Uma Proposta para Melhorar o Rastreamento de Requisitos

Marco Toranzo [1,2,3], Jaelson F. B. Castro [2] e Elton Mello [1]

[1] Universidade Estadual do Oeste do Paraná, Departamento de Informática,
[2] Universidade Federal de Pernambuco, Centro de Informática
[3] Universidade Paranaense, Sistemas de Informação

E-mail: {matc@cin.ufpe.br, jbc@cin.ufpe.br, emello@unioeste.br}

Abstract. Many contributions concerned with software quality (DoD-2167A, ISO 9000-3 and Capability Maturity Model) mandate that requirements traceability be practiced. However, they do not provide a model of what information should be captures and used as a part of requirements traceability model. This paper provides an framework to improve the requirements traceability.

Keywords. Requirement, Requirements traceability, Model and Process.

1. Introdução

A gerência de requisitos enfatiza estabelecer e manter um acordo com o cliente com relação aos requisitos a serem observados no projeto de software [Caputo, 1998]. Algumas das atividades da gerência de requisitos incluem: rastreamento de requisitos, controle de mudança, controle de versão e rastreamento do estado dos requisitos [Wiegers, 1999]. Na última década foram propostos vários trabalhos que visam solucionar muitos dos problemas que afetam o pré e pós-rastreamento de requisitos [Gotel, 1996]. Por exemplo: Gotel centraliza sua atenção no processo de produção e refinamento de requisitos; Pohl ([Pohl, 1996]) propõe um ambiente centrado em processo que inclui a questão do rastreamento; e Pinheiro ([Pinheiro, 1996a]) propõe uma abordagem formal (matemático) para o rastreamento. Em geral, podemos dizer que essas pesquisas:

1. Não fornecem um processo bem definido para desenvolver um modelo de rastreamento usando suas próprias propostas de trabalho;
2. Não fornecem uma classificação das informações que desejam rastrear, e que justifique as perguntas formuladas para identificar as informações a serem rastreadas;
3. Não fornecem e/ou não aplicam seus meta-modelos para desenvolver modelos de rastreamento particulares.

4. Só Ramesh ([Ramesh, 2001]) fornece modelos de referência para o rastreameto de requisitos.

O objetivo deste artigo, é apresentar uma proposta que visa solucionar os problemas citados acima. A Figura 1 apresenta nossas quatro estratégias para melhorar o rastreamento de requisitos.

Fig. 1. Visão da Proposta para melhorar o rastreamento de requisitos

As estratégias do nosso trabalho são: fornecer uma classificação das informações a serem rastreadas; um meta-modelo; um modelo intermediário de requisito que possui muitos dos artefatos que geralmente são encontrados nos modelos de rastreamento; e um processo que reúne e aplica as três estratégias anteriores.

A parte restante do artigo está estruturada como segue. Os trabalhos relacionados são apresentados na Seção 2. A classificação das informações dos modelos de rastreamentos de requisitos é apresentada na Seção 3. Em seguida, o meta-modelo da nossa proposta é apresentado na Seção 4. Um modelo intermediário para o rastreamento de requisitos é discutido na Seção 5. A proposta de um processo para criar um modelo de rastreamento para um projeto é apresentado na Seção 6. Uma validação do trabalho é apresentado na Seção 7. Finalmente, as conclusões são apresentadas na seção 8.

2. Trabalhos relacionados

Para desenvolver este trabalho, selecionamos e estudamos as contribuições dos seguintes pesquisadores: Orlena Gotel ([Gotel, 1996]), Matthias Jarke ([Jarke, 1998]), Balasubramaniam Ramesh ([Ramesh, 2001]), Klaus Pohl ([Pohl, 1996]) e Francisco Pinheiro ([Pinheiro, 1996a]). Em [Toranzo, 2001], foram identificados, definidos e aplicados um conjunto de critérios para comparar os trabalhos desses pesquisadores. A Figura 2 apresenta um quadro comparativo de alguns dos critérios aplicados [Toranzo, 2001]. Algumas das características comuns desses pesquisadores são: assumir que os artefatos que serão rastreados foram elicitados; não fornecem atividades bem definidas para instanciar seus próprios trabalhos; e estão preocupados mais com a questão *Como rastrear*? Este trabalho está mais preocupado com a questão *O que rastrear*?

Critérios de comparação		Identificação dos Pesquisadores					
		Toranzo	Ramesh	Gotel e Filkenstein	Jarke	Pohl	Pinheiro
1	Apresenta um meta-modelo para o rastreamento de requisitos	Sim	Sim	Não	Sim	Sim	Sim
2	Classifica as informações	Sim	Não	Não	Não	Não	Não
3	Propõe tipos de relacionamentos	Sim	Sim	Não	Sim	Sim	Não
4	Propõe modelo intermediário de rastreamento	Sim	Sim	Não	Não	Não	Não
5	Fornece um processo para construir um modelo de rastreamento	Sim	Não	Não	Não	Não	Não
6	Tratamento explícito de aspectos externos (político e econômico) de uma organização	Sim	Não	Não	Não	Não	Não
7	Tratamento explícito com aspectos organizacionais	Sim	Sim	Não	Não	Não	Não
8	Tratamento explícito da gerência de projeto.	Sim	Não	Não	Não	Não	Não

Fig. 2. Quadro comparativo das pesquisas sobre rastreamento de requisitos

A seguir apresentaremos uma classificação das informações a serem rastreadas.

