UNIVERSIDADE NOVE DE JULHO

A MATRIZ DE RASTREABILIDADE COMO FERRAMENTA PARA O GERENCIAMENTO DE REQUISITOS

EDSON TOSHIO NAKAGAWA TOBIAS DA SILVA

SÃO PAULO
2011

EDSON TOSHIO NAKAGAWA TOBIAS DA SILVA

A MATRIZ DE RASTREABILIDADE COMO FERRAMENTA PARA O GERENCIAMENTO DE REQUISITOS

Monografia apresentada ao curso de Pós-Graduação "Lato Sensu" em Engenharia de Software da Universidade Nove de Julho, como requisito parcial à obtenção do título de Especialista em Engenharia de Software.

Orientador: Prof.(MSc). Ernani Marques da Silva

SÃO PAULO
2011

Silva, Edson Toshio Nakagawa Tobias da

????? A Matriz de Rastreabilidade como Ferramenta para o Gerenciamento de Requisitos [manuscrito] / Edson Toshio Nakagawa Tobias da Silva.

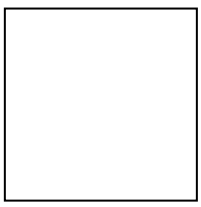
??f., enc.

Orientador: Ernani Marques da Silva

Monografia – Universidade Nove de Julho

Bibliografia: f. 113-117

1. Matriz de Rastreabilidade de Requisitos 2. Engenharia de Software I. TÍTULO II. Silva, Ernani Marques



EDSON TOSHIO NAKAGAWA TOBIAS DA SILVA

A MATRIZ DE RASTREABILIDADE COMO FERRAMENTA PARA O GERENCIAMENTO DE REQUISITOS

Monografia apresentada ao curso de Pós-Graduação "Lato Sensu" em Engenharia de Software da Universidade Nove de Julho, como requisito parcial à obtenção do título de Especialista em Engenharia de Software.

Aprovado em __/__/____

BANCA EXAMINADORA

_____

Prof. XXXXXXXXXX
Universidade Nove de Julho

_____

Prof. XXXXXXXXXX
Universidade Nove de Julho

_____

Prof. XXXXXXXXXX
Universidade Nove de Julho

Dedico este trabalho à minha esposa Marilene Batista Pereira da Silva e aos meus filhos Gabriel Toshio Nakagawa da Silva e Sarah Harumi Nakagawa da Silva, pelo amor e paciência a mim dedicados.

AGRADECIMENTOS

Primeiramente a Deus, que me deu força, saúde e direção para concluir este trabalho.

Ao Prof. Ernani Marques da Silva, pela atenção e apoio em toda a orientação desta monografia.

Aos meus familiares e amigos que sempre me apoiaram nos estudos, incentivando-me a avançar sempre, mais e mais.

À Universidade Nove de Julho – UNINOVE, por todo o suporte fornecido para a obtenção das informações necessárias à conclusão desta obra.

A todos aqueles que, direta ou indiretamente, colaboraram para a realização deste trabalho.

"*Há duas formas de construir um projeto de software: Uma maneira de fazer isso deve ser tão simples que, obviamente, não deixem deficiências, e a outra forma é a de torná-lo tão complicado que não percebam as evidentes deficiências. O primeiro método é muito mais difícil.*"

C.A.R. Hoare

RESUMO

O Gerenciamento de Requisitos é um processo que visa auxiliar os engenheiros de software na identificação, controle e rastreamento dos requisitos de software. Este trabalho trata da utilização da Matriz de Rastreabilidade e sua importância para a realização de um processo de gestão de requisitos bem efetuado e que venha a garantir a entrega de um produto que atenda às necessidades do cliente. Para tal, foi realizada uma revisão da literatura disponível sobre o tema Rastreabilidade de Requisitos e a aplicação da Matriz de Rastreabilidade nos projetos de desenvolvimento de software. Os resultados obtidos permitiram identificar os seus pontos fortes e fracos e os motivos que levam as organizações a não utilizarem este instrumento. Apresenta também as melhores práticas na construção das matrizes e as ferramentas que automatizam este processo, servindo como diretriz para a escolha do modelo a ser adotado para implantação na empresa que visa alcançar a melhoria na qualidade do software desenvolvido.

Palavras-chave: Matriz de Rastreabilidade de Requisitos; Requisitos; Engenharia de Software; Engenharia de Requisitos, Gerenciamento de Requisitos; Qualidade de Software.

ABSTRACT

The Requirements Management is a process designed to assist software engineers in identifying, tracking and tracing of software requirements. This work deals with the use of Traceability Matrix and its importance for the realization of a requirements management process that will ensure the delivery of a product that meets customer needs. To this end, we performed a review of available literature on the subject Requirements Traceability and the application of the Matrix Traceability in software development projects. The results allowed identifying their strengths and weaknesses and the reasons why organizations do not use this instrument. It also presents best practices in the construction of the matrices and tools that automate this process, serving as a guideline for choosing the model to be adopted for deployment in the enterprise that aims to achieve improvements in software quality.

Key words: Requirements Traceability Matrix; Requirements; Software Engineering; Requirements Engineering; Requirements Management; Software Quality.

LISTA DE GRÁFICOS E FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

CASE – *Computer-Aided Software Engineering*

CETIC – Centro de Estudo sobre as Tecnologias da Informação e Comunicação

CMMI - *Capability Maturity Model Integration*

ER – Engenharia de Requisitos

ERS – Especificação de Requisitos de Software

ES – Engenharia de Software

GCS – Gerência de Configuração de Software

GR – Gestão de Requisitos ou Gerenciamento de Requisitos

ID – Identificação

IIBA – *International Institute of Business Analysis*

MPS.BR – Melhoria de Processo do Software Brasileiro

MRR – Matriz de Rastreabilidade de Requisitos

PDL – *Program Description Language*

RR – Rastreabilidade de Requisitos

SADT – *Structured Analysis and Design Technique*

STSC – *Software Technology Support Center*

TIC – Tecnologia da Informação e Comunicação

TR – Tabela de Rastreamento

UML – Unifief Modeling Language

SUMÁRIO

1. INTRODUÇÃO

O mercado de Tecnologia da Informação e Comunicação (TIC) tem crescido a passos largos nos últimos anos, principalmente devido à queda dos preços para aquisição de computadores pessoais e dispositivos móveis, tais como telefones celulares e *notebooks*, facilitando o acesso das pessoas de baixa renda a estes equipamentos. De acordo com uma pesquisa realizada pelo Centro de Estudo sobre as Tecnologias da Informação e Comunicação (CETIC), o uso da Internet no Brasil registrou um crescimento de cerca de 35% entre 2008 e 2009, sendo que 66% dos computadores acessam a Rede Mundial de Computadores a partir da conexão de banda larga, representando um aumento de 32% ao ano, no período compreendido entre 2005 e 2009 (CETIC, 2010). Dessa forma, o país tem contribuído para o incremento das vendas on-line, além da massiva participação em redes sociais.

A realidade apresentada acima faz com que as empresas desenvolvedoras de software tenham que atender à demanda por novos programas e serviços, exigindo não somente inovação e criatividade, mas também o cumprimento correto dos requisitos e dos prazos estipulados em contrato – cada vez menores – a fim de atender aos clientes de forma satisfatória e dentro do orçamento previsto.

Conforme Pressman (2006), uma das tarefas mais difíceis para o engenheiro de software é entender os requisitos de um problema. Isto acontece porque na maioria das vezes o usuário não sabe exatamente o que quer e, quando o sabe, ainda corre o risco de sofrer modificações ao longo do projeto. Devido aos diversos problemas que estas mudanças causam, como aumento de riscos, atrasos no cronograma e diferenças no orçamento, é mister que se busque uma maneira de analisar e gerenciar os requisitos para atender a estes tipos de problemas, minimizando-os e, assim, satisfazer ao cliente, entregando-lhe um produto de qualidade. Entende-se que a qualidade de software está diretamente ligada ao atendimento dos requisitos funcionais e de desempenho explicitamente declarados para o desenvolvimento do software (PRESSMAN, 2006).

A Gestão de Requisitos (GR) é um processo importante da Engenharia de Requisitos (ER), que engloba um conjunto de atividades com o objetivo de identificar, rastrear e controlar os requisitos e suas mudanças. O planejamento desta atividade deve ser realizado ao mesmo tempo em que a elicitação dos requisitos é efetuada e, tão logo exista um esboço da versão do documento de requisitos, o seu gerenciamento deve ser imediatamente iniciado

(SOMMERVILLE, 2003).

Este trabalho trata da atividade de Rastreabilidade de Requisitos (RR), especificamente na utilização da Matriz de Rastreabilidade de Requisitos (MRR), que relaciona os requisitos aos *stakeholders* [1], os requisitos entre si ou mesmo aos módulos do projeto. Elas podem ser construídas usando uma variedade de ferramentas, como um banco de dados, planilhas ou até mesmo tabelas ou hiperlinks criados em um processador de texto, buscando garantir o cumprimento dos requisitos ao mesmo tempo em que controla as suas inevitáveis modificações, analisando o impacto e a viabilidade que poderão atingir o projeto e a implementação do sistema, bem como a consequente implantação destas mudanças.

1.1 Problema

Este estudo investiga o problema relacionado às constantes mudanças que ocorrem nos requisitos e o porquê da não utilização da MRR pelas organizações para lidar com este fato. A fase de requisitos é a menos compreendida dentre todas as que envolvem o desenvolvimento de software. Por isso, expressões tais como rastreabilidade de requisitos e matriz de rastreabilidade são mal-compreendidas, tornando-se comum que as modificações geradas no decorrer do projeto de software não sejam tratadas com a devida formalidade.

1.1.1 Objetivo Geral

O objetivo geral deste trabalho é apresentar a MRR e a sua importância para a GR e o controle das modificações que venham a acontecer.

[1] O *stakeholder* é toda pessoa, órgão ou instituição que possua algum interesse no software a ser desenvolvido ou que faça parte de seu processo de desenvolvimento.

1.1.2 Objetivos Específicos

Como objetivos específicos, destacam-se:

a) Descrever o processo de ER e suas atividades, dando ênfase à GR.

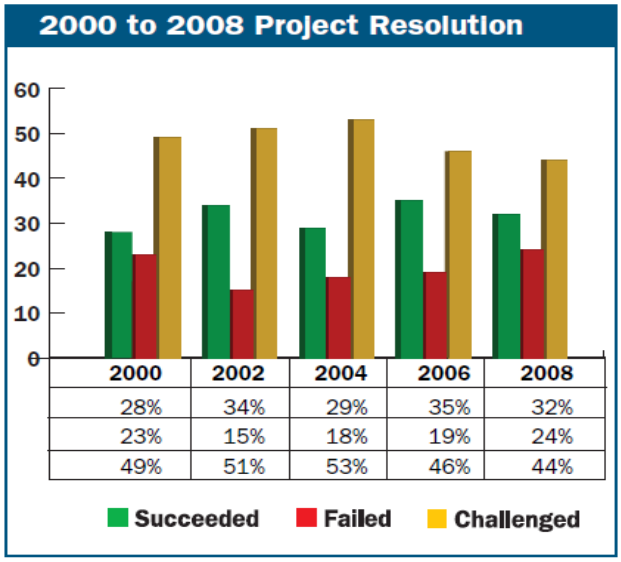
b) Relacionar os pontos positivos e negativos da adoção de um processo formal voltado à identificação, controle e rastreamento dos requisitos, incentivando a utilização das matrizes de rastreabilidade e contribuindo para a melhoria da qualidade dos softwares desenvolvidos no país.

