



PUC

ISSN 0103-9741

Monografias em Ciência da Computação
nº 20/05

Rastreabilidade de Requisitos

Miriam Sayão
Julio Cesar Sampaio do Prado Leite

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO
RUA MARQUÊS DE SÃO VICENTE, 225 - CEP 22453-900
RIO DE JANEIRO - BRASIL

Rastreabilidade de Requisitos*

Miriam Sayão ** Julio Cesar Sampaio do Prado Leite

miriam@inf.puc-rio.br, julio@inf.puc-rio.br

Abstract. The Requirements Management, which is associated to the quality of software development process, it is based on requirements traceability. In this work we deal with requirements traceability, aiming to present some associated aspects to its use in the software development and evolution process. We also seek to make the reader aware of the implications of traceability for the product quality, associating traceability to both technical and managerial aspects. We present two traceability reference models, and heuristics that can guide the development team in a traceability model definition process to the system to be developed.

Keywords: requirements traceability, requirements management, software quality, software development process.

Resumo. A Gerência dos Requisitos, processo associado à qualidade do desenvolvimento de software, é baseada na rastreabilidade de requisitos. Neste trabalho tratamos da rastreabilidade de requisitos, buscando apresentar aspectos associados ao seu uso no contexto do desenvolvimento e da evolução de sistemas de software. Visamos também conscientizar o leitor das implicações da rastreabilidade para a qualidade do produto sendo desenvolvido, associando a rastreabilidade tanto a aspectos técnicos como gerenciais. Apresentamos dois modelos de referência e heurísticas para apoiar a equipe de desenvolvimento no processo de definição do modelo de rastreabilidade para o sistema a ser desenvolvido.

Palavras-chave: rastreabilidade de requisitos, gerencia de requisitos, qualidade de software, processo de desenvolvimento de software.

* Trabalho patrocinado pela PUC-RS e pelo Ministério de Ciência e Tecnologia da Presidência da República Federativa do Brasil.

** Também vinculada ao quadro docente da Pontifícia Universidade Católica do Rio Grande do Sul (PUC-RS)

Responsável por publicações:

Rosane Teles Lins Castilho
Assessoria de Biblioteca, Documentação e Informação
PUC-Rio Departamento de Informática
Rua Marquês de São Vicente, 225 - Gávea
22453-900 Rio de Janeiro RJ Brasil
Tel. +55 21 3114-1516 Fax: +55 21 3114-1530
E-mail: bib-di@inf.puc-rio.br
Web site: <http://bib-di.inf.puc-rio.br/techreports/>

Sumário

1	Qualidade de Software	1
2	Gerência de Requisitos	2
2.1	Rastreabilidade	3
2.2	Impactos da rastreabilidade num projeto de desenvolvimento de software	4
2.3	Tipos de elos de rastreabilidade	7
2.3.1	Meta-modelo proposto por Ramesh&Jarke	7
2.3.2	Meta-modelo proposto por Toranzo	9
2.3.3	Comparação das propostas de Ramesh&Jarke e de Toranzo	10
3	Prática da rastreabilidade de requisitos	11
3.1	Técnicas e ferramentas: suporte à questão da rastreabilidade	12
3.2	Processo associado à rastreabilidade	15
3.3	Heurísticas para definição de um modelo de rastreabilidade	16
4	Conclusões	17
4.1	Contribuições deste artigo	17
4.2	Aspectos em aberto	18
	Referências Bibliográficas	18

1 Qualidade de Software

A preocupação com a qualidade de software levou à criação, no início da década de 90, de um conjunto de conceitos visando medir a maturidade da organização e a capacidade de gerenciamento da engenharia de software, envolvendo processos, qualidade e técnicas de gerenciamento. Para avaliar o processo de desenvolvimento de software, vários modelos de processo foram desenvolvidos, sendo que os mais aplicados até hoje são CMM (*Capability Maturity Model*) [Paulk95] [Paulk93] e ISO 9001 [Weber97]. CMMI (*Capability Maturity Model Integration*) surgiu visando à integração de diversos modelos de qualidade e apresenta uma nova versão para o SW-CMM, que se propõe substituir. CMM e CMMI são mais populares entre empresas que visam o mercado americano, e para o mercado europeu, a certificação para a norma ISO 9001 (uma versão da ISO 9000 específica para serviços) normalmente é mais aceita para medir a qualidade e maturidade do processo de software.

CMM é um modelo projetado em níveis, de forma que os estágios de mais baixo nível provêem progressivamente fundamentos para os estágios de mais alto nível. Para cada nível de maturidade, existem associadas áreas-chave de processo, que identificam os requisitos necessários para atingir cada um dos níveis de maturidade. CMMI apresenta duas modalidades: contínua e em níveis (*staged*), sendo que esta última é uma abordagem semelhante ao CMM, com inclusão de KPAs (*Key Process Areas*, que no CMMI são chamadas de *Process Areas*) e algumas alterações.

No nível 2 de maturidade, conforme definido no CMM, são necessários processos básicos de gerência de projeto para planejamento e acompanhamento de custos, cronogramas, prazos e funcionalidades. Este nível é denominado de *repetível* e possui as seguintes KPAs: Gerência de Requisitos, Planejamento de Projeto de Software, Supervisão e Acompanhamento do Projeto de Software, Garantia de Qualidade de Software, Gerenciamento de Sub-contrato de Software e Gerenciamento de Configuração de Software. Podemos relacionar a meta da KPA Gerência de Requisitos à *estabelecer uma visão comum entre o cliente e a equipe de projeto em relação aos requisitos que serão atendidos pelo projeto de software* [CMM01]. As metas associadas a esta KPA definem a necessidade de documentar e controlar os requisitos (via *baseline*¹) e a manutenção de planos, artefatos e atividades consistentes com os requisitos alocados ao software.

O nível 2, no CMMI *Staged*, é denominado de *Gerenciável* e apresenta as seguintes *Process Areas* (PA): Gerência de Requisitos, Planejamento de Projeto, Monitoração e Controle de Projeto, Gestão de Contratos com Fornecedores, Medida e Análise, Garantia da Qualidade de Produto e Processo e Gerência de Configuração. O propósito da Gerência de Requisitos é gerenciar os requisitos (considerando artefatos e componentes) e identificar inconsistências entre requisitos e os planos e artefatos do software. Podemos observar que a Gerência de Requisitos está presente e possui atividades equivalentes tanto no CMM como no CMMI; além disso, é consenso que as KPAs do nível 2 do CMM estão fortemente relacionadas à ISO 9001 [Paulk95a].

¹ a *baseline* de requisitos é entendida como o conjunto de requisitos que compõe o documento de requisitos de software (Software Requirements Specification - SRS). Estes requisitos são gerados após as várias fases do processo de requisitos (elicitação, modelagem, verificação e validação) e compõem a base para o desenho e a implementação. Alterações em requisitos inseridos numa *baseline* requerem justificativas e aprovação.

A diferença trazida pelo CMMI em relação a requisitos está no nível 3, denominado de *Definido* [CMMI02]. No nível 3 os processos devem estar bem caracterizados e descritos em padrões, métodos, procedimentos e ferramentas. Este nível apresenta a área de Desenvolvimento de Requisitos (*Requirements Development*), cujo propósito é produzir e analisar requisitos relativos ao cliente, ao produto e aos componentes do software. As metas específicas desta área abrangem, portanto, os processos de elicitação, modelagem, verificação e validação dos requisitos a serem atendidos pelo software. Esta área tem como meta genérica institucionalizar um processo para a obtenção e evolução dos requisitos.

Espera-se que a melhoria da qualidade do processo de desenvolvimento, com a adoção de um processo de qualidade como um dos já referidos (CMM, CMMI ou ISO), resulte em aumento de produtividade, redução de custo, maior qualidade e competitividade no produto final [MacFarlane95] [Dion93].

2 Gerência de Requisitos

Atualmente tem-se a convicção que mudanças em requisitos ao longo do processo de desenvolvimento de software fazem parte do processo. Entre os motivos para que ocorram alterações em requisitos encontramos necessidades não identificadas inicialmente, alterações no contexto onde o sistema está inserido, correção de erros detectados por processos de qualidade ou mesmo novas perspectivas por parte dos interessados². Alterações em requisitos podem implicar em mudanças em artefatos de desenho, de código, casos de testes. Todas estas mudanças devem ser acompanhadas, de forma a garantir que todos os artefatos afetados por uma alteração de requisitos sejam corrigidos. As atividades ligadas ao processo de acompanhamento de mudanças são denominadas de Gerência de Requisitos; este é o entendimento comum que encontramos tanto na prática quanto na literatura. No entanto, acreditamos haver certa confusão de conceitos e consideramos necessário fazer aqui o que consideramos uma distinção importante entre a Gerência dos Requisitos e o que denominamos de Gerência por Requisitos.

Gerência dos Requisitos está associada ao processo de acompanhar a evolução dos requisitos ao longo do processo de desenvolvimento; entre outras atividades, implica em manter registro do *status* de cada requisito em relação ao desenvolvimento, em relação a modificações aceitas e *rationale*³ associado. Já a Gerência por Requisitos, comumente denominada de Gerência de Requisitos, está associada ao processo de controlar todo o processo de desenvolvimento tendo como referência a *baseline* de requisitos; os requisitos são a base para a definição da arquitetura do sistema, para a implementação propriamente dita, para geração dos casos de testes e para validação do sistema junto ao usuário.