3. Classificação das Informações a serem Rastreadas

Geralmente, as pesquisas apresentadas na seção anterior, formulam algumas perguntas básicas para identificar os elementos que serão rastreados, mas não fornecem nenhuma classificação dos elementos que justifiquem ou esclareçam as formulações das perguntas. Em [Pohl, 1996], foram estabelecidas algumas perguntas que são insuficientes para identificar todos os elementos de um modelo de rastreamento. Sabemos que os objetivos das perguntas de Pohl, são identificar as informações a serem rastreadas, mas surgem algumas perguntas: Qual tipo de informação identificar? Podemos classificar as informações para entender melhor a atividade de rastreamento? As atividades de gerência do projeto (tarefas) e do aspecto organizacional são consideradas?

Diante do problema identificado, propomos uma classificação composta de quatro níveis de informação: ambiental, organizacional, gerencial e desenvolvimento. A Figura 3 apresenta esses níveis de informação.

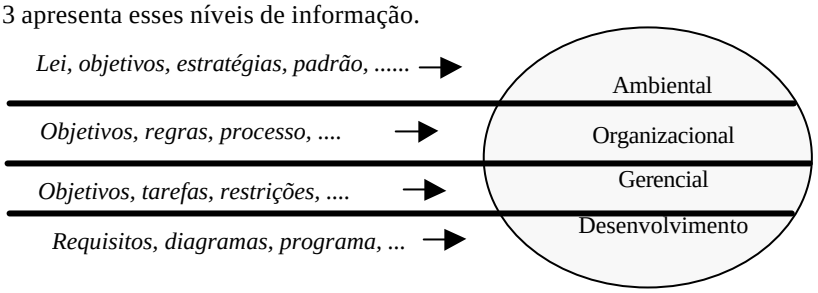

Fig. 3. Classificação da informação do rastreamento

A informação do nível ambiental representa os conceitos relacionados com o contexto político, econômico e padrões que podem afetar uma organização. Por exemplo, a lei da CPMF (um imposto sobre movimentação financeira), afetou vários sistemas financeiros dos bancos brasileiros. Note que no nível de informação ambiental, as informações são expressas geralmente em termos de objetivos e regras políticas e econômicas que precisam ser satisfeitas. Do ponto de vista do rastreamento de requisitos, a inclusão explícita do nível ambiental ajuda a tornar mais claro o que os autores Gotel, Pohl e Ramesh, podem denominar de fontes de informação.

O nível de informação organizacional, representa muito dos conceitos que originam o desenvolvimento e o crescimento de uma organização. Conseqüentemente, a compra e o uso dos sistemas computacionais para uma organização têm suas origens aqui, porque qualquer um deles é usado para agregar valor aos serviços dela. Um dos nossos objetivos relacionados com a introdução do nível de informação organizacional, é tornar explícito o fato de que podem existir alguns conceitos organizacionais, que são importantes para a atividade do rastreamento poder identificar e conhecer algumas das implicações (análise de impacto) desses conceitos sobre os requisitos dos sistemas.

No nível de informação gerencial, nosso trabalho está interessado no relacionamento entre as tarefas e os requisitos do sistema para permitir um melhor acompanhamento e controle dos requisitos do sistema. Alguns dos argumentos que fundamentam o relacionamento entre tarefas e requisitos são:

a) Os relacionamentos entre tarefas e requisitos contribuem à repetição do sucesso de um projeto. Através das ligações (instâncias dos relacionamentos) das tarefas com os requisitos, podemos identificar efetivamente os requisitos trabalhados nas tarefas;
b) Todas as pesquisas sobre o rastreamento de requisitos concentram-se na produção e refinamento de requisitos, e no desenvolvimento de software, sem preocupar-se de como o rastreamento poderia ajudar às outras atividades do desenvolvimento de um sistema;
c) A gerência de projeto de software deveria estar mais relacionada com os requisitos do software [Wiegers, 1999]. Os planos de projetos deveriam estar fundamentados na satisfação dos requisitos dos sistemas e nas mudanças dos requisitos que afetarão esses planos.
d) Para a gerência de projeto, o rastreamento de requisitos é essencial porque fornece um meio para demonstrar ao cliente que todos os requisitos acordados foram satisfeitos e que o trabalho foi concluído [Ramesh, 2001];
e) O rastreamento poderia contribuir para que alguns elementos dos níveis de informação ambiental e organizacional, possam relacionar-se com as tarefas do nível de informação gerencial. Por exemplo, um relacionamento entre as tarefas e os objetivos organizacionais de um sistema, pode contribuir na identificação das tarefas que ajudam a alcançar esses objetivos no decorrer de um projeto.

O nível de informação de desenvolvimento representa os elementos/artefatos produzidos nas diferentes atividades do desenvolvimento de software. Alguns dos elementos/artefatos são: documentos de requisitos, diagramas e programas. Este nível possui as atividades de pré e pós-rastreamento de requisitos, porque aqui estão

geralmente os elementos rastreados pela maioria das pesquisas de rastreamento, por exemplo: [Spence, 2000], [Leffingwell, 2000], [Ramesh, 2001] e [Wiegers, 1999]. A seguir apresentamos nosso meta-modelo de rastreamento.

4. Meta-modelo para o Rastreamento de Requisitos

A Figura 4 apresenta nosso meta-modelo para o rastreamento. Na mesma figura, a meta-classe base do meta-modelo é *Elemento* cujas subclasses são *ElementoGeneralizável* e *Relacionamento*.