1.2 Justificativa

A rastreabilidade ainda não é compreendida e, portanto, não é tratada adequadamente por aqueles envolvidos nos processos de desenvolvimento de software (SAYÃO e LEITE, 2005). Este estudo contribuirá para a disseminação do conhecimento relacionado ao controle das inevitáveis mudanças que ocorrem durante o processo de desenvolvimento de software, acarretando alterações no orçamento e cronograma dos projetos, com a utilização das MRR. Além dos conceitos básicos, serão apresentadas algumas das ferramentas existentes que automatizam a criação das matrizes.

A Figura 1 abaixo mostra o resultado de uma pesquisa realizada em 2009 pelo Standish Group, chamada Chaos Summary 2009 – 10 Laws of Chaos, a qual mostra os resultados de projetos que tiveram sucesso total, parcial ou que falharam. Entende-se por sucesso parcial projetos que embora estejam funcionando, foram entregues sem atender ao custo ou ao cronograma propostos, ou tiveram o escopo parcialmente entregue, com menos recursos e funções do que foram requeridos. É apontada que as principais causas dos problemas estão ligadas aos requisitos (THE STANDISH GROUP, 2009):

Figura 1: Projetos que tiveram sucesso total, parcial e que falharam



Fonte: The Standish Group, 2009

Acredita-se que a devida atenção dada à GR e a correta aplicação dos métodos e ferramentas da MRR farão com que as empresas consigam diminuir esse número. Dessa forma, este trabalho contribuirá substancialmente para que a comunidade empresarial tenha acesso às informações relevantes para a produção de software com qualidade, no que tange à rastreabilidade de requisitos e ao uso de matrizes, agregando também conhecimento à comunidade acadêmica, fornecendo diretrizes às disciplinas relacionadas ao tema em questão.

1.3 Estrutura do Documento

Além do primeiro capítulo, destinado à Introdução do trabalho, o desenvolvimento da monografia está dividido da seguinte forma:

- ✓ ***Capítulo 2:*** trata de todos os aspectos relacionados à ER e suas respectivas atividades, bem como os conceitos gerais de requisitos;

- ✓ ***Capítulo 3:*** apresenta os conceitos relacionados à GR e a importância da rastreabilidade;

- ✓ ***Capítulo 4:*** apresenta os conceitos relacionados à MRR e as ferramentas disponibilizadas para facilitar a criação e o gerenciamento dos requisitos;

- ✓ ***Capítulo 5:*** descreve os métodos e procedimentos utilizados para o desenvolvimento do trabalho;

- ✓ ***Capítulo 6:*** apresenta as conclusões e as recomendações finais.

2. ENGENHARIA DE REQUISITOS

A ER é um dos processos da Engenharia de Software (ES) voltado à criação e manutenção do documento de requisitos de sistema (SOMMERVILLE, 2003). Para tal, a ER possui algumas atividades essenciais para entender o que o cliente realmente deseja do software, qual o impacto que este terá dentro do negócio e como será a interação dos usuários finais após a entrega do produto (PRESSMAN, 2006).

Essas atividades precisam ser adequadas de acordo com a realidade e contexto de cada projeto a ser executado. Para Pressman (2006), a ER envolve sete funções, que são: concepção, levantamento, elaboração, negociação, especificação, validação e gestão. Já Sommerville (2003) acredita que existam quatro atividades básicas de alto nível, sendo estas o estudo de viabilidade do sistema, a obtenção e a análise de requisitos, a especificação de requisitos com a sua devida documentação e a validação dos requisitos. Na maioria dos projetos, os requisitos sofrem mudanças, por isso o autor também cita, separadamente, o gerenciamento de requisitos como uma das atividades do processo de ER.

Embora existam pequenas diferenças entre as citações dos autores, algumas dessas tarefas podem ser realizadas em paralelo, ou seja, em conjunto com outras atividades, sempre se adaptando ao projeto em questão. Todas "(...) tentam definir o que o cliente deseja e servem para estabelecer uma fundação sólida para o projeto e a construção do que o cliente obtém". (PRESSMAN, 2006, p.118).

Neste trabalho, as atividades serão descritas conforme o exposto por Sommerville. Por isso, primeiramente serão apresentados alguns conceitos importantes para facilitar o entendimento de cada uma das atividades da ER. A atividade de Gerenciamento de Requisitos (GR) será analisada separadamente no Capítulo 3.

2.1 Requisitos

É necessário compreender o que é um requisito e como ele pode ser classificado para atender àquilo que foi especificado pelo usuário/cliente, para posteriormente ser inserido no documento de especificação de requisitos. De acordo com o Software Technology Support

Center – STSC (2003, p. 45), “requisitos definem as capacidades ou condições que um sistema possui ou um componente necessário para resolver um problema ou atingir um objetivo”.

Segundo Sommerville (2003), a indústria de software não trata de forma consistente o termo requisito, podendo ser considerado como uma declaração do tipo abstrata, de alto nível, de uma função a ser atendida pelo software ou mesmo de uma restrição.

Devido às diferenças nos níveis das descrições, é interessante que seja feita uma separação entre os requisitos, a fim de se obter uma forma mais clara e compreensível para a sua utilização no processo de engenharia de requisitos, evitando futuros problemas. Para Sommerville (2003), essa separação pode ser realizada da seguinte forma:

- ✓ ***Requisitos do Usuário:*** declaração em linguagem natural para designar os requisitos abstratos de alto nível. Também podem ser utilizados diagramas para mostrar as funções que o sistema deverá oferecer e as restrições de operação. É direcionado para pessoas que não possuem conhecimento técnico detalhado do sistema, como gerentes do cliente e fornecedores.

- ✓ ***Requisitos de Sistema:*** detalham as funções e as restrições do sistema, sendo conhecido como especificação de requisitos de sistema. Deve ser preciso a ponto de servir de contrato entre o comprador e o desenvolvedor. É direcionado para os profissionais técnicos de nível sênior e os gerentes de projetos, mas poderá ser usado pelos gerentes do cliente, fornecedores e usuários finais.

- ✓ ***Especificação de projeto de software:*** descrição abstrata do projeto de software, acrescentando mais detalhes à especificação de requisitos de sistema. É direcionado à implementação, sendo escrito para os engenheiros de software que desenvolverão o sistema.

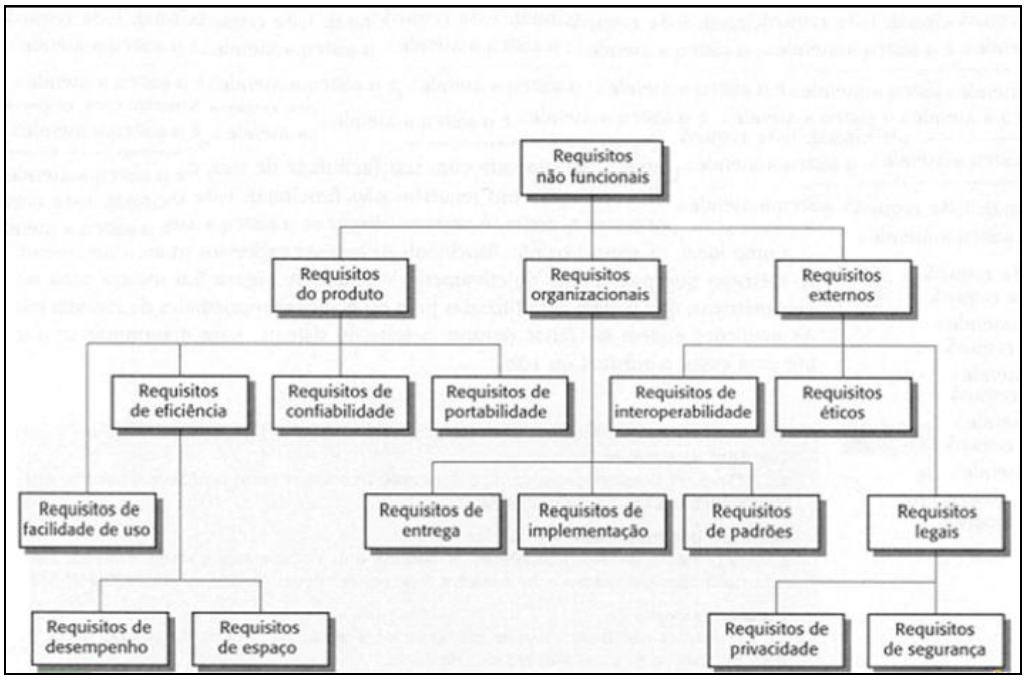
Os requisitos de sistema são normalmente classificados como funcionais, não-funcionais ou de domínio. É importante lembrar que os requisitos devem especificar o que é para ser feito e não o como será feito.

- ✓ ***Funcionais:*** descrevem o que um sistema deve fazer em resposta a um estímulo específico. Dessa forma, especificam o comportamento de entrada e saída de um sistema.

- ✓ ***Não-Funcionais:*** descrevem o desempenho ou restrições do sistema que afetam o desenvolvimento e o projeto. Podem incluir algumas categorias, como: segurança, confiabilidade, ambiente, implementação, pessoas, entre outros (STSC, 2003).

Os requisitos funcionais e não-funcionais podem surgir de acordo com a necessidade dos usuários, seja devido a alterações na legislação vigente que afeta o sistema, seja por restrições no orçamento, nova função a ser desempenhada pelo software, uma estratégia empresarial, como lançar o produto antecipadamente para atender a uma demanda do mercado ou alguma outra condição que obrigue uma intervenção no andamento do projeto. A Figura 2 mostra um tipo de classificação dos tipos de requisitos não-funcionais que podem surgir (SOMMERVILLE, 2003):

Figura 2: Classificação de requisitos não-funcionais



Fonte: Sommerville, 2003

- ✓ ***De domínio:*** podem ser requisitos funcionais ou não-funcionais e são derivados do domínio da aplicação, refletindo características desse domínio, ou seja, não são obtidos a partir das necessidades dos usuários do sistema. Se

esses requisitos não forem satisfeitos, o sistema poderá tornar-se impossível de operar.

2.2 Atividades da Engenharia de Requisitos

2.2.1 Estudo de Viabilidade

O início do processo de ER deve começar através de um estudo de viabilidade. Através deste será possível concluir se é ou não viável, do ponto de vista tecnológico e organizacional, a realização do processo de ER e do desenvolvimento do software. Conforme Sommerville (2003), o estudo de viabilidade tem o objetivo de responder às seguintes perguntas:

a. O sistema contribuirá para os objetivos gerais da organização?
Obs: *Se a reposta a essa pergunta for negativa, não haverá justificativa para a construção do sistema.*

b. Dentro das restrições de custo e prazo e com a tecnologia atualmente utilizada, o sistema poderá ser desenvolvido?

c. É possível a integração com outros sistemas já em operação?

Para a sua realização, além da participação dos engenheiros de software, é necessária a presença e participação dos *stakeholders*, tais como os responsáveis pelas áreas/departamentos que necessitam do sistema e os seus usuários finais, a fim de garantir uma coleta fidedigna das informações sobre o que o software deverá fazer e as restrições a ele impostas.

2.2.2 Levantamento e Análise de Requisitos

Se o projeto for viável, far-se-á a segunda atividade, conhecida como levantamento e análise de requisitos, a qual buscará identificar juntamente com os *stakeholders* quais serão os objetivos do sistema, que tipo de serviço ele deverá fornecer, quais as informações que englobam o domínio da aplicação, as restrições existentes, o desempenho desejado, entre outras coisas igualmente necessárias para o desenvolvimento do software. A análise de requisitos detecta e resolve os conflitos entre requisitos e descobre os limites do software e como ele deve interagir com seu meio ambiente.

