UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

JERÔNIMO BACKES

Rastreabilidade Semi-Automática Através do Mapeamento de Entidades

Dissertação apresentada como requisito parcial
para a obtenção do grau de
Mestre em Ciência da Computação

Prof. Dr. Daltro José Nunes
Orientador

Porto Alegre, novembro de 2008

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Backes, Jerônimo

Rastreabilidade Semi-Automática Através do Mapeamento de Entidades / Jerônimo Backes. – Porto Alegre: PPGC da UFRGS, 2008.

96 f.: il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2008. Orientador: Daltro José Nunes.

1. Engenharia de software. 2. Engenharia de requisitos. 3. Rastreabilidade. I. Nunes, Daltro José. II. Título.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

CMMI	Capability Maturity Model Integration
ER	Entidade-Relacionamento
GQM	Goal Question Metric
ISO	International Organization for Standardization
LSI	Latent Semantic Indexing
MR	Matriz de Rastreabilidade
RAR	Recuperação Automática de Rastreabilidade
RI	Recuperação de Informações
RNF	Requisito Não Funcional
RR	Rastreabilidade de Requisitos
SQL	Structured Query Language
TP	Traceability Path
UML	Unified Modeling Language
VBRT	Value-Based Requirements Tracing
XMI	XML Metadata Interchange
XML	eXtensible Markup Language
XTM	XML Topic Map

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

Entre os fatores que geram o alto custo da rastreabilidade, está a dificuldade na criação e manutenção de relações precisas entre artefatos. Praticamente todas as metodologias existentes preocupam-se em relacionar artefatos diretamente entre si, o que dificulta o uso de processos automatizados na derivação de novos relacionamentos, bem como na manutenção dos já existentes. Com base nestas observações, o presente trabalho propõe o uso de estruturas intermediárias nos relacionamentos, chamadas de entidades, para representar os interesses tratados pelos artefatos, e derivar, automaticamente, relações complexas entre os mesmos. Este modelo foi avaliado por profissionais da indústria e apresentou-se como solução de rastreamento viável, em comparação com as tradicionais matrizes de rastreabilidade. Espera-se que este sirva como base para soluções inovadoras na área, que ainda é considerada problemática.

Palavras-chave: Engenharia de software, engenharia de requisitos, rastreabilidade.

Semi-automated traceability by entity mapping

ABSTRACT

The difficulties in creation and maintenance of precise relationships between artifacts are the root cause of traceability high cost. Virtually all existing methodologies propose solutions that relate artifacts directly among themselves, what hinders the use of automated processes to derivate new relationships, as well as maintaining existing ones automatically. Based on these observations, this work proposes the use of intermediate structures, called entities, to represent the interests present on traced artifacts. These entities can be used to derive, automatically, complex relationships between artifacts. The proposed model was evaluated by industry professionals and was considered by them as a viable solution for traceability when compared to the traditional traceability matrices. It is believed that this model will serve as a basis for innovative research solutions in traceability, which is still considered a problematic field.

Keywords: Software engineering, Requirements engineering, Traceability.

1 INTRODUÇÃO

No desenvolvimento de software, uma vasta gama de artefatos é gerada e mantida: documentação, requisitos, modelos de projeto, casos de teste, etc. Todos utilizados para compreender melhor o sistema (KELLEHER; SIMONSSON, 2006). A evolução destes artefatos e das necessidades identificadas caracteriza o ciclo de vida de qualquer sistema. As atividades relacionadas à evolução de um software ocasionalmente produzem efeitos colaterais nas funcionalidades e artefatos já existentes, tornando-os falhos ou até mesmo inutilizando-os (ANTONIOL et al., 2005).

Esta evolução requer uma melhor compreensão dos requisitos, o que somente pode ser obtido através do rastreamento dos requisitos de volta para suas origens. A rastreabilidade de requisitos (RR) provê a habilidade de associar itens nas especificações de requisitos com itens das especificações de projeto. Com esta capacidade, em casos de necessidade de alteração, é possível determinar custos e prazos mais precisamente, sem exigir que engenheiros de software e programadores conheçam todas as áreas afetadas pela mudança (RAMESH; JARKE, 2001).

Em suma, a RR é um meio importante que facilita, entre outros, a comunicação entre *stakeholders*, a determinação do impacto de alterações e o suporte à realização destas, preservando o conhecimento e as dependências criadas durante o projeto. Este meio é útil na garantia da qualidade, e na prevenção de mal-entendidos (EGYED; GRUNBACHER, 2002; KELLEHER; SIMONSSON, 2006), sendo bastante reconhecida pelos engenheiros de software no suporte a atividades críticas, como validação de requisitos, seleção de casos de teste de regressão, verificação de conformidade, etc. (CLELAND-HUANG, 2006a).

O objetivo do gerenciamento de requisitos e da rastreabilidade é, durante o ciclo de vida do produto, organizar, analisar, e rastrear os requisitos que estes artefatos satisfazem (DELGADO, 2006). O rastreamento pode ser visto como um investimento na tentativa de tornar explícitos e usáveis os relacionamentos entre requisitos e demais artefatos, servindo como base para atividades de gerenciamento e desenvolvimento (HEINDL; BIFFL, 2006).

Aos desenvolvedores e responsáveis pela manutenção do sistema, a RR permite identificar o que acontecerá quando uma alteração for implementada. Com o uso desta técnica, as implicações de uma alteração podem determinadas antes que o reprojeto do sistema seja realizado. Portanto, a ausência da clara rastreabilidade de um projeto para os respectivos requisitos pode criar sérios problemas de manutenção, bem como levar à postergação de prazos de entrega. Os benefícios do uso da RR são óbvios quando o sistema deve ser alterado: A partir do instante em que requisitos de alto nível mudam, descobre-se quais objetos de baixo nível devem mudar. Este ponto, por si só, justifica os trabalhos na área de rastreabilidade (SALEM, 2006).

A RR é o meio mais efetivo de alinhar evolução de sistemas com necessidades de usuários em constante mudança (JARKE, 1998). Assim, esta é definida como a capacidade de descrever e seguir a vida de um requisito, de forma bidirecional através de todos os estágios do desenvolvimento. Quatro tipos de elos de rastreabilidade são distinguidos, levando em conta seu relacionamento com os requisitos:

Adiante, a partir dos requisitos (*Forward from Requirements*) A responsabilidade pelo atendimento a um requisito deve ser atribuída a componentes do sistema, de tal forma que seja possível avaliar o custo e o impacto de alterações sobre o requisito;

De volta para os requisitos (*Backward to Requirements*) Os artefatos desenvolvidos, bem como todo o sistema, devem estar de acordo com os requisitos, evitando o *gold-plating* – elementos do projeto ou do sistema que não possuem requisitos, e que geralmente são desnecessários;

Adiante, em direção aos requisitos (*Forward to Requirements*) Alterações nas necessidades dos *stakeholders*, bem como decisões técnicas, podem requerer a reavaliação da relevância dos requisitos;

De volta, a partir dos requisitos (*Backward from Requirements*) As estruturas que alicerçam os requisitos são cruciais na validação destes, especialmente em ambientes onde a política exerce alta influência.

Os primeiros dois tipos de elo são geralmente classificados como elos de pós-rastreabilidade, pois ligam requisitos ao projeto e implementação, assinalando responsabilidades, permitindo verificar a satisfação dos requisitos e realizar análises de impacto sobre estes. Já os dois últimos habilitam a pré-rastreabilidade, documentando os motivos e o contexto sócio-político a partir dos quais os requisitos emergem.

Os principais objetivos da rastreabilidade, portanto, se resumem a demonstrar que:

- cada requisito foi satisfeito;
- cada componente do sistema satisfaz pelo menos um requisito.

Assim, uma das principais utilidades da RR é servir aos desenvolvedores como prova para o cliente de que os requisitos foram bem compreendidos, que o produto corresponde a estas especificações e não possui funcionalidades desnecessárias. Muitos padrões para desenvolvimento de sistema exigem a prática da RR. Esta é um dos requisitos para uma organização alcançar o CMMI nível 3 – sendo que a maioria das organizações encontram-se nos níveis 1 ou 2 (HEINDL; BIFFL, 2005).

Não é necessário discutir o valor potencial que esta atividade é capaz trazer ao desenvolvimento de sistemas, afinal, a literatura relata que um dos maiores problemas na tarefa de manutenção está relacionado à compreensão de como o sistema funciona: alguns estudos apontam que de 30% a 60% dos custos nesta fase estão relacionados à descoberta do que o sistema faz (TRYGGESETH; NYTRO, 1997). Contudo, a rastreabilidade é raramente utilizada, já que em muitos casos o custo envolvido não cobre os benefícios. Este alto custo vem:

1. da dificuldade em gerar automaticamente as relações, com uma semântica clara e precisa;

2. da heterogeneidade e do grande número de artefatos que são criados durante o desenvolvimento; e

3. a falta de corretude e completude das relações de rastreabilidade (CLELAND-HUANG, 2006a).

Na prática, uma variedade de problemas de rastreabilidade ocorrem pelo uso de métodos informais (SALEM, 2006). Além disso, a pressão para reduzir o tempo de entrega e aumentar a produtividade resulta na adaptação de processos para esta realidade de busca por alto desempenho, que sacrifica atividades consideradas não-produtivas. O rastreamento e manutenção da consistência entre os artefatos de software está entre essas atividades custosas e tediosas, freqüentemente negligenciadas (PENTA; GRADARA; ANTONIOL, 2002).

Como resultado dos problemas de manutenção, um grande número de pesquisadores vem investigando o uso da recuperação automática da rastreabilidade. Porém, não existe processo de recuperação da rastreabilidade completamente automático (CLELAND-HUANG, 2006b). Todos incluem um componente humano, pois as decisões (de aceitação ou rejeição) sobre elos recuperados devem ser tomadas pelo usuário. Além disso, uma vez que os elos de rastreabilidade sejam recuperados, devem ser mantidos de forma consistente, para que possam ser utilizados em outros processos e tarefas .

1.1 Problema

Este estudo investiga o problema relacionado às atividades de manutenção dos relacionamentos de rastreabilidade. Os métodos de RR existentes focam-se na criação de estruturas de relacionamentos. Embora úteis, o problema é evoluir estas estruturas (MURTA; HOEK; WERNER, 2006). Entre as soluções mais aplicadas para encontrar o melhor *tradeoff* entre tamanho e utilidade das informações de RR, estão rastrear somente requisitos críticos e reduzir a granularidade. Infelizmente, nenhuma destas é satisfatória, já que uma pequena alteração pode afetar diversas partes de um sistema, e a análise de impacto deve contar com o maior número de detalhes possível para que os efeitos colaterais possam ser detectados com precisão. Muitos esquemas de rastreabilidade fornecem pouco suporte para identificar o impacto de requisitos totalmente novos (CLELAND-HUANG et al., 2002).

Foram analisadas diversas propostas, em cerca de 250 artigos, tratando do estabelecimento e manutenção de relacionamentos, e percebeu-se que praticamente todas preocupam-se com a criação de relações diretas entre requisitos e artefatos. Porém, constatou-se que relacionamentos diretos ocultam os interesses que ambos artefatos tratam, dificultando a identificação das razões que motivam a existência dos elos, bem como inibindo a eficácia de processos automatizados para criação e manutenção dos mesmos. As decisões sobre o estabelecimento ou não de relacionamentos entre artefatos devem ser tomadas pelo usuário (MARCUS; XIE; POSHYVANYK, 2005), porém, assegurar uma qualidade aceitável nos elos, sem pressionar demasiadamente os desenvolvedores, é um problema desafiador, que carece de maiores pesquisas (NEUMULLER; GRUNBACHER, 2006). A grande maioria das propostas na área utilizam uma das duas técnicas:

1. **Rastreamento Manual:** geralmente usando matrizes de rastreabilidade, que simplesmente exibem relacionamentos entre artefatos. As tentativas de sua utilização pela indústria, resultaram, quase que invariavelmente, em fracaso (CLELAND-HUANG, 2006a). As informações mantidas geralmente restringem-se a simplesmente "um artefato está relacionado a outro", onde qualquer relacionamento pode ser criado, causando grande dificuldade em identificar elos incorretos e até mesmo identificar as razões de sua existência. Outro grave problema é a dificuldade e custo de realizar o rastreamento com alto nível de detalhe.

2. **Recuperação Automática de Relacionamentos:** geralmente utilizando técnicas de indexação LSI (Latent Semantic Indexing), que simplesmente relacionam porções texto, de especificações de requisitos e manuais, com identificadores do código fonte – uma tarefa típica e reconhecimento de padrões (PENTA; GRADARA; ANTONIOL, 2002) – produzindo relacionamentos com as mesmas características citadas no item anterior, com o agravante de que a maioria dos elos gerados devem ser descartados pelo usuário, por não possuírem sentido (LUCIA et al., 2006a). Não obstante, erros de decisão são muito comuns, pois o analista acaba descartando elos verdadeiros, e mantendo elos incorretos (HAYES; DEKHTYAR, 2005). A dificuldade na identificação da natureza das relações pode ser um dos diversos fatores que ocasionam este fenômeno.

Com isto, o resultado é geralmente o mesmo: dificuldade na manutenção dos relacionamentos e sua conseqüente degradação, que por fim se traduz em tempo, esforços e recursos completamente desperdiçados.

1.2 Objetivos

O objetivo geral do presente trabalho é fornecer um modelo para o estabelecimento de relações de rastreabilidade, que permita automatizar grande parde das tarefas relacionadas à manutenção destas relações.

Deste, advém os objetivos específicos:

- Avaliar as propostas de rastreabilidade encontradas na literatura, visando a identificação dos problemas comuns na manutenção de estruturas de rastreabilidade, e soluções aos mesmos;

- Estabelecer um modelo de rastreabilidade, capaz de abranger grande parte dos artefatos produzidos no ciclo de vida do software, facilitando seu estabelecimento e automatizando, quando possível, sua manutenção;

- Avaliar a viabilidade da proposta, comparando, através de experimentação, o modelo proposto com um modelo amplamente adotado pela indústria (Matrizes de rastreabilidade).

1.3 Contribuição Principal do Trabahlo

O presente trabalho apresenta uma forma de estabelecimento de relações com base em estruturas intermediárias, que representam interesses em comum representados por diferentes artefatos. Estas estruturas, chamadas de entidades, têm por

objetivo centralizar interesses para derivar relacionamentos entre os artefatos. Esta estratégia permite a criação e manutenção da estrutura de rastreabilidade de forma semi-automática, tornando-a mais precisa e de produção menos custosa.

1.4 Estrutura do Documento

Este documento está dividido em quatro partes:

1. Capítulo 2: trata de todos os aspectos da rastreabilidade, apresenta as principais técnicas, exemplos e definições formais para os principais conceitos relacionados à rastreabilidade. Por fim, apresenta os diversos problemas da área, considerada extremamente problemática;

2. Capítulo 3: apresenta o modelo proposto para semi-automatizar as atividades de manutenção da estrutura de rastreabilidade;

3. Capítulo 4: apresenta a avaliação quantitativa e qualitativa do modelo, em comparação ao tradicional modelo baseado em matriz;

4. Capítulo 5: apresenta as conclusões deste trabalho e diversas sugestões para trabalhos futuros a partir do modelo proposto;

2 RASTREABILIDADE

Neste capítulo será apresentada uma revisão dos principais métodos existentes: matrizes de rastreabilidade e métodos de recuperação automática de estruturas de rastreabilidade. A seguir, serão apresentados técnicas de rastreabilidade alternativas para o rastreamento de requisitos não funcionais, baseadas em cenários, e considerações sobre a granularidade da rastreabilidade para torná-la mais gerenciável e menos custosa. Por fim, será apresentada uma definição formal da rastreabilidade, identificando as operações básicas sobre elos e artefatos.

2.1 Matrizes de Rastreabilidade

Matrizes de rastreabilidade são geralmente utilizadas para exibir os relacionamentos entre a elicitação de requisitos e a representação destes requisitos em um método particular da engenharia de software. Mesmo em projetos pequenos ou de tamanho moderado, o estabelecimento de elos de rastreabilidade, entre artefatos-chave e modelos, continua sendo uma tarefa desafiadora e cara (NEUMULLER; GRUNBACHER, 2006). Um dos possíveis motivos é que não há forma padronizada de armazenar ou representar elos de rastreabilidade. Tradicionalmente, eles são armazenados em uma matriz e representados como grafos.

Em termos da álgebra linear, elas exibem o mapeamento entre fonte e alvo. Tais mapeamentos são apresentados em um tipo especial de matriz, chamada de matriz de dependência, que representa a relação de dependência entre elementos da fonte e do alvo (fonte x alvo). Nas linhas, ficam os elementos fonte, e nas colunas, os elementos alvo. Nesta matriz, uma célula com o valor 1 denota que o elemento fonte (na linha) é mapeado para o elemento alvo (na coluna). Reciprocamente, isto significa que o elemento alvo depende do elemento fonte (BERG; CONEJERO; HERNÁNDEZ, 2006).

Em sua forma mais simples, a rastreabilidade se manifesta em tabelas cruzadas, nas quais os elementos de um projeto são relacionados aos requisitos que satisfazem (ALMEIDA; ECK; IACOB, 2006). Nesta matriz, elementos-fonte são mapeados para elementos-alvo. A tabela 2.1 (BERG; CONEJERO; HERNÁNDEZ, 2006) apresenta o mapeamento do elemento-fonte f_1 para os elementos-alvo a_1,a_3 e a_4, o que indica que a_1,a_3 e a_4 dependem de f_1. Analogamente, pode-se dizer que f_1 dá origem a a_1,a_3 e a_4.

Esta representação permite visualizar, por exemplo, que vários requisitos são implementados por uma mesma classe, permitindo que a intersecção entre classes responsáveis pela satisfação de um requisito seja não-vazia (ou seja, n classes podem ser comuns entre dois ou mais requisitos). Além disso, é possível visualizar as várias

Tabela 2.1: Exemplo de dependência entre elementos

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

classes que podem ser necessárias para a implementação de um requisito. (ANTONIOL; CIMITILE; CASAZZA, 2000). É possível representar a tabela graficamente, conforme a figura 2.1 (BERG; CONEJERO; HERNÁNDEZ, 2006)

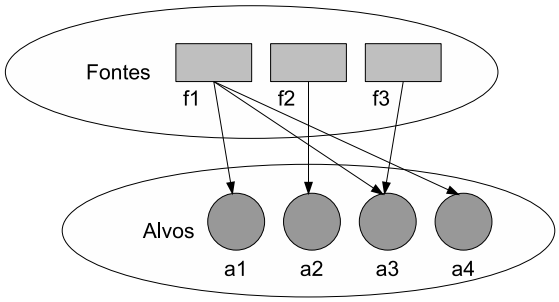


Figura 2.1: Representação gráfica da matriz de rastreabilidade

Elos de rastreabilidade são geralmente estabelecidos pelo relacionamento explícito entre dois artefatos, armazenando-os em tabelas, bancos de dados, ou ferramentas de gerenciamento de requisitos (FLETCHER; CLELAND-HUANG, 2006), e esta ainda é a prática atual (CLELAND-HUANG, 2006b). Existem diversos modelos de rastreabilidade que diferem nos tipos de artefatos que rastreiam. Porém, a maioria dos projetos não utiliza um método sistemático de rastreamento, mas delega, para determinados indivíduos, a atividade de realizar um rastreamento *ad hoc*, quando for necessário (HEINDL; BIFFL, 2006).

Formalmente, as estruturas tradicionais de rastreabilidade são criadas seguindo a definição $Elo(a,a')$. Uma matriz de rastreabilidade (MR) é formada pelo conjunto $\{Elo(a,a')|a$ e a' são artefatos, em qualquer nível de abstração (eg. caso de uso, especificação de requisitos, diagrama de classe, código fonte de uma linguagem específica, etc $\}$ (MALETIC; COLLARD; SIMOES, 2005).

Cada artefato possui um nível de abstração, que pode ser mais ou menos completo do que os outros, e apresenta diferentes pontos de vista sobre as necessidades do sistema. As relações entre estes, do menos completo para o mais completo, até a implementação, deve ser mantida para garantir que a estrutura de rastreabilidade permita:

1. O mapeamento dos requisitos para um modelo posterior, até o código fonte, garantindo que a satisfação dos requisitos esteja atribuída à componentes do sistema (*Forward from Requirements*);

2. O Mapeamento de um modelo qualquer de volta para os requisitos, evitando o *gold-plating* (*Backward to Requirements*);

Tabela 2.2: Matriz de rastreabilidade entre os requisitos R1 a R4

Requisitos	R1	R2	R3	R4
R1		X		X
R2			X	
R3				X
R4				

Grande parte das propostas de rastreabilidade possuem estas características, no entanto, algumas suportam o rastreamento somente vertical (relacionamentos entre artefatos no mesmo nível de abstração), outras somente o horizontal (relacionamentos entre artefatos em níveis diferentes de abstração). Para exemplificar a utilização de matrizes no rastreamento vertical, considere as seguintes regras de negócio (doravante denominados simplesmente como requisitos, por simplicidade):

R1 "Todo cliente tem uma conta, com um determinado limite de crédito";

R2 "O limite de crédito de qualquer conta está restrito a um determinado valor, estabelecido segundo a lei federal XYZ";

R3 "A verificação do limite de crédito é realizada de acordo com os seguintes critérios...";

R4 "Universitários têm limite de R$ 200,00, invariavelmente".

Não é necessário muito esforço de interpretação para perceber que os requisitos estão de fato relacionados, pois tratam de interesses comuns, formando a possível matriz de rastreabilidade, representada pela tabela 2.2.

Note que apesar do pequeno número de requisitos e relacionamentos, não é trivial interpretar a matriz da tabela 2.2 e identificar o motivo por trás da existência dos elos e a inexisência de outros. No caso do exemplo, estes foram criados pelas seguintes interpretações:

R1-R2 "A lei federal XYZ estabelece o limite da conta, citado em R1";

R1-R4 "O limite da conta também pode ser definido pelo valor citado em R4";

R2-R3 "A verificação deve considerar as regras impostas pela legislação";

R3-R4 "A verificação deve considerar o caso das contas universitárias".

Porém, várias interpretações diferentes podem ser tomadas a partir dos requisitos apresentados, formando relacionamentos diferentes. Assim, para evitar este esforço de re-interpretação, é necessário anexar a cada elo as decisões, problemas, suposições e argumentos que motivaram sua criação. Analogamente, é fundamental que as mesmas informações sejam mantidas para os elos que não foram criados, evitando o esforço desnecessário de justificar novamente uma decisão já estabelecida (RAMESH; JARKE, 2001).