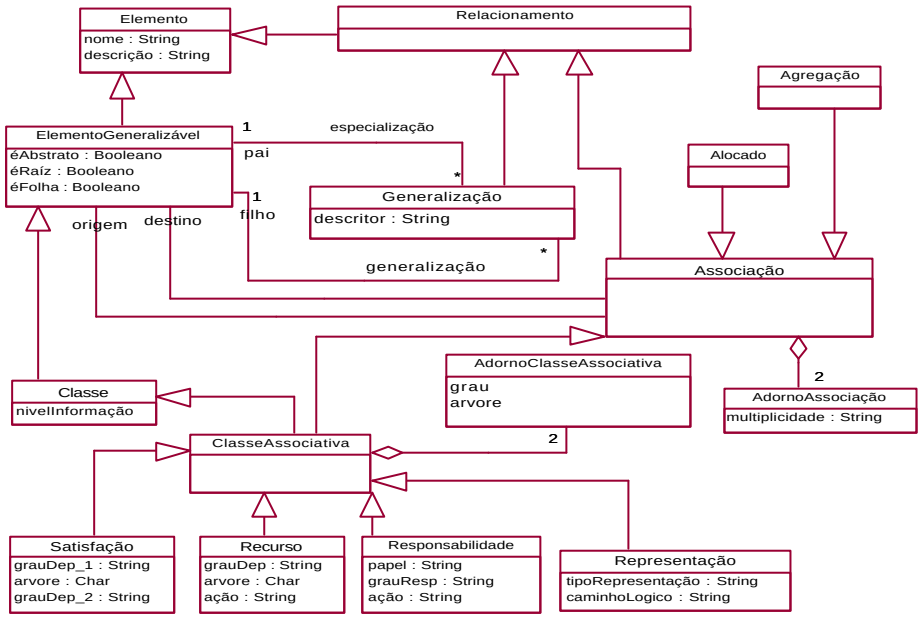

Fig. 4. Meta-modelo para o rastreamento

O *ElementoGeneralizável* classifica e caracteriza as instâncias da meta-classe *Classe* em subclasse, caso a especialização for usada, ou como superclasse, se a generalização for usada. Duas instâncias de *ElementoGeneralizável* se relacionam através da *Generalização* e Associação.

Os relacionamentos usados para rastrear informação são: satisfação, recurso, responsabilidade, representação, alocado, agregação e generalização.

Uma instância da meta-associação *Satisfação* é uma associação de dependência que especifica que a classe origem (início da associação) tem uma dependência de satisfação com a outra classe destino (a classe do outro extremo da associação).

Uma instância da meta-associação *Recurso* é uma associação de dependência que especifica que uma classe origem tem uma dependência de recurso com à classe destino, com o objetivo de manter consistente as informações contidas nas instâncias da classe origem. Entende-se por recurso, um elemento que representa uma

informação ou um elemento físico. Por exemplo, uma proposta de mudança usa como recurso de informação os requisitos do sistema.

Uma instância da meta-associação *Responsabilidade* visa capturar a participação, responsabilidade e ação das pessoas sobre artefatos ou elementos do processo de software. Por exemplo, a participação e responsabilidade de uma pessoa com relação a um susbsistema, pode ser classificada como analista e programador.

A representação dos requisitos nas linguagens de modelagem, ou de programação, é natural no processo de desenvolvimento de software. Para capturar a representação dos requisitos em outras linguagens, propomos uma associação chamada *Representação*.

Uma instância da meta-associação *Alocado* é uma associação que expressa que a classe origem está relacionada com uma instância da classe destino, que representa um subsistema.

Finalmente, o relacionamento de agregação ajuda a expressar que um elemento está composto de outros elementos.

5. Modelo Intermediário para o Rastreamento de Requisitos

Os objetivos desta seção são: fornecer uma visão geral do modelo intermediário e identificar alguns dos seus benefícios. Um modelo intermediário é um modelo de rastreamento, que não é básico, nem avançado para incluir todos os elementos de todos os possíveis modelos de rastreamento de diversos sistemas. Este modelo é resultado de uma combinação de fatores, tais como: boas práticas, estudos de casos, abstração, aplicação do nosso meta-modelo e dos níveis de informação, e uma extensão do trabalho de Ramesh ([Ramesh, 2001]). A Figura 5 apresenta o modelo intermediário. No modelo, existem algumas notações para identificar os diferentes tipos de relacionamentos: Satisfação (<*sat*>), Recurso (<*rec*>), Responsabilidade (<*resp*>), Representação (<*rep*>), e Alocado (<*alo*>).

Na Figura 5, a raiz da estrutura hierárquica é a classe *Informação* e algumas das suas subclasses são: *InformaçãoAmbiental*, *InformaçãoOrganizacional*, *ObjetivoSistema*, *Tarefa*, *Requisito*, *Diagrama*, *Programa* e *Subsistema*.

A classe *InformaçãoAmbiental* é uma abstração de todos os possíveis tipos de informações do nível de informação ambiental. A classe *InformaçãoOrganizacional* é uma abstração das informações do nível de informação organizacional. As classes *Tarefa* e *ObjetivoSistema*, representam algumas das informações do nível de informação gerencial. As classes: *Requisito*, *Diagrama*, *Programa* e *Subsistema*, representam algumas das informações do nível de informação de desenvolvimento.

O relacionamento recursivo *dep_recurso_para*, sobre a classe *Informação*, é herdado pelas suas subclasses. Conseqüentemente, o modelo de rastreamento de um projeto pode expressar diretamente uma relação de dependência de recurso entre as subclasses da classe *Informação*. Por exemplo, os objetivos, estratégias e/ou regras de uma organização podem ser recursos de informação para criar os objetivos de um sistema.

