorece a determinação do impacto que a mudança em um requisito pode gerar nos demais. Além disso, o ambiente **COCAR**, por implementar a técnica TUCCA que estabelece um processo formal para a determinação dos casos de uso a partir dos requisitos funcionais, provê a determinação da dependência entre requisitos e casos de uso de maneira automática. Como se trata de um ambiente de suporte a diversas etapas do processo de desenvolvimento de software, a geração do Indicador de Estabilidade incorpora ao processo de desenvolvimento, uma informação relevante que pode favorecer os gerentes na adoção de medidas corretivas durante a construção do software.

Como trabalhos futuros, pretende-se realizar experimentos para analisar os resultados gerados pelas atividades de gerenciamento implementadas no ambiente **COCAR** comparando-se tais resultados com resultados produzidos por especialistas de forma manual. Além disso, também é importante a utilização do ambiente na indústria, aplicando em sistemas mais complexos e de diferentes portes daqueles avaliados até o momento.

7. Referências

[1] Standish Group - The Chaos Report – 2004 -. URL: <http://www.stsc.hill.af.mil/crosstalk/2004/10/0410Jacobs.pdf>. Acesso em: 12/04/2005

[2] A.M. Salem, "Improving Software Quality through Requirements Traceability Models", The 4th ACS/IEEE International Conference on Computer Systems and Applications (AICCSA 2006), Dubai, Sharjah, UAE, 2006.

[3] A. Belgamo, "GUCCRA: Técnicas de Leitura para Construção de Modelos de Casos de Uso e Análise do Documento de Requisitos". 2004. 157 f. Tese (Mestrado em Ciências da Computação) - Universidade Federal de São Carlos, São Carlos, 2004.

[4] A. Belgamo e S.C.P.F. Fabbri, "Constructing Use Case Model by Using a Systematic Approach: Description of a Study", WER – Workshop em Engenharia de Requisitos, Tandil, Argentina, p. 251 – 262, 2004.

[5] A. Belgamo e S.C.P.F. Fabbri, "GUCCRA: Contribuindo para a Identificação de Defeitos em Documentos de Requisitos Durante a Construção de Modelos de Casos de Uso" WER – Workshop em Engenharia de Requisitos, Tandil, Argentina, p. 100 – 111, 2004.

[6] A. Belgamo e S.C.P.F. Fabbri, "GUCCRA: Técnica de Leitura para apoiar a Construção de Modelos de Caso de Uso e a Análise de Documentos de Requisitos" SBES – 19º Simpósio Brasileiro de Engenharia de Software, Uberlândia, MG, 2005.

[7] A. Belgamo, S.S.P.C. Fabbri e J.C. Maldonado, "Avaliando a Qualidade da Técnica GUCCRA com Técnica de Inspeção", WER – Workshop em Engenharia de Requisitos, Porto, Portugal, 2005.

[8] K.K. Kawai, "Diretrizes para elaboração de Documento de Requisitos com ênfase nos Requisitos Funcionais". 2005. 170 f. Tese (Mestrado em Ciências da Computação) - Universidade Federal de São Carlos, São Carlos, 2005.

[9] Standish Group 2004 - Third Quarter Research Report - . URL: http://www.standishgroup.com/sample_research/PDFpages/q3-spotlight.pdf. Acesso em: 12/04/2005

[10] A. Marcus, X. Xie e D. Poshyanyk, "When and how to visualize traceability links?" Proceedings of the 3rd international workshop on Traceability in emerging forms of software engineering- Automated Software Engineering, Long Beach, USA, 2005

[11] A. Egyed e P. Grünbacher, "Automating Requirements Traceability: Beyond the Record & Replay Paradigm" 17th IEEE International Conference on Automated Software Engineering (ASE'02), Edinburgh, UK , 2002.

[12] I. Sommerville, Engenharia de Software. 6a. edição. – São Paulo - Addison Wesley, 2003, 580 p.