É uma fase difícil de ser realizada devido aos problemas de entendimento em relação aos requisitos. De acordo com Christel e Kang (1992, apud PRESSMAN, 2006, p. 118 e 119), três tipos de problemas auxiliam a compreender a dificuldade supracitada, que são:

- ✓ ***Problemas de escopo:*** Má definição do sistema e de seus limites, impedindo a compreensão dos principais objetivos do sistema. Também pode acontecer se o cliente/usuário especificar detalhes técnicos que não são necessários, trazendo confusão em vez de entendimento.

- ✓ ***Problemas de entendimento:*** Os *stakeholders* não têm certeza do que desejam acerca do sistema, seja relacionado às necessidades que este deverá atender ou ao ambiente computacional que deverá operar. Além disso, informações que para eles são corriqueiras e, portanto, óbvias, como questões voltadas ao domínio da aplicação, ficam sem ser especificadas, acarretando problemas posteriores.

- ✓ ***Problemas de volatilidade:*** Os requisitos sempre mudam ao longo do processo de desenvolvimento. Assim, mesmo que estes tenham sido identificados corretamente, podem sofrer alterações devido a novas necessidades dos *stakeholders.*

Para resolver estes problemas, toda a atividade de levantamento e análise de requisitos deve ser feita de forma organizada, podendo utilizar diversas técnicas para que os requisitos coletados sejam inseridos no documento de especificação de maneira completa e consistente.

Algumas dessas técnicas são brevemente descritas abaixo, haja vista que o foco deste trabalho não é tratá-las de forma detalhada (SOMMERVILLE; BATISTA e CARVALHO, 2003):

- ✓ ***Levantamento orientado a pontos de vista:*** tem a capacidade de descobrir conflitos nos requisitos propostos por diferentes *stakeholders* através do reconhecimento de diferentes pontos de vista.

- ✓ ***Entrevistas e Questionários:*** são os meios mais usuais de se coletar fatos. As entrevistas podem ser estruturadas ou não-estruturadas, isto é, informais. Os questionários servem para se ter uma idéia mais definida de algumas características do software. Ambos necessitam que o engenheiro de software responsável já possua algum conhecimento do problema a fim de elaborá-las.

- ✓ ***Cenários:*** proporcionam maior facilidade para relacionar a realidade com os objetivos dos *stakeholders,* pois se utilizam de exemplos práticos ao invés de declarações abstratas para identificarem os requisitos. Podem ser realizadas técnicas mais estruturadas como cenários de eventos ou de *use-cases,* conhecidas como casos de uso.

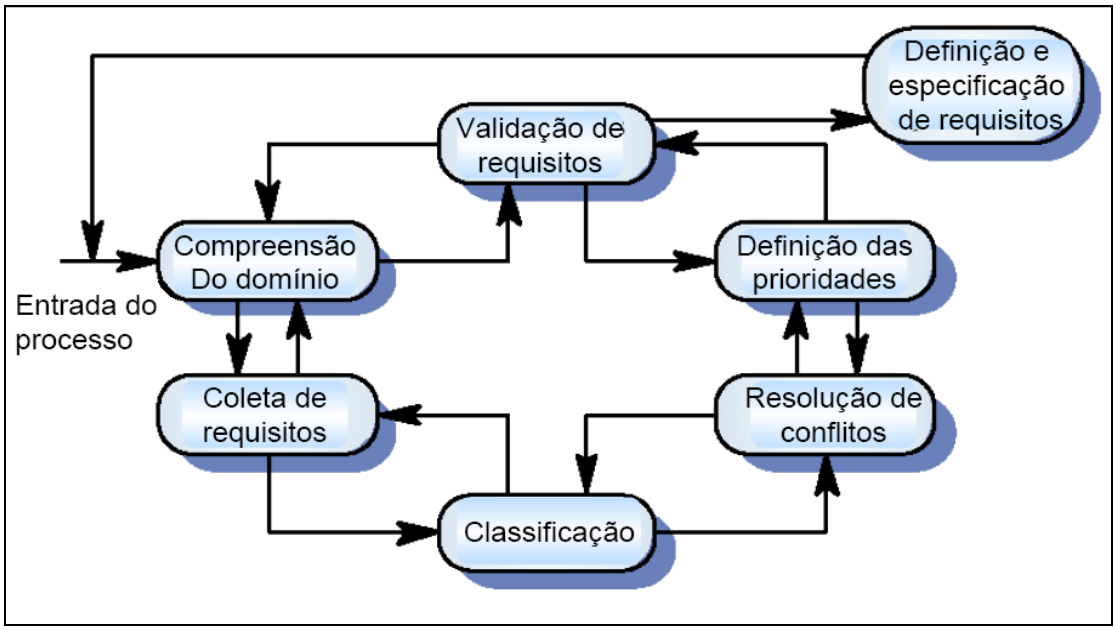
- ✓ ***Prototipação:*** desenvolvimento de parte do sistema, realizado durante todas as fases da ER, utilizando requisitos já identificados que servirão para auxiliar no levantamento de outros igualmente importantes, além de detectar falhas que antes não haviam sido percebidas. O usuário pode interagir com o protótipo e verificar se realmente é aquilo que deseja e se o projeto caminha no rumo certo.

- ✓ ***Etnografia:*** busca identificar os requisitos originários do contexto social e organizacional. É realizada através da inserção de um profissional no mesmo ambiente de trabalho no qual o sistema operará, com o intuito de observar, perceber e anotar requisitos importantes para o entendimento dos processos reais que acontecem no cotidiano daquele lugar e como impactam a vida dos usuários e, por conseguinte, do sistema a ser implementado.

As sub-atividades que compõem a atividade de elicitação e análise de requisitos

podem ser compreendidas em seis tarefas de caráter iterativo, cada uma gerando *feedback* [2] à outra, como mostra a Figura 3. Inicia-se na compreensão do domínio pelos analistas na qual a aplicação há de operar; efetua-se em seguida a coleta dos requisitos em conjunto com os *stakeholders*; faz-se a classificação em grupos coerentes de cada requisito; se conflitos forem encontrados, deverão ser resolvidos o mais rápido possível; defini-se a prioridade de cada requisito em grau de importância para os *stakeholders*; para finalizar, os requisitos são verificados para confirmar se estão em concordância com aquilo que os *stakeholders* desejam do sistema (SOMMERVILLE, 2003).

Figura 3: Processo de Levantamento e Análise de Requisitos



Fonte: Sommerville, 2003

2.2.3 Especificação e Documentação de Requisitos

Segundo Pressman (2006), a especificação pode ter vários significados dependendo do contexto em que se encontra o desenvolvimento de um projeto. Dessa forma, a especificação pode ser um simples documento escrito, um modelo gráfico ou matemático formal, uma série de cenários de uso, bem como um protótipo.

Embora seja usada com freqüência e tenha a facilidade de expressão e compreensão

[2] *Feedback* é o processo de fornecer informações a uma pessoa ou grupo, com o intuito de auxiliar na melhoria de desempenho no sentido de atingir seus objetivos.

como vantagem valiosa, a especificação feita em linguagem natural para descrever a funcionalidade e/ou restrição do software a ser desenvolvido, possui algumas desvantagens, como: falta de clareza, provocando ambigüidades; confusão de requisitos, como a falta de definição precisa de quais são os funcionais e não-funcionais, além dos objetivos do sistema misturados com as informações do projeto; fusão de requisitos, onde um requisito expresso como único engloba na verdade vários outros e de categorias distintas (SOMMERVILLE, 2003).

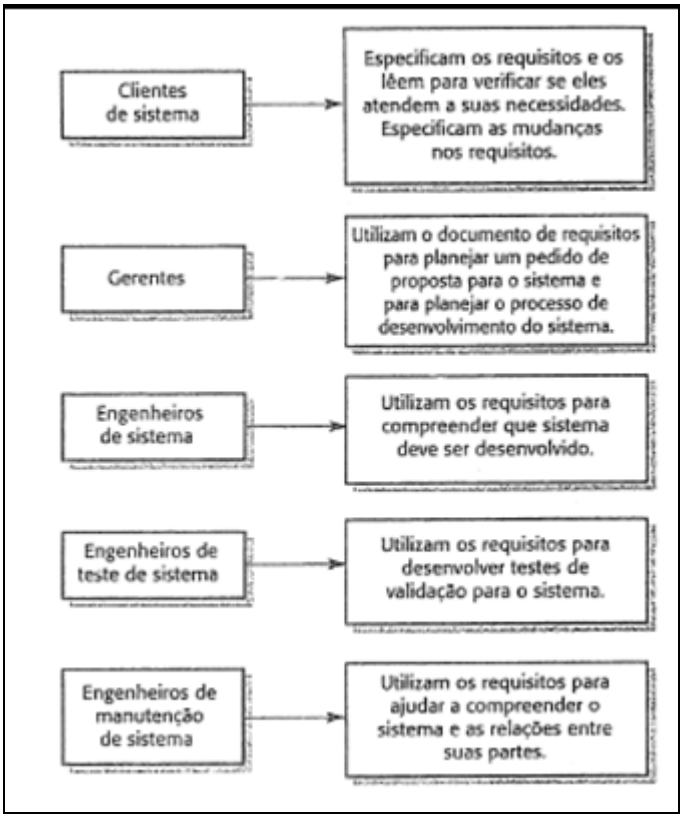
Para resolver estes tipos de problemas, foram criadas alternativas para a especificação, como a utilização da Linguagem Natural Estruturada, que junta à linguagem natural o uso de formulários padrão para garantir maior uniformidade na especificação dos requisitos. Além desta alternativa, há a PDL (*program description language*) ou linguagem de descrição de programa, que é derivada de uma linguagem de programação e possui a vantagem de gerar resultados bem detalhados, evitando as ambiguidades. Porém, necessita que o seu leitor tenha um conhecimento prévio de linguagem de programação.

Podem ainda ser utilizadas notações gráficas como a SADT (*Structured Analysis and Design Technique*) ou Técnica de Análise e Desenho Estruturado e a UML (*Unifief Modeling Language*) ou Linguagem de Modelagem Unificada para descrever as funcionalidades do sistema em conjunto com anotações textuais.

O documento final destas atividades, também chamado de *Software Requirements Specification* ou Especificação de Requisitos de Software (ERS) conterá todos os requisitos identificados nas etapas anteriores e servirá como guia para os desenvolvedores, descrevendo "(...) a função e o desempenho de um sistema baseado em computador e as restrições que governarão o seu desenvolvimento". (PRESSMAN, 2006, p. 120)

A ERS possui usuários diferentes, com conhecimentos técnicos distintos. Por isso, ele deve ser elaborado de forma que todos possam entender as exigências do sistema. Para que essa uniformidade seja alcançada, o documento deve ter separado os requisitos de usuário dos de sistema, devido ao público diferente que cada um atende, conforme descrito em 2.1. A Figura 4 mostra estes diversos usuários e como eles utilizam o documento de requisitos (SOMMERVILLE, 2003):

Figura 4: Usuários de um documento de requisitos



Fonte: Sommerville, 2003

2.2.4 Validação de Requisitos

Um documento mal elaborado pode gerar custos altos no decorrer do processo de desenvolvimento, quando erros e inconsistências nos requisitos podem ser detectados. O custo é maior se o software já estiver em operação. A validação de requisitos verifica se os requisitos estão incompletos, buscando sua completeza e consistência junto aos *stakeholders* para confirmar que aqueles estão de acordo com os padrões da empresa e conforme o que se espera do software. Diferentes *stakeholders,* tanto aqueles que representam o cliente/usuário, como os responsáveis pelo desenvolvimento deverão rever o documento (SWEBOK, 2004).

Uma série de verificações deve ser realizada durante a atividade de validação visando os seguintes atributos (SOMMERVILLE, 2003):

- ✓ ***Validade:*** é preciso que os diferentes usuários que utilizarão o sistema e participaram da elicitação de requisitos entendam que a ERS é um conjunto

conciliatório das diversas funcionalidades exigidas.