Para exemplificar o uso de matrizes no rastreamento horizontal, considere os requisitos R1 a R4, "implementados" no código fonte da listagem 1. Como a listagem implementa os requisitos, pode-se dizer que estes pertencem ao conjunto de relações $\{ Elo(a,r) | a$ é um requisito e r é um trecho de código $\}$.

Tabela 2.3: Matriz de rastreabilidade entre requisitos e código.

Requisito	Classes	Métodos	Atributos
R1	Client, Account		Client.account.limit
R2	Account		Account.limit
R3		System.limitVerification	
R4	Account		Account.{limit, type}

Listagem 1. Exemplo de implementação dos requisitos

```
1  Client{
2          personalData;
3          account;
4  }
5
6  Account{
7          cash;
8          limit;
9          type;
10 }
11
12 System{
13         listOfClients;
14
15         limitVerification(Amount a, Account c){
16                 ...
17         };
18
19         withdraw(Client c, Account acc, Amount a){
20                 limitVerification(a, acc);
21                 ...
22         };
23 }
```

A tabela 2.3 representa as possíveis relações entre os requisitos e partes do código apresentado. O requisito R1 é satisfeito pela implementação das classes *Client* e *Account*, bem como pelo atributo *limit* em *Account*. R2 trata do limite da conta, então a implementação de *Account.limit* pode ser afetada caso R2 seja alterado (e vice-versa). Já R3 é satisfeito pela implementação de *System.limitVerification*, e R4 pelo tratamento de um tipo de conta (Account.type, que pode ser "conta de pessoa física", "conta de pessoa jurídica", "conta universitária", etc). Note que a matriz de rastreabilidade depende muito da interpretação humana, sendo que outras interpretações e necessidades podem originar uma matriz completamente diferente.

Analisando mais profundamente, não é difícil identificar o potencial de crescimento de uma matriz de rastreabilidade. Os problemas de manutenção aparecem quando tenta-se atualizar os elos, especialmente quando se rastreia um grande número de artefatos, com um nível de granularidade que possa ser considerado útil, como no exemplo da tabela 2.3. Por exemplo, projetos com centenas de casos de uso, compostos por inúmeros passos, e relacionados a outros artefatos, são de manutenção extremamente complexa e cara. A matriz de rastreabilidade facilmente assume um tamanho ingerenciável, pois a quantidade de relações tende a crescer exponencialmente (CLELAND-HUANG; ZEMONT; LUKASIK, 2004). A partir deste ponto é que as técnicas de rastreabilidade baseadas em MR falham. Um dos maiores problemas reside na incapacidade de lidar eficientemente com alterações que afetam os relacionamentos estabelecidos (MALETIC; COLLARD; SIMOES, 2005).

Mesmo em projetos pequenos ou de tamanho moderado, o estabelecimento de elos de rastreabilidade, entre artefatos-chave e modelos, continua sendo uma tarefa desafiadora e cara (NEUMULLER; GRUNBACHER, 2006). Um dos possíveis motivos é que não há forma padronizada de armazenar ou representar elos de rastreabilidade. Tradicionalmente, eles são armazenados em uma matriz e representados como grafos. Enquanto a simplicidade é a principal vantagem dos métodos tradicionais, eles são pouco úteis na representação de todas as informações relevantes

(MARCUS; XIE; POSHYVANYK, 2005). Além disso, os esforços dos desenvolvedores são retardados pela falta de suporte ferramental, e há uma percepção geral de que o custo necessário para manter uma matriz de rastreabilidade de requisitos é excessivo, em relação aos seus benefícios (CLELAND-HUANG, 2006b).

2.2 Métodos de Recuperação Automática

Técnicas de geração automática de elos, entre requisitos expressos textualmente e modelos posteriores (geralmente código fonte), vêm emergindo como alternativa para satisfazer o problema da cara e complexa manutenção da rastreabilidade (SPANOUDAKIS, 2002; EGYED et al., 2005a; LORMANS; DEURSEN, 2006; ANTONIOL; CIMITILE; CASAZZA, 2000).

O problema de identificar os documentos relacionados a um dado componente do código fonte pode ser visto como uma tarefa típica e reconhecimento de padrões, ou de recuperação de informações (PENTA; GRADARA; ANTONIOL, 2002). Os elos de rastreabilidade entre artefatos também podem ser identificados utilizando simples regras, baseadas em convenções (NEUMULLER; GRUNBACHER, 2006). As técnicas de recuperação de informações confiam na hipótese de que o uso correto de termos do domínio entre artefatos permite rastreá-los. Nos casos em que isto não acontece, o processo de recuperação automática da rastreabilidade (RAR) torna-se ineficiente (LUCIA et al., 2006b). Assim, os métodos de RAR assumem a conjetura do *programador são* (que produz identificadores com nomes significativos, ou que sigam um padrão); e que para cada requisito, exista pelo menos um elo de rastreabilidade disponível, produzido manualmente. Estes elos já existentes auxiliam na recuperação de elos desconhecidos.

A recuperação de elos funciona na seguinte seqüência (PENTA; GRADARA; ANTONIOL, 2002):

Processamento de requisitos: cada requisito é processado e um dicionário de palavras é montado, com base na freqüencia de cada uma. Esta fase em geral é semi-automática, pois não é possível retirar a ambigüidade da semântica contextual, nem lidar com discursos metafóricos (eg. “deve-se ’matar’ o processo...”);

Processamento do código: para cada classe, uma lista de identificadores contendo suas ocorrências é extraída. Aqui, a intervenção humana é necessária para informar o significado de palavras curtas, com menos de três letras;

Recuperação do mapa de rastreabilidade: é realizada através de um classificador.

Outras técnicas de recuperação são utilizadas por Antoniol *et al.* (ANTONIOL; CIMITILE; CASAZZA, 2000) para recuperar as relações de rastreabilidade de código C++ para páginas de manuais e de código Java para requisitos. Settimi *et al.* (SETTIMI et al., 2004) discutem a obtenção de elos de rastreabilidade entre código e modelos UML, usando uma técnica de recuperação de informações baseada em um modelo de espaço vetorial, mas os resultados apresentaram-se limitados. Porém, a técnica reduziu o tempo necessário na manutenção da rastreabilidade. No estudo realizado por Lucia *et al.* (LUCIA et al., 2006a), em um pequeno projeto, para atingir o objetivo de obter todos os elos corretos, rastreando casos de uso para

classes do código fonte, foi necessário analisar 1013 elos, para identificar, entre estes, 93 elos corretos.

Considerando o extremo esforço necessário para a manutenção de elos, foram propostos métodos de RAR, que assumem a conjetura do *programador são* (que produz identificadores com nomes significativos, ou que sigam um padrão); e que para cada requisito, exista pelo menos um elo de rastreabilidade disponível, produzido manualmente. Estes elos já existentes auxiliam na recuperação de elos desconhecidos. As técnicas de recuperação de informações confiam na hipótese de que o uso correto de termos do domínio entre artefatos permite rastreá-los. Nos casos em que isto não acontece, o processo de recuperação da rastreabilidade torna-se ineficiente (LUCIA et al., 2006b).

Todos os estudos interessados na aplicação de métodos para a RAR mostram que estes não são capazes de recuperar todos os elos corretos, sem também recuperar muitos falsos positivos, que devem ser descartados pelo engenheiro de software. A baixa precisão faz deste método uma tarefa tediosa, já que o engenheiro tem que gastar muito mais tempo para descartar falsos positivos do que rastreando elos corretos. Embora seja possível melhorar a performance destes métodos, eles ainda estão distantes de tornar viável a solução para o problema da recuperação de elos (LUCIA et al., 2006a). Mesmo assim, esta solução vem sendo cada vez mais reconhecida pela indústria como uma solução potencial para o problema da rastreabildade (CLELAND-HUANG, 2006b). Em contrapartida, Hayes e Dekhtyar (HAYES; DEKHTYAR, 2005) estudaram os resultados produzidos por analistas, e descobriram que as decisões tomadas sobre a validade ou invalidade dos elos recuperados nem sempre são corretas, e podem até piorar os resultados: elos corretos acabam sendo descartados, e elos incorretos mantidos.

Como a maioria dos métodos e ferramentas de RR existentes assumem que o desenvolvedor crie e mantenha as relações de rastreabilidade manualmente, ambientes industriais raramente estabelecem um processo de rastreabilidade, já que o estabelecimento dos links de forma manual, além de ser custoso em termos de tempo, é sujeito a erros (SPANOUDAKIS, 2002). Por este motivo, técnicas de geração automática de links entre requisitos expressos textualmente e modelos posteriores (geralmente código fonte), vêm emergindo como alternativa para satisfazer o problema da cara e complexa manutenção da rastreabilidade (SPANOUDAKIS, 2002; EGYED et al., 2005a; LORMANS; DEURSEN, 2006; ANTONIOL; CIMITILE; CASAZZA, 2000).

Técnicas de recuperação de informações (RI) podem ser utilizadas para automatizar o processo de reconstrução de links, como por exemplo Indexação de Semântica Latente (*Latent Semantic Indexing* - LSI). A LSI é uma técnica de RI promissora, pois assume que há uma estrutura semântica latente em qualquer conjunto de documentos (LORMANS; DEURSEN, 2006).

Através da RI, um usuário pesquisa por informações, utilizando palavras-chave ou consultas de texto em linguagem natural. Um sistema de procura obtém um conjunto de objetos candidatos, a partir do qual o usuário seleciona os que considera relevantes. Na maioria dos algoritmos de pesquisa, um índice de similaridade é calculado, de forma que meça o grau de similaridade entre dois documentos. Um valor de aceitabilidade é então estabelecido, de forma que qualquer valor, acima do nível de aceitabilidade, presente no índice, indica que o documento deve ser recuperado (CLELAND-HUANG et al., 2005).

Os algoritmos de RI geralmente são avaliados utilizando as métricas *recall* (re-

torno) e *precision* (precisão). Tais métricas também são muito utilizadas para avaliar a utilidade de diferentes técnicas de rastreabilidade (STIREWALT; DENG; CHENG, 2005). Onde:

$$recall = \frac{\text{Número de documentos relevantes obtidos}}{\text{Número de documentos relevantes}}$$

$$precision = \frac{\text{Número de documentos relevantes obtidos}}{\text{Número total de documentos obtidos}}$$

O *recall* máximo pode ser obtido ao pesquisar todos os documentos de uma coleção, e a *precision* máxima ao pesquisar um único documento que sabe-se ser correto. Por esta razão, um sistema de RI normalmente tenta maximizar *recall* e *precision* simultaneamente.

Em métodos de recuperação da rastreabilidade, uma consulta é geralmente formulada a partir de palavras encontradas em um requisito, e um algoritmo de busca a executa dentro de um espaço de busca de outros artefatos, como outros requisitos, código, casos de teste, ou modelos do projeto. Resultados de vários métodos de RI encontraram sucesso limitado com taxas de *precision* geralmente entre 10% e 60%, e níveis de aceitabilidade definidos para obter de 90% a 100% de *recall*. (CLELAND-HUANG; ZEMONT; LUKASIK, 2004; SPANOUDAKIS, 2002)

Outras técnicas de RI são utilizadas por Antoniol, Cimitile e Casazza (2000) para recuperar as relações de rastreabilidade de código C++ para páginas de manuais e de código Java para requisitos. Settimi *et. al.*(2004) discutem a obtenção de links de rastreabilidade entre código e modelos UML, usando uma técnica de RI baseada em um modelo de espaço vetorial, cujos resultados apresentaram-se limitados. Neste, porém, a técnica de RI reduziu o tempo necessário na manutenção da rastreabilidade. Não obstante, o trabalho fornece razões pelas quais grande parte das técnicas de RI não são completamente eficientes, devido a erros de:

Inclusão (links falsos) Causados pelo alto acoplamento entre artefatos, mesmo que estes artefatos representem links corretamente; e

Omissão (links perdidos, ou não detectados) Causados por falta de terminologia padronizada entre os responsáveis pelos requisitos, projetistas, programadores e testadores.

Outros problemas na detecção de links ocorrem na troca de perspectiva. Por exemplo, um requisito da perspectiva do usuário poderia ser descrito nos termos "o usuário deve ser capaz de ver ...", enquanto que da perspectiva do sistema seria "o sistema deve exibir ...". Assim, a técnica de RI dificilmente associa a mesma ação a partir de diferentes pontos de vista. Ela também é limitada pela diferença no nível de abstração entre artefatos, como entre código fonte e requisitos expressos em linguagem natural.

Embora os resultados sejam influenciados pelo algoritmo utilizado, e possam ser melhorados através de mecanismos de *feedback* e da incorporação de conhecimentos contextuais adicionais, a real limitação parece ser relacionada à qualidade do conjunto de dados. Conjuntos que usam glossários de projeto claramente definidos tendem a produzir resultados melhores do que os que usam glossários definidos sem planejamento (CLELAND-HUANG, 2005).

Glossários de projeto auxiliam no fornecimento de termos importantes do domínio, além de apresentar uma breve explanação sobre o termo em seu contexto (RIEBISCH; HUBNER, 2005).

Embora métodos de rastreabilidade dinâmicos não sejam uma bala de prata, fornecem um meio muito prático na redução de esforço para a obtenção de links. A aplicação de métodos de recuperação de traços no gerenciamento de alterações durante o ciclo de vida de um software reduz o esforço potencial necessário para analisar o impacto de alterações, e provê uma solução alternativa promissora para as atividades de manutenção de links de rastreabilidade (SETTIMI et al., 2004).

2.3 Rastreabilidade de Requisitos Não-Funcionais

Muitas organizações não rastreiam requisitos não-funcionais (RNFs). Alterações funcionais são geralmente implementadas com pouca compreensão de como qualidades do sistema – como segurança e performance – serão afetadas. Não obstante, RNFs são raramente rastreados pois tendem a afetar globalmente o sistema, o que gera a necessidade de manter um exagerado número links de rastreabilidade. Porém, a falha a ao gerenciar a alteração de RNFs pode levar a resultados catastróficos, salientando a importância de se rastrear este tipo de requisito (CLELAND-HUANG et al., 2005).

Devido à sua natureza global, RNFs são vagos e interdependentes Por isso, geralmente só é possível descrevê-los em relação a requisitos funcionais ou abstratos. Além disso, são expressos em um nível de abstração muito alto, e devem ser refinados (PAECH; KERKOW, 2004).

A grande desvantagem dos métodos de rastreabilidade é a sua necessidade de possuir um modelo conceitual completo de todos os elementos da especificação. Ao tratar RNFs, desconhece-se completamente cada tipo de RNF e os relacionamentos entre estes tipos. Outro desafio é a natureza entrelaçada dos RNFs. Os métodos de rastreabilidade atuais são baseados em um conjunto de elementos discretos interrelacionados. Por outro lado, RNFs são aspectos transversais que impactam de forma entrelaçada em diferentes requisitos (PAECH; KERKOW, 2004).

Cleland-Huang (2005) apresenta alguns métodos de rastreabilidade que permitem rastreabilidade de RNFs, apontando seus pontos fracos e fortes:

Matrizes Embora matrizes sejam tecnicamente capazes de suportar a rastreabilidade de RNFs, requerem que os links sejam definidos e mantidos explicitamente, além de terem problemas de escalabilidade ao representar o grande número links que podem ser gerados a partir de um único RNF.

Aspectos Interesses podem ser categorizados como funcionais e não-funcionais. Interesses não-funcionais incluem RNFs como manutenibilidade, performance, segurança, etc. e tendem ser insuficientemente concretos para ser implementados como aspectos. Por outro lado, interesses funcionais e mais concretos como *logging*, autenticação e rastreamento podem ser implementados como aspectos, porque seu comportamento é mais facilmente definido, podendo ser expresso em termos de regras de *weaving* de aspectos.

Métodos de recuperação de informações (RI) Permite que links candidatos sejam gerados dinamicamente. Porém, as métricas *recall* e *precision* dos resultados não são 100% confiáveis, e o usuário deve validar os resultados.

Cenários São de propósito múltiplo e requerem menos trabalho para estabelecer a rastreabilidade, porém dificilmente provêm a cobertura completa da rastreabilidade.

Rastreabilidade baseada em eventos Proposta por Cleland-Huang, Chang e Christensen (2003), utiliza uma arquitetura *publish-subscribe* para definir os links de rastreabilidade que suportam análise de impacto automática sobre RNFs. Os links são estabelecidos entre requisitos de performance quantitativamente definidos, restrições na especificação dos requisitos, e variáveis localizadas em modelos de simulação de performance. Quando uma alteração é proposta, o estado atual do modelo de simulação é salvo, e parâmetros especulativos são passados para o modelo. Este então é novamente executado usando os novos valores. Os resultados são extraídos e reportados, e o modelo é restaurado para o estado inicial. Este método demonstra a capacidade de automatizar a re-execução de um modelo de avaliação arquitetural, em resposta a um evento de alteração, assim que um requisito afetado seja descoberto.

Baseado em *Design Patterns* Fornece um método para rastrear RNFs para o projeto sem depender de uma proliferação descontrolada de links. Porém, técnicas de detecção de padrões retornam resultados imprecisos.

Em qualquer sistema não-trivial, é inviável o custo e esforço necessários para construir uma matriz de rastreabilidade, que possa capturar uma grande rede de relacionamentos entre RNFs e outros artefatos. Assim, métodos de recuperação dinâmica de links de rastreabilidade fornecem uma método que requer esforço menor para obtenção de links (CLELAND-HUANG et al., 2005).

2.4 Cenários

Cenários vem sendo utilizados como um meio alternativo, e por vezes complementar, para expressar requisitos e o comportamento do sistema através das diversas fases de desenvolvimento. Cada cenário pode assumir representações e semânticas diferentes entre cada fase, e estas podem ser relacionadas. Assim, é possível descrever um sistema em níveis de abstração diferentes, mapear os cenários para a arquitetura do software e usá-los em casos de teste (NASLAVSKY et al., 2005).

Para estabelecer relações entre cenários, é preciso compreender seu propósito em cada fase, bem como o que estes possuem ou não em comum entre as fases. Um cenário é uma seqüência de eventos. Um evento pode ser descrito de várias formas: algo que ocorre, uma ação, uma interação, um passo, uma alteração instantânea de um estado para outro, etc.

Os artefatos produzidos no desenvolvimento de um software – tais como descrições de modelo, linguagens diagramáticas, especificações formais, código fonte, etc. – estão altamente relacionados, de tal forma que alterações em um devem afetar os outros. As dependências entre os elos caracterizam seu relacionamento de forma abstrata (EGYED, 2003).

Os links entre cenários servem como artérias entre modelos como diagramas, pois dão vida para descrições de polígonos e setas, de forma a torná-las representações legítimas e úteis de sistemas. Com a ausência da rastreabilidade, ou incerteza sobre

sua precisão, a utilidade dos modelos torna-se severamente limitada. (EGYED, 2001)

O significado semântico de dependências entre elos pode ser inferido através da diferença semântica entre os artefatos ligados por eles. Ou seja, elos são entidades neutras, que simplesmente descrevem dependências. Contudo, a interpretação de tais dependências e de como utilizá-las depende dos artefatos que os elos ligam (EGYED, 2003).

Naslavsky *et. al.*(2005) Identifica diversos tipos de cenários utilizados em diferentes fases do desenvolvimento:

Cenários de Requisitos são utilizados para descrever requisitos após uma fase preliminar de levantamento. É o tipo de cenário mais próximo da compreensão dos *stakeholders*, ao contrário dos utilizados em fases posteriores. Os cenários de requisitos descrevem seqüências de eventos que ocorrem sobre o sistema proposto. Um exemplo deste tipo de cenário são *Use Cases*;

Cenários de Análise adicionam mais detalhes e uma perspectiva da parte interna do sistema aos cenários de requisitos. São identificadas entidades de alto nível, que farão parte do sistema. Da mesma forma, eventos, que antes eram representados por meio de linguagem natural, assumem a forma de interações entre os conceitos do sistema. Um exemplo deste tipo de cenário são diagramas de seqüência da UML, ou um gráfico de no qual agentes que trocam mensagens em um nível similar às classes de um diagrama de classes de análise;

Cenários de Arquitetura acrescentam mais detalhes aos anteriores. Nesta fase, são tomadas decisões sobre quais conceitos tornar-se-ão componentes da arquitetura do sistema. Um exemplo deste tipo de cenário é um diagrama de seqüência da UML, ou um gráfico para seqüenciamento de mensagens, onde agentes que trocam mensagens são componentes arquiteturais e conectores;

Cenários de Projeto são cenários descritos em maiores detalhes do que os de arquitetura. Os conceitos usados neste caso são partes de objetos de componentes arquiteturais, exibindo assim, o funcionamento interno dos componentes;

Cenários de Código descrevem o fluxo de eventos a partir do nível de implementação do sistema, que são na verdade informações de rastreamento em tempo de execução, correspondendo a comandos executados ou invocações de métodos. Assim, cenários de requisitos, análise, arquitetura e de projeto, quando executados, resultam em algum cenário de código;

Cenários de teste aparecem em cada fase do ciclo de desenvolvimento, e são utilizados para testar a implementação do sistema ou outros artefatos. Os cenários definidos acima, para cada fase do ciclo de vida, podem ser utilizados como cenários de teste. Assim, cenários em cada nível de abstração são utilizados para testar se o comportamento de um cenário de nível inferior está de acordo com o comportamento esperado, descrito pelo de nível superior. Já que um cenário de nível superior pode levar a múltiplos cenários de nível inferior, é necessário testar a sua conformidade.

Existem diversas formas de definir o rastreamento entre cenários. Utilizar a mesma linguagem para descrever todos os tipo de cenários facilita o estabelecimento

e manutenção do relacionamento entre eles. Pode-se utilizar cenários para relacionar conceitos e eventos identificados entre outros cenários e artefatos.

Modelar a informação de rastreabilidade de forma precisa é fundamental para manter a consistência dos vários modelos. A falta deste tipo de informação constitui um grande problema, pois reduz a utilidade dos modelos e diagramas ao ponto de os desenvolvedores questionarem a utilidade da modelagem, já que há pouco ou nenhum valor em utilizar modelos de software que não representam consistentemente o sistema real (EGYED, 2003).

2.5 Granularidade da Rastreabilidade

O uso excessivo da rastreabilidade pode resultar em um emaranhado de links de difícil manutenção e utilização. Ou seja, não faz sentido rastrear todos os requisitos com a mesma granularidade (DÖMGES; POHL, 1998).