O conjunto de atividades que compõem o processo de Gerência dos Requisitos é encontrado, no CMMI, na área de Desenvolvimento de Requisitos. Esta área, surpreendentemente, é localizada no nível 3 de maturidade de processo, enquanto que a área de Gerência de Requisitos está situada no nível 2 de maturidade. Isto significa que a organização que busque certificação de qualidade pelo modelo CMMI deverá inicialmente focar sua atenção em gerenciar requisitos e identificar inconsistências entre requisitos e

² denominamos interessados aos *stakeholders* do processo, aqueles que possuem algum interesse no sistema a ser desenvolvido ou que participam no processo de desenvolvimento

³ *rationale* indica as razões, motivações ou intenções para uma determinada ação

os planos e artefatos do software [CMMI02]. Mas como são obtidos os requisitos que devem ser gerenciados? Através das atividades já citadas de elicitação, modelagem, verificação e validação, que estão situadas na área de Desenvolvimento de Requisitos, no nível 3 de maturidade. Ou seja, aqui existe uma discrepância na distribuição das atividades nos níveis de maturidade, pois um gerenciamento de qualidade pressupõe requisitos de qualidade. Esta discrepância já foi apontada por Linscomb [Linscomb03], que apresenta uma proposta de redistribuição das atividades associadas a requisitos entre os dois níveis iniciais do modelo CMMI.

A Gerência por Requisitos (ou Gerência de Requisitos) trata de um aspecto fundamental e crítico em qualquer processo de software: estabelecer uma visão comum entre o cliente e a equipe de projeto em relação aos requisitos que serão atendidos pelo software [CMM01]. As metas a serem atingidas pela gerência por requisitos são: (i) documentar e controlar os requisitos alocados para estabelecer uma baseline para uso gerencial e da engenharia de software e (ii) manter planos, artefatos e atividades de software consistentes com os requisitos alocados [Santander00]. A baseline referida, segundo Leite [Leite97], deve incorporar descrições em linguagem natural. Uma possível estrutura, segundo Leite [Leite97] é aquela composta de: um modelo de léxico (apoio à elicitação da linguagem do macrosistema), um modelo básico (diagramas Entidade-Relacionamento para requisitos externos), modelo de cenários (situações de comportamento da aplicação em momentos específicos), um modelo de hipertexto e um modelo de configuração (estes dois últimos são na verdade serviços de suporte aos demais componentes da baseline).

As atividades ligadas ao Gerenciamento por Requisitos incluem o controle de mudanças (registro e análise de novas solicitações ou de alterações em requisitos já definidos), identificação de relacionamentos entre requisitos, suas origens e componentes que os implementam (o que é possibilitado pelo rastreamento de requisitos), análises de impacto (estimativas de impactos em custos e prazos devido a novas solicitações), controle de versões dos diferentes artefatos e acompanhamento do estado dos requisitos.

Indispensável à tarefa de Gerência por Requisitos (gerenciamento por requisitos) é a disponibilidade de facilidades de rastreamento, que iremos detalhar a seguir. Detalharemos o rastreamento de requisitos, que é vital ao gerenciamento do processo [Järke98]; requisitos não podem ser efetivamente gerenciados sem rastreabilidade. *Um requisito é rastreável se é possível descobrir quem sugeriu o requisito (a fonte), por que o requisito existe (rationale), que outros requisitos estão relacionados a ele (dependência entre requisitos) e como o requisito se relaciona com outras informações tais como desenho do sistema, implementação e documentação do usuário* [Sommerville98].

2.1 Rastreabilidade

O rastreamento de requisitos é utilizado para prover relacionamentos entre requisitos, arquitetura e implementação final do sistema e possibilita uma adequada compreensão dos relacionamentos de dependência entre requisitos e através dos artefatos de requisitos, de arquitetura e de implementação. A rastreabilidade pode ser implementada por um conjunto de elos ou ligações (*links*) entre requisitos inter-relacionados, entre requisitos e suas fontes, e entre requisitos e os componentes que os implementam.

A rastreabilidade de requisitos tem sido identificada na literatura como fator de qualidade, característica que um sistema pode possuir e incluir como requisito não funcional [Ramesh01]. Diversos trabalhos apresentam a rastreabilidade como um dos mais importantes pré-requisitos para o desenvolvimento de software de qualidade

[Ramesh01] [Jarke98] [Kotonya98] [Palmer97] [Spanoudakis02] [Toranzo99]. As principais características da rastreabilidade são apresentadas a seguir.

Rastreabilidade pode ser definida como sendo a técnica usada para prover relacionamento entre requisitos, arquitetura e implementação final do sistema [Edwards91]. Ela auxilia na compreensão dos relacionamentos existentes entre requisitos do software ou entre artefatos de requisitos, arquitetura e implementação. Esses relacionamentos permitem aos projetistas mostrar que o projeto atende aos requisitos. A rastreabilidade também apóia a detecção precoce daqueles requisitos não atendidos pelo software [Palmer97].

A rastreabilidade de requisitos pode ser vista como a habilidade de acompanhar e descrever a vida de um requisito, em ambas as direções; pré-rastreabilidade documenta a o contexto a partir do qual emergem os requisitos; pós-rastreabilidade vincula os requisitos ao desenho do sistema e sua implementação [Davis93]. A Figura 1 mostra como as ligações possibilitam acompanhar a "vida" de um requisito, em ambas as direções.

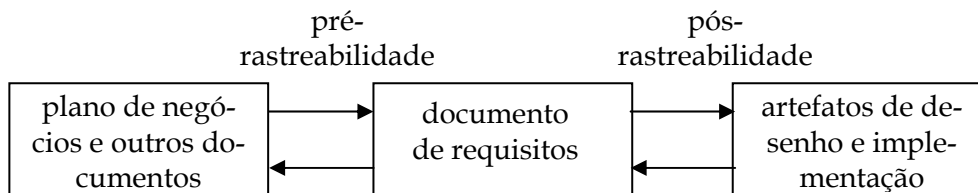


Fig. 1 – Rastreabilidade de Requisitos

No primeiro tipo temos a rastreabilidade *forward-to* (para frente), que liga documentos obtidos no processo de elicitação a requisitos relevantes, e a rastreabilidade *backward-from* (para trás), que liga requisitos às suas fontes. No segundo tipo temos rastreabilidade *forward-from*, que liga requisitos a artefatos de desenho e implementação e rastreabilidade *backward-to*, que liga artefatos de desenho e implementação de volta a requisitos.

Rastreabilidade também pode ser identificada como a habilidade de descobrir a história de toda característica do sistema, dado que os impactos de mudanças nos requisitos podem ser identificados [Hamilton91]; ou ainda como a habilidade de permitir que mudanças em qualquer artefato - requisitos, especificação e implementação - sejam rastreadas através do sistema [Greenspan78].

Rastreabilidade de requisitos é uma característica de sistemas nos quais requisitos são claramente ligados às suas fontes e aos artefatos criados durante o ciclo de vida de desenvolvimento do sistema baseado nesses requisitos. Rastreabilidade de requisitos estabelece um elo entre mudanças das necessidades dos usuários e evolução dos sistemas de computação, sendo uma base para o gerenciamento do conhecimento organizacional [Ramesh01].

2.2 Impactos da rastreabilidade num projeto de desenvolvimento de software

Ao longo do tempo, diversos modelos genéricos foram propostos para guiar o processo de desenvolvimento de um sistema de software. O modelo conhecido por cascata (*waterfall*), um dos primeiros a ser proposto, apresentava o problema de não prever tratamento para alterações nos requisitos durante o próprio processo de desenvolvimento. Este processo é dinâmico: após a definição de requisitos segue-se o desenho e a imple-

mentação do sistema, e a chegada de novos requisitos ou alterações naqueles já definidos pode justificar a retomada de tarefas associadas à fase anterior. Levantamentos indicam que em projetos com relativa estabilidade esperam-se alterações da ordem de 1% ao mês [Hammer98]; outros relatos apontam para um percentual de alteração de até 50% dos requisitos antes que o sistema seja colocado em produção [Kotonya98]. Outra fonte de alterações são os problemas identificados quando tem início a fase de implementação propriamente dita: nesta etapa podem emergir problemas na arquitetura do sistema e o processo de desenho ser retomado.

A *baseline* de requisitos, construída durante o processo de requisitos, evolui durante o processo de desenvolvimento: alterações devem ser incorporadas à *baseline* para que esta reflita sempre o conjunto de requisitos acordado entre cliente e desenvolvedores [Leite97]. O modelo - registrado na *baseline* de requisitos - deve evoluir em consonância com o sistema e o Universo de Informações (UdI)⁴. A *baseline* é referência para todo o processo de desenvolvimento, guiando a equipe nas várias atividades relacionadas à geração do sistema desejado. Na Figura 2 [Leite97] mostramos que uma *baseline* é contextualizada num eixo com duas dimensões: o eixo horizontal mostra a evolução ao longo do processo de desenvolvimento, e o eixo longitudinal registra a evolução em cada etapa do processo (por exemplo a *baseline* da especificação).

Uma das principais fontes geradoras de alterações é o próprio conjunto de interessados, que inicialmente não tem muita clareza nos objetivos a alcançar, e à medida que o desenvolvimento evolui vai descobrindo novas possibilidades ou funcionalidades a incluir no projeto. Um adequado gerenciamento de requisitos necessita da rastreabilidade dos requisitos para controlar e documentar as modificações; além disso, a rastreabilidade ajuda a estimar variações em cronogramas e alterações de custos no desenvolvimento (análise de impacto [Palmer97] [Pinheiro04] [Davis93] [Dömges98] [Egyed04]).