O relacionamento *dep_informação* (de *InformaçãoOrganizacional* para *InformaçãoAmbiental*) é uma abstração das possíveis dependências de recurso das

informações do nível de informação organizacional com as do nível de informação ambiental. Por exemplo, para o rastreamento de um sistema de biblioteca, um tipo de informação do nível de informação ambiental pode ser o *MARC* [Marc, 2002], que é um formato de visualização de obras.

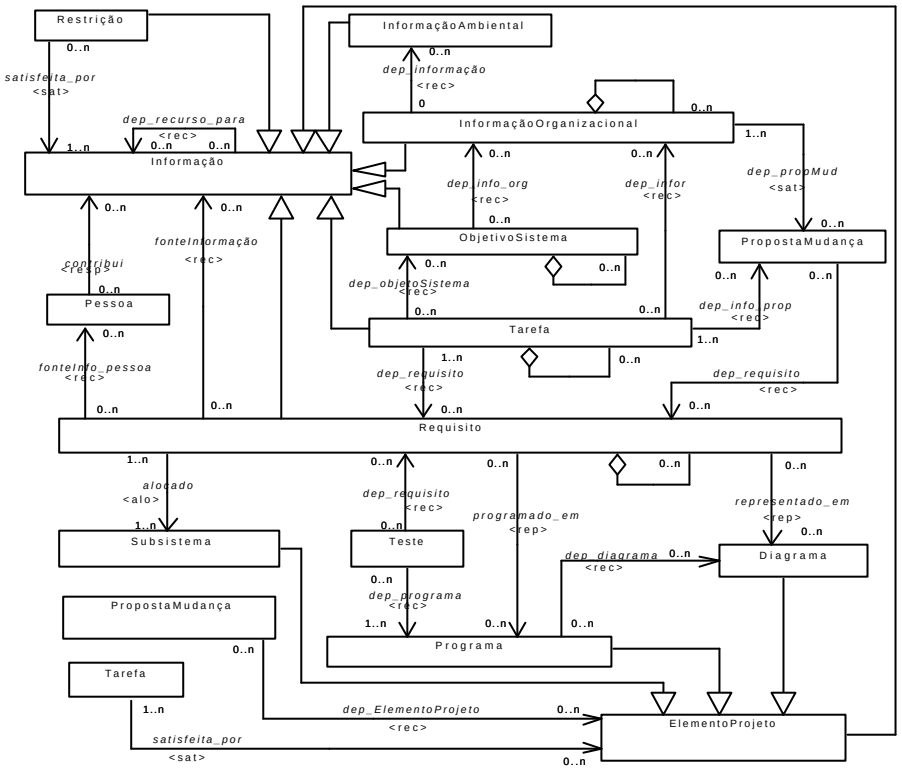

Fig. 5. Modelo intermediário para o rastreamento requisitos

O relacionamento *satisfeita_por* (de *Restrição* para *Informação*) expressa que as instâncias da classe *Restrição* devem ser satisfeitas por algumas instâncias das subclasses de *Informação*. O importante disso, é que devemos considerar as restrições impostas ao projeto de software. Por exemplo, uma restrição pode restringir que todos os requisitos que envolvam cálculos matemáticos, incluam as fórmulas.

O relacionamento *contribui* (de *Pessoa* para *Informação*) expressa que pessoas contribuem, ou são responsáveis pelas instâncias das subclasses da classe *Informação.* Por exemplo, pessoas são responsáveis pelo desenvolvimento de programas e requisitos.

Sabemos que a elaboração de um modelo de rastreamento pode necessitar de classes, relacionamentos e adornos não definidos no modelo intermediário. De fato, isto pode acontecer, porque o modelo intermediário não fornece todas as possíveis classes e relacionamentos usados em todos os modelos de rastreamentos dos

diferentes projetos de software. Entretanto, isto não invalida ou inviabiliza a aplicação do modelo intermediário, porque o mesmo pode ser melhorado e adaptado às necessidades dos projetos de uma organização.

Alguns dos principais objetivos do modelo intermediário são:

1. Servir como um ponto de partida para identificação, discussão e construção de um modelo de rastreamento.
2. Identificar as informações geralmente encontradas nos modelos de rastreamento dos diferentes projetos. Por exemplo, o modelo intermediário fornece uma visão do rastreamento mais próxima da realidade de um projetista de software, porque o modelo identifica os artefatos comuns da fase de projeto.
3. Contribuir na derivação dos modelos de rastreamento dos projetos de software.
4. Realçar a importância dos tipos de relacionamentos do meta-modelo, para incrementar o significado e entendimento dos modelos intermediários e particulares de um projeto. No modelo de rastreamento de projeto proposto por Ramesh ([Ramesh, 2001]), o significado atribuído aos relacionamentos é feito pela nomeação do próprio relacionamento. Por exemplo, você como leitor entende o significado dos relacionamentos chamados *direciona*, *faz* e *modifica*? Provavelmente não, mas poderá entende-los se observar o modelo de Ramesh. Com o exemplo, desejamos ilustrar que o significado atribuído a um relacionamento pelo nome definido pelo usuário pode ser claro para o seu autor, mas não necessariamente para outros usuários.

Na seção seguinte, apresentaremos e aplicaremos um processo para desenvolver um modelo de rastreamento.

6. Processo para Desenvolver um Modelo de Rastreamento

Os objetivos desta seção são apresentar e aplicar um conjunto de diretrizes do nosso processo, para desenvolver um modelo de rastreamento de requisitos. Os três importantes componentes do processo são:

1. A aplicação dos quatro níveis de informações.
2. O meta-modelo de rastreamento
3. O uso do modelo intermediário para o rastreamento de requisitos.