- ✓ ***Consistência:*** corrigir todo o tipo de conflito que exista entre os requisitos.
- ✓ ***Completeza:*** a ERS deve incluir todos os requisitos que descrevam a funcionalidade do software e as suas restrições.
- ✓ ***Realismo:*** o software precisa ser implementado dentro das restrições existentes de custos, prazos e tecnologias a serem utilizadas para sua produção.
- ✓ ***Facilidade de Verificação:*** a forma em que os requisitos são escritos deve ser tal que permita que tanto o cliente quanto o fornecedor possam verificar se os mesmos estão de acordo com o solicitado.

Algumas técnicas de validação podem ser utilizadas para auxiliar a verificação dos atributos supracitados, como a revisão de requisitos por uma equipe de revisores, já citado segundo o SWEBOK nesta mesma seção. Além desta, a prototipação poderá ser realizada para criar ao usuário um modelo executável do sistema apenas para verificação de que os requisitos descritos na ERS atenderão ao solicitado. Casos de teste também podem ser gerados para testar os requisitos antes mesmo de estes serem implementados, permitindo a correção de falhas de forma antecipada, economizando tempo e dinheiro. Outra forma é o uso de Ferramentas CASE (*Computer-Aided Software Engineering*) ou Engenharia de Software Assistida por Computador, que podem criar banco de dados de requisitos e utilizar métodos de notação formal para verificar possíveis inconsistências através de uma análise automatizada. Mesmo diante de todas as técnicas conhecidas para corrigir os erros no documento de requisitos, é inevitável que este venha a apresentar alguma inconsistência posteriormente, mesmo já tendo sido aceito (SOMMERVILLE, 2003).

Devido a estes fatos, o SWEBOK (2004) incentiva que a ERS seja também um dos artefatos a serem colocados sob o controle da Gerência de Configuração de Software (GCS) com o intuito de controlar as futuras modificações.

3. GERENCIAMENTO E RASTREABILIDADE DE REQUISITOS

Neste capítulo serão apresentados os motivos que provocam as mudanças nos requisitos, bem como a sua classificação neste quesito. A seguir, serão mostrados o objetivo e as atividades ligadas ao GR. Por fim, será apresentado o conceito de rastreabilidade e a sua importância para o GR.

3.1 As Mudanças nos Requisitos

O fato de a ERS precisar ser controlada pelo processo de GCS mostra que os requisitos não só podem ter sido levantados de forma equivocada ou incompleta, mas que também aqueles corretamente identificados poderão sofrer algum tipo de modificação. Leite (2001) entende que a construção de software será cada vez mais baseada no conceito de evolução, sendo que o próprio processo de identificação de requisitos gera um *feedback* que promoverá mudanças nos requisitos. Por isso, acredita ser falsa a idéia de congelamento de requisitos (HAZAN e LEITE, 2003).

As inevitáveis mudanças nos requisitos acontecem por vários motivos. Felici (2004) cita os seguintes:

- ✓ Melhor compreensão das necessidades do sistema por parte dos *stakeholders*, os quais refinam a descrição dos requisitos de forma a torná-los mais precisos.

- ✓ Descoberta de novos requisitos que foram esquecidos durante o início do processo devido à falta de informações e dependências de requisitos.

- ✓ *Feedback* fornecido pelas atividades de concepção, desenho, implementação e teste em relação ao requisitos, propiciando novas mudanças.

É interessante classificar os requisitos segundo a perspectiva de evolução, ou seja, se são permanentes ou voláteis. Segundo Sommerville (2003), os requisitos permanentes são relativamente estáveis e referem-se à atividade principal do cliente, estando diretamente

ligados ao domínio do sistema. Já os requisitos voláteis são aqueles com alto grau de probabilidade de se modificarem ao longo do processo de desenvolvimento do sistema ou quando este já estiver em uso.

Os requisitos voláteis podem ainda ser divididos em quatro tipos (SOMMERVILLE, 2003):

1. ***Requisitos Mutáveis:*** modificam-se conforme a mudança do ambiente em que a organização opera.

2. ***Requisitos Emergentes:*** surgem de acordo com o aumento do entendimento do cliente a respeito do sistema.

3. ***Requisitos Consequentes:*** a introdução do sistema de computação na empresa pode mudar os seus processos e criar novas formas de trabalho, gerando novos requisitos.

4. ***Requisitos de Compatibilidade:*** são aqueles que dependem de outros sistemas ou processos de negócios. Se estes forem modificados, os requisitos de compatibilidade também deverão evoluir.

De acordo com a Melhoria de Processo do Software Brasileiro – MPS.BR (2009), os eventos que geram a modificação nos requisitos podem exigir uma nova análise para confirmar a viabilidade de continuidade do projeto.

3.2 Objetivo e Atividades do Gerenciamento de Requisitos

O objetivo do GR é o de controlar as mudanças dos requisitos, de modo a medir o impacto que elas poderão causar no sistema, os riscos associados, possíveis alterações no orçamento e cronograma do projeto. O resultado deste processo permitirá aos *stakeholders* tomarem a decisão correta quanto à implementação ou não de determinada modificação ou inclusão de requisitos, estabelecendo assim um acordo entre o cliente e o desenvolvedor a

respeito de quais funcionalidades e restrições serão inseridas na *baseline* [3] do produto.

Os requisitos devem ser identificados de forma única, a fim de facilitar a sua rastreabilidade e o controle de mudanças (DINIZ, 2007). Cada organização adota uma maneira específica para criar essa identificação (ID), podendo ser desde uma simples numeração em ordem crescente, como os gerados em bancos de dados, até um conjunto de letras e números que compõem o requisito.

Segundo Sayão e Leite (2005), para que possa alcançar seu objetivo, o processo de GR é composto de algumas atividades, tais como o controle de mudanças, que consiste no registro de novas solicitações e/ou alterações em requisitos já definidos; a identificação da origem e dos relacionamentos entre um ou mais requisitos e seus respectivos componentes; as análises de impacto em relação aos orçamentos e cronogramas devido às mudanças; controle de versões de todos os artefatos utilizando a GCS e acompanhamento do estado dos requisitos.

Essas atividades não constituem um padrão a ser seguido, ficando cada organização responsável por adaptá-las à sua realidade. Como exemplo, o Serviço Federal de Processamento de Dados (SERPRO) adotou as atividades mostradas na Tabela 1 (SERPRO, 2002, apud HAZAN e LEITE, 2003, p. 3):

Tabela 1: Atividades do Gerenciamento de Requisitos da SERPRO

ATIVIDADES	DESCRIÇÃO
Receber as solicitações de alteração de requisitos	O grupo de engenharia de requisitos recebe as solicitações de alteração de requisitos, ou por formulário padronizado, ou por meio de um sistema de solicitação de demandas.
Registrar novos requisitos	Novos requisitos também devem ser recebidos formalmente, seja por formulário padronizado, ou por meio de controle sistemático.
Analisar impacto da mudança de requisitos	Uma análise criteriosa deve ser conduzida para avaliar o impacto do requisito a ser incluído, alterado ou excluído sobre cada um dos seus requisitos relacionados, os quais podem ser identificados por meio das matrizes de rastreabilidade. Caso o impacto seja significativo, os requisitos (analisado e relacionado) devem ser revistos.
Elaborar relatório de impacto	Deve ser mantido um histórico de alterações para cada requisito, permitindo uma visão cronológica das principais mudanças nos requisitos.
Notificar os envolvidos	Os envolvidos são um conjunto de pessoas para as quais pode haver um impacto devido à alteração de requisitos (alteração, inclusão ou exclusão de requisitos) e devem ser notificados.
Coletar métricas	As métricas devem ser utilizadas e coletadas periodicamente para o acompanhamento das atividades de Gerência de Requisitos.

Fonte: Hazan e Leite, 2003

[3] *Baseline* é um conjunto de especificações ou produtos de trabalho que foram formalmente revisados e sobre os quais foi feito um acordo, que serve como base para desenvolvimento posterior e que pode ser modificado somente através dos procedimentos de controle de mudanças

O ideal é que todas as solicitações de inclusão, alteração ou exclusão de requisitos sejam primeiramente registradas e devidamente analisadas. O custo da mudança pode abranger o documento de requisitos e, dependendo do momento da solicitação, envolver também o projeto de sistemas e a implementação. Isso inclui as modificações consideradas urgentes pelos *stakeholders*, pois é comum que os desenvolvedores implementem esse tipo de modificação diretamente no sistema e, somente depois, caso se lembrem, inserem a especificação no documento de requisitos. Essa atitude normalmente provoca inconsistências em relação ao que está no sistema daquilo que está na documentação (SOMMERVILLE, 2003).

3.3 Rastreabilidade de Requisitos

Como informado acima, a identificação única dada aos requisitos possui um papel fundamental, pois tem o objetivo de facilitar o seu rastreamento e controlar de forma organizada as suas possíveis alterações. Por isso, a RR deve assegurar que a evolução do sistema estará de acordo com as necessidades dos *stakeholders* (RAMESH e JARKE, 1999).

A RR refere-se à:

> (...) capacidade de descrever e seguir a vida de um requisito, em ambos os sentidos, para frente e para trás, isto é, desde as suas origens, através do seu desenvolvimento e especificação, para a sua posterior implementação e utilização, e através de todos os períodos em curso de refinamento e interação em qualquer uma destas fases. (GOTEL e FINKELSTEIN, 1994, p. 97).

De acordo com o IIBA (2009), a rastreabilidade é usada para ajudar a garantir a conformidade da solução com os requisitos e para auxiliar no alcance e gestão de mudanças, riscos, tempo, custos e comunicação.

Entende-se dessa forma, que a RR é uma atividade do GR destinada a definir e utilizar as relações entre os requisitos e os artefatos produzidos no processo de desenvolvimento do software (EGYED e GRÜNBACHER, 2005). Assim, deve ser possível identificar qual o requisito de origem que resultou em determinada aplicação, bem como o tipo de implementação que há de ser gerada por determinado requisito. Além disso, também é

possível identificar a relação deste com outros requisitos e suas respectivas dependências, de modo que gerem o produto final.

Para Ramesh e Jarke (1999), na ER, o maior desafio é a vinculação de *rationales*[4] e fontes aos requisitos. Para aliviar estes problemas, a RR facilita a comunicação entre os *stakeholders*, bem como a avaliação do impacto das mudanças dos requisitos e o suporte à sua realização, preservando o conhecimento e as dependências criadas no decorrer do projeto (BACKES, 2008).

3.3.1 Tipos de Rastreabilidade

Gotel e Finkelstein (1994) sugeriram dois tipos de rastreabilidade: a chamada pré-rastreabilidade – que se preocupa com os aspectos da vida de um requisito antes de sua inclusão na ERS – e a pós-rastreabilidade – que se preocupa com os aspectos da vida de um requisito que resultam de sua inclusão na ERS. Dentro destes tipos, existe a seguinte classificação de rastreabilidade (FELICI, 2004; BACKES, 2008):

Na Pré-Rastreabilidade:

Forward to Requirements **(Adiante, em direção aos requisitos)** – relaciona outros documentos, anteriores ao documento de requisitos. Mudanças nas necessidades dos *stakeholders*, bem como em pressupostos técnicos, podem levar a uma reavaliação radical da relevância dos requisitos.

Backward from Requirements **(De volta, a partir dos requisitos)** – relaciona os requisitos às suas fontes, em outros documentos ou pessoas. As estruturas em que os requisitos se baseiam são cruciais na sua validação.