A captura de interdependências entre os artefatos do processo de desenvolvimento é de alta complexidade, e o esforço necessário para manter o rastreamento torna o processo de RR muito custoso na prática. Além disso, o esforço para criar elos ligando requisitos ao código fonte é bastante alto (o estudo realizado por Heindl e Biffl (2005) apresenta um tempo de cerca de 45 minutos por requisito), porém este tipo de traço é o que fornece as informações mais úteis para a rastreabilidade.

Mesmo que ferramentas tentem automatizar o processo de RR, ainda não conseguem reduzir o esforço necessário para a sua manutenção, tornando a RR uma atividade custosa para implementação prática. A principal razão é que os métodos existentes não levam em consideração a diferença entre requisitos valiosos – do ponto de vista da rastreabilidade – e entre requisitos de baixíssimo valor (HEINDL; BIFFL, 2005).

Egyed *et. al.*(2005a) acreditam que considerações sobre o valor dos artefatos são necessárias, permitindo realizar o planejamento da rastreabilidade de forma sustentável. A motivação da engenharia de software baseada em valores, é que na maior parte das vezes a prática e a pesquisa consideram circunstâncias nas quais o valor dos artefatos não é levado em conta. Assim, cada requisito, caso de uso, objeto, defeito, etc., são tratados com igual importância. O desenvolvimento de software baseado em valor tem como premissa classificar os artefatos de software com alguma medida de valor, considerando sua relevância, e portanto, tratando-os de forma diferente.

No contexto da rastreabilidade de software, pode-se então concluir que nem todos os links de rastreabilidade são igualmente importantes. Por exemplo: requisitos podem ser classificados como “críticos”, “importantes” e “de interesse”. Mesmo que alguns requisitos “de interesse” sejam implementados, sua corretude não possui a importância de requisitos considerados “críticos”.

Há pouco retorno econômico em melhorar o nível de detalhe de links de rastreabilidade além de um certo limite. Algumas aplicações podem não necessitar rastrear certos requisitos com alta granularidade. Portanto, nem todo link é necessário, e gerá-los ou mantê-los representa desperdício quando não são necessários de fato. Da mesma forma, se alguma aplicação necessitar somente de um certo nível de qualidade, gerar ou manter links mais detalhados é um desperdício se tais melhorias não se traduzirem em benefícios (EGYED et al., 2005b).

Heindl e Biffl (2005) introduziram o processo de rastreamento de requisitos baseado no valor (*Value-Based Requirements Tracing - VBRT*), cujo objetivo é identificar

elos baseados em requisitos priorizados e assim identificar quais são mais importantes e valiosos em detrimento de outros menos interessantes.

O processo VBRT consiste de cinco passos distintos:

Definição de requisitos: o engenheiro de requisitos analisa a especificação de requisitos do software e identifica requisitos atômicos, atribuindo identificadores únicos a cada um. Após este passo, obtém-se uma lista de requisitos e seus identificadores;

Priorização de requisitos: todos os *stakeholders* avaliam os requisitos e estimam o valor, risco e esforço de cada. O resultado deste passo é uma lista ordenada de requisitos onde estes são classificados sob três níveis de prioridade;

Empacotamento de requisitos: este passo é opcional e permite identificar grupos de requisitos relacionados, com o objetivo de refiná-los para uma arquitetura intermediária;

Linkagem de artefatos: são estabelecidos os links entre requisitos e artefatos. Requisitos importantes são rastreados mais detalhadamente do que os menos importantes, de acordo com os níveis de intensidade de rastreamento definidos (o VBRT cita três). O resultado deste passo é um plano de rastreabilidade;

Avaliação: pode-se utilizar os elos para os mais variados propósitos, como por exemplo: estimar o impacto de alterações sobre determinados requisitos.

Os resultados no caso de estudo apresentado por Heindl e Biffl (2005) mostraram que o esforço necessário para estabelecer links, considerando o valor dos requisitos, foi 35% do esforço necessário para estabelecer a rastreabilidade completa, sem afetar a qualidade das informações de rastreabilidade. Obviamente, tais resultados dependem de diversos fatores como: documentação e complexidade dos artefatos do sistema.

Dando suporte às conclusões deste trabalho, o método proposto por Riebisch e Hubner (2005) considera a manutenção de um número pequeno de rastros como fator-chave para o sucesso do processo de rastreabilidade.

2.6 Modelo Formal da Rastreabilidade

Cleland-Huang, Chang e Christensen (2003) apresentam, formalmente, definições para:

- artefatos;
- elos;
- operações sobre artefatos;
- rastro;
- tipos de erro;
- operações sobre requisitos.

2.6.1 Artefatos e elos

Artefatos incluem objetos como requisitos, módulos de código, projetos, casos de teste e seus respectivos resultados, além de várias outras entidades.

Elos de rastreabilidade definem as relações que existem entre os vários artefatos do processo de engenharia de software.

Definição 1 *Um **artefato** é uma parte da informação produzida ou modificada pelo processo de engenharia de software. Este pode tomar uma variedade de formas, incluindo modelos, documentos, código fonte, casos de teste e executáveis. Todos os artefatos, total ou parcialmente, formam os objetos rastreáveis do sistema. Seja* $A = \{a_1, a_2, a_3, \ldots, a_n\}$ *o conjunto de todos os artefatos identificados no sistema. Seja* $R = \{r_1, r_2, r_3, \ldots, r_n\} \subset A$ *o conjunto de todos os artefatos que são requisitos.*

Definição 2 *Um **elo** Elo*(a, a') *representa um relacionamento explícito, definido entre dois artefatos* a *e* a'*. Se* a *e* a' *estão diretamente ligados como em* $a \rightarrow a'$*, onde* $\rightarrow$ *indica um elo, então o elo é **direto**. Se* a *e* a'' *estão indiretamente ligados através de um ou mais artefatos intermediários, como em* $a \rightarrow a' \rightarrow a''$*, então o elo é **indireto**. Seja* $a \Rightarrow a''$*, a fórmula representa um elo indireto de* a *até* a''*. Um artefato (como por exemplo* a'*) através do qual um elo indireto é estabelecido é chamado de **intermediário**. A direção do elo indica que este é estabelecido a partir do artefato à esquerda (LHS) para aquele à direita (RHS), tal que o artefato LHS apresente uma **dependência** sobre o artefato RHS.*

Para compreender por que a manutenção da rastreabilidade é considerada tão difícil (vide seção 2.7), é necessário primeiro examinar como a infraestrutura de rastreabilidade é afetada por alterações.

Definição 3 *Uma **alteração** C pode ser introduzida sobre um artefato a em uma de duas fases. Uma alteração **proposta** implica que uma análise de impacto deve ser realizada para determinar como a alteração C afetará o sistema atual. Enquanto que uma alteração **implementada** implica que todos os artefatos afetados e seus elos relacionados devem ser atualizados para refletir a alteração. Uma alteração proposta não resulta, necessariamente, em uma alteração implementada.*

Quando uma alteração C é introduzida em um artefato a, os artefatos restantes do sistema podem ser categorizados, de acordo com seu relacionamento com a, das seguintes formas:

Definição 4 (Artefato ligado) *Seja Linked*$(a) \subseteq A$ *o conjunto de todos os artefatos ligados direta ou indiretamente ao artefato a, através de um ou mais elos de rastreabilidade explicitamente definidos.*

Definição 5 (Artefatos relacionados) *Seja Related*$(a) \subseteq A$ *o conjunto de todos os artefatos intrinsecamente relacionadas ao artefato a direta ou indiretamente, tendo ou não um elo de rastreabilidade estabelecido entre eles. Há dois motivos pelos quais Linked*$(a) \neq$ *Related*(a)*. (1) Não é desejável representar cada relacionamento concebível entre artefatos com um elo, já que isto produziria um "novelo" de elos ingerenciável, muitos dos quais provavelmente nunca seriam utilizados. As decisões sobre quais artefatos relacionados serão ligados devem ser baseadas em estratégias bem*

definidas, em nível de projeto. Portanto, seja StrategyRelated(a) $\subseteq$ Related(a) o conjunto de artefatos que, de acordo com as estratégias definidas no projeto, devem ser explicitamente ligados a um artefato a. Idealmente, Linked(a) = StrategyRelated(a). (2) Certos elos que devem existir podem estar ausentes devido à falhas na atualização e manutenção da infraestrutura de rastreabilidade. Por esta razão, Linked(a) $\subset$ StrategyRelated(a) é geralmente o caso.

Definição 6 (Artefatos afetados) *Seja Impacted(C,a) $\subseteq$ Related(a) o conjunto de todos os artefatos afetados por uma alteração proposta C sobre o artefato a.*

Definição 7 (Artefatos identificados) *Seja Identified(C,a) $\subseteq A$ o conjunto de todos os artefatos identificados como afetados por uma alteração proposta C sobre o artefato a. A menos que o desenvolvedor identifique manualmente artefatos externos ao esquema de rastreabilidade, Identified(C,a) deve ser um subconjunto de Linked(a)*

Definição 8 (Artefatos atualizados) *Seja Updated(C,a,t) $\subseteq$ Identified(C,a) o conjunto de artefatos atualizados, como resultado da implementação da alteração C sobre o artefato a, em um instante de tempo t no qual todos os artefatos e elos relacionados à alteração C estão atualizados.*

Além disso, certos artefatos desempenham um papel mais crítico na análise de impacto do que outros. Estes estão situados em posições centrais dentro de um grafo ou árvore de rastreabilidade, e quando inadequadamente mantidos, podem inibir a capacidade de identificar os artefatos corretos em análises de impacto futuras. Estes artefatos são definidos em termos de sua posição dentre uma série de caminhos de rastreabilidade.

Definição 9 *Um **caminho de rastreabilidade** TP é um conjunto ordenado de elos, no qual artefatos indiretamente ligados a_j e a_{j+n} estão conectados através de uma série de artefatos intermediários diretamente ligados, tal que $TP(a_j, a_{j+n}) = \{Elo(a_j, a_{j+1}), Elo(a_{j+1}, a_{j+2}), \ldots, Elo(a_{j+n-1}, a_{j+n})\}$ representa o caminho de rastreabilidade do artefato a_j até a_{j+n}. Um **caminho de rastreabilidade de requisitos** representa o caso especial para o qual a_j é um requisito. O comprimento do menor caminho de rastreabilidade a partir de um artefato para o seu requisito mais próximo especifica o **nível** deste artefato. Um artefato diretamente ligado a um requisito é dito de nível 1, enquanto que um artefato com um artefato intermediário é dito do nível 2, e assim por diante. Um artefato é dito **crítico** se está posicionado como um artefato intermediário em múltiplos caminhos de rastreabilidade, pois qualquer falha ao mantê-lo resultará na falha da manutenção de outros artefatos em níveis menores, no mesmo caminho de rastreabilidade.*

2.7 Problemas e desafios da rastreabilidade

Embora a rastreabilidade seja legalmente requerida na maioria das aplicações de segurança crítica, e seja reconhecida como um componente de muitas iniciativas de melhoria em processos de software, as organizações ainda lutam para implementá-la de modo que ofereça um bom custo-benefício (CLELAND-HUANG, 2006b). Padrões aceitos como o CMMI nível 3 e ISO 15504 exigem que práticas básicas de rastreabilidade sejam utilizadas (CLELAND-HUANG, 2006b; NEUMULLER; GRUNBACHER, 2006), mas as atividades de rastreamento e gerenciamento de requisitos

podem trazer diversos custos inesperados, que em alguns casos excedem os benefícios (DELGADO, 2006). Infelizmente, a tarefa de construir uma matriz de RR leva tempo, é árdua, cara, e sujeita a erros (CLELAND-HUANG, 2006b). Por conta disto, com o desenvolvimento do software e sua manutenção, geralmente as informações de rastreabilidade não representam mais a realidade, ou nem existem mais (LUCIA; OLIVETO; SGUEGLIA, 2006).

Os métodos de rastreabilidade existentes focam-se na criação de estruturas de relacionamentos. Embora úteis, o problema é evoluir estas estruturas (MURTA; HOEK; WERNER, 2006). Entre as soluções mais aplicadas para encontrar o melhor *tradeoff* entre o tamanho e a utilidade das informações de rastreabilidade, estão rastrear somente requisitos críticos e reduzir a granularidade (por exemplo, a tabela 2.3 poderia relacionar somente requisitos e classes). Infelizmente, nenhuma destas é satisfatória, já que uma pequena alteração pode afetar diversas partes de um sistema, e a análise de impacto deve contar com o maior número de detalhes possível para que os efeitos colaterais possam ser detectados com precisão. Além disso, muitos esquemas de rastreabilidade fornecem pouco suporte para identificar o impacto de requisitos totalmente novos (CLELAND-HUANG et al., 2002).

Não é necessário discutir o valor potencial que a RR pode trazer ao desenvolvimento de sistemas, afinal, a literatura relata que um dos maiores problemas na tarefa de manutenção está relacionado à compreensão de como o sistema funciona: alguns estudos apontam que de 30% a 60% dos custos nesta fase estão relacionados à descoberta do que o sistema faz (TRYGGESETH; NYTRO, 1997). Contudo, a RR é raramente utilizada, já que em muitos casos o custo envolvido não cobre os benefícios. Este alto custo vem: (1) da dificuldade em gerar automaticamente as relações de RR, com uma semântica clara e precisa; (2) da heterogeneidade e do grande número de artefatos que são criados durante o desenvolvimento; e (3) a falta de corretude e completude das relações de rastreabilidade (CLELAND-HUANG, 2006a).

Na prática, uma variedade de problemas de rastreabilidade geralmente ocorrem pelo uso de métodos informais de RR, falha na cooperação entre as pessoas responsáveis pela coordenação do processo, dificuldade na obtenção das informações necessárias para dar suporte ao processo, e falta de treino em práticas de RR (SALEM, 2006). Além disso, a pressão para reduzir o tempo de entrega e aumentar a produtividade resulta na adaptação de processos para esta realidade de busca por alto desempenho, que geralmente sacrifica atividades não-produtivas. O estabelecimento e manutenção da RR, bem como a manutenção da consistência entre os artefatos de software, são algumas destas atividades custosas e tediosas freqüentemente negligenciadas (PENTA; GRADARA; ANTONIOL, 2002).

Como resultado dos problemas de manutenção, um grande número de pesquisadores vem investigando o uso da recuperação automática da rastreabilidade (RAR), utilizando métodos de recuperação de informações para gerar elos dinamicamente, tais como modelos de espaço vetorial, indexação semântica, ou modelos probabilísticos de rede. Mesmo que não seja ainda uma bala de prata, esta técnica vem sendo cada vez mais reconhecida pela indústria como uma solução potencial para a rastreabildade (CLELAND-HUANG, 2006b). Porém, não existe processo de recuperação da rastreabilidade completamente automático. Todos incluem um componente humano, pois as decisões (de aceitação ou rejeição) sobre elos recuperados devem ser tomadas pelo usuário. Além disso, uma vez que os elos de rastreabilidade sejam

recuperados, devem ser mantidos de forma consistente, a fim de ser utilizados em outros processos e tarefas (MARCUS; XIE; POSHYVANYK, 2005). Portanto, um problema desafiador a ser pesquisado está em assegurar uma qualidade aceitável nos elos, sem pressionar demasiadamente os desenvolvedores (NEUMULLER; GRUNBACHER, 2006).

O problema da manutenção da rastreabilidade persiste, bem como a resistência dos desenvolvedores em aplicá-la a seus projetos. Entre os diversos fatores que geram seu alto custo, está a dificuldade em gerar elos automaticamente, com uma semântica clara e precisa (CLELAND-HUANG, 2006a). Ao ignorar a semântica das informações de RR, limita-se a capacidade de processos automáticos de dedução (RAMESH, 1998).

Enquanto a simplicidade é a principal vantagem dos métodos tradicionais, eles são pouco úteis na representação de todas as informações relevantes (MARCUS; XIE; POSHYVANYK, 2005). Além disso, os esforços dos desenvolvedores são retardados pela falta de supore ferramental, e há uma percepção geral de que o custo necessário para manter uma matriz de rastreabilidade de requisitos é excessivo, em relação aos seus benefícios (CLELAND-HUANG, 2006b; DELGADO, 2006).

Todos os estudos interessados na aplicação de métodos para a RAR mostram que estes não são capazes de recuperar todos os elos corretos, sem também recuperar muitos falsos positivos, que devem ser descartados pelo engenheiro de software. A baixa precisão faz deste método uma tarefa tediosa, já que o engenheiro tem que gastar muito mais tempo para descartar falsos positivos do que rastreando elos corretos. Embora seja possível melhorar a performance destes métodos, eles ainda estão distantes de tornar viável a solução para o problema da recuperação de elos (LUCIA et al., 2006a). Mesmo assim, esta solução vem sendo cada vez mais reconhecida pela indústria como uma solução potencial para o problema da rastreabildade (CLELAND-HUANG, 2006b). Em contrapartida, Hayes e Dekhtyar (HAYES; DEKHTYAR, 2005) estudaram os resultados produzidos por analistas, e descobriram que as decisões tomadas sobre a validade ou invalidade dos elos recuperados nem sempre são corretas, e podem até piorar os resultados: elos corretos acabam sendo descartados, e elos incorretos mantidos.

A maioria das ferramentas e métodos de rastreabilidade suportam a identificação de artefatos afetados, quando existe um conjunto preciso de elos. Contudo, grande parte destes não fornece nenhum tipo de suporte para assegurar que os artefatos e elos relacionados, afetados por alguma alteração, sejam atualizados rapidamente. A atualização é tipicamente manual, e como resultado, surgem erros. Estes podem afetar a capacidade de, futuramente, identificar os impactos de novas alterações (CLELAND-HUANG; CHANG; CHRISTENSEN, 2003).

Mesmo quando uma organização implementa um processo de rastreabilidade, sua eficiência é geralmente influenciada negativamente por dificuldades no estabelecimento, manutenção e completa utilização de elos de rastreabilidade. Entre outros problemas, os desenvolvedores freqüentemente falham em manter os elos devido às dificuldades inerentes na coordenação de membros da equipe, além da impressão de que o custo de implementar a RR excede suas vantagens (CLELAND-HUANG; ZEMONT; LUKASIK, 2004). Não obstante, a falta de percepção do benefício por parte dos desenvolvedores faz com que as tarefas de rastreabilidade tenham baixíssima prioridade, o que resulta em dados incompletos e incorretos (ARKLEY; RIDDLE, 2005). Assim, um dos maiores problemas da rastreabilidade é integrá-la ao processo

de desenvolvimento de forma que forneça benefícios tangíveis e imediatos.

O suporte ferramental eficiente para a rastreabilidade é um fator-chave para torná-la viável. Devem ser desenvolvidos métodos nos quais a rastreabilidade seja mantida automaticamente, enquanto o usuário continua seu trabalho normal. Por exemplo, um usuário pode dizer "Estou trabalhando na satisfação deste requisito", e a partir daí, tudo o que ele tocar é rastreado para aquele requisito (CLELAND-HUANG, 2006a). Além disso, para reduzir a quantidade de recursos e tempo necessários para rastrear requisitos, é necessário embutir as informações de cobertura dos requisitos no próprio software que irá atendê-los (DELGADO, 2006).

Atualmente, é muito difícil gerenciar um conjunto de requisitos de tal forma que seja possível visualizar onde estes são atendidos. Informações suficientes para isto querem que uma matriz de rastreabilidade seja mantida, estabelecendo relações entre requisitos, casos de teste, decisões de projeto, etc (LORMANS; DEURSEN, 2005).

Além disso, estabelecer e manter os traços traz uma grande sobrecarga sobre os engenheiros de software. Existem ferramentas que fornecem a infraestrutura para gerenciá-los, contudo estas não libertam o usuário de identificar links, ou de assegurar sua validade com o passar do tempo (EGYED et al., 2005a). Conseqüentemente, a RR é raramente estabelecida em ambientes industriais, onde sistemas são documentados por grandes coleções de artefatos, geralmente expressos em linguagem natural (SPANOUDAKIS, 2002).

A falta de percepção do benefício faz com que as tarefas de rastreabilidade tenham baixíssima prioridade, o que resulta em dados incompletos e incorretos. Porém, esta impressão revela-se falsa, pois com a apropriada gerência de um processo de RR, a equipe de desenvolvimento torna-se capaz de: (ARKLEY; RIDDLE, 2005)

- demonstrar que todos os requisitos foram testados;
- identificar quais partes genéricas de um software podem ser alteradas para satisfazer outros requisitos;
- justificar decisões de projeto acerca da criação de variantes de um produto;
- medir o progresso do desenvolvimento;
- entre outros.

Outro grande problema da rastreabilidade, portanto, é integrá-la ao processo de desenvolvimento de forma que forneça benefícios tangíveis e imediatos.

O trabalho de Daneva (2003) apresenta a seguinte lição: assegurar que as políticas de rastreabilidade sejam obedecidas. Negligenciá-las, especialmente na manutenção de informações de rastreabilidade, trouxe problemas de gerenciamento de informação em seis de treze projetos acompanhados no trabalho citado. A atualização esporádica de requisitos fez com que a substancial quantidade de informação não pôde ser mantida, aumentando o custo e o tempo de um processo já caro.

Tantas limitações praticamente impossibilitam o uso da rastreabilidade fácil e eficientemente, portanto, o presente trabalho propõe um novo modelo de rastreamento, cujo objetivo é mitigar os problemas relacionados à dificuldade de criação e manutenção de estruturas de rastreabilidade com maior precisão e utilidade.

3 MODELO PROPOSTO

Analisando as diversas técnicas para o estabelecimento de estruturas de rastreabilidade, brevemente apresentadas no início do capítulo 2, constatou-se que o maior problema das técnicas de rastreabilidade está na dificuldade em indentificar os motivos pelos quais as relações existem. A simples informação de que um artefato a está relacionado a outro exige grande esforço de interpretação humana para identificar esses motivos. Por conta disso, as técnicas de recuperação automática também não são eficientes, afinal, não é possível esperar que um computador interprete corretamente o significado de textos produzidos em linguagem natural e crie elos consistentes. A utilização de técnicas automatizadas também oculta dos motivos por trás da existência das relações. Assim, não trás surpresa o fato de que o estabelecimento manual das relações é a forma mais precisa de criação de uma estrutura de rastreabilidade, pois somente um humano é capaz de interpretar corretamente o conteúdo de artefatos cujo significado depende de interpretação (especialmente requisitos expressos em linguagem natural), e identificar a relação entre estes e os demais artefatos, geralmente representados por modelos heterogêneos, com significados e objetivos distintos.