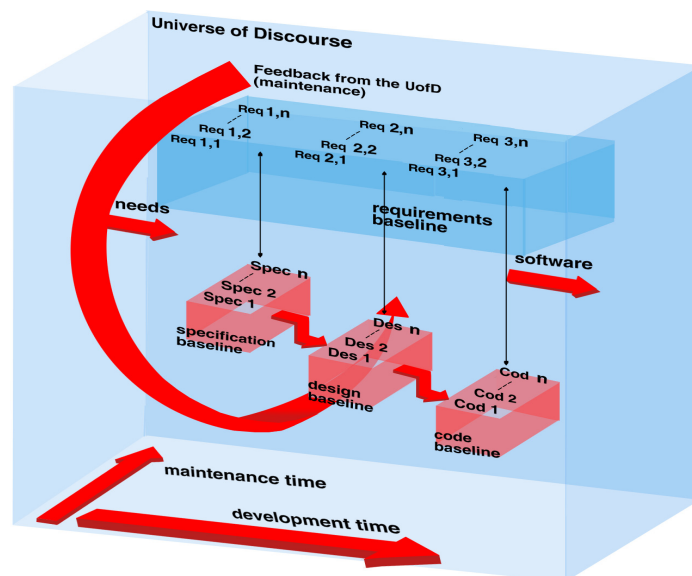


Fig. 2: Evolução da *baseline* de requisitos e do processo de desenvolvimento [Leite97]

A rastreabilidade pode auxiliar gerentes e desenvolvedores em várias situações, entre as quais destacamos:

⁴ contexto geral no qual o software será desenvolvido e operado; inclui todas as fontes de informação e pessoas relacionadas ao software. Também identificado como Universo do Discurso.

- verificação da alocação de requisitos a componentes do software: a avaliação dos elos de rastreabilidade de requisitos a artefatos de desenho e implementação identifica requisitos ainda não alocados ou implementados;
- resolução de requisitos em conflito: durante o processo de elicitação é comum que diferentes representantes do cliente ou usuário interajam com a equipe de desenvolvimento, trazendo suas necessidades em relação ao sistema a ser desenvolvido. Essas necessidades irão gerar requisitos, e a análise de requisitos pode apontar necessidades ou requisitos conflitantes. A rastreabilidade auxilia a identificação das origens dos requisitos em conflito, para busca de uma solução para o problema detectado;
- verificação: no processo de análise de cobertura de requisitos nos testes, a avaliação da rastreabilidade entre requisitos e casos de testes permite identificar requisitos (ou funcionalidades) para os quais não foram previstos casos de testes, indicando à equipe de testes a necessidade de complementar a massa de testes;
- correção de defeitos: após a identificação do componente que originou o erro, a análise do problema pode indicar que a origem do defeito não está no código propriamente dito. Os elos de rastreabilidade indicarão quais artefatos, além do código, deverão ser revistos para a identificação do erro; os casos de testes também deverão ser verificados e/ou corrigidos, caso ocorra modificação em outros artefatos;
- validação: a etapa final de validação do sistema criado junto ao conjunto de clientes e usuários se beneficia da rastreabilidade, permitindo mostrar se a implementação atende ao conjunto de requisitos acordados entre clientes e desenvolvedores;
- análise de impacto na evolução do sistema: a existência de elos de rastreabilidade entre requisitos e componentes facilita a identificação dos componentes afetados por mudanças em um requisito ou mesmo por inclusão de novos requisitos, sem que haja necessidade de consultas a diferentes artefatos (por exemplo projeto da arquitetura, modelo de classes, código de componentes). Caso o requisito sendo alterado esteja ligado a outros requisitos (dependência entre requisitos), estes também deverão ser avaliados;
- previsão de custos e prazos: quando uma nova funcionalidade deve ser incluída no sistema em implementação ou quando uma mudança num requisito já implementado é solicitada, o gerente de projeto necessita de estimativas confiáveis para poder negociar custos e prazos junto ao cliente. No processo de evolução ou de manutenção a mesma situação ocorre com frequência. Este aspecto é particularmente crítico quando o desenvolvimento é terceirizado; estimativas serão mais rapidamente obtidas com os elos de rastreabilidade identificando os componentes atingidos pela solicitação;
- gerenciamento de riscos: a análise de riscos de um projeto inclui a identificação de riscos associados a custos e cronograma e de fatores contextuais que possam impactar em requisitos (restrições legais, por exemplo). A rastreabilidade apóia a identificação de artefatos atingidos por cada fator de risco, possibilitando a elaboração de estratégias para tratamento ou mitigação dos riscos;
- upgrade de hardware e/ou ambiente operacional: em sistemas embarcados (desenvolvidos para um hardware específico) é provável que existam relacionamentos muito fortes entre componentes do hardware e do software. Esta situação também ocorre em utilitários ou software correlatos: funcionalidades podem fa-

zer chamadas a funções do sistema ou ambiente operacional, numa versão específica. Na mudança de versão do ambiente operacional ou na troca do hardware alvo, elos de rastreabilidade agilizam a identificação dos componentes atingidos [Ramesh95];

- **reuso de componentes:** obter ativos (*assets*) reusáveis a partir de sistemas existentes tem incrementado o reuso na indústria; uma abordagem que propicia este incremento utiliza a recuperação de elos de rastreabilidade entre código e documentos escritos em linguagem natural (manual de usuário, documentos de especificação de requisitos, documentos da organização) facilitando a identificação de componentes relacionados a documentos em linguagem natural. Estes elos podem ajudar na criação de bibliotecas de componentes, possibilitando a obtenção de parte da informação necessária para inserção destes componentes em bibliotecas direcionadas ao reuso [Antoniol02].

O conjunto de situações apresentado não pretende ser exaustivo, mas sem dúvida nos permite concluir que a rastreabilidade é fundamental para um gerenciamento de qualidade. A existência dos elos de rastreabilidade possibilita identificar as origens de cada funcionalidade presente no sistema; procedimentos de verificação e validação utilizam a rastreabilidade para garantir que o sistema foi adequadamente testado e que atende às necessidades dos usuários. Elos de *rationale* registram decisões de projeto e possibilitam o efetivo gerenciamento dos riscos e o controle de mudanças utiliza a rastreabilidade para identificar artefatos afetados por uma solicitação de mudança.

2.3 Tipos de elos de rastreabilidade

A literatura apresenta diversos trabalhos enfocando os tipos de relacionamentos associados à rastreabilidade. Selecionamos dois artigos e apresentamos os elos de rastreabilidade neles propostos.

2.3.1 Meta-modelo proposto por Ramesh&Jarke

Em seu artigo sobre rastreabilidade [Ramesh01], Ramesh&Jarke propõem um meta-modelo para a rastreabilidade que possibilita a captura de informações relacionadas a agentes, fontes e objetos - as três dimensões dos modelos de rastreabilidade. Nesse meta-modelo os interessados são ligados através de estruturas de contribuição [Gotel94] aos objetos conceituais que eles influenciam e a documentos onde tais objetos são registrados [Jarke98]. A Figura 3, adaptada de [Jarke98], apresenta a visão geral desse meta-modelo; note que são apresentados objetos (relacionados ao produto sendo elaborado) e artefatos que são gerados pelo próprio processo de desenvolvimento. As três dimensões consideradas correspondem a:

- a) *Source* (fontes): documentos que remetem à origem dos requisitos (normas, padrões, legislação pertinente, atas de reuniões, ...);
- b) *StakeHolders* (interessados): são as pessoas envolvidas no Processo de Requisitos e que também possuem algum grau de interesse na rastreabilidade;
- c) *Objects* (objetos) ou artefatos: correspondem a objetos conceituais relacionados ao produto ou a artefatos gerados no processo de desenvolvimento.

Ramesh&Jarke ponderam que mesmo existindo uma grande variedade de elos de rastreabilidade, eles podem ser agrupados em duas categorias básicas:

- a) **relacionados ao produto:** elos que descrevem propriedades e relacionamentos dos objetos; são subdivididos em elos de *satisfação* e elos de *dependência*;
- b) **relacionados ao processo:** elos relacionados ao histórico de ações executadas no próprio processo; são subdivididos em elos de *evolução* e elos de *rationale*.

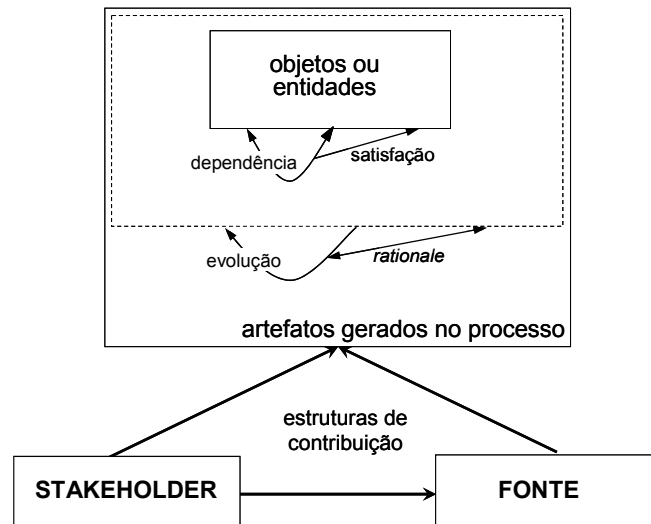


Fig. 3 – Meta-modelo para a rastreabilidade de requisitos [Jarke98]

O propósito dos elos de *satisfação* é assegurar que os requisitos sejam atendidos pelo sistema, ou seja, a cada requisito foi associado um componente que deverá atendê-lo. Este tipo de elo é utilizado para registrar os desenhos criados para satisfazer requisitos e os componentes para os quais requisitos são alocados, para assegurar que todo componente satisfaz a requisitos, registrar fatores críticos de sucesso associados a requisitos e assegurar consistência entre saídas das diferentes fases do ciclo de vida.

O propósito dos elos de *evolução* é registrar relacionamentos que levam de objetos existentes para objetos novos ou modificados. Este tipo de elo é útil para identificar as origens dos objetos para melhor compreensão dos requisitos e outros objetos (através de sua história) e para registrar as modificações e histórico de refinamentos dos vários objetos.