A apresentação do processo será feita sobre um sistema de biblioteca que está sendo desenvolvido e implantado em quatro campus da Universidade Estadual do Oeste do Paraná (UNIOESTE).

O sistema de biblioteca visa gerenciar e integrar os dados bibliográficos de todas as bibliotecas das diferentes sedes da UNIOESTE, localizadas em diversas cidades do Estado do Paraná. O sistema gerenciará todas as informações dos seus usuários (alunos, professores e funcionários), operadores (atendentes, bibliotecária, administradores), das obras (livros, multimídia, etc), circulação de obras (por exemplo, empréstimo, reserva, devolução), e fornecerá dados estatísticos referentes ao acervo e circulação das obras.

A matriz da Figura 6 apresenta as classes candidatas, identificadas para o modelo de rastreamento do sistema de biblioteca.

	Análise	Projeto	Implementação	Teste
Ambiental	InformaçãoFormato			
Organizacional	ObjetivoOrganizacional			
Gerencial	Pessoa ObjetivoSistema			
Desenvolvimento	Requisito	Diagrama, Subsistema	Programa	

Fig. 6. Organizando as classes candidatas

A primeira linha da matriz identifica as fases genéricas de um processo de software. A primeira coluna identifica os quatro níveis de informação. A justificativa da inclusão de cada uma das classes, da Figura 6, será fornecida na próxima Subseção.

6.1 Identificação e aplicação de Diretrizes para Desenvolvver um modelo de Rastreamento.

Uma das primeiras atividades para desenvolver e representar um conjunto de relacionamentos, é fornecer algumas convenções usadas no trabalho. Algumas delas são:

1. O nome e tipo da associação representada na matriz são colocados na parte superior esquerda da matriz. A representação do tipo de uma associação no modelo de rasteamento é expresso através de um rótulo. Por exemplo, a Figura 7 apresenta o rotulo "<*rec*>" para representar uma associação de tipo de recurso.
2. Junto com o nome da associação colocaremos o símbolo de uma seta para identificar as instâncias da classe destino na associação. Se for usado o símbolo "$\leftarrow$", então as instâncias declaradas na primeira fila da matriz são instâncias da classe destino. Caso contrário ("$\rightarrow$"), as instâncias declaradas na primeira coluna da matriz são instâncias da classe destino. Por exemplo, a Figura 7 indica que o nome do relacionamento representado do modelo de rastreamento do sistema de biblioteca (Figura 13) é dep_rec_org.
3. Na primeira coluna e primeira fila das matrizes de rastreamento, usaremos expressões para identificar as instâncias de uma classe.

A seguir, apresentamos as expressões usadas para identificar as diferentes instâncias das classes que formaram parte do modelo de rastreamento.

[OBO] = Objetivos organizacionais
[OBS] = Objetivos do sistema
[PRO] = Programa do sistema
[DGM] = Diagramas

[REQ] = Requisitos do sistema [FUN] = Funcionário
[FMT] = Formato de catalogação [SUB]= Subsistema do sistema

A Figura 7 apresenta um exemplo do uso das expressões.

Dep_rec_org:<rec> ←	[OBS-2]	[OBS-4]	[OBS-5]	[OBS-6]	[OBS-8]
[OBO-1]	<A;A;cri>				
[OBO-2]		<A;A;cri>			
[OBO-3]					
[OBO-4]			<A;A;cri>		
[OBO-7]					<A;A;cri>
[OBO-8]					
[OBO-9]				<A;A;cri>	
[OBO-10]					

Fig. 7. Exemplo do uso da nomesclatura

Na matriz da Figura 7, as expressões *[OBO-1]*, *[OBO-2],…,[OBO-10]* são instâncias da classe *ObjetivoOrganizacional*, que representa os objetivos organizacionais do sistema de biblioteca. As expressões *[OBS-2]*, *[OBS-4],..,[OBS-8]* são instâncias de uma classe *ObjetivoSistema*, que representa objetivos do sistema. Na mesma figura, a interseção da primeira fila e da primeira coluna identifica o nome do relacionamento (*dep_rec_org*) e seu tipo (recurso). Na mesma interseção, a seta indica que a classe destino é a classe *ObjetivoOrganizacional*.

Considerando o modelo intermediário de rastreamento de requisitos, preenche-se a tabela da Figura 6 com os resultados obtidos com as seguintes diretrizes.

Diretriz 1: Identificar as informações que podem afetar o sistema. O objetivo da diretriz é identificar classes no nível de informação ambiental que podem afetar o sistema.

O resultado da aplicação dessa diretriz, foi a inclusão da classe *InformaçãoFormato* para referenciar as informações dos formatos de visualização e de catalogação de obras, chamados *MARC* (*MAchine-Readable Cataloging*) e *AACR* (*Anglo-American Cataloging Rules*).

Diretriz 2: Identificar os objetivos, estratégias ou regras de negócio que serão rastreadas. O objetivo dessa diretriz é identificar os objetivos, estratégias ou regras registradas que podem afetar o sistema desejado.

Após aplicar essa diretriz, foi decidido incluir a classe *ObjetivoOrganizacional* do modelo intermediário, no modelo de rastreamento do sistema de biblioteca, porque os objetivos organizacionais foram registrados e disponibilizados. Alguns exemplos dos objetivos organizacionais do sistema de biblioteca são apresentados na Figura 8.