Para mitigar estes problemas, e considerando especialmente as dificuldades no estabelecimento e manutenção de uma estrutura de rastreabilidade, decidiu-se introduzir um modelo cujo objetivo é facilitar o processo de criação de elos, semi-automatizando parte do processo. Espera-se que este reduza o esforço de interpretação necessário para identificar as relações, ao mesmo tempo que permita atualizações com o mínimo possível de intervenção humana.

3.1 Requisitos

Os requisitos que o modelo proposto devem atender são parte do relatório técnico do *Center of Excellence for Traceability* (CLELAND-HUANG et al., 2006), que descreve problemas e desafios relacionados aos vários aspectos da rastreabilidade em sistemas.

Suporte à evolução: Deve evoluir automaticamente com os artefatos, e maximizar o reuso dos elos de rastreabilidade quando os artefatos forem reusados em um novo produto;

Escalabilidade: Deve permitir o rastreamento através de conjuntos de dados heterogêneos; além disso deve permitir o rastreamento através de conjuntos de dados fracamente estruturados;

Automação: Deve facilitar a geração e manutenção da rastreabilidade das informações.

3.2 Visão geral

O processo de desenvolvimento de software compreende a criação de diferentes abstrações da realidade, usadas para representar e controlar a arquitetura do sistema, avaliar oportunidades e identificar riscos. O termo "modelo" é geralmente usado para se referir à estas abstrações (NOLL; RIBEIRO, 2007). Um modelo de requisitos, por exemplo, expressa idéias, conceitos. Estes conceitos, relacionados em um contexto, formam um significado. Para facilitar o entendimento deste significado, pode-se usar vários modelos, em um ou mais níveis de abstração, para expressar diferentes pontos de vista.

O modelo proposto visa o rastreamento desses modelos, e conseqüentemente, de diferentes níveis de abstração e pontos de vista sobre um determinado interesse. Como resultado, torna-se possível identificar onde o interesse é satisfeito, qual a influência de outros interesses sobre este, etc. Por exemplo, o requisito R1, na seção 2.1, trata dos conceitos de "cliente", "conta", "crédito" e "limite". Estes são relacionados de forma a sugerir a idéia de que "O cliente tem uma determinada conta, com um determinado crédito, que está restrito a um determinado limite". Basicamente, um requisito como: "o cliente não pode creditar um valor superior ao limite de sua conta", relaciona os conceitos supracitados, de forma a expressar uma necessidade que o sistema deve atender.

Outros requisitos expressam outras necessidades, com base em alguns dos mesmos conceitos. Por exemplo, os requisitos R2, R3 e R4, apesar de tratarem de necessidades diferentes, tratam de interesses em comum. Estes podem ser usados para identificar relações, divergentes ou complementares, entre os requisitos e demais abstrações do sistema (modelos diversos, código fonte, etc). Por exemplo, considerando os requisitos R1 e R4, pode-se identificar conceitos tratados por ambos, e conseqüentemente, uma ou mais relações entre R1 e R4. A figura 3.1 ilustra como estas relações podem ser formadas.

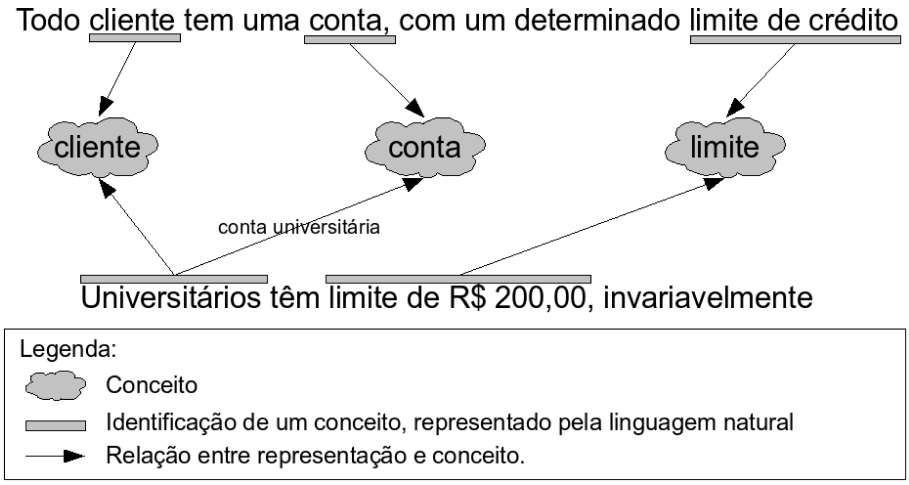


Figura 3.1: Relações formadas pelos conceitos encontrados nos requisitos R1 e R4

Os demais artefatos que compõe o sistema devem satisfazer os requisitos, e por-

tanto, certamente tratam dos mesmos interesses de alguma forma. Assim, ao mapear os interesses identificados nesses artefatos, deverá ser possível derivar, automaticamente, relações complexas entre artefatos, utilizando como referência os interesses que compartilham. Dessa forma, pode-se relacionar todos os artefatos e seus elementos aos interesses que representam, permitindo recuperar elos de rastreabilidade implícitos (NOLL; RIBEIRO, 2007).

O presente trabalho propõe um modelo de rastreabilidade baseado na identificação destes interesses, denominados entidades. O modelo permite registrar as áreas de interesse que dão origem às entidades, bem como classificá-las de acordo com suas características.

O modelo de entidades extende o proposto por Jiang et al (JIANG et al., 2007), que usa centralizadores de elos de rastreabilidade (denominados âncoras), identificando áreas de interesse comuns a mais de um artefato, conforme a figura 3.2.

Figura 3.2: Âncoras representam áreas de interesse dos artefatos

3.2.1 Entidades

O modelo proposto permite classificar as áreas de interesse de acordo com a interpretação do analista sobre o conteúdo identificado como "de interesse":

- **Agentes** causadores de modificações sobre o estado do sistema (eg. cliente);
- *Atributos* pertencentes a agentes, que contêm algum valor (eg. saldo, limite);
- ***Valores*** de atributos (eg. R$200, status bloqueado, etc);
- Ações que manipulam os atributos de um ou mais agentes (eg. creditar, debitar, verificar limite);
- Restrições que identificam limites para os valores dos atributos de um ou mais agentes, ou restringem suas ações (eg. saldo não pode ser inferior ao limite);
- `Entidades sem classificação definida` usadas para simplesmente rastrear áreas de interesse identificadas nos artefatos.

Entidades, além de atuarem como centralizadores, são usadas para abstrair, classificar e relacionar as áreas de interesse identificadas nos requisitos. A figura 3.3 exibe as áreas de interesse (na forma de elipses) que são identificadas nos artefatos, e que dão origem às entidades.

Utilizando esta forma de classificação, as relações entre entidades ganham significado, conforme representa a figura 3.4. Um agente está relacionado a uma restrição, este a um atributo, o atributo a uma entidade, uma entidade à uma ação, etc.

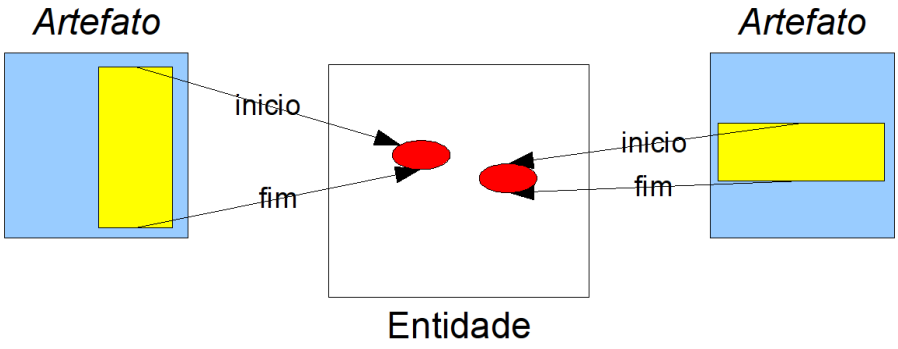


Figura 3.3: Entidades são âncoras, indentificando áreas de interesse

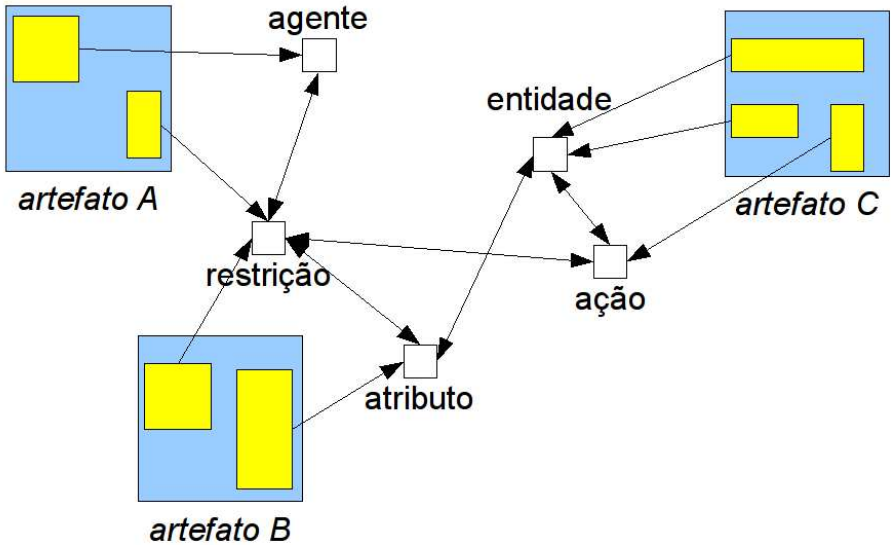


Figura 3.4: A classificação de entidades facilita a interpretação da estrutura de rastreabilidade

3.2.2 Atributos nas relações

O trabalho de Ramesh *et. al.*(2001) apresenta modelos de referência para a rastreabilidade de requisitos, utilizados em diversos trabalhos subseqüentes (exemplos: (EGYED; GRUNBACHER, 2002; JARKE, 1998; RAMESH, 1998; SPANOUDAKIS, 2002; ESPINOZA; ALARCON; GARBAJOSA, 2006)). Os aspectos essenciais da rastreabilidade são capturados em diversos meta-modelos, que fornecem as primitivas básicas para categorizar e descrever modelos de rastreabilidade em detalhe. Entre um dos aspectos mais importantes, está a necessidade de relacionar decisões, motivos, responsáveis, etc, a cada relacionamento. Para satisfazer tal requisito, o modelo proposto permite associar atributos a quaisquer relações:

Entre áreas de interesse e entidades cada área de interesse, identificando uma entidade, poderá ser associada a diversos atributos, conforme a necessidade do usuário do modelo. A figura 3.5 apresenta esta associação.

Entre entidades cada relação entre entidades poderá ser associada a diversos atributos, conforme a necessidade do usuário. A figura 3.6 apresenta esta associação.

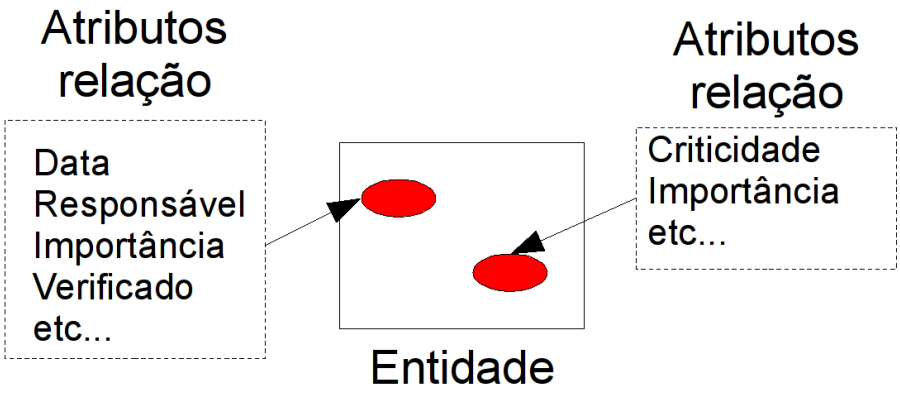


Figura 3.5: Atributos associados a áreas de interesse

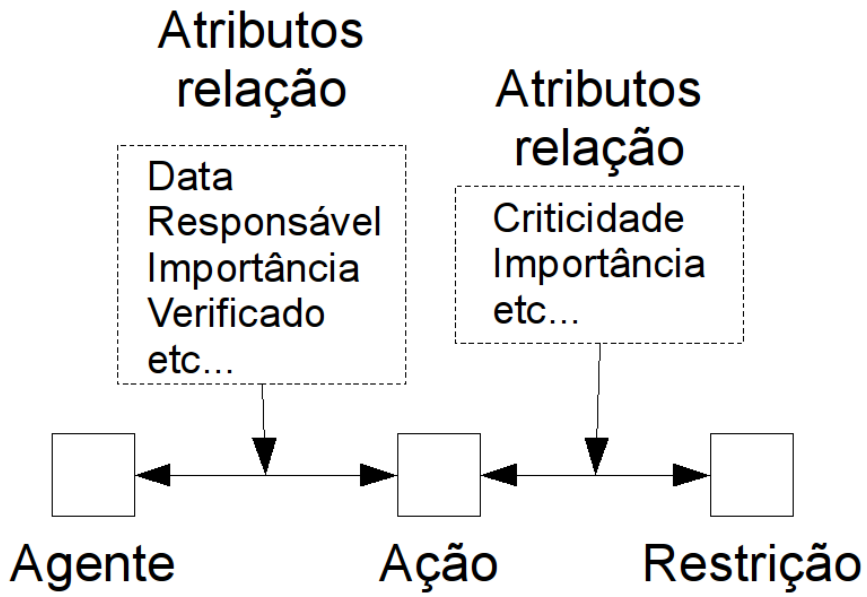


Figura 3.6: Atributos associados a relações entre entidades

Note que os atributos são definidos pelo usuário, e não são determinados pelo modelo. Este somente fornece o meio pelo qual o usuário poderá registar atributos a áreas de interesse e relações entre entidades. Somente o usuário, em seu contexto, poderá determinar quais são as informações importantes para seu problema.

3.2.3 Tipos de relação

Os relacionamentos entre artefatos podem ser tipados (ESPINOZA; ALARCON; GARBAJOSA, 2006), dando maior semântica a estes, e facilitando a compreensão da estrutura de rastreabilidade (RAMESH; JARKE, 2001). O modelo proposto segue a mesma idéia, permitindo identificar tipos de relação entre artefatos e entidades:

defines: relação entre uma área de interesse em um artefato e uma entidade (lê-se: a área de interesse identifica a entidade);

defined_by: relação entre uma área de interesse em um artefato e um agente (lê-se: a entidade é identificada pela área de interesse);

No caso de relacionamentos entre entidades somente, são definidos os seguintes tipos de relação:

related_to: relação de uma entidade qualquer à outra;

attribute_of: relação de um atributo a um agente;

has_attribute: relação de um agente a um atributo;

restriction_of: relação de uma restrição a um agente;

has_restriction: relação de um agente a uma restrição;

action_of: relação de uma ação a um agente;

has_action: relação de um agente a uma ação;

value_of: relação de um valor a um atributo;

has_value: relação de um atributo a um valor;

restricted_by: relação de uma entidade a uma restrição;

affected_by: relação de uma entidade a uma ação;

based_on: relação de uma ação ou restrição a uma entidade;

base_of: relação de uma entidade a uma ação ou restrição;

affected_by: relação de uma entidade a uma ação;

affects: relação de uma ação a uma entidade; e

restricts: relação de uma restrição a uma entidade.

A classificação das relações entre as entidades tem por objetivo facilitar a consulta à estrutura de rastreabilidade. A figura 3.7 apresenta as relações entre as entidades:

Ao realizar uma consulta na estrutura de rastreabilidade, pode-se, por exemplo, pesquisar por todas as áreas de interesse relacionadas ao agente afetados por (*affected_by*) uma ação. Outras relações, com significado mais específico, podem ser definidas conforme desejado. A vantagem de definir tipos de relações é que as consultas na estrutura de rastreabilidade podem ser realizadas de forma mais precisa e compreensível para seres humanos.

3.2.4 Modelo ER da proposta

O modelo proposto pode ser representado na forma de um diagrama de entidade-relacionamento (ER). A figura 3.8 apresenta este diagrama. Através da implementação do modelo em um banco de dados, as consultas à estrutura de rastreabilidade podem ser facilmentes realizadas com comandos SQL.

A seção a seguir trata da aplicação do modelo através de simples exemplos, sendo seguida por considerações diversas sobre o modelo.

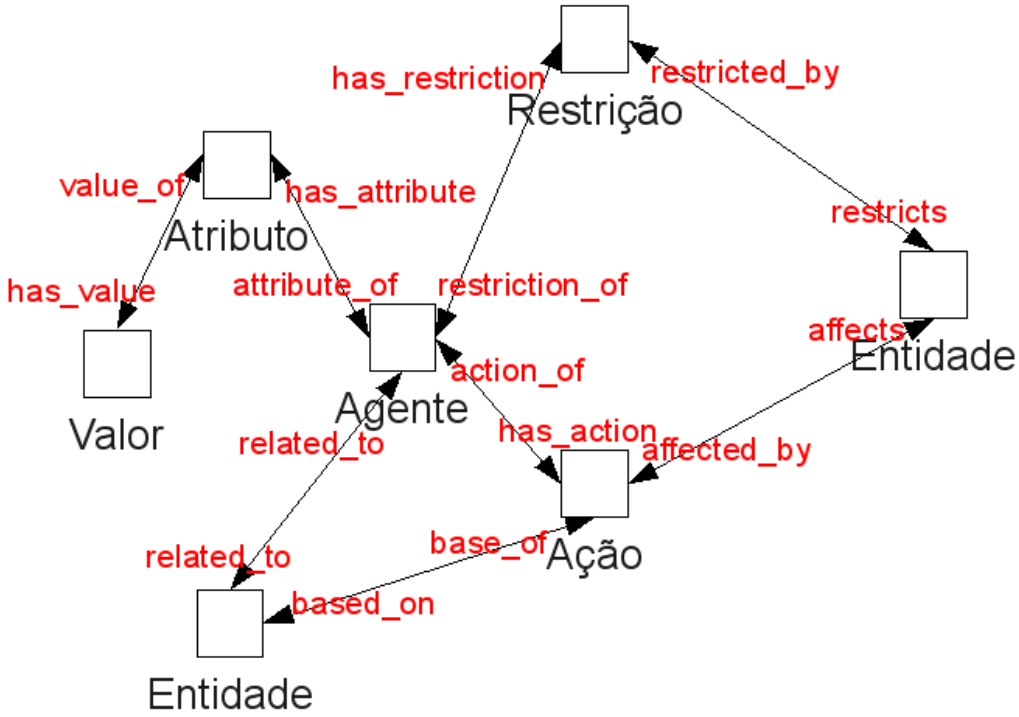


Figura 3.7: Relações entre entidades

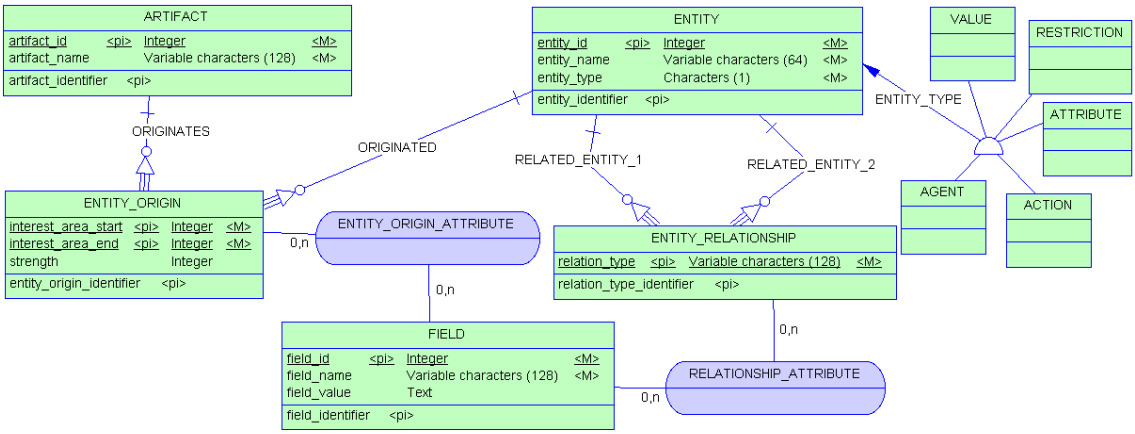


Figura 3.8: Modelo de rastreabilidade proposto

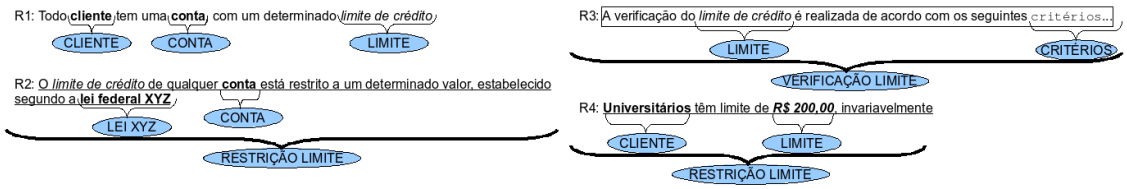


Figura 3.9: Entidades identificadas nos requisitos

3.3 Aplicação do modelo

Analisando os requisitos R1, R2, R3 e R4 da seção 2.1, pode-se identificar diversas entidades, conforme ilustra a figura 3.9.

É interessante ressaltar que as entidades representam, de forma abstrata, ele-

Tabela 3.1: Relacionamentos entre as entidades identificadas

Agente	Cliente	Conta	Lei XYZ
Atributos	conta	limite	
Ações		verificação limite	
Restrições		restrição limite	
Ação	verificação limite		
Baseada em	critérios		
Afeta	limite		
Restrição	restrição limite		
Baseada em	lei XYZ, cliente, conta, limite		
Restringe	limite		

mentos que fazem parte dos requisitos funcionais, e que de alguma forma são implementados no sistema, e representados em diferentes artefatos. Esta representação é exclusivamente conceitual, e não tem por objetivo representar parte da modelagem do sistema. Da mesma forma que uma matriz de rastreabilidade, estas estruturas servem apenas como uma representação que permite identificar relacionamentos entre artefatos diversos.

Após identificadas as entidades de interesse, e suas origens nos requisitos ou demais artefatos, é possível derivar relacionamentos complexos semi-automaticamente, conforme descrito na seção 3.3.1, a seguir.