O propósito dos elos de *rationale* é representar as motivações subjacentes aos objetos existentes ou documentar as razões para evolução. Estes elos são utilizados para encontrar as justificativas para criação ou modificação de objetos, registrar suposições utilizadas no processo de decisão e identificar o contexto de criação de objetos. Estes elos possibilitam registrar aspectos do processo decisório, incluindo alternativas descartadas, de forma a providenciar clara compreensão da solução escolhida, facilitando a manutenção e o reuso. Em outras palavras, os elos de *rationale* auxiliam a gerenciar o desenvolvimento do sistema de acordo com as necessidades e objetivos organizacionais. Estes elos representam a área de atuação dos interessados, registrando as origens e o contexto no qual os objetos são desenvolvidos.

E finalmente, os elos de *dependência* têm por propósito apoiar o gerenciamento de dependências entre objetos, freqüentemente impostas por restrições de recurso, de competência ou de compatibilidade, sendo úteis para registrar a composição e hierarquia dos objetos e apoiar o gerenciamento do impacto das alterações num objeto sobre os objetos que dele dependem.

2.3.2 Meta-modelo proposto por Toranzo

A proposta de Toranzo [Toranzo02] para a rastreabilidade utiliza quatro estratégias de trabalho: apresenta uma classificação das informações a serem rastreadas, propõe um meta-modelo para a rastreabilidade e um modelo intermediário para o rastreamento de requisitos e ainda inclui um processo para guiar o engenheiro de requisitos na aplicação das outras três estratégias.

As informações a serem rastreadas seriam classificadas em quatro níveis: *ambiental*, *organizacional*, *gerencial* e *desenvolvimento*. O nível *ambiental* congrega informações oriundas do contexto ambiental onde a organização está inserida e que podem afetar o sistema sendo desenvolvido; o nível *organizacional* reúne informações relacionadas à organização (missão, objetivos, metas e padrões) e que podem impactar os requisitos do sistema; o nível *gerencial* agrega informações que permitem associar tarefas a requisitos, e que podem auxiliar a gerência do projeto; e finalmente o nível *desenvolvimento* abarca informações relacionadas aos diversos artefatos gerados no processo de desenvolvimento (documento de requisitos, diagramas, programas, casos de testes, ...).

O meta-modelo proposto pode ser graficamente visualizado na Figura 4, apresentada em [Toranzo02]. Neste meta-modelo a meta-classe base *Elemento* possui as sub-classes *ElementoGeneralizável* e *Relacionamento*; instâncias de *ElementoGeneralizável* se relacionam através de *Generalização* e *Associação*.

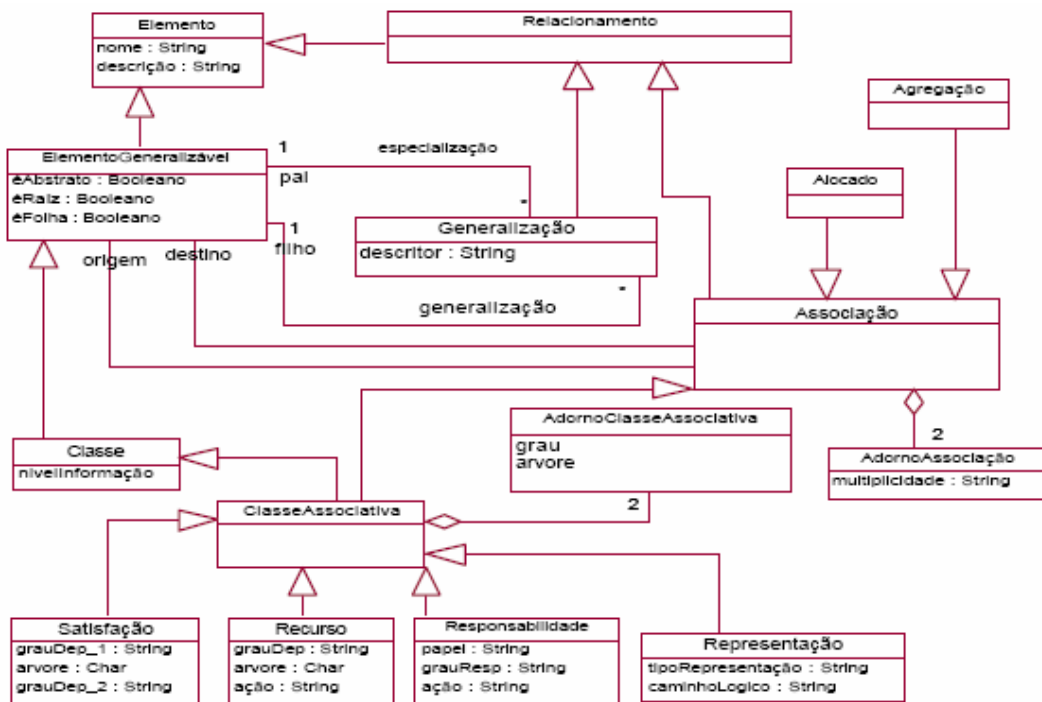


Fig. 4 – Meta-modelo para a rastreabilidade de requisitos [Toranzo02]

Neste meta-modelo, o suporte à rastreabilidade identifica os seguintes tipos de elos:

- satisfação*: indica que a classe de origem tem dependência de satisfação com classe de destino;
- recurso*: indica que a classe de origem tem dependência de recurso com classe de destino;

- c) *responsabilidade*: registra a participação, responsabilidade e ação de pessoas sobre artefatos;
- d) *representação*: captura a representação ou modelagem dos requisitos em outras linguagens;
- e) *alocado*: classe de origem está relacionada à classe de destino, que representa um subsistema;
- f) *agregação*: indica composição de elementos.

Considerando as dificuldades enfrentadas pelos desenvolvedores na definição de um modelo para a rastreabilidade, Toranzo propõe também um modelo intermediário de rastreamento, que consolida observações da prática, de estudos de caso e da aplicação das duas estratégias já apresentadas - classificação das informações e o meta-modelo de rastreabilidade. A idéia é que este modelo intermediário possa ser encarado como base para discussão e definição de um modelo de rastreabilidade para o projeto a ser desenvolvido. E finalmente a proposta apresenta também o processo associado ao modelo proposto, que inicia pela classificação das informações a serem rastreadas nos níveis propostos, passando pelo meta-modelo e chegando ao modelo intermediário, dirigido ao projeto em desenvolvimento. O processo é apresentado detalhadamente, com a aplicação do modelo proposto a um sistema de automação de biblioteca, mostrando exemplos e diretrizes para sua aplicação.

2.3.3 Comparação das propostas de Ramesh&Jarke e de Toranzo

Se considerarmos apenas os tipos de elos, observamos que as propostas de Ramesh&Jarke [Ramesh01] e Toranzo [Toranzo02] coincidem nos elos de:

- a) **satisfação**: os elos classificados como de *satisfação* para Ramesh&Jarke, e que indicam que requisitos são atendidos pelo sistema, correspondem ao que Toranzo denomina de *alocado* e que indica alocação de requisitos a sub-sistema, ou ainda à representação do requisito em outras linguagens (por exemplo, diagramas UML);
- b) **dependência**: os elos deste tipo, para Ramesh&Jarke, indicam que objetos dependem uns dos outros por restrições de recursos, competência ou compatibilidade, registrando a composição e hierarquia dos objetos. Toranzo denomina tais elos de *recurso* (dependência devida a um recurso), *satisfação* (dependência entre instâncias de duas classes) ou de *agregação* (composição de elementos).

Os tipos de elos que Ramesh&Jarke relacionam ao processo de desenvolvimento, ou seja, elos de *evolução* e de *rationale*, e que tem o propósito de registrar tanto a evolução dos objetos como a motivação para as mudanças, não estão presentes no meta-modelo de Toranzo conforme apresentado em [Toranzo02]. Aspectos gerenciais do desenvolvimento do projeto recebem maior ênfase no modelo de Toranzo, que cria o elo do tipo *responsabilidade* com o intuito específico de registrar as responsabilidades dos interessados por artefatos e objetos gerados no processo de desenvolvimento.

Ambas as propostas analisadas possibilitam a instanciação do meta-modelo proposto, permitindo atender tanto a aspectos básicos da rastreabilidade (registro das ligações entre requisitos, suas origens e artefatos de desenho e implementação) quanto a modelos mais sofisticados. No nosso entendimento, o mérito maior da proposta de Ramesh&Jarke está na simplicidade do meta-modelo e na atenção colocada nos aspectos de evolução, registrando alterações sofridas pelos objetos rastreados, e as motivações para tais mudanças. Já a atenção colocada por Toranzo nos aspectos gerenciais do projeto é uma de suas contribuições, dado que o Gerenciamento de Requisitos está intimamente ligado à gerência do projeto. Isto é observado tanto nos elos de responsabilidade como

na classificação proposta para as informações a serem rastreadas, enfatizando as influências dos contextos ambiental, organizacional e gerencial no projeto em desenvolvimento.

O registro de todos os tipos de rastreabilidade, num projeto, pode representar um grande volume de informações para ser gerenciado, exigindo automação do processo e impactando custos [Ramesh01] [Egyed01]; como exemplo, o Departamento de Defesa Americano estima que seus custos em rastreabilidade sejam da ordem de 4% do orçamento da área de Tecnologia da Informação [Ramesh01]. É tarefa do gerente do projeto definir políticas que estabeleçam quais informações de rastreabilidade devem ser registradas; na próxima seção serão apresentadas algumas heurísticas que podem auxiliar nesta tarefa.