[OBO-1]	Integrar as informações do acervo bibliográfico das bibliotecas das diferentes sedes da UNIOESTE
[OBO-2]	Controle dos diversos tipos de materiais existentes na biblioteca (Monografias, Periódicos, Multimídias)
[OBO-3]	Gerenciar e centralizar todas as informações de todos os usuários do sistema de biblioteca
[OBO-4]	Controlar o acesso e uso do sistema de biblioteca

Fig. 8. Exemplo de objetivos organizacionais do sistema de biblioteca

Diretriz 3: Incluir classes de informação da gerência de projeto no modelo de rastreamento. O objetivo dessa diretriz é recomendar a inclusão das classes que representarão as tarefas do cronograma do projeto, os objetivos do sistema e as Pessoas (desenvolvedores), no modelo de rastreamento do projeto. Com isso desejamos reduzir o *gap* da gerência de projeto com a gerência de requisitos. A classe *Tarefa* não foi incluída porque não foram registradas as atividades para desenvolver o sistema. A Figura 9 apresenta algumas informações dos objetivos do sistema.

[OBS-1]	Controlar débitos referentes ao empréstimo de obras
[OBS-5]	Gerenciar o controle de acesso.
[OBS-6]	Gerenciar o empréstimo e reserva das obras.
[OBS-8]	Emissão de relatórios estatísticos sobre a circulação dos materiais do acervo da biblioteca.
[OBS-11]	Gerenciar as classificações dos diversos materiais do acervo, atendendo as tabelas CDD (*Classificação decimal Dewey*) ou CDU (Classificação decimal universal)
[OBS-12]	Gerenciar a catalogação de obras, atendendo as normas especificadas no AACR (Anglo-American Cataloging Rules) e MARC (*Machine-Readable Cataloging*)

Fig. 9. Objetivos do sistema

Diretriz 4: Identificar subsistemas. O objetivo dessa diretriz é identificar os subsistemas que compõem o sistema. O resultado da aplicação da diretriz, foi a inclusão da classe *Subsistema* para registrar e representar os subsistemas do sistema de biblioteca. A Figura 10 apresenta alguns dos subsistemas do sistema de biblioteca.

Código	Identificação	Descrição
[SUB-1]	Gerência de Obra	Manipulaçao das obras da biblioteca
[SUB-2]	Gerência de Usuário	Controla as informações dos usuários
[SUB-3]	Gerência Circulação de obra	Controla os empréstimos, reserva, devolução e consultas locais

Fig. 10. Subsistemas do sistema de biblioteca

Diretriz 5: Identificar requisitos. É obrigatória a inclusão da classe Requisito, do modelo intermediário, no modelo de rastreamento do sistema de biblioteca. Portanto, foi incluída a classe Requisito. A Figura 11 apresenta alguns requisitos do sistema.

[REQ-1]	O sistema deverá permitir a inclusão de usuários com os seguintes campos de informação: Código de Pessoa, Nome Completo, Endereço,……
[REQ-2]	O sistema deverá permitir a inclusão de dados referentes ao usuário, caso este seja um operador do sistema de bibliotecas com o seguinte campo de formação: Nível de acesso.
[REQ-3]	O sistema deve permitir a consulta dos dados do usuário pelos campos: Nome, Sobrenome ou Nº de identificação. O retorno da consulta mostra todos os dados do usuário.
[REQ-4]	O sistema deve permitir alterar os dados do usuário. O único campo que não pode ser alterado é o Código da Pessoa Física.

Fig. 11. Requisitos do sistema

Diretriz 6: Identificar diagramas. Os objetivos desta diretriz são: identificar os tipos de diagramas nos quais foram ou serão representados os requisitos do sistema, e definir os caminhos lógicos para acessar as representações dos requisitos nos diferentes tipos de diagrama.

O resultado da aplicação da diretriz foi a inclusão da classe *Diagrama*, que representará todos os diagramas nos quais foram modelados os requisitos. A Figura 12 apresenta a definição do caminho lógico para acessar um requisito em um diagrama de classe. No caminho lógico dos requisitos para os diagramas de classes somente é colocado o nome da classe.

Tipo referência	Caminho lógico
de requisito para classe	"C:nome_classe", onde "C" é uma abreviação para classe e "nome_classe" é o nome de uma classe

Fig. 12. Definição do caminho lógico para acessar requisitos no diagrama de classe

Diretriz 7: Identificar programas. O objetivo dessa diretriz é identificar os programas nos quais foram representados (implementados) os requisitos do sistema e definir um caminho lógico para acessar a representação dos requisitos nos programas.

O resultado da diretriz é a inclusão da classe *Programa* e a elaboração de caminhos lógicos para acessar a implementação dos requisitos.

Diretriz 8: Identificar teste. O objetivo desta diretriz é identificar os documentos, ou os fragmentos de documentos, que especificam como os requisitos foram ou serão testados. Não foi incluída a classe *Teste* porque os testes não foram documentados no desenvolvimento do sistema biblioteca.

Diretriz 9: Remover as classes de informação irrelevantes.
Diretriz 10: Integrar as classes com o mesmo significado.
Diretriz 11: Integrar novas classes.

As diretrizes 9, 10 e 11 foram aplicadas para desenvolver e identificar as classes candidatas do modelo de rastreamento.

Diretriz 12: Organizar as classes. O objetivo dessa diretriz é organizar e estruturar as classes candidatas em forma hierárquica. Veja a Figura 13.

Diretriz 13: Estabelecer relacionamentos. O objetivo é relacionar as classes candidatas. Veja a Figura 13.