Na Pós-Rastreabilidade:

Forward from Requirements **(Adiante, a partir dos requisitos)** – relaciona os

[4] Rationale indica as razões, motivações ou intenções para uma determinada ação.

requisitos ao desenho e à implementação, atribuindo a responsabilidade do requisito a um desenho específico e componentes de aplicação. Permite avaliar o impacto das mudanças de requisitos nestes processos.

Backward to Requirements **(De volta para os requisitos)** – relaciona o desenho e a implementação de volta aos requisitos, avaliando se aqueles cumprem os requisitos de alto nível. Permite identificar se o que foi desenvolvido estava especificado ou não.

Felici (2004) cita como exemplos dentro da pré-rastreabilidade os requisitos-fontes, que vinculam os requisitos às pessoas ou documentos que realizaram a sua especificação. Além disso, existem os requisitos-*rationale*, que vinculam os requisitos com a descrição do por que aquele requisito foi especificado. Já a pós-rastreabilidade envolve os requisitos-arquitetura, que ligam os requisitos com os subsistemas que os implementam; os requisitos-desenho, que vinculam os requisitos a um hardware específico ou um componente de software no sistema que implementará os requisitos e, por fim, os requisitos-interface, que relacionam os requisitos às interfaces de sistemas externos que fornecem as exigências.

Ao efetuar o rastreamento dos requisitos, o relacionamento entre eles deve ser identificado de modo a permitir sua análise e implementação. De acordo com Marques (2010), os principais relacionamentos podem ser descritos como de:

- ✓ ***Necessidade*** – um requisito é importante somente se existir a inclusão de outro requisito.
- ✓ ***Esforço*** – qual o requisito que pode ser implementado mais facilmente se outro também o for.
- ✓ ***Subdivisão*** – um requisito existe como parte de outro requisito.
- ✓ ***Cobertura*** – representa um caso especial de subconjunto, sendo o contrário da Subdivisão.
- ✓ ***Valor*** – quando um requisito altera a percepção do *stakeholder* em relação a outro requisito, ou quando modifica uma funcionalidade ou desempenho já existente.

3.3.2 Problemas Relacionados à Rastreabilidade

A GR e a RR são práticas exigidas para que as organizações obtenham certificados nacionais e internacionais de qualidade, como o MPS.BR e o CMMI. Todos os requerimentos são passíveis de serem incluídos no rastreamento de requisitos. Porém, a realidade praticada nas empresas, bem como muitos estudos realizados nesta área de conhecimento têm mostrado que implantar a rastreabilidade de requisitos dentro do processo de desenvolvimento de software não garante uma relação de custo/benefício que justifique o seu uso.

Embora exista uma ampla literatura que comprove os benefícios do uso da rastreabilidade de requisitos, os custos relacionados à sua implantação são altos, normalmente devido à dificuldade de gerar os relacionamentos da RR automaticamente e de forma clara e precisa. Além disso, a heterogeneidade e os muitos artefatos produzidos durante o desenvolvimento e a falta de corretude e completude das relações de rastreabilidade também elevam o investimento a ser aplicado para a implantação de uma política de rastreamento realmente eficaz (BACKES, 2008).

Para resolver este tipo de problema, algumas organizações separam os requisitos, bem como os artefatos produzidos, como código-fonte, documentos, casos de uso, entre outros com importâncias diferenciadas. Conhecido como Engenharia de Software Baseada em Valor, esse método permite classificar a relevância dos artefatos de software, tratando-os como itens críticos, importantes ou de interesse. Assim, os itens de interesse não possuem o mesmo peso que os críticos, permitindo que a análise dos *links* de rastreabilidade possa ser adaptada conforme a importância dos artefatos que os requisitos hão de gerar (EGYED, 2005).

Dessa forma, Backes (2008) afirma que o foco do trabalho de manutenção dos relacionamentos e seus respectivos *links* volta-se naquilo que é de maior impacto nas necessidades dos clientes, evitando desperdício de tempo no detalhamento de *links* de rastreabilidade que não se traduzam em benefícios ao projeto.

De acordo com os estudos realizados por Gotel e Finkelstein (1994), muitos problemas ligados a uma pobre RR devem-se também à realização inadequada da Pré-Rastreabilidade, a qual faz com que os requisitos sejam desenvolvidos sem a devida completude, reduzindo assim a qualidade do produto final.

4. MATRIZ DE RASTREABILIDADE DE REQUISITOS

Uma das ferramentas mais utilizadas pela indústria de software para a realização do rastreamento dos requisitos é a Matriz de Rastreabilidade de Requisitos (MRR). A MRR tem o objetivo de garantir que o objeto dos requisitos esteja em conformidade com o que foi especificado na ERS, associando cada requerimento com o objeto a ser produzido. Além disso, é utilizada para verificar se o *Forward Trace* foi devidamente realizado, ou seja, se todos os requisitos estabelecidos e derivados são atribuídos aos componentes do sistema e outros produtos. Determina também a origem dos requisitos, a conhecida *Backward Trace* e é utilizada para garantir ao cliente que todos os requerimentos serão cumpridos (LUDWIG CONSULTING SERVICES, 2008).

Através do seu uso, é possível verificar quais componentes serão afetados em casos de mudanças de requisitos, permitindo assim que as estimativas de prazo, custo, desvantagens e vantagens das mudanças sejam realizadas mais facilmente, explicitando o impacto de tais modificações.

4.1 Tipos de Matrizes de Rastreabilidade

Após a atribuição de um identificador único a cada requisito, para que a GR seja efetuada de forma completa é preciso criar as MRR ou Tabelas de Rastreamento (TR). Entre elas, Pressman (2006) cita as seguintes:

- ✓ ***TR de Características*** – demonstra o relacionamento dos requisitos com características importantes do sistema vistas pelo cliente.

- ✓ ***TR de Fontes*** – A fonte de cada requisito é identificada.

- ✓ ***TR de Dependência*** – Mostra como os requisitos relacionam-se uns com os outros.

- ✓ ***TR de Subsistemas*** – Os requisitos são caracterizados pelos subsistemas que

eles governam.

- ✓ ***TR de Interface*** – Demonstra o relacionamento dos requisitos com as interfaces internas e externas do sistema.

4.2 Modelos e Ferramentas para Construção da MRR

Em sua forma mais simplificada, uma MRR pode ser manifestada em tabelas cruzadas, onde os elementos-fonte são mapeados para elementos-alvo. A Figura 5 mostra como f_1 dá origem à a_1, a_3 e a_4 (BACKES, 2008):

Tabela 2: Exemplo de Dependência entre Elementos

		Alvos			
		a_1	a_2	a_3	a_4
Fontes	f_1	1	0	1	1
	f_2	0	1	0	0
	f_3	0	0	1	0

Fonte: Backes, 2008

Sommerville (2003) apresenta um exemplo de matriz de facilidade de rastreamento que vincula os requisitos a outros requisitos, sendo que cada um é representado por uma linha e por uma coluna na matriz. As dependências entre os requisitos são demonstradas na Tabela 3, onde a letra U indica que o requisito na linha utiliza os recursos do requisito nomeado na coluna. A letra R indica que há uma relação fraca entre os requisitos representados na matriz.

Tabela 3: Matriz de Facilidade de Rastreamento de Sommerville

Req. id	**1.1**	**1.2**	**1.3**	**2.1**	**2.2**	**2.3**	**3.1**	**3.2**
1.1		U	R					
1.2			U			R		U
1.3	R			R				
2.1			R		U			U
2.2								U
2.3		R		U				
3.1								R
3.2							R	

Fonte: Sommerville, 2003.

Além das vantagens já descritas neste trabalho sobre o uso da rastreabilidade e da MRR, pode-se acrescentar o benefício de se utilizar as matrizes para mapear os requisitos de usuário com os casos de teste. Dessa forma, é possível certificar-se que todos os requisitos foram cobertos pelos casos de teste, sem deixar que alguma funcionalidade seja esquecida.

A inexistência da MRR ou o seu uso de forma incompleta resulta em testes pobres e, consequentemente, em mais defeitos encontrados na produção. Isso acontece porque alguns erros passam despercebidos nos primeiros ciclos de teste, podendo surgir nos ciclos finais, causando problemas para a equipe reparar tais erros, como desentendimentos, retrabalho e atrasos no cronograma. Por outro lado, o uso das matrizes proporciona a facilidade de identificar funcionalidades esquecidas, provê a certeza da inclusão dos requisitos nos casos de teste, além de servir de guia aos desenvolvedores de que não estão desenvolvendo algo que não foi solicitado. Em caso de mudança nos requisitos, é possível encontrar facilmente quais casos de teste deverão ser atualizados (SOFTWARE TESTING, 2011).

As MRR podem ser criadas com a utilização de tabelas e hiperlinks em processadores de textos, com o uso de planilhas eletrônicas, banco de dados ou em ferramentas de gerenciamento de requisitos.

Um modelo simples pode ser feito com o uso do *Microsoft Excel* seguindo os seguintes parâmetros:

- ✓ ID Requisito
- ✓ Descrição do Requisito
- ✓ Caso de Teste 001
- ✓ Caso de Teste 002, em diante.

Tabela 4: Modelo Simples de Matriz de Rastreabilidade com Casos de Teste

ID REQUISITO	DESCRIÇÃO DO REQUISITO	CASO DE TESTE 001	CASO DE TESTE 002	CASO DE TESTE NNN
1.1	O sistema deve permitir o cadastro dos dados pessais dos clientes	Sim		
1.2	O sistema deve emitir relatório à gerência	Sim	Sim	
1.3	Ao clicar no botão Enviar o sistema deverá enviar uma mensagem de confirmação		Sim	
1.4	A mensagem de confirmação deverá ser enviada em até dois segundos.	Sim		

Para uma matriz mais completa, que venha a abranger maiores detalhes de rastreamento, podem-se acrescentar outras colunas para tornar a matriz mais eficaz, de acordo com a realidade do projeto em execução, como por exemplo:

- ✓ ID Requisito
- ✓ Descrição do Requisito
- ✓ Necessidades do Cliente
- ✓ Requisito Funcional
- ✓ *Status*
- ✓ Prioridade
- ✓ Documento da Arquitetura/Desenho
- ✓ Especificação Técnica
- ✓ Componentes do Sistema
- ✓ Módulos de Software
- ✓ Número do Caso de Teste
- ✓ Testado?
- ✓ Implementado?
- ✓ Verificação
- ✓ Comentários Adicionais