3.3.1 Relacionamentos

Pode-se estabelecer relacionamentos entre as entidades conforme suas características. Por exemplo, a entidade "restrição limite" é uma restrição baseada em "lei XYZ", que se aplica a "limite". Já a entidade "verificação limite" é uma ação que se baseia em "critérios" para manipular "limite". A tabela 3.1 exibe os relacionamentos formados entre as entidades criadas com os requisitos R1 a R4.

Analisando as entidades identificadas na figura 3.9, percebe-se que há entidades comuns a mais de um requisito (limite, conta, cliente e restrição limite). Usando estas entidades como âncoras, pode-se derivar, automaticamente, diversos relacionamentos entre os requisitos:

1. Relacionamentos diretos:

   $\{a \leftrightarrow b | a \in \{R1, R2, R3, R4\}, b \in \{R1, R2, R3, R4\}\}$**:** Relacionados entre si por originarem a entidade "limite";

   $R1 \leftrightarrow R2$**:** por tratarem da entidade "conta" ;

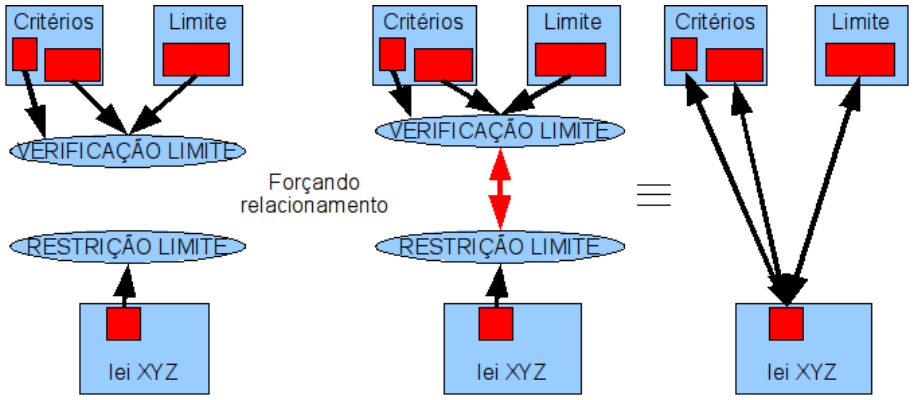
   $R1 \leftrightarrow R4$**:** por tratarem da entidade "cliente";

   $R2 \leftrightarrow R4$**:** por tratarem da entidade "restrição limite";

2. Relacionamentos indiretos:

   $\{R2 \leftrightarrow a | a \in \{R1, R2, R3, R4\}\}$**:** pois a entidade "restrição limite" restringe a entidade "limite", sendo que esta última é originada por todos os requisitos. Além disso, essa restrição se baseia em "Lei XYZ" e "Conta"(R2), e também em "Cliente" (R4);

Figura 3.10: Relacionamento forçado entre entidades



$\{R3 \leftrightarrow a | a \in \{R1, R2, R3, R4\}\}$**:** pois a entidade "verificação limite" afeta a entidade "limite", como no caso anterior;

Outros: como "limite" é atributo de "conta", qualquer requisito que origine um estará indiretamente ligado a qualquer requisito que origine o outro.

3. Relacionamentos forçados: É possível estabelecer relacionamentos entre as entidades manualmente. Por exemplo, considere que a entidade "Lei XYZ" seja relacionada a "Critérios", será possível derivar novos relacionamentos automaticamente:

   $R2 \leftrightarrow R3$**:** pois R2 dá origem à entidade "Lei XYZ" e R3 à "Critérios", implicando que, em uma análise de impacto dos efeitos de alguma alteração na lei, os critérios especificados pelo requisito R3 sejam apontados para revisão. A figura 3.3.1 apresenta um exemplo deste tipo de relacionamento.

Note que o crescimento dos relacionamentos derivados (entre requisitos e entre demais artefatos, tanto horizontal, quanto verticalmente) é exponencial, enquanto o crescimento dos relacionamentos manuais, entre requisitos e entidades, é linear. O modelo proposto requer o estabelecimento manual apenas de relações simples (área de interesse para entidade, ou entidade para entidade), permitindo que as relações complexas entre artefatos (áreas de interesse de um artefato para áreas de interesse de outros artefatos) sejam derivadas automaticamente.

Em relação à manutenção dos relacionamentos entre artefatos e entidades, todas as alterações realizadas sobre um artefato, que removam as áreas de interesse relacionadas a alguma entidade, farão o relacionamento desaparecer. Para que isto ocorra, o modelo exige que toda área de interesse seja identificada no próprio artefato, para que alterações neste atualizem as áreas de interesse automaticamente. Para reduzir a quantidade dos recursos e tempo necessários para o rastreamento, é necessário embutir as informações de cobertura dos requisitos no próprio software que irá atendê-los (DELGADO, 2006). Por exemplo, para identificar as entidades no texto de um documento de requisitos D, pode-se usar uma sintaxe abstrata com a forma:

$$
\begin{array}{lll}
D & ::= & palavra\ D | TAG\ D | \varepsilon \\
TAG & ::= & < ENTITY\_ID > D < \backslash ENTITY\_ID > \\
ENTITY\_ID & ::= & \mathbb{N}
\end{array}
$$

Ou seja, um documento de requisitos é formado por palavras e também por *tags*, que determinam quais são as entidades identificadas no texto. O não-terminal TAG mantém as informações das entidades identificadas no texto que cinge, identificando uma área de interesse. Na prática, este modelo requer um editor especial, que "esconda" as tags e renderize o texto entre estas. Assim, ao apagar um trecho de texto que contenha tags, o relacionamento com as entidades será simplesmente descartado. Caso uma entidade não possua relacionamento com nenhum artefato, esta será removida, bem como as relações que possui com outras entidades. A idéia por trás deste formato de definição da entidades é que o analista simplesmente diga: "Estou definindo o agente X, ou parte dele", e todos os artefatos criados a partir daí sejam relacionados ao "agente X", automaticamente, conforme idealizou Cleland-Huang ((CLELAND-HUANG, 2006a)), sendo considerada pela pesquisadora a "solução ideal" para o problema da rastreabilidade.

Note que a única tarefa de manutenção da estrutura de rastreabilidade é identificar novas entidades em novas versões dos artefatos, sendo todos os relacionamentos entre artefatos derivados automaticamente. O relacionamento entre entidades e artefatos é tal que caso os artefatos mudem, as entidades e suas relações podem ser reconstruídas automaticamente. Basta comparar o estado anterior da estrutura com o atual para identificar todos os artefatos possivelmente afetados pela alteração. Isto deixa como tarefa manual apenas o rastreamento de novas entidades que porventura sejam introduzidas.

Dessa forma, todos os artefatos produzidos no ciclo de desenvolvimento podem ser utilizados para identificar entidades, basta criar um formato de representação de uma área de interesse, compatível com o modelo do artefato, como feito com o documento de requisitos ***D***. Isto permite que a estrutura de rastreabilidade seja automaticamente atualizada à medida que os artefatos ou requisitos mudam.

3.3.1.1 Rastreamento horizontal

Utilizando o modelo proposto, basta relacionar suas áreas de interesse às entidades já existentes, ou a uma nova, caso necessário. Por exemplo, analisando a listagem 1, é possível relacionar áreas de interesse às entidades já identificadas, ilustradas na figura 3.9. A figura 3.11 apresenta estas relações.

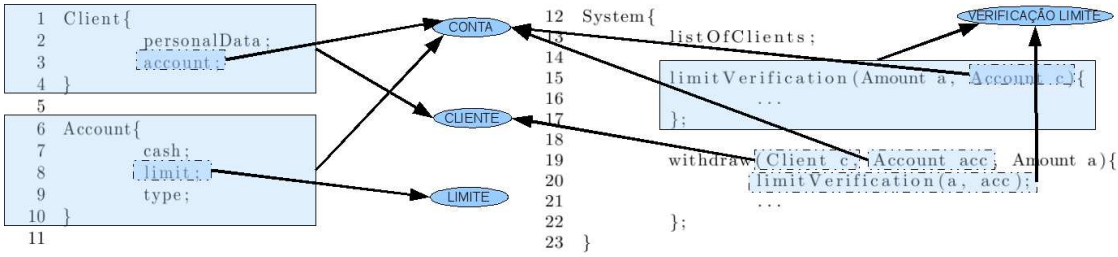


Figura 3.11: Relação entre áreas de interesse no código e entidades

Utilizando esta estrutura, caso um requisito seja alterado, e o analista interpretar que a ação de verificação deva ser realizada de forma diferente, todos os atributos e agentes relacionados serão identificados automaticamente como possíveis entidades afetadas. Esta identificação será propagada para os demais artefatos (possivelmente afetados) que as entidades representam.

Por exemplo, considere que a lei XYZ seja alterada e o sistema deva se adequar à nova lei. Que partes do sistema serão afetados por esta alteração? Tendo identificada a lei XYZ como uma entidade, pode-se pesquisar quais ações e restrições se baseiam em "lei XYZ". Na tabela 3.1, tem-se que "restrição limite" se baseia em "lei XYZ", e conforme a figura 3.3.1, a entidade "verificação limite" está relacionada a "restrição limite". Basta identificar, na implementação, as áreas de interesse relacionadas a "verificação limite". A figura 3.11 identifica as linhas 15 a 17, e a linha 20, que possivelmente serão impactadas pela alteração na lei.

Se for desejado ir mais a fundo na estrutura de rastreabilidade, pode-se identificar outras áreas que podem sofrer algum impacto pela alteração. Por exemplo, a verificação de limite afeta o atributo limite (linha 8), que pertence a uma conta (linhas 6 a 10). Esta, por sua vez, pertence a um cliente (linhas 1 a 4), e assim por diante. O quanto deve-se percorrer a estrutura de rastreabilidade, para encontrar possíveis impactos, depende da criticidade da alteração. No exemplo citado, é improvável que seja necessário ir além das áreas de interesse relacionadas a "verificação limite".

3.4 Considerações Sobre o Modelo

Note que depois de estabelecidas as relações de rastreabilidade, estas são automaticamente mantidas, da seguinte forma:

Alteração nos requisitos: entidades identificadas no texto antigo dos requisitos, que não existem mais no atual, automaticamente têm suas relações desabilitadas, sendo que estas relações, e todos os caminhos de rastreabilidade que passam por esta entidade, devem ser verificados (CLELAND-HUANG; CHANG; CHRISTENSEN, 2003). As entidades atuais, que ainda existirem nos requisitos, permanecerão habilitadas, bem como suas relações. Caso novas sejam identificadas, estas deverão ser criadas e relacionadas com as existentes manualmente.

Alteração nas entidades: se uma entidade for desabilitada, todas as relações da qual faz parte também serão, e os caminhos de rastreabilidade que passam por esta deverão ser verificados. As porções de texto nos requisitos que identificam a entidade serão relacionadas como desabilitadas. O mesmo ocorrerá com os artefatos que a representam, e as relações entre os mesmos.

Alteração nos elos entre entidades: Caso somente um elo entre entidades seja desabilitado, os requisitos e artefatos associados às entidades automaticamente não mais terão elos entre si.

Alteração direta nos artefatos: Depende da semântica dos artefatos. Por exemplo, no caso de modelos como diagramas de classe, caso um conjunto de classes não esteja de forma alguma relacionado à outro, então não haverá relação entre as entidades representadas pelas classes. Caso for criado um relacionamento entre algumas destas classes, as entidades serão relacionadas automaticamente, assim como os requisitos.

Alteração direta no código: no caso de qualquer alteração no código, basta recuperar automaticamente as áreas identificadas pelos comentários de rastrea-

mento, e remontar toda a estrutura de rastreabilidade, isto é, recriar entidades e seus relacionamentos. Como o código fonte é a fonte mais confiável para validação, que representa a realidade nua e crua (EGYED, 2002), a estrutura de rastreabilidade recuperada nunca ficará desatualizada. É possível automaticamente compará-la com a anterior, e desabilitar elos e entidades que não mais existem.

Note que além de automatizar a manutenção da rastreabilidade, o modelo também automatiza parte de seu estabelecimento.

O estabelecimento dos elos pode ser realizado livremente, com o grau de granularidade dependendo das necessidades do projeto:

1. as entidades podem ser mais genéricas ou mais específicas, dependendo da necessidade;

2. as relações podem ser estabelecidas entre quaisquer artefatos, produzidos em quaisquer fases do ciclo de vida;

3. as informações de rastreamento no código fonte podem se relacionar a quaisquer regiões do código: declarações de funções e métodos, classes inteiras, e até mesmo linhas.

Além disso, o modelo fornece integração com o processo de desenvolvimento, podendo ser utilizado desde o início do ciclo de vida do software, e no gerenciamento de todo o processo. Por exemplo, é possível identificar quantos requisitos já foram satisfeitos, e quantos ainda faltam para o término do trabalho, entre outras informações.

O processo de desenvolvimento de software é um processo de decomposição de uma abstração, partindo de um modelo totalmente conceitual para um formal, computável. Os conceitos envolvidos neste processo são sempre apresentados em diferentes representações do sistema. O uso de conceitos para rastrear artefatos é uma excelente estratégia para este tipo de cenário. Ao fazer isso, a granularidade é reduzida, permitindo mais precisão na descoberta de artefatos relacionados (NOLL; RIBEIRO, 2007).

A proposta deste modelo é permitir a semi-automatização do trabalho de manutenção de elos, permitindo que, após estabelecidas as relações entre artefatos e entidades, estas sejam automaticamente mantidas. A identificação das áreas de interesse dentro dos próprios artefatos dá esta capacidade ao modelo.

Além disso, a proposta não visa alterar as práticas atuais de desenvolvimento, e sim complementá-las, quando a utilização da rastreabilidade for desejada, ou mesmo imposta. Além de automatizar a manutenção dos elos existentes, é também possível automatizar parte de sua criação, eliminando a necessidade de trabalho repetitivo na criação de relações óbvias (NEUMULLER; GRUNBACHER, 2006).

Diferente de muitas soluções de rastreabilidade existentes, o modelo proposto leva em conta as três dimensões da rastreabilidade (BIANCHI; VISAGGIO; FASOLINO, 2000):

Vertical e horizontal: distingue itens interconectados pertencentes ao mesmo modelo (rastreabilidade vertical), e pertencentes a modelos diferentes (rastreabilidade horizontal);

Implícita e explícita: elos explícitos são pré-definidos, ou outras formas de referências implementadas entre itens. Já elos implícitos ocorrem em todas as situações nas quais não existe relação explícita, mas é possível utilizar outras técnicas para este propósito (por exemplo, rastreamento em modelos diferentes, relacionando entidades cujos nomes são idênticos em ambos);

Estrutural e cognitiva: elos estruturais são aqueles onde existe um relacionamento de dependência sintática entre artefatos. Já os elos cognitivos são produzidos pelo conhecimento semântico associado a decisões de projeto.

O estabelecimento dos elos pode ser realizado livremente, com o grau de granularidade dependendo das necessidades do projeto:

1. as entidades podem ser mais genéricas ou mais específicas, dependendo da necessidade;

2. as relações podem ser estabelecidas entre diversos artefatos, produzidos em quaisquer fases do ciclo de vida, desde que permitam a identificação de áreas de interesse;

3. as informações de rastreamento no código fonte podem se relacionar a quaisquer regiões do código: declarações de funções e métodos, classes inteiras, e até mesmo linhas.

Além disso, o modelo fornece integração com o processo de desenvolvimento, podendo ser utilizado em qualquer estágio do ciclo de vida do software. Basta começar a rastrear as entidades de interesse, que os elos de rastreabilidade derivados das entidades formarão automaticamente os relacionamentos complexos entre artefatos, necessários às atividades de gerenciamento.

Assim, pode-se reduzir a granularidade da rastreabilidade, fornecendo elos mais precisos e com mais semântica do que os métodos baseados somente em relações diretas (NOLL; RIBEIRO, 2007).

4 AVALIAÇÃO DA PROPOSTA

A presente proposta apresenta duas perspectivas para sua avaliação. A primeira está relacionada com a necessidade de desenvolver uma metodologia que comprove a viabilidade do uso de entidades no rastreamento de artefatos, apresentada no capítulo anterior. A segunda perspectiva está relacionada com o processo de utilização da proposta e seus benefícios. Para se avaliar uma metologia, é necessário que pessoas o utilizem. Neste contexto, estudos experimentais são determinantes para uma boa avaliação, provendo uma disciplinada, sistemática, quantificada e controlada forma de avaliar as atividades desenvolvidas por humanos.

Existe uma discussão na comunidade de Engenharia de Software sobre sua consideração como engenharia ou ciência. Esta discussão é devido ao seu caráter multidisciplinar que correlaciona questões técnicas, tais como linguagens de programação, sistemas operacionais, sintaxe e semântica, com questões sociais e psicológicas, característica da engenharia e da produção (WOHLIN et al., 2000).

Para avaliação de processo onde o fator humano é considerado, a literatura provê algumas abordagens baseadas em uma estratégia experimental. Em (PFLEEGER; ATLEE, 2005), as seguintes abordagens são definidas para avaliação de processos, produtos e recursos:

- Análise das características: é a mais simples e corresponde a uma abordagem subjetiva, utilizada para atribuir um valor e classificar os atributos de vários métodos, visando decidir qual utilizar. Esta abordagem corresponde a um estudo em retrospectiva e é útil para estreitar o leque de opções a serem escolhidas, porém não avalia o comportamento em termos de causa e efeito;

- Pesquisa de opinião (survey): é um estudo em retrospectiva que visa documentar as relações e os resultados de certa situação. Durante a realização da pesquisa, registram-se as informações sobre uma situação, comparando-as com informações semelhantes. Não ocorre a manipulação de variáveis neste estudo;

- Estudo de Caso: ao contrário das anteriores, esta abordagem define previamente o que se deseja investigar, identificando os principais fatores que possam afetar o resultado de uma atividade. Após sua execução, é realizada a documentação de suas entradas, restrições, recursos e saídas. Esta abordagem é utilizada principalmente para observar projetos ou atividades, sem muito controle sobre o objeto de estudo;

- Experimento: representam o tipo de estudo mais controlado, geralmente realizado em laboratórios. Nesta abordagem, os valores das variáveis independentes

(entradas do processo de experimentação) são manipulados para se observar as mudanças nos valores das variáveis dependentes (saídas do processo de experimentação). Ao término da execução do experimento, os resultados são analisados, interpretados, apresentados e por fim, empacotados.

Em (KITCHENHAM; PICKARD; PFLEEGER, 1995), é observado que as diferenças entre os métodos de pesquisa são refletidas em suas escalas. Por sua natureza, como os experimentos requerem bastante controle, eles tendem a ser pequenos, envolvendo um reduzido número de pessoas ou eventos. Pode-se pensar em experimentos como "pesquisas em um ambiente restrito". Os estudos de caso geralmente abordam um projeto típico em vez de tentar obter informações sobre todos os possíveis casos; eles podem ser considerados como "pesquisas em um ambiente típico".

O objetivo deste trabalho consiste em investigar métodos alternativos para rastreabilidade, identificando relações como "melhor do que" ou "mais preciso que". É possível, assim, isolar as variáveis que determinam esta relatividade do resto do processo e manipulá-las, avaliando os resultados a partir dos tipos de combinações possíveis. Para isso, torna-se necessário um controle sobre as variáveis independentes do experimento.

Frente às observações aqui apresentadas, optou-se pela utilização de um experimento para avaliação da proposta. Devido ao experimento ser puramente quantitativo (WOHLIN et al., 2000), é necessário uma abordagem adicional para a avaliação. Para este fim, será utilizada a pesquisa de opinião integrada ao experimento.

4.1 Engenharia de software experimental

O processo de experimentação requer um preparo para conduzir e analisar corretamente o objeto de estudo. Este esforço representa seu maior benefício: a capacidade de controlar as variáveis relacionadas às entradas do exercício, permitindo a geração de conclusões significativas para o experimento. Na presente avaliação, faz-se necessário o controle dos indivíduos que estarão aplicando a proposta de rastreabilidade, utilizando tanto o método proposto quanto o manual, utilzando matrizes de rastreabilidade. Para conduzir o experimento, será utilizado como guia a proposta de (WOHLIN et al., 2000).

O processo de experimentação adotado compreende os seguintes passos:

Definição: corresponde à definição clara das hipóteses e dos objetivos a partir do problema a ser resolvido. É proposto um framework para suas constituintes, que compreende: objeto de estudo, propósito, foco na qualidade, perspectiva e contexto;

Planejamento: corresponde a fundamentação do experimento, onde seu contexto é explicitado em detalhes. Este passo compreende a formalização da hipótese nula (hipótese fundamental na qual o experimento objetiva rejeitar em favor das demais) e das alternativas (demais hipóteses que o experimento visa apoiar), determinação das variáveis independentes e dependentes, seleção dos participantes, preparação conceitual da experimentação e a consideração sobre a validade do experimento;

Execução: este passo compreende a preparação, execução e validação dos dados. É durante esta etapa que ocorre a interação humana, sendo necessário preparar

os participantes sob o ponto de vista moral e metodológico, evitando assim resultados errôneos ou desinteressados.

Análise e Interpretação: com a entrada de dados no passo anterior, é possível a análise e interpretação do experimento. Para tanto, ocorre à compreensão dos dados e a verificação das hipóteses utilizando estatística descritiva, possibilitando assim conclusões sobre a validade do experimento;

Empacotamento: este passo compreende a documentação dos resultados e a estruturação do experimento, visando possibilitar sua replicação.

É importante esclarecer que um experimento nunca irá proporcionar uma resposta final para uma questão, porém uma resposta específica para uma determinada configuração. O empacotamento é importante para que o experimento seja replicado em circunstâncias diversas, provendo um conhecimento adicional sobre os conceitos estudados. Nas próximas seções, serão detalhados os passos para condução do experimento.

4.1.1 Definição

A fase de definição corresponde à fundamentação dos objetivos do experimento e a definição de seu escopo, identificando os objetos e os grupos de estudo envolvidos. Será utilizada a abordagem de (BASILI; CALDIERA; ROMBACH, 1994) chamada Goal Question Metric (GQM), que corresponde a uma medição para melhoria de processo de software dirigida por objetivos. Esta abordagem parte da definição dos objetivos (nível conceitual) para o estabelecimento de questões (nível operacional), que tentam caracterizar o processo em termos de métricas (nível quantitativo).

4.1.2 Planejamento

Para que o processo de experimentação seja bem sucedido, é necessário um planejamento de todas as atividades envolvidas. A fase de planejamento pode ser dividida em sete passos:

1. Seleção do contexto: com base em sua definição, este passo seleciona o ambiente no qual o experimento será executado;

2. Formulação das hipóteses: um experimento é normalmente formulado através de hipóteses, que representam a base para a análise estatística.