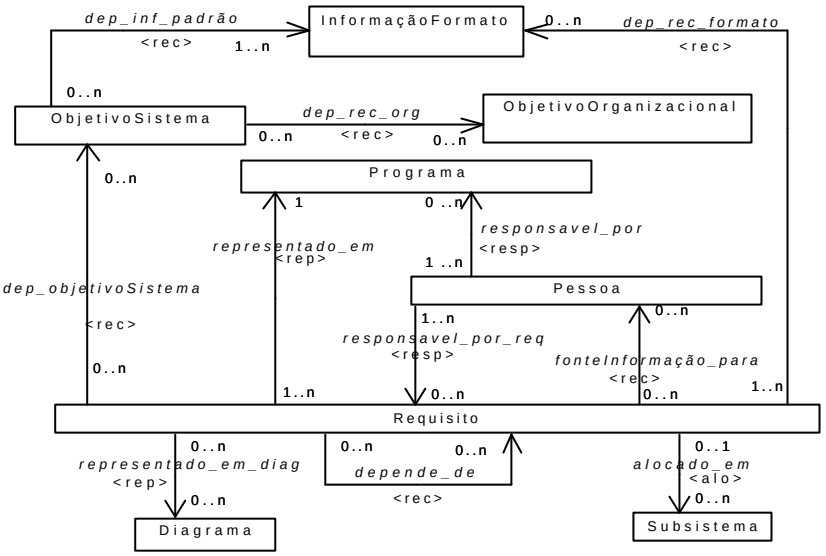

Fig. 13. Modelo de rastreamento para o sistema de biblioteca

Diretriz 14: Recomendar atributos sobre as classes. O objetivo é sugerir alguns atributos geralmente encontrados nos elementos rastreados.

Diretriz 15: Definir uma matriz para cada relacionamento do modelo. O objetivo desta diretriz é desenvolver várias matrizes ou listas para representar os relacionamentos do modelo de rastreamento. A Figura 7 é um exemplo da representação matricial de um relacionamento.

7. Validação das Matrizes

A validação das matrizes consistirá em fazer análise de impacto. Para isso, considere o seguinte cenário que diz que a UNIOESTE decidiu pela retirada do padrão *MARC* do sistema de biblioteca porque será usado um outro padrão. Portanto, deveremos identificar as partes afetadas do sistema. A seguir, aplicamos alguns passos para identificar os elementos afetados.

1. Revisar a representação matricial do relacionamento *dep_rec_org* (Figura 7). Após uma revisão foi determinado que os objetivos do sistema afetados foram [OBO-11] e [OBO-12]. Logo, deve-se examinar a representação matricial dos relacionamentos *dep_rec_formato* e dep_objetivoSistema, para identificar os requisitos afetados.
2. Após revisar os objetivos [OBO-11] e [OBO-12] da representação matricial do relacionamento *dep_rec_formato*, identificamos que os requisitos afetados foram: [REQ-102], [REQ-106], [REQ-501], [REQ-502], [REQ-503], [REQ-504], [REQ-302] e [REQ-401]. Muitos dos requisitos não foram incluídos por limitações de espaço do artigo. Uma versão mais completa estará disponível em www.inf.unioeste/~toranzo.
3. Com a identificação dos requisitos afetados, podemos identificar os diagramas e programas afetados. Para identificar os programas afetados, deve-se revisar a representação matricial do relacionamento *representado_em* (Figura 13). A Figura 14 apresenta os requisitos e programas afetados. Nos caminhos lógicos, a expressão "*imp*" expressa que se trata da implementação de requisitos. O símbolo "U" representa o nome da unidade definida pelo usuário no programa. O símbolo "P" representa o nome do procedimento dentro da unidade. Por exemplo, o requisito [REQ-102] foi implementado na unidade *fConsObra*, especificamente, no procedimento *TF_ConsObra.EditConsTituloKeyPress*.

Requisitos	Programas (Caminho lógico)
[REQ-102]	<imp,U: fConsObra; P: TF_ConsObra.EditConsTituloKeyPress(Sender: TObject; var Key: Char);
[REQ-106]	<imp,U: fRelItemClassificacao; P: TF_RelItemClassificacao.BtRelacionarClick(Sender: TObject)>
[REQ-302]	<imp,U: fRelatorioObra; P: TF_RelatorioObra.Frame_Botao_ImprimirEdBtImprimirClick(Sender: TObject)>
[REQ-401]	<imp,U: fRelatorioFicha; P: TF_RelatorioFicha.BtImprimeMatrizClick(Sender: Tobject)>
[REQ-501]	<imp,U:unit fCadClassificacao; P:TF_CadClassificacao.Frame_Botoes_CadastroBtNovoClick (Sender: TObject);
[REQ-502]	<imp,U:unit fConsClassificacao; P:TF_ConsClassificacao.EditPesquisaChange(Sender: TObject);
[REQ-503]	<imp,U:unit fCadClassificacao; P:TF_CadClassificacao.Frame_Botoes_CadastroBtEditarClick(Sender: TObject);
[REQ-504]	<imp,U:unit fCadClassificacao; P: TF_CadClassificacao.Frame_Botoes_CadastroBtExcluirClick(Sender: TObject);

Fig. 14. Lista de requisitos e programas afetados

8. Conclusões

As conclusões do nosso trabalho podem ser resumidas como seguem:

1. Fornecemos uma classificação das informações que podem ser rastreadas. A mesma classificação pode servir como um ponto de partida para elicitar e validar as informações. Estas podem formar parte de um modelo de rastreamento de um sistema.
2. Fornecemos um meta-modelo que indica os tipos de relacionamentos que podem ser estabelecidos entre as classes que representam as informações rastreadas de um sistema.
3. Definimos tipos de relacionamentos que podem ser usados para desenvolver um modelo de rastreamento. Somente Ramesh ([Ramesh, 2001]) trata este problema, mas, mesmo assim, o próprio autor não mostra como usar os relacionamentos. O benefício dos tipos dos relacionamentos é que seu uso evita que o significado atribuído aos relacionamentos nos modelos de rastreamento sejam definidos por um nome definido pelo usuário, porque o mesmo autor pode ter claro o significado, mas não outras pessoas. Como exemplo, é só examinar e tentar entender os diferentes relacionamentos dos modelos de rastreamento existentes na literatura.
4. O trabalho apresentou um modelo intermediário para o rastreamento de requisitos. Em lugar de partir do zero todas as vezes que deve ser desenvolvido um modelo de rastreamento, o modelo intermediário propõe classes de informações que podem ajudar a reduzir e/ou melhorar a elicitação das informações que formarão parte de um modelo de rastreamento. Porém, o modelo intermediário por si mesmo não é suficiente, é necessário um processo que organize as atividades necessárias para desenvolver um modelo de rastreamento.
5. Apresentamos e ilustramos um processo para desenvolver um modelo de rastreamento para um sistema de Biblioteca, que está sendo implantado na Universidade Estadual do Oeste do Paraná. Através do processo, mostramos que é possível desenvolver com passos simples e sistemáticos, um modelo de rastreamento para um projeto.

Finalmente, este trabalho foi aplicado em vários projetos desenvolvidos com e sem orientação a objeto. Em ambos, os estudos de casos têm contribuído para melhorar várias partes do nosso trabalho. Futuros trabalhos visam relatar os resultados obtidos em nossos estudos de casos e nas aplicações feitas em [Castro, 2002] e [Pinto, 2001].

Referências

[Caputo, 1998] K.Caputo. CMM Implementation Guide: Choreographing Software Process Improvement. Reading, MA: Addison-Wesley, 19998

[Castro, 2002] J. Castro, R. Pinto, A. Castor, " Requirements Traceability in Agent Oriented Development: The Tropos Case". Submetido ao ICSE´2002

[Gotel, 1996] O. Gotel. Contribution Structures for Requirements Engineering. Ph.D Thesis. Department of Computing, Imperial College of Science, Technology, and Medicine, London, U.K., 1996.

[Jardim, 1999] N. Jardim, M. Toranzo, J. Cunha, J. Castro, S. Kovacecic, R. Dave, J. Tarby, M. Collinscope, and M. Harmelen. Workshop an Interactive Systems Design and Object Models (Wisdom'99). In: 13th Conference European on Object, 1999, Lectures on Computer Ciences. 1999. v.1. p.34-58.

[Jarke, 1998] M. Jarke, "Requirements Tracing," Communication ACM, vol. 41, pp. 32-36, Dec. 1998.

[Leffingwell, 2000] D. Leffingwell, and D. Widrig. Managing Software Requirements: A unified Approach. Addison-Wesley. 2000.

[Marc,2002] MARC 21 Concise Format. Library of Congress Network Development and MARC Standards Office. http://lcweb.loc.gov/marc/concise/concise.html#general_intro. Pagina visitada em 2/2/2002

[Paulk, 1993] M. Paulk, M. Chissis, and C. Weber. Capability Maturity Model for Software: Version V1.1. Technical report SEI-93-TR-24. Software Engineering Institute, Carnegie Mellon University, Pennsylvania, USA, Feb. 1993

[Pinheiro, 1996a] F. A. C. Pinheiro. Design of a Hyper-Environment for Tracing Object-Oriented Requirements. Ph.D Thesis. Department of Computing, University of Oxford, Oxford, U.K., 1996.

[Pinto, 2001] Pinto, R. C. C., Castro, J. F. B., Toranzo, M. A., "Requirements Traceability in Agent Oriented Development" SELMAS2002 - International Workshop on Software Engineering for Large-Scale Multi-Agent Systems, 2002, Orlando

[Pohl, 1996] K. Pohl. Process-Centered Requirements Engineering. Advanced Software Development Series, John Wiley and Sons Ltd., 1996.

[Ramesh, 2001] B. Ramesh and M. Jarke, "Towards Reference Models For Requirements Traceability," IEEE Transacions on Software Eng., vol. 27, pp. 58-93, Jan. 2001

[Spence, 2000] I. Spence. And L. Probasco. Traceability Strategies for Requirements Management with Use Cases . White Papers, 2000. Rational Software Corporation

[Toranzo, 1999] Toranzo, M., Castro, J.(1999a), A Comprehensive Traceability Model to Support the Design of Interactive Systems, WISDOM'99 Workshop, http://math.uma.pt/wisdom99. in European Conference on Object-Oriented Programming- ECOOP'99

[Toranzo, 2000] M.Toranzo, "Towards to Reference Models for Requirements Traceability," III Fórum de Tecnologia: X Seminário Regional de Informática. Universidade Regional Integrada do Alto Uruguai e das Missões, 2000

[Toranzo, 2001] Toranzo, M. *A Framework to Improve Requirements Traceability* (in Portuguese: Um Framework para Melhorar o Rastreamento de Requisitos). Ph.D thesis, forthcoming, Centro de Informática da UFPE, Brazil, 2002.

[Wiegers, 999] K. Wiegers. Software Requirements. Microsoft Press, 1999.