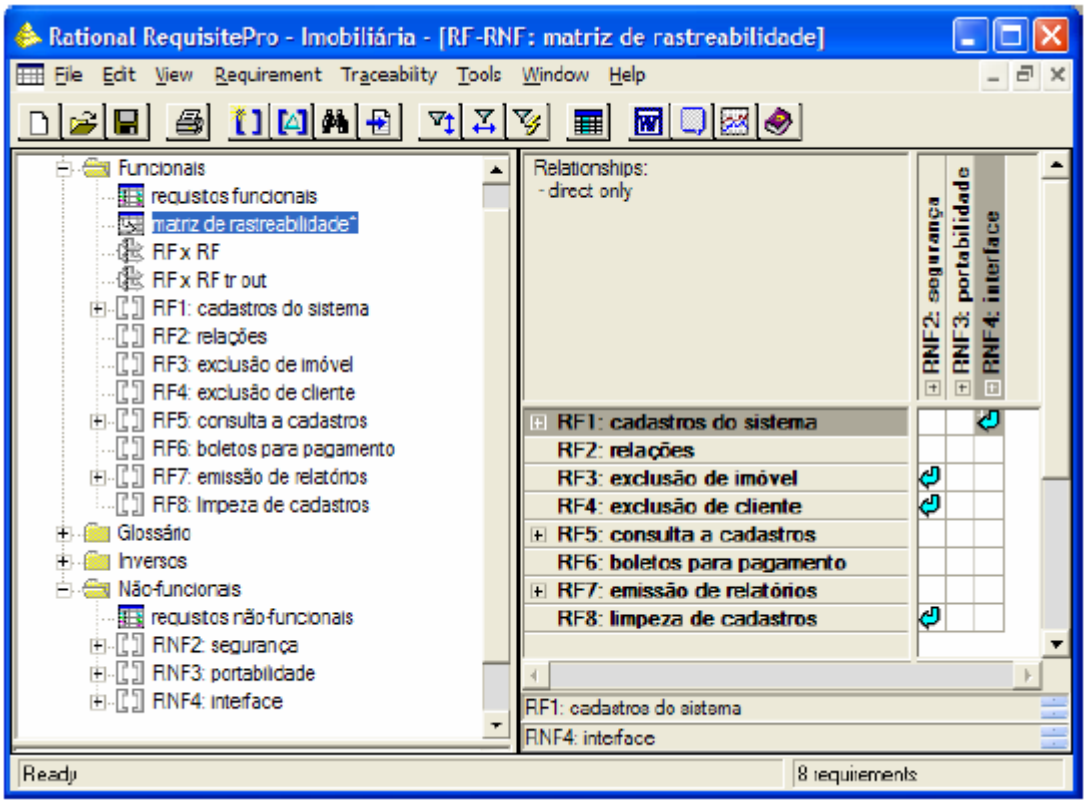
Tabela 5: Modelo Avançado de Matriz de Rastreabilidade

REQUIREMENTS TRACEABILITY MATRIX								REQUIREMENTS TRACEABILITY MATRIX					
Project Name:		<optional>						Project Name:	<optional>				
National Center:		<required>						National Center:	<required>				
Project Manager Name:		<required>						Project Manager Name:	<required>				
Project Description:		<required>						Project Description:	<required>				
ID	*Assoc ID*	*Technical Assumption*	*Functional Requirement*	*Status*	*Architectural/Design*	*Technical Specificati*	*System Component(*	*Software Module(s)*	Test Case Number	Tested In	Implemented In	Verification	Additional Comments
001	1.1.1												
002	2.2.2												
003	3.3.3												
004	4.4.4												
005	5.5.5												

Fonte: Centers for Disease Control and Prevention, 2011.

O mercado apresenta ferramentas comerciais e iniciativas de software livre, todas com o intuito de auxiliar os desenvolvedores no gerenciamento de requisitos e construção da MRR, proporcionando o controle desejado das mudanças que venham a ocorrer durante os projetos. Em seguida, serão apresentadas algumas das ferramentas mais utilizadas para este processo:

Figura 5: Rational RequisitePro – Matriz de Rastreabilidade

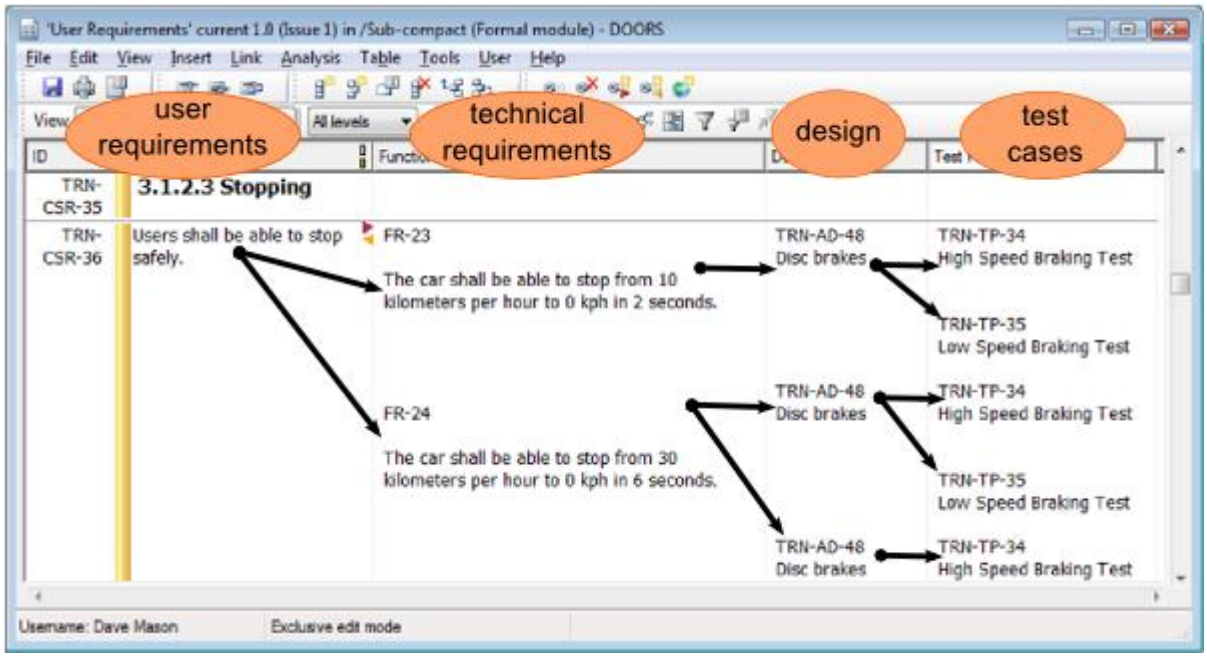


Fonte: Sayão e Leite, 2005

O *IBM Rational® RequisitePro®* da IBM é uma ferramenta comercial de GR que ajuda a equipe do projeto a gerenciar seus requisitos, escrever casos de uso, melhorar a rastreabilidade, fortalecer a colaboração, reduzir o retrabalho e aumentar a qualidade. Além disso, possui integração em tempo real com o *Microsoft Word* e interface *Web* escalável para colaboração de equipes espalhadas geograficamente, exibe os relacionamentos detalhados dos requisitos, permitindo gerenciar a complexidade da rastreabilidade, auxiliando na redução de riscos na ocorrência de mudanças de requisitos (IBM, 2011). Disponível versão *Trial* por quinze dias no site da fabricante em http://www.ibm.com/developerworks/downloads/r/rrp/?S_CMP=RRP.

Outra ferramenta comercial conhecida foi adquirida pela IBM através da compra da empresa Telelogic. Agora chamado *IBM® Rational® DOORS®*, esse software possui características muito parecidas com as do *RequisitePro*, possuindo uma versão *Trial* para ser usada sem instalação, via *web*, por três horas em http://www.ibm.com/developerworks/downloads/r/doorswebaccess/. A Figura 6 mostra parte da interface deste software (IBM, 2011):

Figura 6: Rational DOORS da IBM



Fonte: ECCAM, 2011

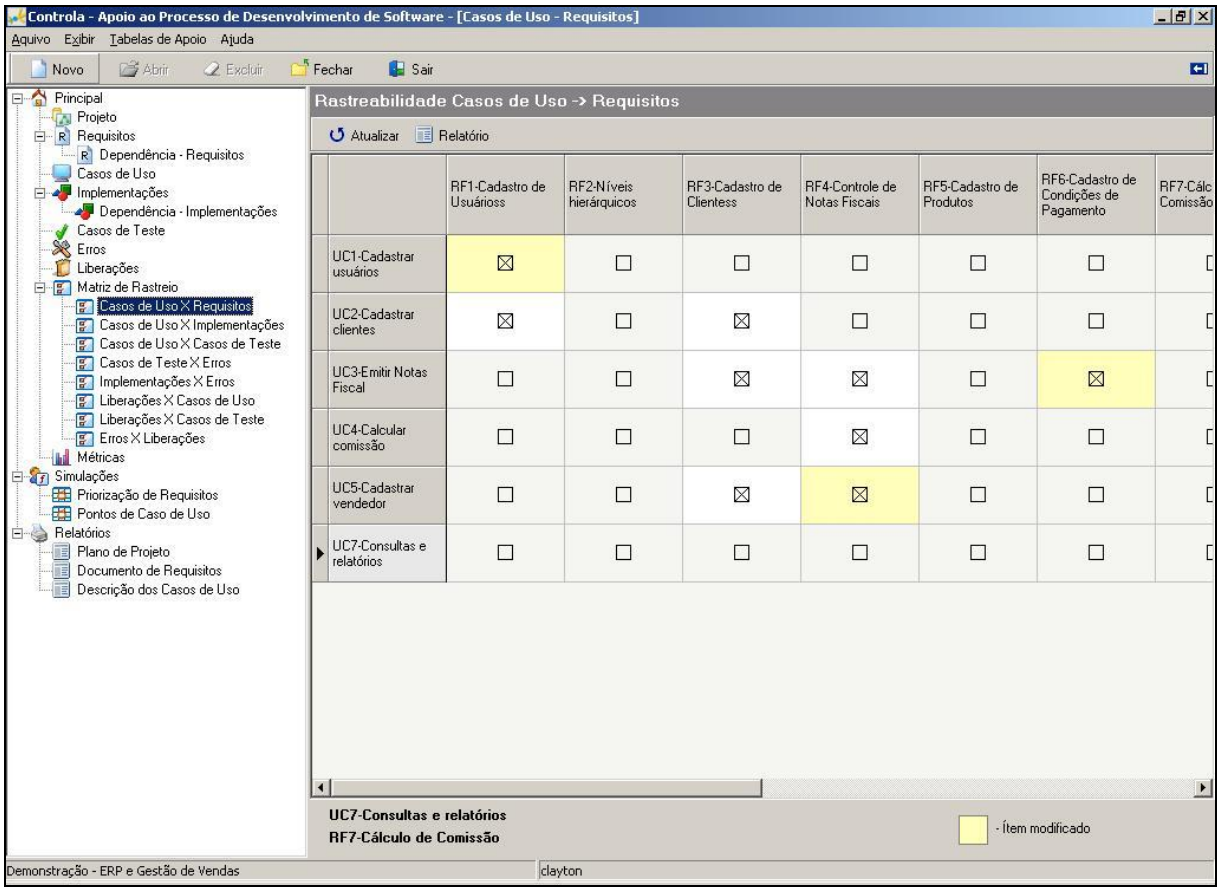
Existem algumas ferramentas gratuitas para auxiliar o desenvolvimento de software e a GR. Entre elas, pode-se destacar o software Controla, desenvolvido na Faculdade de Viçosa (FDV) – MG. O Controla oferece importantes recursos, sendo que os mais importantes são (DEVMEDIA, 2010):

- ✓ Gerenciamento de Requisitos;
- ✓ Gerenciamento de Casos de Uso;
- ✓ Gerenciamento de Casos de Teste e Erros;
- ✓ Planejamento de Liberações;
- ✓ Gerenciamento de Implementações;
- ✓ Controle de Dependência entre implementações;
- ✓ Matriz de Rastreabilidade (*Traceability Matrix*):
- ✓ Rastreabilidade dos requisitos;
- ✓ Rastreabilidade de projeto;
- ✓ Casos de Uso X Implementações;
- ✓ Casos de Uso X Casos de Teste;
- ✓ Casos de Teste X Erros;
- ✓ Implementações X Erros;
- ✓ Liberações X Casos de Uso;

- ✓ Liberações X Casos de Teste;
- ✓ Erros X Liberações;
- ✓ Registro de Métricas para todos os artefatos;
- ✓ Ferramenta de estimativa de tamanho de software por Pontos de Casos de Uso;
- ✓ Ferramenta para priorização de Requisitos;
- ✓ Emissão de documentos:
- ✓ Documento de Plano de Projeto;
- ✓ Documento de Casos de Uso;
- ✓ Documento de Especificação de Requisitos;
- ✓ Exportação de dados.

O software pode ser baixado no site http://www.cientec.net/scripts/controla/download.asp.