3. Seleção das variáveis: este passo compreende a definição das variáveis independentes e dependentes do experimento, isto é, as entradas e saídas do processo que será observado;

4. Seleção dos indivíduos: esta atividade se relaciona com a generalização dos resultados do experimento, com o objetivo de definir um conjunto representativo de indivíduos;

5. Projeto: determina a forma na qual o experimento será conduzido, incluindo seus objetos e seus participantes. A escolha do projeto correto é crucial para a validade das conclusões;

6. Instrumentação: representa a implementação prática, fornecendo os meios de como conduzir e monitorar o experimento;

7. Análise da Validade: realiza a avaliação dos resultados do experimento. Esta análise inclui a validade interna e externa, além da validade de sua construção e de sua conclusão.

4.1.2.1 Seleção do Contexto

O contexto é composto pelas condições na qual o experimento será executado. A seleção adotada contempla as seguintes dimensões:

Processo: • In-vitro ou In vivo: o primeiro corresponde à experimentação controlada em um laboratório e o segundo em um projeto real;

Participantes: • Alunos ou Profissionais: define os indivíduos que farão parte do experimento;

Realidade: • Problema desenvolvido em sala de aula ou Problema real: define o tamanho do problema que será estudado;

Generalidade: • Específico ou Geral: apresenta o escopo da validade do experimento, em âmbito específico e contextualizado ou em todo o domínio da Engenharia de Software;

4.1.2.2 Formulação das Hipóteses

A formulação das hipóteses corresponde à definição formal sobre o que se pretende com o experimento. Conforme já apresentado, a hipótese fundamental se chama Hipótese Nula (H_0) e representa a não-ocorrência da relação de causa e de efeito do experimento, ou seja, a não-derivação de seus objetivos. Portanto, o objetivo do experimento é rejeitá-la em favor de uma ou mais hipóteses alternativas ($H_1, H_2, ..., Hn$).

4.1.2.3 Seleção das variáveis

A escolha das variáveis para o experimento não é uma tarefa trivial e geralmente requer certo conhecimento sobre o domínio do problema. É necessário que as variáveis independentes (entrada da experimentação) sejam controladas e que exerçam alguma influência sobre as variáveis dependentes (saída da experimentação).

Um experimento compreende observações sobre a alteração de uma ou mais variáveis independentes, também chamadas de fatores. Durante o estudo, é definido um fator, e as demais variáveis independentes são fixadas. Isso é necessário para saber qual fator é responsável pelo fenômeno observado, também chamado de tratamento.

Os tratamentos são aplicados para a combinação de indivíduos e objetos. Para exemplificar em termos práticos, pode-se considerar um objeto como o conjunto de artefatos recuperados pela metodologia proposta, ou pelo método tradicional. Os indivíduos podem ser representados pelos participantes que definem os elos de rastreabilidade e a aquisição do conjunto de artefatos rastreados. Ambos indivíduos e objetos são variáveis independentes do experimento. Um experimento é estruturado através de testes, onde a combinação de tratamentos, indivíduos e objetos são avaliados.

4.1.2.4 Seleção dos Indivíduos

A seleção dos indivíduos é particularmente importante para a geração de resultados relevantes durante o experimento. Para tanto, é necessário escolher uma população representativa.

4.1.2.5 Projeto do experimento

Para se obter conclusões significativas em um experimento, é necessário recuperar cuidadosamente os dados e aplicar métodos de análise estatística. O projeto do experimento corresponde à forma na qual todas as atividades serão planejadas e conduzidas para obtenção destas conclusões.

O experimento consiste em uma série de testes sobre os tratamentos de variáveis independentes. A atenção nesta fase de projeto consiste em identificar o número de vezes que os testes serão executados para que se torne visível os efeitos dos tratamentos sobre as variáveis.

4.1.2.6 Princípios genéricos de projeto

A literatura provê alguns princípios genéricos para o projeto do experimento:

Aleatoriedade: este princípio estabelece uma ordem aleatória para a alocação de indivíduos e objetos;

Obstrução: este princípio é estabelecido pela premissa de que algumas vezes um fator de probabilidade possa exercer algum efeito indesejável sobre o resultado do experimento. Caso o fator do efeito seja visível, é possível utilizar uma técnica que o bloqueie, aumentando assim a precisão dos resultados;

Balanceamento: este princípio estabelece que cada tratamento sobre as variáveis possua o mesmo número de indivíduos.

4.1.2.7 Padrão para tipo de projeto

Na maioria dos experimentos, são formuladas hipóteses e é definido um método para analisá-las estatisticamente. Existem duas abordagens para sua execução, que são:

Projeto completamente aleatório: consiste em avaliar cada instância do fator aleatoriamente entre os dois tratamentos, isto é, cada instância será aplicada em apenas um dos tratamentos, definido aleatoriamente. Sugere- se que exista um balanceamento na distribuição dos tratamentos entre os fatores;

Projeto de comparação pareado: consiste em avaliar todas as instâncias do fator com os dois tratamentos. O objetivo é aumentar a precisão do experimento através do pareamento do objeto experimentado. Esta abordagem demanda um cuidado extra sobre o efeito da ordem do experimento. Sugere-se a utilização de escolha aleatória e balanceada para a execução do fator sobre os tratamentos.

Durante o projeto, define-se a tabela de contingência que representam a distribuição do fator sobre os tratamentos, definindo se o projeto será aleatório ou pareado. Conforme o projeto escolhido, existem testes específicos para a avaliação das hipóteses.

4.1.2.8 Instrumentação

O objetivo da instrumentação é proporcionar os meios para condução do experimento e sua análise. Sugerem-se as seguintes definições de instrumentos:

Objetos: os objetos podem ser, por exemplo, documentos de especificação ou código fonte;

Guias: são especialmente úteis para apoiar os participantes no experimento e incluem, por exemplo, descrição de processos, tutoriais e checklists;

Métricas: são obtidas no experimento através da coleta de dados, normalmente através de entrevistas ou formulários preenchidos pelos participantes.

4.1.2.9 Análise da Validade

Um ponto crítico durante o experimento é a análise de sua validade. Sugere-se que esta preocupação ocorra desde o seu planejamento, prevendo algumas questões sobre a avaliação do experimento. A literatura sugere quatro tipos de validação dos resultados.

Validade interna esta validade define se o relacionamento observado entre o tratamento e o resultado é casual e não resultado de algum fator não previsto. A atenção principal é dada aos participantes durante a condução do experimento;

Validade externa: esta validade sugere a generalização dos resultados obtidos durante o experimento em práticas industriais. É avaliada a representatividade dos participantes com relação ao público alvo;

Validade de construção: a validade de construção avalia a relação de causa e efeito na qual o experimento é idealizado. A atenção desta validade se concentra em mapear a teoria, que motiva o experimento, nos indivíduos. Ao término, são observados os efeitos da experimentação;

Validade da conclusão: esta validade corresponde à capacidade de chegar a uma conclusão correta a respeito dos tratamentos e dos resultados do experimento. Para tanto, é necessário escolher os testes estatísticos, os participantes e a confiabilidade das medidas e da implementação dos tratamentos.

4.1.3 Execução

A execução compreende a etapa operacional da experimentação, onde ocorre o envolvimento direto dos participantes. Ainda que o experimento seja perfeitamente planejado e os dados coletados sejam analisados com os métodos mais apropriados, o resultado será inválido se os indivíduos não se engajarem de maneira séria com o experimento.

A execução do experimento compreende três passos:

Preparação: corresponde a escolha dos participantes e dos materiais preparados para a experimentação;

Execução: é quando os participantes executam as tarefas do experimento, de acordo com os diferentes tratamentos previstos;

Validação dos dados: corresponde a validação dos dados coletados após a execução do experimento.

4.1.4 Preparação

Existem alguns preparativos que devem ser cautelosamente analisados antes da execução do experimento. Dois pontos são avaliados nesta etapa: prover a informação aos participantes e preparar os materiais necessários ao experimento, tais como formulários e ferramentas.

Execução A execução de um experimento pode se dar de diferentes formas. Sua duração é variável e pode ocorrer em um período de tempo curto, no qual o responsável está presente em todos os detalhes da execução. Outra perspectiva é quando o tempo é longo, tornando inviável a participação do responsável em cada detalhe da execução do experimento.

Validação dos dados Após a coleta de dados, é necessário verificar se os dados são razoáveis e se foram coletados corretamente. Isso lida com aspectos específicos, como o entendimento dos participantes sobre os formulários e seu correto preenchimento, com a seriedade em que os participantes conduziram o experimento, com a remoção de dados a partir da análise, etc. É importante revisar se o experimento foi conduzido conforme o planejado e, para cada erro percebido, os dados devem ser considerados inválidos.

4.1.5 Análise e Interpretação

Após a coleta de dados experimentais gerados na fase anterior, é necessário extrair conclusões. Para se recuperar conclusões válidas, é necessário interpretar os dados experimentais.

A primeira observação sobre os dados brutos obtidos no experimento diz respeito à medida de suas escalas. As escalas determinam quais operações que podem ser executadas sobre os valores das variáveis. A literatura define os seguintes tipos de escalas:

Nominal: representam diferentes valores que não possuem interpretação numérica nem ordenação;

Ordinal: representam diferentes valores que podem ser ordenados, porém não possuem uma representação numérica;

Intervalar: representam diferentes valores que podem ser ordenados e a distância entre os números possui a mesma interpretação;

Razão: representam diferentes valores que podem ser ordenados e a distância e razão entre os mesmos pode ser interpretada.

Após a análise das medidas das escalas para as variáveis, é importante avaliar a apresentação e o processamento numérico dos dados obtidos através da estatística descritiva.Posteriormente, sugere-se a eliminação de distorções que comprometam a validade das conclusões. Por fim, é realizado o teste das hipóteses para extração das conclusões do experimento.

A estatística descritiva lida com a apresentação e o processamento numérico de um conjunto de dados. Para este fim, sugere-se representar graficamente estes dados para identificação de alguns aspectos interessantes, como os indicadores de medidas.

O objetivo da estatística descritiva é analisar a distribuição geral de um conjunto de dados. Esta etapa deve ser executada antes da validação das hipóteses para compreender a natureza dos dados e identificar os dados anormais ou os valores extremos (outliers).

Com relação ao teste das hipóteses, existem duas formas:

Paramétricos: utilizam formas fechadas, derivadas de propriedades de distribuições de freqüências conhecidas e exigem que os dados possuam normalidade e homocedasticidade;

Não-Paramétricos: devem ser usados quando os dados não atendem os requisitos de normalidade e de homocedasticidade. Esta abordagem utiliza rankings de valores observados como parâmetro ao invés dos valores propriamente ditos.

A escolha do teste das hipóteses adequado demanda a análise de dois requisitos:

Normalidade: os valores tendem a se concentrar próximos de uma média, e quanto maior a distância dessa média, menor a freqüência das observações;

Homocedasticidade: implica em uma variância constante entre o conjunto de dados que serão testados.

Depois de avaliada a normalidade e a homocedasticidade, sugerem-se dois tipos de teste de acordo com os supostos paramétricos, em um projeto com um fator e dois tratamentos:

Teste T: este teste é utilizado para comparar duas médias a partir de uma hipótese nula. Esta hipótese pressupõe que não existem diferenças significativas entre dois grupos;

Mann-Whitney: este teste serve para provar se dois grupos independentes procedem da mesma população.

4.1.6 Empacotamento

Após a análise e interpretação, sugere-se o empacotamento do experimento devido à sua necessidade de replicação. Ao executar o experimento em contextos distintos, possibilita-se a aquisição de novos conhecimentos a respeito dos conceitos estudados. Para isso, é importante documentar os diversos aspectos relacionados com o experimento, entre eles:

Comunidade: apresenta os pesquisadores e participantes relacionados com o experimento, além dos interessados que possam aproveitar os resultados obtidos;

Organização: apresenta o planejamento, projeto e demais informações referentes à preparação, diretrizes, execução e instrumentação do experimento;

Artefatos: apresenta os artefatos definidos durante a instrumentação do experimento;

Resultados: incluem a descrição detalhada dos resultados recebidos, com os dados brutos, refinados (pela eliminação dos outliers) e analisados. Os dados analisados são utilizados para testar as hipóteses e fazer as conclusões sobre o experimento.

O empacotamento do experimento proposto está documentado na Seção 4.2 onde estão presentes todas as informações necessárias para a sua replicação.

4.2 Estudo Experimental

4.2.1 Definição

Foi utilizada a abordagem GQM para a definição do estudo, estabelecendo o objetivo global, os objetivos de estudo e de medição. Por fim, serão apresentadas as questões e suas métricas.

4.2.1.1 Objetivo Global

Comparar a precisão e o esforço da rastreabilidade de artefatos através de entidades, com relação a tradicional proposta de indexação por matrizes.

4.2.1.2 Objetivo do estudo

Comparar o modelo proposto com o modelo matricial, com o propósito de caracterizar o tempo despendido no uso e os elementos recuperados por cada uma das abordagens, com foco no esforço e na precisão, sob o ponto de vista do arquiteto de software, no contexto de manutenção de um sistema de informação desenvolvido por profissionais, no domínio de uma aplicação bancária (sub-conjunto de um exemplo real).

4.2.1.3 Objetivo da medição

Em uma manutenção de um sistema de informação, caracterizar:

- Qual o esforço (medido em minutos por cada participante) necessário para relacionar os artefatos utilizando a rastreabilidade apoiada por entidades e por matrizes;
- Qual o esforço (medido em minutos por cada participante) necessário para atualizar as estruturas de rastreabilidade criadas com cada técnica;
- Qual a precisão definida através da completude e corretude dos artefatos recuperados através da rastreabilidade utilizando entidades e matrizes, com relação aos artefatos que realmente se relacionam no sistema.

4.2.1.4 Questões

1. O esforço para definição dos relacionamentos através de entidades é igual ao esforço para definição e manutenção dos relacionamentos em matrizes?
2. O esforço para atualização dos relacionamentos através de entidades é igual ao esforço para definição e manutenção dos relacionamentos em matrizes?
3. A precisão na recuperação de artefatos utilizando a entidades é igual à precisão utilizando matrizes de rastreabilidade?

4.2.1.5 Métricas

A métrica associada às Questões 1 e 2 corresponde ao esforço medido pela relação do tempo gasto em minutos por cada participante durante a definição e manutenção dos elos de rastreabilidade em cada abordagem.

A métrica relacionada à Questão 3 corresponde à precisão dos elos de rastreabilidade de cada abordagem, com relação à completude e corretude dos artefatos

recuperados. A precisão foi definida, no estudo, como a razão entre o conjunto correto de artefatos recuperados e o conjunto total de artefatos relacionados. O conjunto correto recuperado por determinada técnica compreende a intersecção entre o conjunto total de artefatos recuperados por esta técnica e o conjunto ideal, que representa o relacionamento real entre os artefatos (gabarito). O conjunto ideal foi definido por um especialista através da inspeção do sistema. O conjunto total de artefatos relacionados é a união entre o conjunto de artefatos recuperados com o conjunto ideal. O cálculo da precisão se dá por:

$$P = \frac{qtdElementos(R \cap I)}{qtdElementos(R \cup I)}$$

Onde:

R: Conjunto de artefatos recuperados utilizando determinada técnica;

I: Conjunto ideal de artefatos relacionados (gabarito);

qtdElementos(): função que recupera a quantidade de elementos do conjunto.

4.2.2 Planejamento

4.2.2.1 Seleção do Contexto

Para a condução do experimento de rastreabilidade, foi escolhido o contexto de uma empresa, já que diversos autores argumentam sobre a necessidade de conduzir experimentos em ambientes realistas (SJOBERG et al., 2002), semelhantes aos encontrados na indústria. Dentro das dimensões apresentadas, caracterizam-se:

Processo: será utilizada à abordagem In-vitro, na qual o conjunto de participantes executa o experimento em um ambiente controlado. Este experimento não se deu durante o desenvolvimento de software industrial,isto é, ele foi off-line;

Participantes: o experimento foi conduzido por analistas e desenvolvedores terceirizados por diversas empresas, trabalhando em um projeto de grande porte, em Porto Alegre - RS.

Realidade: o problema estudado é um sub-conjunto dos sistemas desenvolvidos, e corresponde a uma pequena fração de um sistema modelado e desenvolvido pela empresa. O objetivo desta escolha é a utilização de um sistema real, e que foi desenvolvido sem a intervenção do pesquisador;

Generalidade: o experimento é específico e com validade apenas no escopo do presente estudo.

4.2.2.2 Formulação das Hipóteses

No presente experimento, duas hipóteses foram definidas informalmente:

1. A rastreabilidade impõe o chamado Princípio de Circularidade: Tanto para a aquisição dos elos de rastreabilidade, quanto para sua manutenção, é necessário um esforço. Sugere-se que o esforço gasto com a definição e manutenção dos elos utilizando requisitos seja igual à definição dos elos utilizando entidades. Esta hipótese mapeia as Questões 1 e 2, definidas anteriormente.

2. Sugere-se que a rastreabilidade matricial recupera o mesmo conjunto de artefatos que o mapeamento por entidades, com relação ao conjunto ideal de artefatos relacionados. Esta hipótese mapeia a Questão 3;

Com base na definição informal, viabiliza-se a formalização das hipóteses e a definição de suas medidas para avaliação.

Hipótese Nula, H_0: O esforço envolvido com a definição dos elos de rastreabilidade utilizando matrizes é igual ao esforço envolvido utilizando entidades.

Medidas: O esforço foi avaliado pelo tempo gasto em minutos com a definição dos elos de rastreabilidade em cada abordagem, isto é, a diferença entre o tempo final e o tempo inicial de cada abordagem, onde:

δt_{mat}**:** representa a variação de tempo gasto em minutos para estabelecimento e manutenção utilizando matrizes;

δt_{ent}**:** representa a variação de tempo gasto em minutos para estabelecimento e manutenção utilizando entidades;

$H_0 : \delta t_{mat} = \delta t_{ent}$

Hipótese Alternativa, H_1: O esforço envolvido com a definição e manutenção dos elos de rastreabilidade utilizando entidades é maior do que o esforço envolvido utilizando matrizes.

$$H_1 : \delta t_{ent}\, \delta t_{mat}$$

Hipótese Alternativa, H_2: O esforço envolvido com a definição e manutenção dos elos de rastreabilidade utilizando matrizes é maior do que o esforço envolvido utilizando entidades.

$$H_2 : \delta t_{mat} > \delta t_{ent}$$

Hipótese Nula, H_0: A precisão na definição dos elos de rastreabilidade, relacionados em matrizes é igual a dos elos relacionados a entidades. A precisão é avaliada pela relação entre o conjunto correto de artefatos recuperados e o conjunto total, onde:

P_{mat}**:** Precisão associada às relações matriciais;

P_{ent}**:** Precisão associada às relações entre entidades;

$$H_0 : P_{mat} = P_{ent}$$

Hipótese Alternativa, H_1: A precisão na aquisição dos elos de rastreabilidade, relacionados em matrizes é maior do que os elos relacionados a entidades.

$$H_1 : P_{mat} > P_{ent}$$

Hipótese Alternativa, H_2: A precisão na aquisição dos elos de rastreabilidade, relacionados a entidades, é maior do que os relacionados em matrizes.

$$H_2 : P_{ent} > P_{mat}$$

4.2.2.3 Seleção das variáveis

Variáveis Independentes Assumiram-se como variáveis independentes:

- Técnica para definição dos elos de rastreabilidade;

Variáveis Dependentes Assumiram-se como variáveis dependentes:

- Esforço para a definição e manutenção dos elos de rastreabilidade;
- Precisão através do número de artefatos recuperados utilizando determinada técnica com relação ao conjunto total de artefatos.
- Experiência da equipe de manutenção

4.2.2.4 Seleção dos Indivíduos

Conforme já apresentado, a população definida para o experimento é formada por profissionais experientes, no total de dez profissionais, sendo:

- Quatro analistas;
- Quatro programadores;
- Dois projetistas;

Não foi utilizada uma amostragem probabilística para seleção dos indivíduos, mas uma amostragem não probabilística:

Amostragem por conveniência: foram escolhidas as pessoas mais convenientes para o experimento;

Amostragem por quota: esta abordagem implica na escolha de indivíduos de diferentes populações, no caso, analistas, programadores e projetistas.

Pressupõe-se que a experiência com a modelagem de sistemas e com o processo de desenvolvimento entre estas duas populações seja diferente.

4.2.2.5 Projeto do experimento

Dentre os princípios genéricos para o projeto do experimento, caracterizam-se:

Aleatoriedade: a aleatoriedade é utilizada para definir quais participantes devem executar cada abordagem de rastreabilidade (matrizes ou entidades);

Obstrução: durante a experimentação, muitos dos participantes não possuíam o mesmo nível de experiência acadêmica e profissional. Para minimizar o efeito da experiência sobre o experimento, os indivíduos foram selecionados utilizando o critério de quota e conveniência;

Balanceamento: este princípio foi utilizado no experimento para que cada proposta de rastreabilidade seja executada pela mesma quantidade de participantes.

Tabela 4.1: Tabela de contingência.

Participante	μ_{ent}	μ_{mat}
Analista 1	X	
Analista 2	X	
Analista 3		X
Analista 4		X
Programador 1		X
Programador 2	X	
Programador 3	X	
Programador 4		X
Projetista 1	X	
Projetista 2		X

4.2.2.6 Padrão para tipo de projeto

Para cada hipótese, foram utilizadas as seguintes notações:

μ_{mat}: Rastreabilidade utilizando matrizes;

μ_{ent}: Rastreabilidade utilizando entidades;

O tipo de projeto apresentado procura investigar se μ_{ent} possui o mesmo esforço e precisão que μ_{mat}. Para este fim, foi utilizada uma abordagem chamada um fator com dois tratamentos. O fator, neste experimento, consiste na técnica que será utilizada e os tratamentos consistem nas relações a entidades e a matrizes.

Com relação às abordagens apresentadas no referencial teórico, é possível escolher tanto o projeto pareado quanto o aleatório, porém a complexidade associada ao primeiro é maior do que o segundo. Caso se opte por um projeto pareado no contexto deste experimento, é interessante considerar um novo tratamento que represente a experiência dos participantes na execução dos demais tratamentos. Adicionalmente, o ganho com relação ao grau de liberdade entre as duas abordagens não apresenta significante diferença. Por motivos de projeto e complexidade, o projeto pareado descrito não será utilizado, mas sim o projeto completamente aleatório, onde cada participante executará apenas uma abordagem definida aleatoriamente.