3 Prática da rastreabilidade de requisitos

Problemas relacionados ao registro e uso da rastreabilidade não são recentes: em artigo de 1994, Gotel e Filkelstein [Gotel94a] observam que a rastreabilidade de requisitos permanece citada como problema-chave entre indústria e comunidade de engenharia de requisitos. Boa parte desses problemas é de caráter informacional e ocorre principalmente quando as ferramentas disponíveis não são utilizadas para controlar informações sobre os requisitos que se deseja rastrear. Foi verificado o uso inadequado da pré-rastreabilidade como razão para problemas de rastreabilidade a longo prazo; também foi observada a falta de acordo a respeito de quantidade e tipo de informação que os desenvolvedores desejam rastrear e a extrema importância por eles atribuída a contato pessoal e comunicação informal.

O já referido trabalho de Ramesh&Jarke [Ramesh01], mais recente, apresenta resultado de uma longa pesquisa que tem a rastreabilidade como tema central. Este trabalho reforçou a divisão entre usuários (aqui se referindo a desenvolvedores) sofisticados e normais (*high-end* e *low-end users*) [Jarke98] [Ramesh98], e buscou um modelo de referência para implementação da rastreabilidade em repositórios manuais ou automatizados; verificou-se que os modelos de referência utilizados pelas duas categorias de usuários diferem significativamente. Usuários normais utilizam esquemas simples de rastreabilidade: registro de dependências entre requisitos, ligações entre requisitos e componentes que os atendem e ligações entre requisitos e procedimentos de verificação (como testes e simulações). Usuários deste tipo (*low-end users*) desconsideram importantes informações relacionadas ao processo de rastreabilidade, como por exemplo o *rationale* subjacente a artefatos e à evolução desses mesmos artefatos no processo de desenvolvimento. Estes usuários entendem a rastreabilidade como uma imposição dos responsáveis pelo projeto.

O modelo de referência de usuários sofisticados (*high-end users*) mostra que eles empregam esquemas de rastreabilidade mais ricos e que também utilizam as informações de rastreabilidade de formas mais enriquecedoras, enfatizando aspectos relacionados ao *processo*. O meta-modelo destes usuários envolve atividades associadas ao gerenciamento de requisitos, ao desenho e alocação, verificação e validação e gerenciamento do *rationale*. Os modelos de referência para esses dois tipos de usuários estão detalhados em [Ramesh98] e [Ramesh01]; os autores ressaltam que, mesmo em empresas onde a prática esteja mais próxima do primeiro tipo, é possível encontrar esquemas mais maduros de rastreabilidade em algumas áreas.

A rastreabilidade também é vista como inadequadamente tratada no processo distribuído de desenvolvimento [Zowghi01] [Damian03]. Mesmo em organizações onde a

preocupação com a qualidade do processo de desenvolvimento leva a processos que atendem a preceitos como os preconizados por modelos de qualidade (CMM, CMMI ou ISO), há dificuldades no registro e na manutenção dos elos de rastreabilidade. Como registrado por Zowghi em [Zowghi01], *qualquer rastreabilidade útil, como por exemplo de requisitos para cliente ou desenho para requisitos, é próxima da não-existência*.

Linscomb [Linscomb03] registra que nunca encontrou empresas certificadas ao nível 2 do CMM onde fossem trabalhadas matrizes de rastreabilidade completas, indicando que cada requisito foi atendido por objetos de desenho, implementação e testes. Egyed [Egyed01] relata um importante problema relacionado a falhas na documentação de sistemas, com impacto na rastreabilidade: a evolução separada do sistema e da documentação é causa da dissociação entre implementação e modelos. Se esta dissociação acontece, desenvolvedores entendem que o esforço por eles dedicado às tarefas associadas à rastreabilidade foi inútil; a consequência última deste fato é que atividades relacionadas ao registro e evolução das ligações de rastreabilidade serão negligenciadas.

Esses relatos, que envolvem observações de uma década, relatam uso inadequado da rastreabilidade por parte dos desenvolvedores; muitas vezes, a rastreabilidade é registrada simplesmente para atender a padrões da organização [Ramesh01] ou exigências dos modelos de processos utilizados [Egyed01]. A rastreabilidade é valorizada e adequadamente utilizada por desenvolvedores que possuem experiência suficiente para conhecer, pela prática, a importância do registro das ligações das fontes aos requisitos, destes aos componentes e aos artefatos de software, e das possibilidades abertas por estas ligações quando surgem alterações no desenho [Ramesh01].

3.1 Técnicas e ferramentas: suporte à questão da rastreabilidade

Uma possível classificação para as técnicas de rastreabilidade mais comuns está relacionada ao foco principal: referências cruzadas ou documentos. Técnicas centradas em referências cruzadas são simples de entender e utilizar e podem ser suportadas pelo uso de hipertexto, esquemas de numeração, indexação ou uso de *tags* ou ainda por matrizes de rastreabilidade. Técnicas centradas em documentos utilizam modelos (*templates*) para documentos e artefatos utilizados no processo de desenvolvimento e incorporam mecanismos para transformação e integração de documentos.

Uma das técnicas mais comuns, o uso de referências cruzadas, pode utilizar uma matriz de rastreabilidade, implementada com uso de uma ferramenta de uso geral, como um editor de textos ou uma planilha eletrônica (muitas das ferramentas comercialmente disponíveis para a fase de requisitos utilizam alguma forma de matriz de rastreabilidade). A Tabela 1 apresenta um modelo simplificado de matriz de rastreabilidade.

Tab. 1 - Rastreabilidade entre requisitos e entidades geradas no processo de desenvolvimento

Projeto <nome_projeto> - Matriz de Rastreabilidade				
Requisito	Documento fonte	Arquitetura	Componente	Caso de teste

No exemplo da Tabela 1, a primeira coluna da matriz deverá ser preenchida com os requisitos; normalmente os requisitos são expressos em linguagem natural e numerados seqüencialmente, mas também podem ser utilizadas outras formas de representa-

ção. As demais colunas devem representar artefatos gerados durante o processo de desenvolvimento; a correspondência nem sempre é da ordem de um para um (por exemplo, um requisito pode estar sendo verificado em diversos casos de teste, e vice-versa).

A mesma técnica de referências cruzadas com uso de matriz de rastreabilidade também pode indicar dependências entre requisitos; a Figura 5 apresenta a ferramenta RequisitePro® registrando, na matriz de rastreabilidade, interdependências entre requisitos funcionais e não funcionais em um sistema de controle de imobiliárias [Felicissimo02]. A matriz registra que cadastros do sistema devem atender a requisitos não funcionais de interface, e requisitos funcionais de exclusão de imóvel, exclusão de cliente e limpeza de cadastros devem atender ao requisito não funcional de segurança.

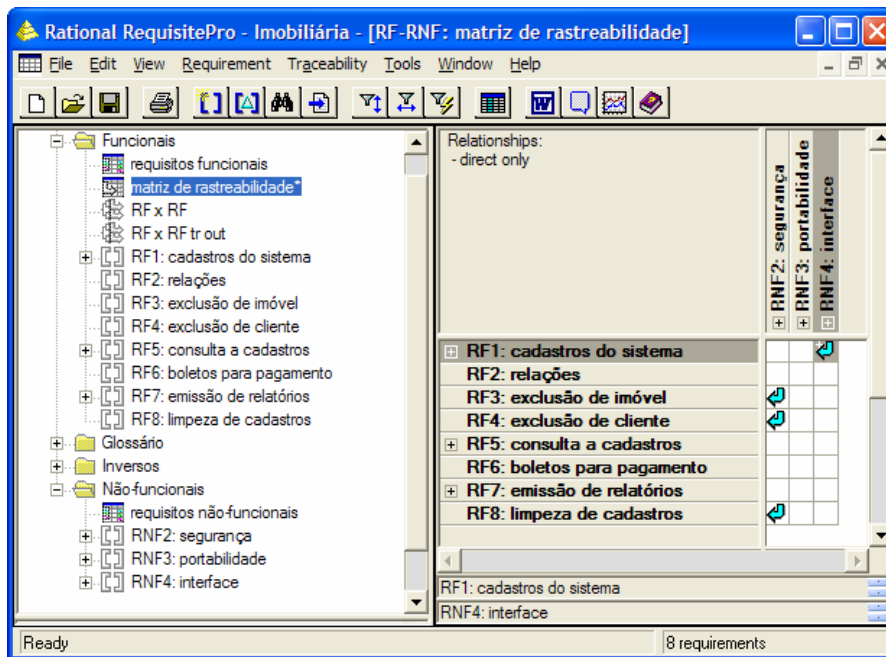


Fig. 5 - Matriz de rastreabilidade entre requisitos funcionais e não-funcionais, no Req-Pro®

A técnica de referências cruzadas pode ainda ser implementada com o uso de elos de hipertexto, mostrando relacionamentos entre artefatos [Palmer97] [Oinas-Kukkonen96] [Leite97] [Alexander03]. Este tipo de rastreabilidade é encontrado no software C&L - Cenários e Léxico, ferramenta de software livre disponível através do endereço <http://sl.les.inf.puc-rio.br/cel/>, desenvolvida pelo Grupo ER - Engenharia de Requisitos da PUC-Rio. Esta ferramenta, destinada a apoiar o trabalho do engenheiro de software nas atividades do processo de desenvolvimento, cria automaticamente elos de rastreabilidade entre símbolos do léxico e cenários da aplicação. O léxico é uma representação dos termos particulares da linguagem de uma dada aplicação, e é utilizado para facilitar a comunicação e a compreensão de palavras ou frases peculiares a um Universo de Informação entre interessados [Leite97]. Cenários representam situações comuns no cotidiano dos clientes e usuários; o método utilizado no processo de construção de cenários segue o princípio de conservação dos nomes, com âncora no léxico. Isto facilita a automação da criação dos elos, já que cenários fazem referência a símbolos registrados no léxico; a ferramenta percorre os cenários, identificando os símbolos presentes no léxico e criando os *hiperlinks* entre cenários e léxico. Como um termo do léxico pode estar presente em mais de um cenário, cria-se uma "teia" de referências cruzadas entre léxico e cenários.