Figura 7: Controla – Apoio ao Processo de Desenvolvimento de Software

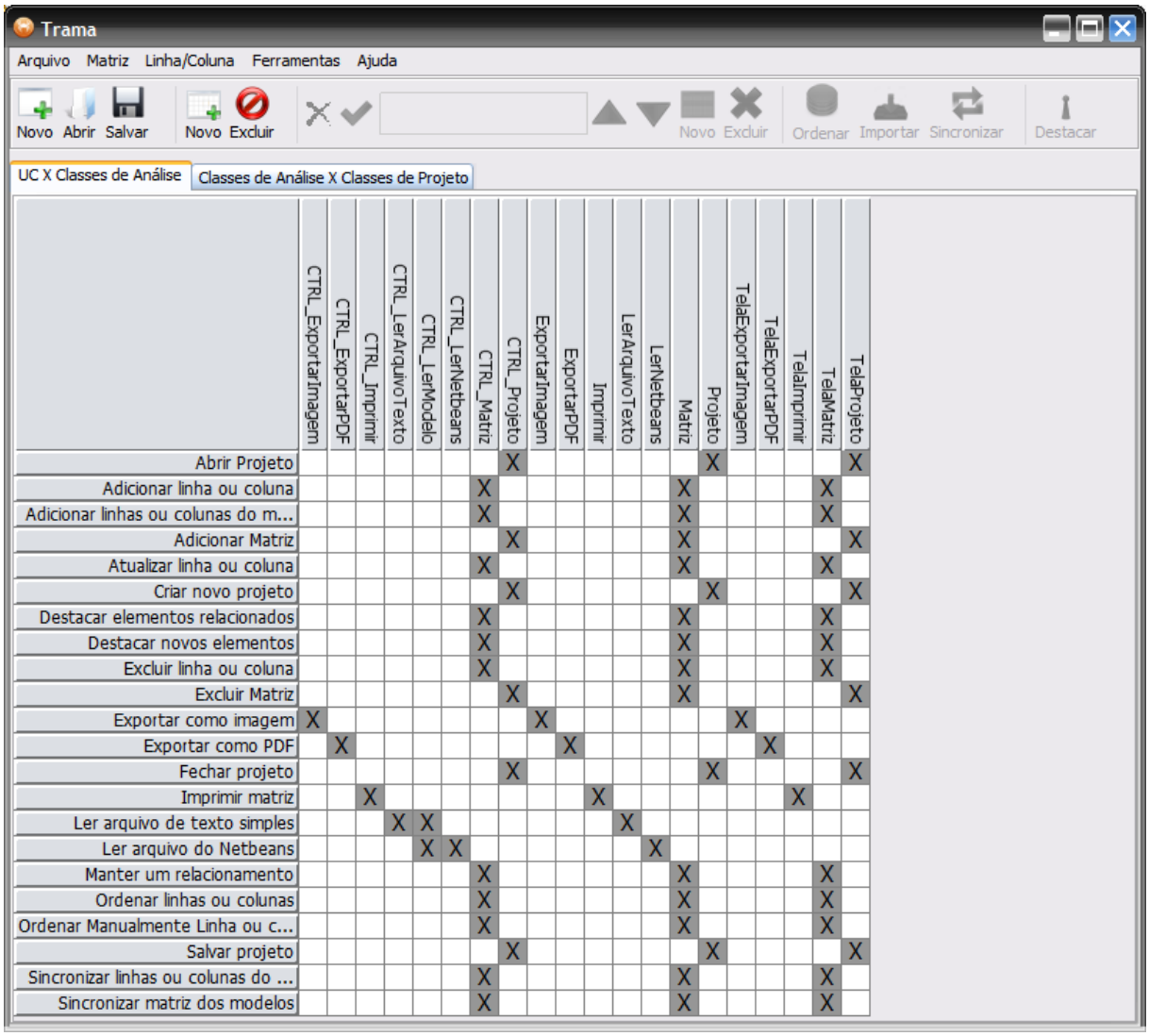


Fonte: DEVMEDIA, 2010.

Outra opção *open source* pode ser encontrada na ferramenta Trama, disponibilizada sob a licença *GNU*. O software Trama procura oferecer suporte e auxílio à criação, edição e checagem de relacionamentos entre os diversos artefatos dos diagramas UML. Pode ler os arquivos gerados pelo *plugin* UML do *Netbeans*, utilizando um sistema de *plugins* (MARMITT, 2008).

O software pode ser baixado no site http://trama.googlecode.com/files/Trama.zip.

Figura 8: Trama – Uma Ferramenta de Auxílio ao Trabalho com Matrizes de Rastreabilidade



Fonte: Google Code, 2011.

4.3 Prática de Utilização da MRR

Como visto no capítulo anterior, a rastreabilidade gera custos e mais esforços de trabalho para o desenvolvimento do software. Por isso, deve ser tomada a decisão da utilização ou não da mesma para o processo e, em caso positivo, qual o nível de detalhe estará envolvido no estabelecimento dos *links* de rastreamento (MARQUES, 2010).

Os problemas apresentados na RR se refletem quando uma organização busca utilizar alguma ferramenta para a execução desta atividade. Muitos desenvolvedores não têm a cultura de utilizar a MRR ou, quando a usam, não a atualizam com a mesma freqüência das mudanças que ocorrem nos requisitos, o que normalmente impacta na qualidade final do produto.

Linscomb (2003) registrou nunca ter visto uma organização com nível de maturidade dois no *Capability Maturity Model Integration* (CMMI) que realizasse um bom trabalho com a utilização da MRR de forma completa, ainda que esta seja uma das práticas requeridas para obtenção de tal nível. Nota-se não apenas um problema técnico relacionado à manutenção das matrizes, mas também cultural dentro das empresas.

Para Backes (2008), o fato de não existir uma padronização de armazenamento ou representação dos *links* de rastreabilidade, faz com que a utilização das MRR continue sendo uma tarefa desafiadora e cara, mesmo quando se trata de projetos pequenos ou de médio porte. Semelhantemente, Sommerville (2003) afirma que as MRR podem ser utilizadas quando o número de requisitos a ser gerenciado é pequeno, mas a sua manutenção torna-se dispendiosa ao tratar de sistemas com grande número de requisitos.

É possível dividir os desenvolvedores em duas classes de usuários em relação ao uso da rastreabilidade nos projetos em que estão envolvidos (RAMESH e JARKE, 1999):

- ✓ ***Low-end users*** – Possuem poucos anos de experiência em rastreabilidade e enxergam a atividade como algo obrigatório imposto pelos responsáveis do projeto ou como cumprimento de normas.

- ✓ ***High-end users*** – Possuem maior tempo de experiência em rastreabilidade e a enxergam como uma grande oportunidade de alcançar a satisfação do cliente, além da criação do conhecimento durante todo o ciclo de vida do sistema. Conforme cresce a complexidade do sistema, mais reconhecida fica a importância do uso da rastreabilidade nos projetos.

Para enfrentar esses problemas, é preciso que seja criada uma cultura organizacional no que se refere à utilização de um processo de desenvolvimento de software de forma sistemática, disciplinada e quantitativa. É necessário que todos os envolvidos entendam que a criação de documentos, entre eles a MRR, serve exatamente para diminuir o retrabalho gerado pelas inconsistências encontradas nos estágios mais avançados do desenvolvimento do produto.

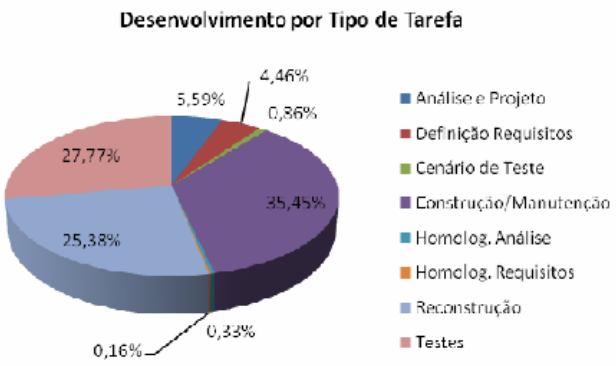
Além disso, muitas ferramentas CASE têm sido desenvolvidas com o intuito de automatizar a manutenção das MRR (SOMMERVILLE, 2003), como aquelas descritas em 4.2. Porém, ainda há a necessidade da intervenção manual para atualização de determinados *links* que vinculam os diversos relacionamentos dos requisitos, pois os softwares não têm a capacidade de interpretação da linguagem natural que os humanos possuem para efetuarem os elos de forma correta e precisa (BACKES, 2008).

Um estudo de caso realizado por Schumacher e Souza (2010) mostrou que a utilização da MRR em um processo de manutenção de um produto de software resultou na otimização deste processo, "(...) reduzindo o tempo da execução dos ciclos de manutenções e minimizando esforços das etapas de testes de software e reconstrução". (SCHUMACHER e SOUZA, 2010, p. 8).

As Figuras 9 e 10 mostram os resultados sintéticos deste estudo de caso. A primeira apresenta os resultados do processo realizado sem a utilização do GR e as MRR. A segunda já demonstra os resultados com o uso MRR como ferramenta do processo, além da aplicação da GR.

Figura 9: Resultados Sintéticos sem a Utilização da Gestão de Requisitos e Matriz de Rastreabilidade

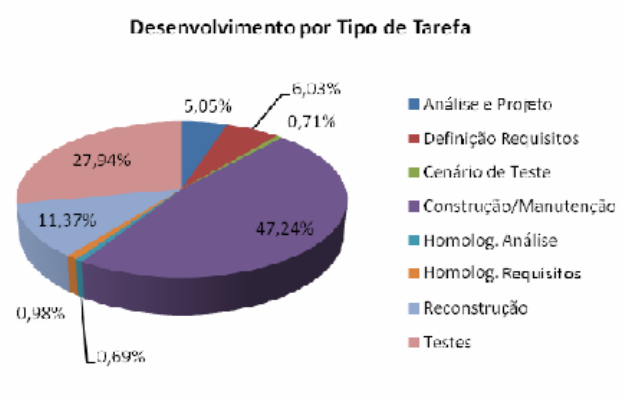
Tipo de Tarefa	Horas Trabalhadas
Análise e Projeto	102,38
Definição Requisitos	81,77
Homolog. Análise	6
Homolog. Requisitos	3
Construção/Manutenção	649,61
Cenário de Teste	15,75
Testes	508,84
Reconstrução	465,02
Total	**1832,37**



Fonte: Schumacher e Souza, 2010.

Figura 10: Resultados Sintéticos com a Utilização da Gestão de Requisitos e Matriz de Rastreabilidade

Tipo de Tarefa	Horas Trabalhadas
Análise e Projeto	87,92
Definição Requisitos	104,83
Homolog. Análise	11,92
Homolog. Requisitos	17
Construção/Manutenção	821,65
Cenário de Teste	12,33
Testes	485,98
Reconstrução	197,73
Total	**1739,36**



Fonte: Schumacher e Souza, 2010.

Notou-se também o aumento na motivação e confiança da equipe envolvida, devido à maior segurança e qualidade mostradas na execução do processo. Assim, justifica-se o trabalho gerado na utilização e manutenção da MRR desde os processos iniciais do desenvolvimento do projeto, haja vista que essa ferramenta, quando bem usada, garante a conformidade àquilo que foi especificado, permitindo um gerenciamento eficaz do ciclo de vida dos requisitos e seus respectivos relacionamentos.

5. METODOLOGIA

Para a realização deste trabalho foi adotada a pesquisa teórica, sendo realizada a consulta a fontes bibliográficas, como livros, teses e dissertações relacionadas ao tema da Rastreabilidade de Requisitos e a utilização da Matriz de Rastreabilidade. Também foram realizadas pesquisas em fontes paralelas, como as localizadas na rede Internet e em revistas especializadas, com o intuito de angariar informações a respeito das técnicas ligadas ao Gerenciamento de Requisitos e os resultados advindos da sua utilização no mercado de software nacional e internacional.