A Tabela 4.1 é chamada de tabela de contingência e representa a distribuição do fator sobre os dois tratamentos.

A Tabela 4.1 representa a forma na qual o experimento foi executado. A ordem foi definida de forma aleatória e balanceada entre os fatores. Para a realização dos testes das hipóteses, em um contexto de um fator e dois tratamentos aleatórios, a literatura sugere o teste de significância chamado Teste T para duas amostras independentes, caso seja realizado um teste paramétrico, ou Mann-Whitney, caso o teste seja não-paramétrico. A definição do teste a ser aplicado ocorre após a análise da normalidade (através do teste de Shapiro-Wilk) e variância dos dados obtidos pela execução do experimento (Teste de Levene).

4.2.2.7 Instrumentação

No presente experimento, foram utilizados:

Objetos: a descrição dos casos de uso de parte de um sistema, a partir dos quais devem ser definidos os elos de rastreabilidade, as entidades e suas relações ao caso de uso e ao código-fonte. Para o estabelecimento dos elos de rastreabilidade, foi fornecido o apoio ferramental para cada abordagem: o protótipo ReMan para identificação das entidades, e matrizes de rastreabilidade, utilizando a ferramenta *Enterprise Architect* (EA), para indexação dos requisitos. Foi prevista, para as duas abordagens, uma normalização dos esforços visando não causar distorções nos resultados pela utilização de ferramentas distintas;

Guias: foi fornecido um treinamento para cada grupo de participantes, apresentando:

- as duas abordagens de rastreabilidade, o contexto do experimento e a motivação da equipe.
- Métricas: Os dados foram recuperados através de formulários preenchidos pelos participantes, com base na ferramenta ReMan e na matriz definida no EA.

4.2.2.8 Análise da Validade

Validade interna Foram avaliados alguns critérios, tais como:

Histórico: a data de aplicação do experimento (25/04/2008) foi criteriosamente definida, evitando períodos nos quais os participantes poderiam sofrer influências externas;

Seleção dos grupos: foi utilizada uma abordagem para nivelar o conhecimento dos participantes através de um treinamento sobre as técnicas. A execução das atividades foi individual;

Difusão: Deve-se evitar a interação entre os participantes. Assim, houve um policiamento durante a experimentação para evitar este tipo de influência.

4.2.2.9 Validade externa

Para esta avaliação, foi adotada a interação da seleção, ou seja, os participantes que foram selecionados possuem um perfil apto aos tratamentos do experimento, apresentando, em sua maioria, conhecimento prévio sobre processo de desenvolvimento de software e modelagem de sistemas, além de experiência na indústria.

4.2.2.10 Validade de construção

Durante o experimento, foram avaliados:

Inadequada explicação pré-operacional: consiste na explicação operacional do experimento, visando maturar a forma na qual serão definidos os elos de rastreabilidade e a extração dos dados;

Adivinhação de hipóteses: devido ao fato dos participantes serem humanos, é possível sua interação com o experimento, sugerindo novas hipóteses e exercitando a criatividade. É importante manter o foco no estudo planejado;

Expectativas do condutor do experimento ao se conduzir um experimento, o responsável pode exercer influências sobre as variáveis envolvidas e sobre o material elaborado. Durante a presente proposta, todo o material utilizado foi previamente avaliado por outro responsável.

4.2.2.11 Validade da conclusão

Foram avaliadas as seguintes perspectivas:

Manipulação dos dados: como os dados resultantes do experimento foram manipulados pelo pesquisador, é possível que os mesmos sofram algumas variações, tal como o coeficiente de significância para validação dos resultados;

Confiabilidade das medidas: esta perspectiva sugere que medidas subjetivas possam ser influenciadas pelo pesquisador. Na proposta as medidas foram objetivamente definidas, não dependendo do critério humano;

Confiabilidade na implementação dos tratamentos: consiste no risco em que diferentes participantes possam implementar de forma distinta os processos estabelecidos pelo experimento. Este risco não foi evitado no estudo, visto que não se pode interferir no caráter subjetivo de relacionamento entre artefatos. Possivelmente, diferentes participantes definirão elos de rastreabilidade distintos;

Configurações do ambiente do experimento: consiste nas interferências externas do ambiente que podem influenciar os resultados durante a execução do experimento. O experimento foi executado após o horário de trabalho, onde a interação externa era reduzida.

Heterogeneidade aleatória dos participantes: a escolha de diferentes participantes com diferentes experiências pode exercer um risco na variação dos resultados.

4.3 Execução

4.3.1 Preparação

Para preparar a execução do experimento, atentou-se para:

Consenso com o experimento: de acordo com Wohlin *et. al.*(2000), se os participantes não concordam com os objetivos da pesquisa ou não têm conhecimento sobre o experimento, corre-se o risco de que sua participação não ocorra em encontro aos objetivos. Durante a experimentação, a preparação dos participantes forneceu o embasamento necessário sobre o experimento, clarificando quais são os objetivos e metas almejadas;

Resultados sensitivos: é possível que o resultado obtido pelo experimento seja influenciado por questões pessoais, como a sensibilidade dos participantes por estarem sendo avaliados. Foi adotada uma postura de anonimato dos participantes em toda a descrição da experimentação.

Tabela 4.2: Escalas das variáveis

Variáveis	**Nome**	**Escala**
Dependentes	Esforço (tempo)	Razão
	Precisão	Razão
Independentes	Técnica de rastreabilidade	Nominal

Com relação à instrumentação, todas as variáveis e recursos foram criteriosamente estabelecidos antes da execução do experimento. Foi apresentado um treinamento específico para cada grupo (rastreabilidade utilizando entidades e matrizes), contextualizando os objetivos, a técnica, a motivação e o procedimento técnico para condução do experimento. Adicionalmente, para apoiar os participantes em suas tarefas foi fornecido um tutorial sobre como executar as atividades para a rastreabilidade utilizando entidades e matrizes. Os próprios participantes foram responsáveis pela coleta dos dados, utilizando o EA para relacionar artefatos em matrizes, e o protótipo ReMan para indexar artefatos a entidades. Outro critério a ser considerado é a questão do anonimato, onde os nomes dos participantes não foram registrados.

4.3.1.1 Execução

A presente proposta estrutura o experimento em um curto período de tempo, na qual o pesquisador ficou envolvido em todos os detalhes da execução. Durante a execução, foi definida a coleta dos dados e o ambiente no qual ocorrereu o experimento. A coleta de dados é de responsabilidade dos participantes, através do uso dos softwares supracitados, e dos formulários para avaliação quantitativa apresentados nos Anexos C e D. Esta opção é justificada, pois o pesquisador não pode se envolver na definição dos dados resultados do experimento.

Durante a execução do experimento, o responsável esteve à disposição dos participantes para o esclarecimento das dúvidas que surgiam ao longo do processo.

4.4 Análise e Interpretação

A primeira análise apresentada diz respeito à classificação das escalas das variáveis definidas no experimento, apresentada na tabela 4.2. Com esta classificação, é possível determinar as operações que podem ser aplicadas sobre as variáveis.

4.4.1 Análise Tabular e Gráfica

O experimento obteve como resultado os dados apresentados na Tabela 4.3.

Os cálculos estatísticos e gráficos foram produzidos com a utilização do software estatístico R (DALGAARD et al., 2008), (DALGAARD, 2002), (MAINDONALD; BRAUN, 2003) e (VERZANI, 2005). A figura 4.1 representa o gráfico de barras com o esforço de cada participante para a definição dos elos de rastreabilidade. Já a figura 4.2 representa o gráfico de barras com o esforço para a alteração dos mesmos.

A figura 4.3 representa o gráfico de barras com a precisão dos artefatos recuperados pelos participantes.

Tabela 4.3: Tabulação dos valores brutos obtidos após a execução do experimento.

Abordagem	**Participante**	**Esforço Estabelecimento** (minutos)	**Esforço Manutenção** (minutos)	**Precisão(%)**
Entidades	E1	13	0,5	1
Entidades	E2	8	1,5	1
Entidades	E3	10	0,5	1
Entidades	E4	14	1,5	1
Entidades	E5	15	1	1
Matrizes	M1	34	5	0,8
Matrizes	M2	23	4	1
Matrizes	M3	38	7	0,8
Matrizes	M4	29	5	1
Matrizes	M5	41	9	0,75

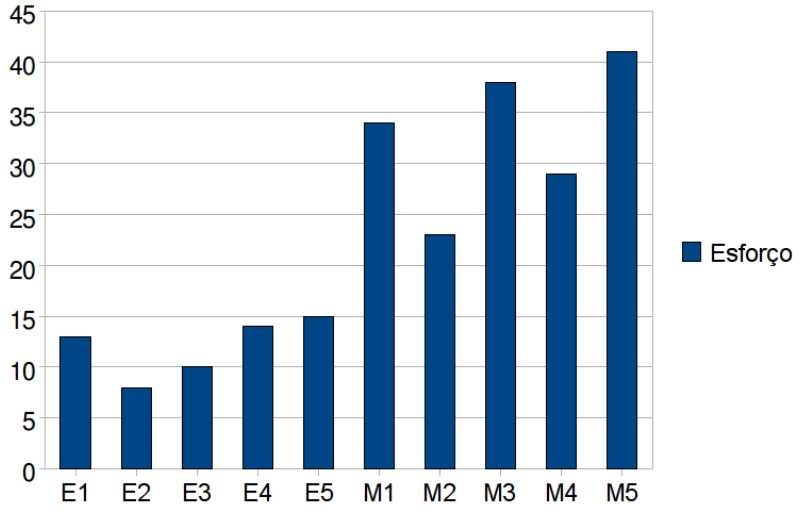


Figura 4.1: Gráfico de barras relativo ao esforço para definição dos elos de rastreabilidade.

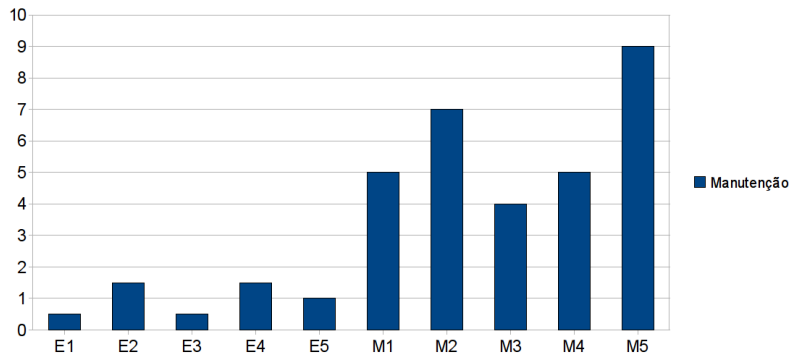


Figura 4.2: Gráfico de barras relativo ao esforço para manutenção dos elos de rastreabilidade.

4.4.2 Estatística Descritiva

Conforme apresentado, as variáveis dependentes estão caracterizadas na escala razão, o que permite o cálculo da normalidade e homocedasticidade, necessária para

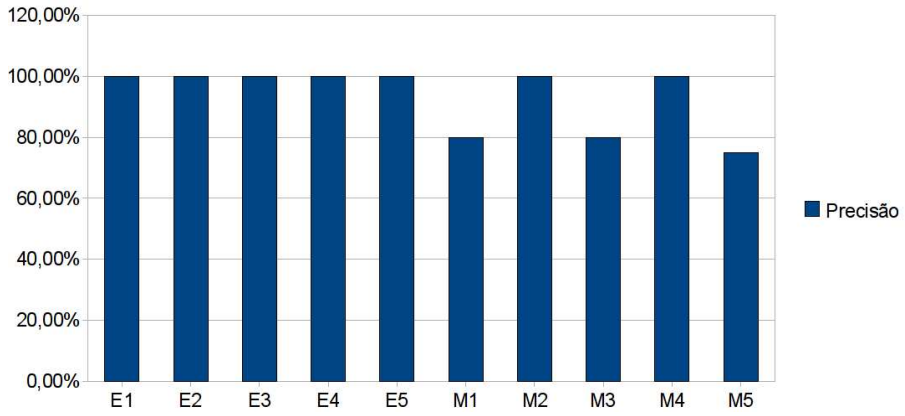


Figura 4.3: Gráfico de barras relativo à precisão na recuperação dos elos de rastreabilidade.

Tabela 4.4: Média e desvio padrão para a variável esforço na criação.

Abordagem	**Média**	**Desvio Padrão**
Entidades	12	2,915476
Matrizes	33	7,17635

definir o tipo de teste das hipóteses (paramétrico ou não-paramétrico). Conforme definido no projeto do experimento, o padrão para tipo de teste previsto é o Teste T para duas amostras independentes, caso o teste empregado seja paramétrico, ou Mann-Whitney, caso seja não paramétrico.

A avaliação será executada para as hipóteses: esforço para a criação da estrutura, sua manutenção e precisão. Para cada hipótese, os dados serão caracterizados, visualizando tendências centrais e dispersões. Posteriormente, sugere-se a eliminação de dados anormais ou incertos, que distorcem a integridade da conclusão, através da redução do intervalo de dados. Por último, será realizado o teste das hipóteses que compreende a avaliação estatística dos dados até certo nível de significância. O nível de significância adotado (p-value) para todos os testes é de 5%. O p-value compreende o menor nível de significância com que se pode rejeitar a hipótese nula.

4.4.2.1 Primeira Hipótese: Esforço para estabelecimento da estrutura de rastreabilidade

Uma análise inicial da distribuição é eficiente para avaliar o comportamento das amostras. O gráfico de dispersão boxplot, apresentado na figura 4.4, é utilizado para identificação dos outliers.

Conforme apresentado na figura 4.4, a variável esforço na criação na possui outliers moderados. A Tabela 4.4 apresenta algumas medidas estatísticas para a variável esforço na criação, agrupadas por abordagem.

Por critério de projeto, estabeleceu-se que os valores extremos que não atingirem a média com mais de dois desvios padrões, serão removidos da amostra. Neste contexto, nenhum participante foi eliminado do conjunto de dados.

. A próxima etapa consiste em identificar se os dados seguem uma distribuição normal. Para se avaliar a normalidade, é definida uma hipótese nula e uma hipótese alternativa, conforme:

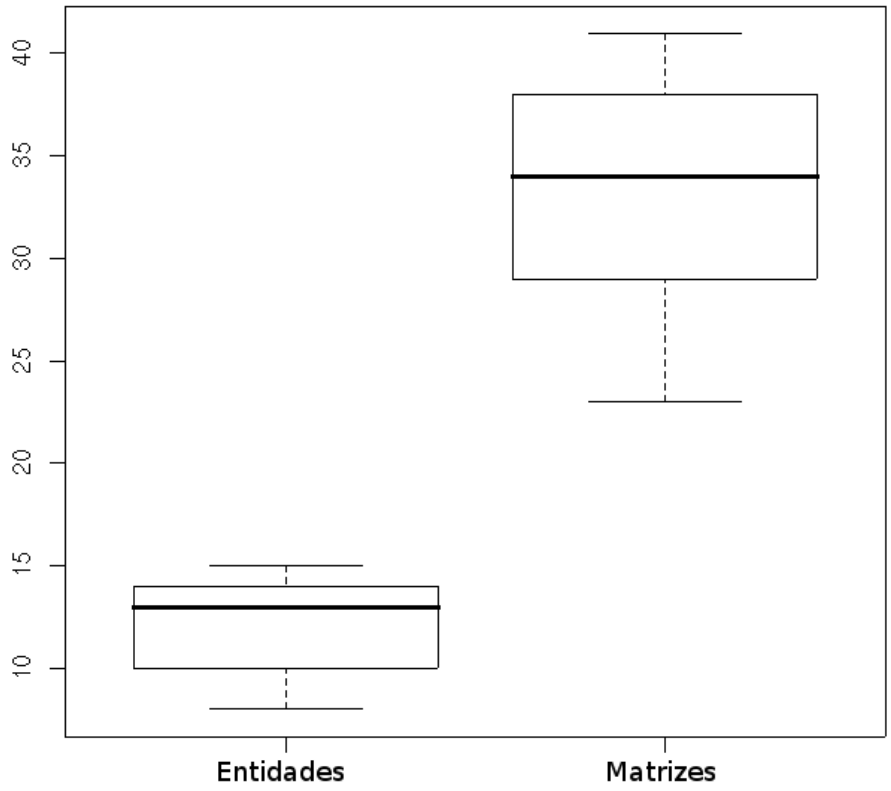


Figura 4.4: Gráfico de dispersão para a variável esforço na criação.

Tabela 4.5: Teste de normalidade Shapiro-Wilk para a variável esforço na criação.

Variável	**Abordagem**	**Estatística**	**Grau de Liberdade**	**Significância**
Esforço	Entidade	0,9283	5	0,5846
	Matriz	0,97	5	0,8752

H_0 : a distribuição é normal;

H_1 : a distribuição não é normal.

Existem duas formas para se avaliar a distribuição normal dos dados, que compreendem o Teste de Kolmogorov-Smirnov e o Teste de Shapiro-Wilk. O primeiro é utilizado para identificar a normalidade em variáveis com pelo menos 30 valores e o segundo em variáveis com menos de 50 valores. A Tabela 4.5 apresenta os testes de normalidades para a amostra utilizando o Teste de Shapiro-Wilk.

Com base na Tabela 4.5, observa-se que a significância dos dados do teste de Shapiro-Wilk é superior, em ambas as amostras (entidades e matrizes), ao nível de significância definido (0,05 ou 5%). Com esta informação, não há indícios para rejeitar a hipótese nula sobre a distribuição da normalidade, conseguindo assim o primeiro requisito para utilização de teste paramétrico para duas amostras independentes.

O segundo requisito requer a análise da homocedasticidade, tornando necessário analisar a variância das duas amostras. Com este objetivo, definem-se duas hipóteses:

Tabela 4.6: Teste de Levene para igualdade das variâncias sobre a variável esforço.

Variável	**Variâncias Iguais**	**Significância**
Esforço	Assumido	0,1483
	Não assumindo	0,0000

Tabela 4.7: Teste T para duas amostras independentes para a variável esforço, agrupadas por entidades e matrizes.

Grau Sign.	**Variável**	**Variâncias Iguais**	**T**	**Liberdade (bicaudal)**
Esforço	Assumindo	-6,0622	8	0,000302
	Não assumindo	-6,0622	5,285	0,001449

H_0**:** As variâncias são iguais;

H_1**:** As variâncias não são iguais.

O teste da hipótese acima é realizado com a significância obtida diretamente através do Teste de Levene. O Teste de Levene é usado para testar se k amostras têm a mesma variância. A Tabela 4.6 apresenta os resultados obtidos para este teste.

Com base na tabela 4.6, verifica-se que o nível de significância para variâncias iguais (0,1483) é superior ao nível de significância definido (0,05 ou 5%). Com esta informação, não se consegue rejeitar a hipótese nula para variâncias, conseguindo o segundo requisito para utilização do teste paramétrico.

Conforme definido no planejamento do projeto do experimento, o teste indicado para avaliação das hipóteses é o Teste T para duas amostras independentes. Consegue-se validar sua utilização por ser um teste paramétrico, no qual seus requisitos foram explicitamente atendidos.

Com base na declaração das hipóteses, tem-se:

$$H_0 : \mu_{mat} = \mu_{ent} H_1 : \mu_{ent}\ \mu_{mat} H_2 : \mu_{mat}\ \mu_{ent}$$

O critério para rejeição de H_0 em favor de H_1 é:

H_1**:** ($\mu_{ent}\ \mu_{mat}$): rejeita-se H_0 se $t_0 > t_{\alpha,n+m-2}$, onde:

t_0: é o valor t obtido através da aplicação do Teste T.

$t\alpha, n+m-2$ é o valor obtido pela tabela de Distribuição de T, onde $n+m\check{}2$ representa o grau de liberdade, sendo n o número de participantes de uma abordagem e m o número de participantes da outra abordagem. O grau de liberdade da amostra é 10.

O Teste T para duas amostras independentes foi aplicado neste contexto e o resultado está apresentado na Tabela 4.7.

Com base na Tabela 4.7, obtém-se o valor de t_0 $(-6,0622)$ e, com base no Anexo B, obtém-se o valor de $t_{\alpha,n+m-2}(=2,31)$. Como $t_0 < t_{\alpha,n+m-2}$, então não se consegue rejeitar a hipótese nula a um nível de significância de 5% em favor de H_1: $\mu_{ent}\ \mu_{mat}$.

O critério para rejeição de H_0 em favor de H_2 é H_2: ($\mu_{mat}\ \mu_{ent}$) rejeita-se H_0 se $t_0\, t\alpha, n+m-2$.

Tabela 4.8: Teste T para duas amostras independentes para a variável esforço, agrupada por entidades e matrizes.

Grau Sign.	Variável	Variâncias Iguais	T	Liberdade (bicaudal)
Esforço	Assumindo	6,0622	8	0,000302
	Não assumindo	6,0622	5,282	0,001449

O Teste T foi aplicado neste contexto e o resultado está apresentado na tabela 4.8

Com base na tabela 4.8, obtém-se o valor de t_0 $(6,0622)$ e o valor de $t_{\alpha,n+m-2}$ $(2,31)$. Como $t_0 t_{\alpha,n+m-2}$, consegue-se rejeitar a hipótese nula a um nível de significância de 5% em favor de H_2: $\mu_{mat}\,\mu_{ent}$.

Pelas análises apresentadas, pode-se concluir que existe diferença em relação ao esforço na definição dos elos de rastreabilidade utilizando entidades e utilizando matrizes. O esforço necessário para indexação através de matrizes é maior do que através de entidades.

4.4.3 Segunda Hipótese: Esforço para manutenção da estrutura de rastreabilidade

Utilizando a mesma forma de análise de distribuição para avaliar o comportamento das amostras sobre o esforço na manutenção, obtém-se o gráfico de dispersão boxplot, apresentado na figura 4.5, para identificar o outliers.