[13] I. Alexander "SemiAutomatic Tracing of Requirement Versions to Use Cases Experiences & Challenges" Second International Workshop on Traceability, Montreal, 2003. URL: <http://www soi.city.ac.uk/~gespan/paper6.pdf>. Acesso em 10/07/2006.

[14] A. Zisman e G. Spanoudakis, "Software Traceability: Past, Present, and Future", The Newsletter of the Requirements Engineering Specialist Group of the British Computer Society – URL : <http://www.resg.org.uk/archive/rq33.pdf>. Acesso em: 05/03/2005

- [15] E. V. Munson, e T. N. Nguyen, "Concordance, conformance, versions, and traceability"; Proceedings of the 3rd international workshop on Traceability in emerging forms of software engineering, Long Beach, California, 2005.
- [16] T. Hammer, L. Rosenberg; L. Huffman e L. Hyatt, "Requirement Metrics – Value Added", ISRE'97 Third International Symposium on Requirements Engineering. Ied. USA: IEEE Comput. Soc, Los Alamitos, CA. Proceedings, 1997.
- [17] A.C. Rocha, J.C. MALDONADO e K.C. WEBER, "Qualidade de Software – teoria e prática" Prentice Hall, 1a. edição, p.320, 2001.
- [18] WIEGERS, K. Software Requirements, Microsoft Press, 1999. 1a edição.
- [19] A. Belgamo, S.C.P.F. Fabbri e J.C. Maldonado, J. C. "TUCCA: Improving the Effectiveness of Use Case Construction and Requirement Analysis". Proc. of the 4th International Symposium on Empirical Software Engineering, Noosa Heads, Australia, 2005.
- [20] SERPRO. "Processo SERPRO de Desenvolvimento de Soluções (PSDS)", 2002 . URL : <http://www.serpro.gov.br/publicacao/tematec/2002/ttec60>
Acesso em: 04/02/2005
- [21] C. Hazan, J.C.S.P. Leite, "Indicadores para a Gerência de Requisitos", WER - Workshop em Engenharia de Requisitos; Piracicaba, Brasil, 2003.
- [22] B. Ramesh e M. Jarke, "Towards Reference Models For Requirements Traceability", IEEE Transactions on Software Eng., vol. 27, pp. 58-93, 2001
- [23] IEEE Computer Society Professional Practices Committee. Guide to the Software Engineering Body of Knowledge SWEBOK. Los Alamitos, California: BOURQUE, P., DUPUIS, R., 2004. 202 p.
- [24] J. Kelleher, "A reusable traceability framework using patterns" Proceedings of the 3rd international workshop on Traceability in emerging forms of software engineering- Automated Software Engineering. Long Beach, USA, 2005.
- [25] K. Wieggers, "Automating Requirements Management". URL: http://www.processimpact.com/articles/rm_tools.html.
Acesso em: 21/04/2005
- [26] I. Sommerville, "Requirements Engineering – A good practice guide", John Wiley, 1997.
- [27] P. Letelier, "A Framework for Requirements Traceability in UML-based Projects". 1st International Workshop on Traceability in Emerging Forms of Software Engineering, Edinburgh, U.K, p32-41, 2002
- [28] M.D.C. Martins, "Geração de Pontos de Casos de Uso no Ambiente Scorpion". 2007. 157 f. Tese (Mestrado em Ciências da Computação) - Universidade Federal de São Carlos, São Carlos, 2007.
- [29] INCOSE Requirements Management Tools Survey, Disponível em: <http://www.paper-review.com/tools/rms/read.php>. Acesso em: 12/03/2005
- [30] CaliberRM –Gerenciamento de Requisito para Engenharia de Software, Disponível em : <http://www.borland.com/br/products/caliber/index.html>
Acesso em: 28/01/2007
- [31] RequisitePro – IBM Rational RequisitePro http://www-306.ibm.com/software/info/ecatalog/pt_BR/products/U107428M40511T21.html
Acesso em: 28/01/2007
- [32] Doors – Requirements Management Traceability solutions <http://www.telelogic.com/products/doors/index.cfm>
Acesso em: 28/01/2007