Alexander [Alexander03] relata o uso de uma ferramenta comercialmente disponível, o DOORS, associada a um software desenvolvido especificamente para atender a necessidades de registro de elos entre requisitos e casos de uso, possibilitando múltiplas versões dos requisitos e dos casos de uso e exportando informações para páginas *html*; isto é interessante, por exemplo, para facilitar a manipulação de requisitos para diferentes versões de um mesmo produto.

É importante registrar que as técnicas de referência cruzada para registro da rastreabilidade são adequadas para registro dos elos relacionados ao produto em desenvolvimento, ou seja, aqueles denominados por Ramesh&Jarke como de *satisfação* e *dependência*, e todos os tipos de elos relacionados por Toranzo. Os elos que Ramesh&Jarke associaram ao processo, ou seja, os elos de *evolução* e de *rationale*, não são suportados pela técnica de referência cruzada.

Pesquisas relatam trabalhos que têm buscado responder às necessidades de registro do *rationale* subjacente aos requisitos ou a modelos do sistema: o método gIBIS para a captura de *rationale*, baseado na técnica IBIS, trabalha com repetição de ciclos onde são apresentados e discutidos temas até que se chegue à solução desejada. É feito o registro dos argumentos e posicionamentos para viabilizar a recuperação do *rationale* - motivações para decisões [Conklin88]. Em [Leite95] é apresentado um modelo conceitual para a *baseline* de requisitos, que incorpora um modelo de versionamento para requisitos onde são registradas as motivações para mudanças. Um *framework* para suportar a evolução de cenários é apresentado em [Breitman00]: o registro do *rationale* associado à evolução dos cenários pode ser efetuado segundo um dos cinco modelos possíveis. O registro do *rationale* associado a decisões de desenho de software foi explorado por Potts e Bruns [Potts88] e serviu de base para a construção do ambiente JSD/PUC [Lucena96].

A análise de ferramentas destinadas a apoiar as tarefas de um processo de desenvolvimento mostra a pouca adequação das ferramentas de rastreabilidade na integração ao processo de desenvolvimento [Spanoudakis02] [Toranzo02] [Alexander03]. Abordagens e ferramentas comercialmente disponíveis para o processo de requisitos não fornecem o suporte necessário ao registro dos elos de rastreabilidade, forçando os desenvolvedores a fazer manualmente o trabalho de registrar e manter atualizados os elos de rastreabilidade [Palmer97] [Pinheiro99] [Mohan02] [Egyed02]. Tal trabalho é considerado oneroso pelos desenvolvedores, o que provoca a ausência de registros de rastreabilidade em ambientes de desenvolvimento. Contribui para este fato o ambiente altamente competitivo onde as empresas se inserem, no qual o *time-to-market* (pressão de comercialização) é muitas vezes crucial para o sucesso de um projeto [Leon00].

Entre as ferramentas comerciais disponíveis, citamos a suíte da IBM/Rational, que inclui o RequisitePro® (centrado em documentos e utilizando matriz de rastreabilidade) para o Processo de Requisitos e outras ferramentas como o AnalystStudio para o gerenciamento de requisitos. Da mesma forma, DOORS® (centrado em documentos e utilizando *hyperlinks* para rastreabilidade) apóia tarefas relacionadas ao gerenciamento de requisitos. Registramos que RequisitePro® propicia também o registro do *rationale* subjacente aos requisitos e sua evolução e a hierarquia entre requisitos de alto nível aos seus derivados, e que DOORS® associado ao *toolkit* ScenarioPlus possibilita elos entre requisitos e casos de uso, em múltiplas versões, para diferentes configurações de um mesmo sistema.

O INCOSE, International Council on Systems Engineering, mantém uma página, periodicamente atualizada, com informações sobre as ferramentas para apoio ao registro da rastreabilidade (veja em <http://www.incose.org/ProductsPubs/pro->

ducts/SEtools/tooltax/reqtrace_tools.html). Outra página, mantida pela mesma organização, apresenta um resumo sobre essas ferramentas, com informações fornecidas pelos fabricantes (<http://www.paper-review.com/tools/rms/read.php>).

3.2 Processo associado à rastreabilidade

No desenvolvimento de um sistema de software onde a rastreabilidade é utilizada, encontramos atividades associadas à rastreabilidade em quatro momentos: na fase inicial do trabalho são definidos os objetos e artefatos a serem rastreados. Também nesta fase devem ser estabelecidos os tipos de elos que serão registrados. Durante todo o processo de desenvolvimento, os objetos e artefatos a serem rastreados deverão relacionados através de elos, de acordo com os tipos definidos anteriormente. Esse processo tanto pode ser realizado manualmente como contar com auxílio de ferramentas. A recuperação desses relacionamentos deverá ser realizada para atender necessidades dos interessados nas diferentes atividades: resolução de requisitos em conflito, identificação da origem de requisitos, alocação de requisitos a componentes, entre outros. Como qualquer outro produto gerado no processo de desenvolvimento de software, espera-se que também os elos sofram evolução, e portanto o processo de manutenção dos mesmos também deve ser realizado.

As etapas desse processo, esquematizadas do diagrama SADT [Ross77] da Figura 6, foram bem definidas em [Pinheiro04] e [Knethen02], e estão sumarizadas a seguir:

- a) **definição:** envolve a definição das entidades a serem rastreadas (artefatos, objetos), os tipos de relacionamentos a serem estabelecidos e sob que condições os elos deverão ser registrados. Esta definição pode ter como base um modelo de referência como os apresentados anteriormente, e deve considerar as necessidades da equipe e as restrições contextuais como prazos e custos;
- b) **registro dos elos:** envolve o registro explícito dos relacionamentos, envolvendo artefatos e entidades, nas situações e condições definidos anteriormente. O uso de uma ferramenta específica pode apoiar este processo, sendo em alguns casos imprescindível devido ao volume de informações a ser manipulado;
- c) **recuperação ou extração:** dado um objeto ou artefato rastreável, e um tipo de relacionamento específico, implica em encontrar as entidades associadas;
- d) **evolução:** processo de atualização dos elos de rastreabilidade, conforme surjam modificações nas entidades relacionadas ou nos próprios relacionamentos estabelecidos entre elas, ao longo do processo de desenvolvimento e de evolução do software.

Ramesh&Jarke [Ramesh01] [Ramesh98] reforçam a importância do envolvimento dos interessados na criação da documentação de requisitos; esta importância cresce à medida que aumenta a duração e complexidade do projeto. Sistemas de gerenciamento de configuração e versão são fundamentais para o adequado gerenciamento dos documentos que são a origem das informações de rastreabilidade. A natureza dinâmica da rastreabilidade exige uma ferramenta para apoio ao processo de rastreabilidade. Sem tal ferramenta, esta atividade pode ser impossível de atingir com qualidade, dado o volume de informações que poderá ser manipulado [Ramesh98] [Dömges98] [Eged01].

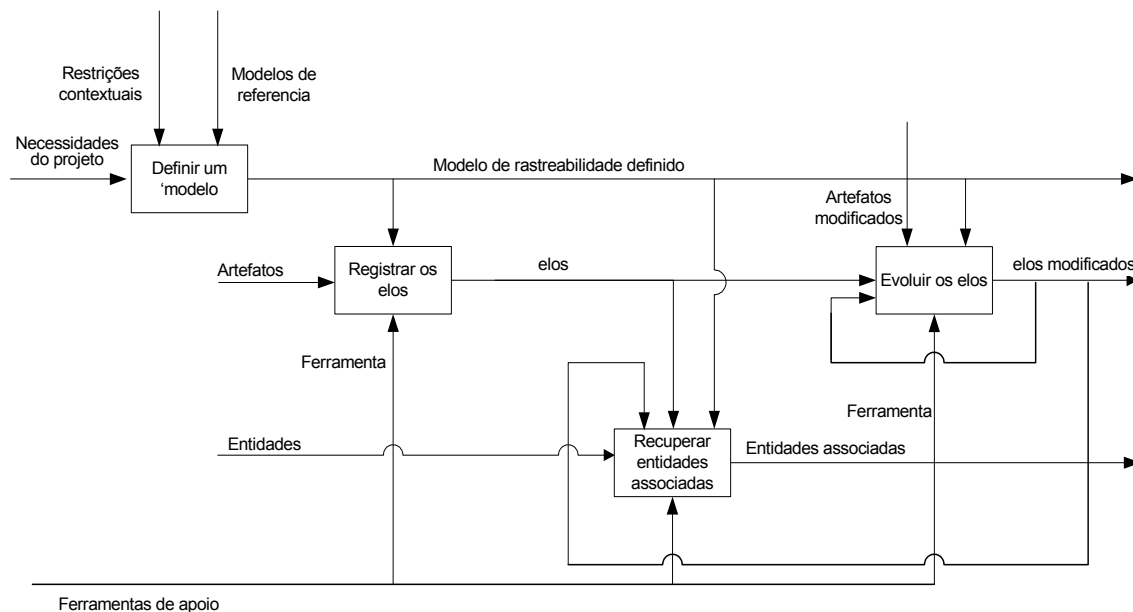


Fig. 6 – Diagrama SADT⁵ para o processo associado à rastreabilidade

3.3 Heurísticas para definição de um modelo de rastreabilidade

A captura e uso dos elos de rastreabilidade devem ser adaptados às necessidades específicas de cada projeto, possibilitando uma relação custo-benefício positiva e evitando uma massa excessiva de informações relacionadas à rastreabilidade [Dömges98] [Ramesh01]. A definição dos elos a serem capturados deve considerar prazos e custos do projeto em pauta, além de processos e padrões em uso na organização. Propomos a seguir, apoiados pela nossa experiência, algumas heurísticas que podem auxiliar gerente de projeto e equipe na tarefa de definir um modelo de rastreabilidade para o processo de desenvolvimento de um software específico:

1. definir no início do projeto, considerando a aplicação a ser desenvolvida, os tipos de elos a serem utilizados e explicitamente registrados;
2. identificar as ferramentas que apoiarão o processo de rastreabilidade;
3. conscientizar a equipe da importância do processo de rastreabilidade (*considere que desenvolvedores não são exatamente conhecidos por seu amor à documentação [Jarke98] e que a falta de comprometimento da organização com a atividade de rastreabilidade é responsável pela falta de interesse de seus desenvolvedores nessa tarefa [Ramesh98]*);
4. estabelecer as entidades (artefatos, componentes, objetos, requisitos) a serem rastreadas e os pontos onde o registro deverá ser realizado;
5. durante o processo de desenvolvimento, verificar se os elos de rastreabilidade estão sendo registrados pela equipe;
6. utilizar e avaliar o mecanismo de extração de elos, em relação às expectativas realizadas;
7. após a liberação do software, analisar criticamente com a equipe a efetividade do modelo de rastreabilidade adotado, corrigindo possíveis distorções e melhorando-o para os próximos projetos.