6. CONCLUSÃO

O desenvolvimento de software tem crescido em número e complexidade no mundo inteiro. As organizações precisam atender as constantes solicitações dos clientes dentro de prazos e custos que não refletem a realidade do desenvolvimento do produto de software, o que acarreta em soluções com baixa qualidade, entregues fora do tempo determinado e com o orçamento acima do especificado, causando a insatisfação dos clientes.

Para melhorar o processo de desenvolvimento, as empresas têm buscado se adequar a normas e padrões nacionais e internacionais, como o MPS.BR e o CMMI, a fim de que estas as auxiliem a seguir um modelo específico que lhes garanta um produto final de qualidade superior, atendendo aos custos e prazos fixados. Dentre as melhores práticas utilizadas, a Engenharia e o Gerenciamento de Requisitos ditam alguns procedimentos com o intuito de ajudar as corporações a assumirem de forma eficaz o controle e a gestão das constantes mudanças que ocorrem em projetos de software, especialmente nos requisitos outrora levantados e documentados, sendo de suma importância para a conclusão de um produto que satisfaça as especificações dos clientes.

Este trabalho apresentou os conceitos da Engenharia e Gestão de Requisitos, focando-se na importância da implantação do processo de Rastreabilidade de Requisitos e o uso da Matriz de Rastreabilidade como uma importante ferramenta para alcançar o controle dos requisitos, seus relacionamentos e suas modificações. A adoção de um método formal de gerenciamento de requisitos permite que todas as propostas de mudanças sejam tratadas de forma consistente e as variações no documento de requisitos sejam efetuadas de maneira controlada (SOMMERVILLE, 2003).

Embora existam certas dificuldades em relação à manutenção dos *links* de relacionamento das matrizes e o investimento atrelado ao seu uso seja alto, vários estudos mostram a importância de se implantar a Rastreabilidade de Requisitos e a Matriz de Rastreabilidade para a diminuição do retrabalho, a redução dos custos e a melhoria da qualidade.

Os softwares destinados à criação das MRR têm sido aperfeiçoados e sua adoção é uma opção eficaz no atendimento a projetos complexos, resolvendo boa parte dos problemas de manutenção de *links* com suporte automatizado, mas necessitando ainda da interferência humana para verificação de inconsistências. Além disso, é preciso que as empresas invistam

no treinamento das suas equipes para a utilização plena destes instrumentos e disseminem a cultura de um trabalho centrado em processos bem definidos para alcançarem os seus objetivos organizacionais.

Espera-se que este trabalho sirva como base para o conhecimento dos métodos que envolvem a GR e incentive as organizações a adotarem estas atividades em seus processos de desenvolvimento, entendendo que este é um investimento realizado para obter o retorno de médio a longo prazo e que trará benefícios concretos ao atenderem os requisitos dos clientes de forma completa, contribuindo na melhoria da qualidade do software brasileiro e fortalecendo suas marcas no mercado pelo trabalho realizado com excelência.

7. REFERÊNCIAS

BACKES, Jerônimo. Rastreabilidade Semi-Automática Através do Mapeamento de Entidades. 2008. 96 f. Dissertação – Universidade Federal do Rio Grande do Sul, Porto Alegre. (On-line). Disponível: http://www.lume.ufrgs.br/bitstream/handle/10183/15308/000677241.pdf?sequence=1, (30 de janeiro de 2011).

BATISTA, E. A. e CARVALHO, A. M. B. R. (2003). *Uma Taxonomia Facetada para Técnicas de Elicitação de Requisitos*. Anais do WER03 – Workshop em Engenharia de Requisitos. (On-line). Disponível: http://wer.inf.puc-rio.br/wer03/artigos/edinelson_batista.pdf, (25 de janeiro de 2011).

CETIC (2010). *Pesquisa sobre o uso das tecnologias da informação e da comunicação no Brasil 2009*. (On-line). CGI.BR. Disponível: http://www.cetic.br/tic/2009/index.htm, (29 de dezembro de 2010).

CHRISTEL, M. G. e KANG, K. C. *Issues in Requirements Elicitation,* 1992. In: PRESSMAN, Roger S. *Engenharia de Software*. 6. ed. São Paulo: Makron Books, 2006, p. 118 e 119.

DEVMEDIA (2010). *Controla – Ferramenta de Apoio ao Processo de Desenvolvimento de Software em Pequenas Empresas*. (On-line). Disponível: http://www.devmedia.com.br/articles/viewcomp.asp?comp=283, (13 de fevereiro de 2011).

DINIZ, Alfraino de Souza. *PROREQ – Um Guia Facilitador para a Implantação dos Processos de Gestão de Requisitos*. 2007. 92 f. Dissertação – ICMC-USP, São Carlos. (On-line). Disponível: http://www.teses.usp.br/teses/disponiveis/55/55134/tde-31032008-161043/pt-br.php, (30 de janeiro de 2011).

EGYED, Alexander (2005). *Tailoring Software Traceability to Value-Based Needs.* (On-line). Disponível: http://www.alexander-egyed.com/publications/Tailoring_Software_Traceability_to_Value-Based_Needs.pdf, (13 de fevereiro de 2011).

EGYED, Alexander e GRÜNBACHER, Paul (2005). *Supporting Software Understanding with Automated Requirements Traceability*. (On-line). Disponível: http://www.alexander-egyed.com/publications/Supporting_Software_Understanding_with_Automated_Requirements_Traceability.pdf, (04 de janeiro de 2011).

FELICI, Massimo. (2004). *Observational Models of Requirements Evolution.* Doctoral Thesis. (On-line). Disponível: http://homepages.inf.ed.ac.uk/mfelici/doc/IP040037.pdf, (02 de janeiro de 2011).

GOTEL, O. C. Z. e FINKELSTEIN, A. C. W. (1994). *An Analysis of the Requirements Traceability Problem.* In: Proceedings of the First International Conference on Requirements Engineering, Colorado Springs, Co, 1994, 94-101. (On-line). Disponível: eprints.ucl.ac.uk/749/1/2.2_rtprob.pdf, (04 de janeiro de 2011).

HAZAN, Claudia e LEITE, J. C. S. P. (2003) *Indicadores para a Gerência de Requisitos*. Anais do WER03 – Workshop em Engenharia de Requisitos. (On-line). Disponível: http://wer.inf.puc-rio.br/wer03/artigos/claudia_hazan.pdf, (02 de janeiro de 2011).

IBM (2011). *Rational RequisitePro – A Requirements Management Tool. (On-line).* Disponível: http://www-01.ibm.com/software/awdtools/reqpro/, (13 de fevereiro de 2011).

IBM (2011). *Rational DOORS – Features and Benefits. (On-line).* Disponível: http://www-01.ibm.com/software/awdtools/doors/features/index.html, (13 de fevereiro de 2011).

IIBA (2009). *A Guide to the Business Analysis Body of Knowledge (BABOK Guide)*. Version 2.0. Toronto: International Institute of Business Analysis, 2009. (On-line). Disponível em: http://books.google.com.br/books?id=5ZLikcDw68YC&printsec=frontcover&dq=babok+2.0&cd=1#v=onepage&q&f=false, (30 de janeiro de 2011).

LEITE, J. C. S. P. *Gerenciando a Qualidade de Software com Base em Requisitos*. In: ROCHA, A. R.; MALDONADO, J. C. e WEBER, K. C. *Qualidade de Software: Teoria e Prática*. Prentice Hall, 2001, Capítulo 17.

LINSCOMB, Dennis (2003). *Requirements Engineering Maturity in the CMMI*. (On-line). Disponível: http://www.crosstalkonline.org/storage/issue-archives/2003/200312/200312-Linscomb.pdf, (27 de fevereiro de 2011).

LUDWIG CONSULTING SERVICES, LLC (2008). *Requirements Traceability Matrix*. (On-line). Disponível: http://www.jiludwig.com/RTM.html, (13 de fevereiro de 2011).

MARMITT, Fabio (2008). *Manual Trama*. (On-line). Disponível: http://trama.googlecode.com/files/Manual%20Trama.pdf, (13 de fevereiro de 2011).

MARQUES, Ernani. *CBAP®PREP - Certified Business Analysis Professional: Seu guia para o Certificado Profissional em Análise de Negócio*. São Paulo: All Print, 2010.

MELHORIA DE PROCESSO DO SOFTWARE BRASILEIRO (2009). *Guia de Implementação – Parte 1: Fundamentação para Implementação do Nível G do MR-MPS.* (On-line). Disponível: www.softex.br/mpsbr/_guias/guias/MPS.BR_Guia_de_Implementacao_Parte_1_2009.pdf, (02 de janeiro de 2011).

PRESSMAN, Roger S. *Engenharia de Software*. 6. ed. São Paulo: Makron Books, 2006.

RAMESH, Bala e JARKE, Matthias. (1999). *Towards Reference Models for Requirements Traceability*. IEEE Transactions on Software Engineering, vol. 27, no. 1, pp. 58-93, Jan. 2001. (On-line). Disponível: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.40.6602&rep=rep1&type=pdf, (04 de janeiro de 2011).

SAYÃO, Miriam e LEITE, J. C. S. P. *Rastreabilidade de Requisitos*. 2005. 22 f. Monografia (Departamento de Informática) – PUC-RIO, Rio de Janeiro. (On-line). Disponível: http://www.dbd.puc-rio.br/depto_informatica/05_20_sayao.pdf, (29 de dezembro de 2010).

SCHUMACHER, Marcelo e SOUZA, V. C. (2010). *Gerenciamento de Requisitos como Alternativa de Otimização na Manutenção de Softwares já Implantados: Um Estudo de Caso*. (On-line). Disponível: http://www.unisinos.br/eventos/uniinfo/images/gerenciam_de_requisitos_como_alternativa_de_otimizaca.pdf, (27 de fevereiro de 2011).

SERPRO. *Processo SERPRO de Desenvolvimento de Soluções (PSDS)* (2002). In: HAZAN, Claudia e LEITE, J. C. S. P. (2003) *Indicadores para a Gerência de Requisitos*. Anais do WER03 – Workshop em Engenharia de Requisitos, p. 3. (On-line). Disponível: http://wer.inf.puc-rio.br/wer03/artigos/claudia_hazan.pdf, (25 de janeiro de 2011).

SOFTWARE TECHNOLOGY SUPPORT CENTER (2003). *Guidelines for Successful Acquisition and Management of Software-Intensive Systems: Weapon Systems, Command and Control Systems, Management Information Systems - Condensed Version 4.0 February 2003*. (On-line). Disponível: http://www.stsc.hill.af.mil/resources/tech_docs/gsam4/GSAMv4_1.ZIP, (29 de dezembro de 2010).

SOFTWARE TESTING (2011). *Traceability Matrix from Software Testing Perspective*. (On-line). Disponível: http://www.softwaretestingtimes.com/2010/04/traceability-matrix-from-software.html, (27 de fevereiro de 2011).

SOMMERVILLE, Ian. *Engenharia de Software*. 6. ed. São Paulo: Pearson Addison Wesley, 2003.

SWEBOK (2004). *Guide to the Software Engineering Body of Knowledge 2004 Version.* (On-line). Disponível: http://www.computer.org/portal/web/swebok/htmlformat, (02 de janeiro de 2011).

THE STANDISH GROUP (2009). *Chaos Summary 2009 – The 10 Laws of Chaos*. (On-line). Disponível: http://www.portal.state.pa.us/portal/server.pt/document/690719/chaos_summary_2009_pdf, (29 de dezembro de 2010).