⁵ Notação SADT: atividades nomeadas nos retângulos possuem entradas representadas por setas à esquerda e saídas à direita. Controles são representados por setas descendentes e mecanismos são representados por setas ascendentes.

4 Conclusões

Este trabalho apresentou diferentes aspectos relacionados à rastreabilidade, buscando apresentar ao leitor uma visão ampla da rastreabilidade e suas implicações para o processo de gerenciamento de requisitos e mesmo do gerenciamento do desenvolvimento do software. Do ponto de vista da qualidade do produto, a rastreabilidade é importante pois apóia a análise de cobertura de testes de funcionalidade e a correção de defeitos. Do ponto de vista da gerência do desenvolvimento, rastreabilidade possibilita identificar rapidamente artefatos de desenho, projeto e implementação afetados por uma solicitação de mudança, possibilitando avaliar o impacto das alterações necessárias e negociar alterações de prazos e custos de desenvolvimento junto ao cliente. Rastreabilidade auxilia a validação do software liberado junto ao cliente, mostrando a completude da implementação em relação aos requisitos estabelecidos.

Em sistemas desenvolvidos para ambientes em constante evolução, como é o caso de aplicações para Internet, e onde a pressão pelo cumprimento de prazos é forte, a análise da rastreabilidade pode ajudar a detectar requisitos ainda não alocados a componentes, alertando a gerência ainda a tempo de solucionar o problema.

A análise da rastreabilidade é fundamental em sistemas onde os processos organizacionais que devem ser suportados são complexos [Mohan02], de grande porte [Palmer97] ou naqueles com prazos de desenvolvimento muito curtos [Leon00]. Rastreabilidade apóia a detecção de conflitos e é essencial para o gerenciamento do desenvolvimento e controle de riscos [Palmer97]. Sua aplicação, porém, impacta nos custos do projeto e o gerente do projeto deve escolher os requisitos que serão rastreados [Dömges98] [Ramesh01].

Apesar dos benefícios de sua aplicação, e da disponibilidade de ferramentas para tratar a rastreabilidade de requisitos, nossa experiência e relatos encontrados na literatura mostram que, por diversas razões, a rastreabilidade não tem sido adequadamente trabalhada por parte dos desenvolvedores [Gotel94a] [Leon00] [Ramesh01] [Zowghi01] [Egyed01] [Damian03] [Linscomb03] [Felici04]. Acreditamos que isso se deve, em parte, à pouca flexibilidade das ferramentas disponíveis no mercado, o que obriga os desenvolvedores a registrar manualmente as informações de rastreabilidade. Acreditamos que esse problema possa ser solucionado com a apresentação de ferramentas mais poderosas, que possibilitem o registro das informações de rastreabilidade e sua integração com demais artefatos gerados por outras ferramentas utilizadas no processo de desenvolvimento. Outro aspecto que deverá ser trabalhado é o treinamento e a disseminação dos ganhos que a rastreabilidade pode trazer aos processos de desenvolvimento e evolução de software, possibilitando rapidamente a elaboração de análises de impacto, previsões de alterações em cronogramas e custos, auxiliando os procedimentos de verificação e validação e mesmo incrementando o reuso.

4.1 Contribuições deste artigo

Neste trabalho ressaltamos a importância da rastreabilidade no contexto do processo de desenvolvimento de software, associando-a tanto a aspectos técnicos como gerenciais; além de apresentar uma visão ampla do processo de rastreabilidade, buscamos conscientizar o leitor das implicações da rastreabilidade para a qualidade do produto sendo desenvolvido. Esta associação da rastreabilidade à qualidade do produto e do processo é encontrada também nos modelos de avaliação da maturidade do processo de desenvolvimento, como CMM, CMMI e ISO 9001, que colocam a rastreabilidade como atividade essencial ao gerenciamento por requisitos.

Enquanto a literatura técnica sobre rastreabilidade tende a apresentar apenas aspectos diretamente relacionados à rastreabilidade, neste artigo procuramos apresentar uma visão mais abrangente do processo de construção de software, onde se insere a rastreabilidade. Nosso objetivo envolveu levar ao leitor uma visão ampla dos aspectos relacionados ao gerenciamento de requisitos e ao próprio processo de desenvolvimento, passando também por modelos de maturidade no desenvolvimento de software e suas ligações à rastreabilidade.

Modelos de referência são úteis para servir como guias no processo de definir um modelo que atenda a necessidades e restrições específicas para um dado problema; neste trabalho apresentamos dois diferentes modelos de referência, oriundos da literatura pesquisada, que podem servir como base para o leitor na definição ou escolha de um modelo de rastreabilidade.

A prática da rastreabilidade exige a definição de um processo adaptado a cada sistema de software a ser desenvolvido; apresentamos heurísticas para auxiliar gerente e equipe na tarefa de estabelecer um processo para a rastreabilidade durante o desenvolvimento. Esperamos que essas heurísticas guiem a equipe no caminho da definição, utilização e evolução de elos de rastreabilidade. Também apresentamos um conjunto abrangente de tarefas que podem se beneficiar da rastreabilidade, buscando com isso conscientizar o leitor da importância do registro da rastreabilidade e sua evolução durante o processo de desenvolvimento.

4.2 Aspectos em aberto

Um importante aspecto pouco trabalhado na literatura pesquisada e analisada é a ligação entre evolução dos requisitos e rastreabilidade: alguns poucos trabalhos, como os relatados em [Leite95] e [Ramesh01] endereçam especificamente tipos de ligações que são diretamente relacionados à evolução dos requisitos e dos artefatos. Também a maioria das ferramentas disponíveis comercialmente pouco oferece em termos de apoio ao tratamento dos elos de evolução e *rational*, que são aqueles relacionados ao *processo* de desenvolvimento [Ramesh01].

A infra-estrutura de apoio ao processo de rastreabilidade inclui ferramentas que manipulem a *baseline* de requisitos e os elos de rastreabilidade. A *baseline* de requisitos em uso pelo projeto deve ter flexibilidade para propiciar não apenas o registro e evolução dos elos, mas também a recuperação ou extração das entidades associadas através de um dos tipos de elos de rastreabilidade estabelecidos para o projeto. Uma ferramenta que atenda a essas necessidades deve incorporar também características usualmente presentes em sistemas de controle de versões e de gerenciamento de configuração.

O caráter inevitável das mudanças em requisitos ao longo do processo de desenvolvimento [Leite95] exige políticas de gerenciamento de mudanças que incluam o tratamento da evolução dos elos de rastreabilidade. Mais que isso, exige também por parte dos desenvolvedores confiança nos resultados do esforço investido no processo de criação e atualização dos elos de rastreabilidade.

Referências Bibliográficas

[Alexander03] Alexander, I. "SemiAutomatic Tracing of Requirement Versions to Use Cases Experiences & Challenges". In: Second International Workshop on Traceability, Montreal, October 2003. **Proceedings**. Disponível em <http://easyweb.easynet.co.uk/~iany>. Acesso em 11.10.2004.

- [Antoniol02] Antoniol, Giuliano; Canfora, Gerardo; Casazza, Gerardo; De Lucia, Andrea & Merlo, Ettore. "Recovering Traceability Links between Code and Documentation". **IEEE Transactions on Software Engineering**, vol. 28 (10), October 2002. pp. 970-983.
- [Breitman00] Breitman, Karin k. **Evolução de cenários**. 2000. Doutorado (tese). Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática, Rio de Janeiro. 2000.
- [Conklin88] Conklin, J. & Begeman, M. "gIBIS: A hypertext tool for exploratory policy discussion". In: Second Conference on Computer-Supported Cooperative Work (Portland, Oreg., Sept. 26-28). ACM, New York, 1988. **Proceedings**. pp. 140-152.
- [CMM01] Modelo de Maturidade de Capabilidade de Software (CMM) versão 1.2-tradução do CMU/SEI-93-TR-24-CMM V1.1, por José Marcos Gonçalves e André Villas Boas.
- [CMMI02] Capability Maturity Model® Integration (CMMI), Version 1.1 - for Software Engineering (CMMI-SW, V1.1), Staged Representation.
- [Damian03] Damian, D.; Chisan, J.; Vaidyanathasamy, L. & Pal, Y. "An Industrial Case Study of the Impact of Requirements Engineering on Downstream Development". In: 2003 International Symposium on Empirical Software Engineering (ISESE'03). **Proceedings**.
- [Davis93] Davis, A. M., "Software Requirements: Objects, Functions and States". Englewood Cliffs, New Jersey: Prentice Hall. 1993.
- [Dion93] Dion, Raymond. "Process Improvement and the Corporate Balance Sheet", **IEEE Software**, vol 10 (4), July 1993. pp. 28-35.
- [Dömges98] Dömges, R. & Pohl, K. "Adapting traceability environments to project-specific needs". **Communications of ACM**, vol. 41(12), 1998. pp. 54-62.
- [Edwards91] Edwards, M. & Howell, S., "A Methodology for System Requirements Specification and Traceability for Large Real-Time Complex Systems", technical report, U.S. Naval Surface Warfare Center-Dahlgren Division, Dahlgren, Va., 1991.
- [Egyed01] Egyed, Alexander. "A Scenario-Driven Approach to Traceability". In: 23rd International Conference on Software Engineering - ICSE 2001. **Proceedings**. pp. 123-132.
- [Egyed02] Egyed, Alexander & Grünbacher, Paul. "Automating Requirements Traceability: Beyond the Record & Replay Paradigm". In: 17th IEEE International Conference on Automated Software Engineering (ASE), Edinburgh, Scotland, UK, September 2002, **Proceedings**. pp. 163-171.
- [Egyed04] Egyed, Alexander. "Software Tracing". Research project. Disponível em http://sunset.usc.edu/~aegyed/research/model_traceability.html. Acesso em 09.03.2005.
- [Felici04] Felici, Massimo "Observational Models of Requirements Evolution". Doctoral thesis. Disponível em <http://homepages.inf.ed.ac.uk/mfelici/doc/IP040037.pdf>. Acesso em 19.03.2005.
- [Felicissimo02] Felicíssimo, C.; Maria, B. A.; Sugar, M. & Silva, S. B. "Sistema de Administração de Aluguéis". Trabalho final da disciplina INF 1813-Engenharia de Requisitos, com o prof. Julio Cesar S. P. Leite, PUC-Rio, 2001.

- [Gotel94] Gotel, Orlena & Filkenstein, A., "Modelling the Contribution Structure Underlying Requirements". In: First International Workshop on Requirements Engineering, 1994. **Proceedings**. pp. 71-81.
- [Gotel94a] Gotel, Orlena & Finkelstein, A. "An Analysis of the Requirements Traceability Problem". In: IEEE International Conference on Requirements Engineering, IEEE Computer Society Press, Colorado Springs, Colorado, 1994. **Proceedings**. pp. 94-101.
- [Greenspan78] Greenspan, S. & McGowan, C., "Structuring Software Development for Reliability," **Microelectronics and Reliability**, vol. 17, 1978.
- [Hamilton91] Hamilton, V.L. & Beeby, M.L., "Issues of Traceability in Integrating Tools". In: IEEE Colloquium Tools and Techniques for Maintaining Traceability during Design, 1991. **Proceedings**.
- [Hammer98] Hammer et al. "Doing Requirements Right the First Time". In: Software Technology Conference '98, April 1998, Salt Lake City, Utah. **Proceedings**. Disponível em http://satc.gsfc.nasa.gov/support/STC_APR98/do_reqmnt/do_reqmnt.pdf. Acesso em 23.09.2002.
- [Jarke98] Jarke, Mathias, "Requirements Tracing", **Communications of the ACM**, vol.41 (12), 1998. pp. 32-36.
- [Knethen02] Knethen, A. von & Paech, B. "A Survey on Tracing Approaches in Practice and Research". IESE-Report n° 095.01/E, 2002.
- [Kotonya98] Kotonya, Gerald & Sommerville, Ian, "Requirements Engineering", John Wiley & Sons, 1998. cap. 5, pp. 113-136.
- [Leite95] Leite, J. C. S. P. & Oliveira, A. P. "A client oriented requirements baseline". In: Second IEEE International Symposium on Requirements Engineering 95. **Proceedings**. pp. 108-115.
- [Leite97] Leite, Julio C. S. P. et al. "Enhancing a Requirements Baseline with Scenarios". **Requirements Engineering Journal**, vol. 2, n° 4 - Springer Verlag - December, 1998. pp. 184-198.
- [Leon00] Leon, Marco. "Staying on track". Intelligent Enterprise, September 2000. Disponível em http://www.intelligententerprise.com/000908/e_business.jhtml. Acesso em 31.03.2004.
- [Linscomb03] Linscomb, D. "Requirements Engineering Maturity in the CMMI". In: **CrossTalk**, dec 2003. pp. 25-28.
- [Lucena96] Lucena, C. J. P.; Leite, J.C.S.P.; Fernandes, J. R.; Gheiner, M. & Prado, A. F. "JSD/PUC: Um ambiente Experimental para Estudo do Processo de Automatização de Desenvolvimento de Software". In: **SBA Controle e Automação**, vol. 7, n° 3, 1996. pp. 126-146.
- [MacFarlane95] MacFarlane, I. A. & Reilly, Ian. "Requirements Traceability in an Integrated Development Environment". In: 2nd International Symposium on Requirements Engineering, York, England, 1995. **Proceedings**. IEEE Computer Society Press.
- [Mohan02] Mohan, K. & Ramesh, B. "Managing Variability with Traceability in Product and Service Families". In: **Proceedings** of the 35th Annual Hawaii International Conference on System Sciences (HICSS-35.02). Disponível em

<http://csdl.computer.org/comp/proceedings/hicss/2002/1435/03/14350076.pdf>. Acesso em 13.09.2004.

- [Oinas-Kukkonen96] Oinas-Kukkonen, H. "Hypermedia functionality in modeling tools". In: Second International Workshop on Incorporating Hypertext Functionality into Software Systems. **Proceedings**. Disponível em <http://www.sigweb.org/resources/HTF/HTFII/Oinas-Kukkonen.html>. Acesso em 11.10.2004.
- [Palmer97] Palmer, J.D., "Traceability". In: **Software Requirements Eng.**, R.H. Thayer and M. Dorfman, eds., 1997. pp. 364-374.
- [Paulk93] Paulk, M.C., et al.: "Key Practices of the Capability Maturity Model", Version 1.1 (CMU/SEI-93-TR-25). Pittsburgh, Pa.: Software Engineering Institute, Carnegie Mellon University, 1993.
- [Paulk95] Paulk, M.C. et al: **The Capability Maturity Model - Guidelines for Improving the Software Process**. Addison Wesley, Reading, Massachusetts, 1995.
- [Paulk95a] Paulk, Mark C "How ISO 9001 compares to CMM". **IEEE Software**, vol. 12 (1), Jan. 1995. pp. 74-83.
- [Pinheiro99] Pinheiro, F. "An Object Oriented Library for Tracing Requirements". In: WER99 - Workshop em Engenharia de Requisitos, Buenos Aires, Argentina, Setembro 9-10, 1999. **Anais**. pp 187-197.
- [Pinheiro04] Pinheiro, F. "Requirements Traceability". In: **Perspectives on Software Requirements** (Kluwer International Series in Engineering and Computer Science), by Julio Cesar Sampaio do Prado Leite, Jorge Horacio Doorn (Editors). Cap.5, pp. 91-113.
- [Potts88] Potts, C. & Bruns, G. "Recording the reasons for design decisions". In: 10th International Conference on Software Engineering, April 11-15, 1988, Singapore. **Proceedings**. pp. 418-427.
- [Ramesh01] Ramesh, B. & Jarke, M., "Towards reference Models for Requirements Traceability", **IEEE Trans. Software Eng.**, vol. 27(1), 2001. pp. 58-93.
- [Ramesh95] Ramesh, B.; Stubbs, C.; Powers, T. & Edwards, M. "Lessons Learned from Implementing Requirements Traceability". In: **CrossTalk**, abril de 1995. Disponível em <http://www.stsc.hill.af.mil/crosstalk/frames.asp?uri=1995/04/Lessons.asp>. Acesso em 31.10.2002.
- [Ramesh98] Ramesh, B., "Factors Influencing Requirements Traceability Practice", **Communications of the ACM**, vol. 41 (12), 1998. pp. 37-44.
- [Ross77] Ross, Douglas & Schoman, A. "Structured analysis for requirements definition". *IEEE Transactions on Software Engineering*. 1977. Vol. 3(1), pp. 6-15.
- [Santander00] Santander, Victor F. A. & Vasconcelos, Alexandre M.L., "Mapeando o Processo Unificado em Relação ao CMM - Nível 2". In: XI CITS - Qualidade de Software 18-21 Junho de 2000, Curitiba-PR, pp. 120-137. **Anais**.
- [Spanoudakis02] Spanoudakis, G. "Plausible and adaptive requirement traceability structures". In: 14th International Conference on Software Engineering and Knowledge Engineering, Ischia, Italy, jul 2002. **Proceedings**. pp. 135-142.
- [Sommerville98] Sommerville, I. **Software Engineering**, Addison-Wesley, Reading, MA, 1998.

- [Toranzo99] Toranzo, Marco & Castro, Jaelson, "The Multiview++ Environment: Requirements Traceability from perspective of stakeholders". In: WER' 99 - Workshop em Engenharia de Requisitos, Buenos Aires, Argentina, 1999. **Anais**. pp. 198-216.
- [Toranzo02] Toranzo, M.; Castro, J. & Mello, E. "Uma proposta para melhorar o rastreamento de requisitos". In: WER02 - Workshop em Engenharia de Requisitos, Valencia, Espanha, Novembro 11-12, 2002. **Anais**. pp. 194-209.
- [Weber97] Weber, K. C. **Qualidade e Produtividade em Software**, Makron Books, São Paulo, Brasil, 1997.
- [Zowghi01] Zowghi, D.; Damian, D. & Offen, R. "Field Studies of Requirements Engineering in a Multi-Site Software Development Organization: Research in Progress". In: Australian Workshop on Requirements Eng., Univ. of New South Wales, 2001. **Proceedings**. Disponível em http://www.cs.uvic.ca/~danielad/AWRE/Zowghi_AWRE.pdf. Acesso em 12.11.03.