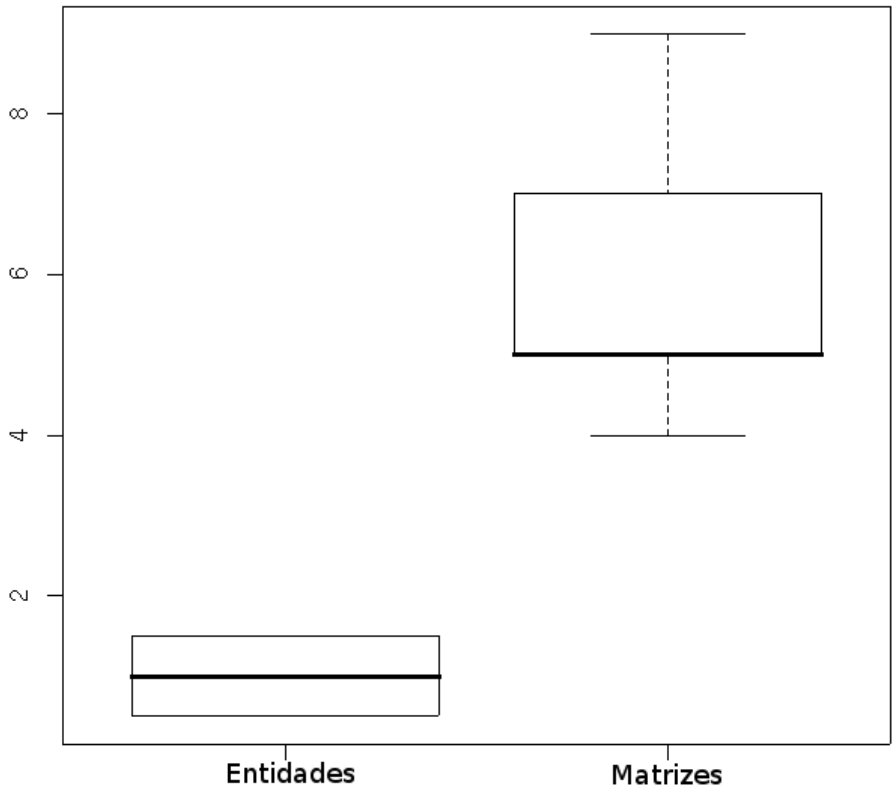


Figura 4.5: Gráfico de dispersão para a variável esforço na manutenção.

Conforme apresentado na figura 4.5, a variável esforço na manutenção possui

Tabela 4.9: Média e desvio padrão para a variável esforço na manutenção.

Abordagem	**Média**	**Desvio Padrão**
Entidades	1	0,5
Matrizes	6	2

Tabela 4.10: Teste de normalidade Shapiro-Wilk para a variável esforço na manutenção

Variável	**Abordagem**	**Estatística**	**Grau de Liberdade**	**Significância**
Esforço	Entidade	0,8208	5	0,1185
	Matriz	0,8949	4	0,4064

outliers moderados, exceto o participante M5, que será retirado da amostra para evitar distorções, pelos critérios para eliminação de outliers estabelecidos anteriormente. A Tabela 4.9 apresenta algumas medidas estatísticas para a variável esforço na manutenção, agrupadas por abordagem.

É necessário, avaliar, novamente a normalidade, através de uma hipótese nula e uma hipótese alternativa, conforme realizado anteriormente. A Tabela 4.10 apresenta os testes de normalidades para a amostra utilizando o Teste de Shapiro-Wilk.

Com base na Tabela 4.10, observa-se que a significância dos dados do teste de Shapiro-Wilk é superior, em ambas as amostras (entidades e matrizes), ao nível de significância definido (0,05 ou 5%). Novamente, não há indícios para rejeitar a hipótese nula. Utilizando as mesmas hipóteses para analisar a variância das amostras, a Tabela 4.11 apresenta os resultados obtidos para o teste de Levene.

Com base na tabela 4.11, verifica-se que o nível de significância para variâncias iguais (0,646) é superior ao nível de significância definido (0,05 ou 5%). Portanto, não se consegue rejeitar a hipótese nula para variâncias. Pode-se prosseguir com a utilização do teste paramétrico (Test T). Utilizando-se as mesmas hipóteses e critério para rejeição de H_0 em favor de H_1, o resultado do teste T está apresentado na Tabela 4.12.

Com base na Tabela 4.12, obtém-se o valor de t_0 $(-5,68765)$ e, com base no Anexo B, obtém-se o valor de $t_{\alpha,n+m-2}(=2,37)$. Como $t_0 < t_{\alpha,n+m-2}$, então não se consegue rejeitar a hipótese nula a um nível de significância de 5% em favor de H_1: $\mu_{ent}\ \mu_{mat}$.

O critério para rejeição de H_0 em favor de H_2 é H_2: $(\mu_{mat}\ \mu_{ent})$ rejeita-se H_0 se $t_0\, t\, \alpha, n+m-2$.

O Teste T foi aplicado neste contexto e o resultado está apresentado na tabela 4.8

Tabela 4.11: Teste de Levene para igualdade das variâncias sobre a variável esforço na manutenção.

Variável	**Variâncias Iguais**	**Significância**
Esforço	Assumido	0,646
	Não assumindo	0,000

Tabela 4.12: Teste T para duas amostras independentes para a variável manutencão, agrupadas por entidades e matrizes.

Grau Sign.	Variável	Variâncias Iguais	T	Liberdade (bicaudal)
Esforço	Assumindo	-5,68765	7	0,0002134
	Não assumindo	-6,3651	3,761	0,003842

Tabela 4.13: Teste T para duas amostras independentes para a variável manutenção, agrupada por entidades e matrizes.

Grau Sign.	Variável	Variâncias Iguais	T	Liberdade (bicaudal)
Esforço	Assumindo	6,9903	7	0,0002134
	Não assumido	6,3651	3,761	0,003842

Com base na tabela 4.13, obtém-se o valor de t_0 $(6,9903)$ e o valor de $t_{\alpha,n+m-2}$ $(2,37)$. Como $t_0 t_{\alpha,n+m-2}$, consegue-se rejeitar a hipótese nula a um nível de significância de 5% em favor de H_2: $\mu_{mat}\, \mu_{ent}$.

Pelas análises apresentadas, pode-se concluir que existe diferença em relação ao esforço na manutenção dos elos de rastreabilidade utilizando entidades e utilizando matrizes. O esforço necessário para manutenção através de matrizes é maior do que através de entidades.

4.4.4 Terceira Hipótese: Precisão

Da mesma forma que na análise da primeira e segunda hipóteses, utiliza-se o gráfico de dispersão boxplot para identificação dos outliers, apresentado na figura 4.6

De acordo com a figura 4.6, a variável precisão possui um outlier. Para escolha de quais participantes serão excluídos da amostra, optou-se pela identificação numérica. A Tabela 4.14 apresenta algumas medidas estatísticas para a variável precisão, agrupadas por abordagem.

Por critério de projeto, estabeleceu-se que os valores extremos que não atingirem a média com mais de dois desvios padrões, serão removidos da amostra. Neste contexto, nenhum participante foi eliminado do conjunto de dados.

A próxima etapa consiste em identificar se os dados seguem uma distribuição normal. Definem-se assim as hipóteses:

H_0: a distribuição é normal;

H_1: a distribuição não é normal.

Com base na tabela 4.15, observa-se que a significância dos dados do teste de Shapiro-Wilk é inferior, na abordagem matricial, ao nível de significância definido

Tabela 4.14: Média e desvio padrão para a variável precisão.

Abordagem	Média	Desvio Padrão
Entidades	1	0
Matrizes	0,87	0,1204159

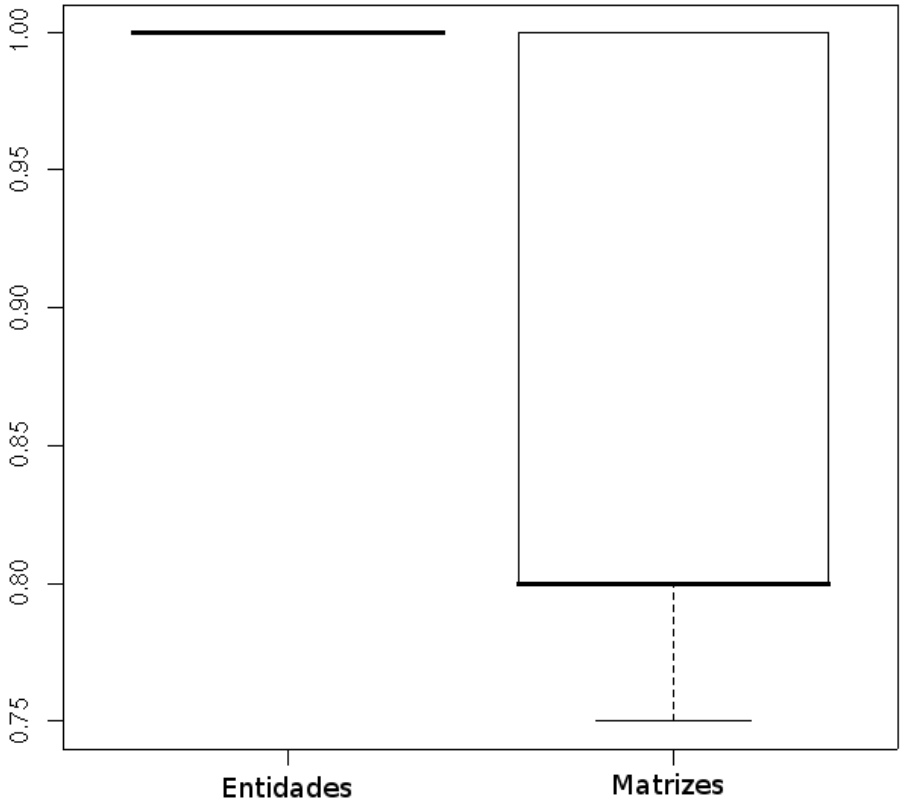


Figura 4.6: Gráfico de dispersão para a variável precisão.

Tabela 4.15: Teste de normalidade Shapiro-Wilk para a variável precisão.

Variável	**Abordagem**	**Estatística**	**Grau de Liberdade**	**Significância**
Precisão	Matrizes	0,7928	6	0,07067

(0,05 ou 5%). Sendo assim, há indícios para rejeitar a hipótese nula e, conseqüentemente, não se pode aplicar um teste paramétrico para avaliação das hipóteses.

Optou-se por aplicar o teste Mann-Whitney, para duas amostras independentes, por se tratar de uma alternativa não-paramétrica para o Teste T. O teste de Mann-Whitney para duas amostras independentes é utilizado para comprovar se as diferenças entre as médias observadas nos dois grupos independentes são estatisticamente significativas. Com base na declaração das hipóteses, sugere-se:

H_0: Não há diferença entre as médias ($\mu_{mat} = \mu_{ent}$)

H_1: Há diferença entre as médias ($\mu_{mat} \neq \mu_{ent}$)

O resultado do teste Mann-Whitney foi aplicado sobre as amostras e está apresentado na Tabela 4.16.

Tabela 4.16: Teste não paramétrico de Mann-Whitney para a variável precisão.

Variável	**U de Mann-Whitney**	**W de Wilcoxon**	**Z**	**Sig. Assimpt.**
Precisão	0,000	21,000	-3,857	0,004

Tabela 4.17: Estatística descritiva para a variável precisão.

Abordagem	**Média**
Matrizes	0,87
Entidades	1,000

Como o grau de significação associado (Sig. Assimpt.) é 0,004 e é menor que a significância assumida de 0,005, deve-se rejeitar H_0. Frente aos resultados apresentados para a variável precisão, existe diferença de média entre a rastreabilidade apoiada por matrizes e por entidades.

Pela análise estatística dos dados, consegue-se recuperar duas informações:

- A distribuição da precisão não é normal, o que implica na execução de testes não paramétricos;
- Utilizando o teste Mann-Whitney, conseguiu-se verificar que existem diferenças entre as médias das duas amostras μ_{mat} e μ_{ent}.

Utilizando o teste de Mann-Whitney, conseguiu-se apenas rejeitar a hipótese nula, porém não foi possível avaliar as hipóteses alternativas, pois não é possível extrair relações de "maior do que" com o teste aplicado. Porém, sugere-se comparar a análise descritiva das médias da amostra conforme a Tabela 4.17.

Comparando as médias apresentadas, e com base nas médias das duas abordagens, observa-se que a precisão na aquisição dos elos de rastreabilidade por entidades é maior do que por matrizes.

4.5 Avaliação Qualitativa

Conforme verificado no referencial teórico deste capítulo, um experimento é responsável apenas por uma avaliação quantitativa. Para a análise qualitativa das duas abordagens de rastreabilidade, foi desenvolvida uma pesquisa de opinião integrada ao experimento. Ao término da execução, cada participante respondeu um questionário composto de 6 questões (Q1 a Q6), conforme Anexo E. Como critério de resposta, foi utilizado a escala Likert, de 5 pontos para o grau de satisfação.

Os objetivos almejados com a pesquisa foram apresentar a opinião dos participantes quanto à usabilidade, utilidade e esforço necessários em cada abordagem. Os dados brutos obtidos como resultados estão apresentados na Tabela 4.18.

Em uma primeira análise, é explicitada a média aritmética das questões em cada uma das abordagens, apresentada na Tabela 4.19.

É possível observar que a média para todas as questões em uma abordagem por entidades é maior do que por matrizes. Pela utilização de uma escala de Likert variando de 1 a 5, com grau de satisfação progressiva, verificou-se que a avaliação qualitativa da proposta de rastreabilidade por entidades foi superior, em termos gerais, à rastreabilidade utilizando-se matrizes.

Em geral, todos os participantes consideraram o modelo proposto uma alternativa viável ao rastreamento de requisitos, por considerarem que este não demanda muito esforço de interpretação na identificação de relações entre artefatos em diversos níveis de abstração. Além disso, consideraram que o processo de rastreamento

Tabela 4.18: Resultados da avaliação qualitativa.

Abordagem	**Participante**	**Q1**	**Q2**	**Q3**	**Q4**	**Q5**	**Q6**
Entidades	E1	5	5	5	5	5	3
Entidades	E2	4	5	5	5	5	4
Entidades	E3	5	5	5	5	5	4
Entidades	E4	4	5	5	5	5	3
Entidades	E5	5	5	5	5	5	2
Matrizes	M1	3	3	1	2	4	2
Matrizes	M2	4	4	2	1	5	2
Matrizes	M3	3	3	2	2	4	3
Matrizes	M4	4	4	2	2	5	2
Matrizes	M5	3	2	1	1	4	2

Tabela 4.19: Média da satisfação das questões sobre as abordagens.

Abordagem	**Q1**	**Q2**	**Q3**	**Q4**	**Q5**	**Q6**
Entidades	4,6	5	5	5	5	3,2
Matrizes	3,4	3,2	1,6	1,6	4,4	2,2

auxiliou na revisão dos requisitos, por automatizar o processo de busca dos mesmos em documentos diversos, além de facilitar a localização de inconsistências e conflitos. No entanto, o protótipo foi considerado de utilização difícil, pois não realizava o rastreamento automático do texto para as entidades que já foram identificadas pelas mesmas palavras (o que reduziria significativamente o tempo para finalizar o rastreamento). Porém, por se tratar de um problema de interface do protótipo, e não do modelo de rastreabilidade propriamente dito, ao ser comparado com o modelo matricial, foi considerado "mais barato (em termos de tempo e esforço), mais preciso e mais fácil".

Não obstante, a avaliação foi realizada sobre uma pequena parte de um sistema, com pequenos conjuntos de casos de uso e código fonte. Assim, não há certeza quanto à escalabilidade do modelo. Porém, os envolvidos na avaliação consideraram a proposta "significativamente mais viável", e acreditam que o modelo seja aplicável em projetos de grande porte, desde que exista suporte ferramental adequado, e maior automação nas atividades de identificação de entidades.

4.5.1 Considerações

Este capítulo apresentou uma avaliação quantitativa e qualitativa da proposta de rastreabilidade por entidades, utilizando um estudo experimental e uma pesquisa de opinião. O objetivo foi verificar a aplicabilidade e a relevância da proposta quando comparada a tradicional forma de relacionamento entre artefatos através de matrizes. É importante clarificar que os resultados obtidos através da experimentação não são generalizáveis para todo o processo de desenvolvimento de software. O contexto escolhido para o experimento é relativo à rastreabilidade de código fonte e requisitos associados à entidades, durante a manutenção do sistema em um sub-domínio de uma aplicação bancária.

Durante a indexação matricial, o esforço para definição dos elos de rastreabilidade correspondeu percorrer a descrição dos casos de uso, e identificar o código fonte associado a cada passo do caso de uso. Durante a definição dos elos a partir de entidades, o esforço compreendeu a identificação de entidades do sistema, através do modelo de entidades propostos, relacionando cada porção do código e dos requisitos à sua respectiva entidade.

Os dados relativos ao esforço obtiveram uma distribuição normal com uma diferença estatisticamente insignificante entre suas variâncias (homocedasticidade). Diante desta circunstância, foi executado o Teste T (paramétrico) e se verificou a validade da hipótese alternativa que estabelece que o esforço relativo à indexação e manutenção dos casos de uso através das entidades identificadas é menor do que a indexação matricial, diretamente entre casos de uso e código fonte.

A precisão apresentou um fenômeno distinto do esforço, onde a amostra não apresentou uma distribuição normal, impossibilitando a utilização do Teste T. O teste Mann-Whitney foi a escolha definida para avaliação das hipóteses. Este teste apenas informa se dois grupos independentes procedem da mesma população. O resultado foi que existem diferenças entre as médias das duas amostras de entidades e matrizes. Neste contexto, não foi possível avaliar estatisticamente as hipóteses alternativas e se optou por comparar a análise descritiva das médias. Após comparar as médias, verificou-se que a precisão na recuperação da indexação por entidades é maior que por matrizes.

Durante a análise da precisão, os resultados indicaram um fenômeno bastante positivo para o rastreamento através de entidades em comparação com as matrizes. A razão deste fenômeno pode ser que, pela redução da granularidade dos elementos que dão origem às entidades, obtiveram-se elos mais apurados.

Este fenômeno é explicado porque um caso de uso pode englobar mais de uma classe, por exemplo. Ao se recuperar as porções de código fonte (no caso, métodos e classes), relacionadas em matrizes, corre-se o risco de recuperar outros elementos que não se relacionam diretamente com o trecho de caso de uso consultado, isto é, falsos positivos e falsos negativos.

É possível também que o resultado seja reflexo de um planejamento e treinamento não balanceados ou inadequados de ambas as propostas de rastreabilidade, gerando como conseqüência o efeito chamado *Experiência de Hawthorne*. Este efeito é observado em uma experimentação como conseqüência da percepção dos participantes sobre a abordagem e não pela abordagem em si. É possível que os participantes induzissem que a abordagem por entidades seja mais eficiente que por matrizes e distorcessem a realidade.

Para aumentar o conhecimento sobre o esforço e precisão em diferentes contextos, definindo a validade da experimentação, sugere-se replicações do experimento em modelos distintos e mais completos, indexando diferentes estruturas que não apenas casos de uso e código fonte.

Sugere-se avaliar, por exemplo, diagramas de atividades, seqüência, componente, etc. Além disso, é interessante sua replicação em elementos rastreáveis com uma granularidade mais específica, relacionando, por exemplo, atributos, métodos, associações, interações, objetos, etc. Generalizando o experimento, possibilita-se a extração de novas informações sobre a proposta em diferentes perspectivas do processo de desenvolvimento.

5 CONCLUSÃO

O custo da manutenção de uma estrutura de rastreabilidade é uma das maiores barreiras para seu efetivo uso. As técnicas tradicionais exigem que isto seja feito manualmente, pois não é possível esperar que uma máquina interprete a semântica de requisitos em linguagem natural, para que estes sejam relacionados aos diversos artefatos que os satisfazem. Técnicas de recuperação automática tentam realizar este tipo de atividade, mas ainda não há relatos de sucesso no estabelecimento da rastreabilidade através de seu uso: os resultados são pouco precisos, e os métodos pelos quais os elos recuperados devem ser mantidos após sua descoberta ainda são manuais. Até a publicação deste trabalho, o autor não tomou conhecimento de outras alternativas.

Com o intuito de mudar a realidade de alto custo na manutenção da rastreabilidade, e fornecer uma solução para os problemas inerentes ao uso de relações diretas, este trabalho apresenta uma inovação sobre os tradicionais métodos: a concretização do raciocínio por trás da interpretação dos requisitos em um modelo mais detalhado que o matricial, através de entidades identificadas nos artefatos. Este é utilizado para representar os demais artefatos produzidos no ciclo de vida do sistema, rastrear os requisitos a estes, e identificar relacionamentos entre todos os artefatos automaticamente. Assim, a manutenção dos elos de rastreabilidade já existentes torna-se automática, salvo casos de inclusão de novos elos, devido à semântica que o modelo fornece, permitindo derivar relações.

Para avaliar a aplicabilidade do modelo, o protótipo ReMan foi implementado, adaptando o conceito à orientação a objetos. Conforme as avaliações, alterações nos requisitos e no código realmente permite que os elos existentes sejam atualizados semi-automaticamente, mantendo consistentes as relações entre requisitos e implementação.

Espera-se que este trabalho sirva como base para diversas novas idéias e conceitos, na ainda problemática área da rastreabilidade de requisitos. Acredita-se que o modelo apresentado serve como um passo à frente nas pesquisas da área, para que um dia faça-se referência não ao "problema da rastreabilidade", mas sim à "solução da rastreabilidade".

5.1 Trabalhos futuros

O modelo proposto não tem por objetivo ser completo, mas sim, extensível, para permitir o surgimento de soluções mais específicas para determinados problemas. Por exemplo, outra classificação para as entidades pode ser proposta, bem como outras formas de relacionamento e de identificação destes.

Além disso, seria interessante identificar formas de utilização do modelo proposto sobre os diversos modelos e diagramas utilizados na indústria, para representação de entidades.

Como toda proposta na área de rastreabilidade, o suporte ferramental é considerado essencial. Espera-se que implementações aplicáveis no mundo real sejam desenvolvidas a partir do modelo proposto. O protótipo ReMan já desenvolvido é apenas uma prova-de-conceito, mas que provavelmente pode ser utilizado como base na compreensão e melhoria do modelo aqui apresentado.

Em especial, devem ser realizadas pesquisas na área de recuperação automática de rastreabilidade, que possibilitem a criação automática de entidades. Acredita-se que a combinação do método proposto com as técnicas de recuperação já desenvolvidas possam facilitar ainda mais o processo de criação de elos. Esta idéia é baseada na hipótese de que relações simples, entre artefatos e entidades, possam ser criadas com grande precisão através do uso de tais técnicas.

Por fim, a semântica dos modelos, utilizados para representar os diferentes pontos de vista sobre o sistema, poderia ser aproveitada na determinação dos elos. Isto permitiria consultas mais eficientes na estrutura e descoberta de relações com muito mais precisão.

REFERÊNCIAS

ALMEIDA, J. P.; ECK, P. van; IACOB, M.-E. Requirements Traceability and Transformation Conformance in Model-Driven Development. In: IEEE INTERNATIONAL ENTERPRISE DISTRIBUTED OBJECT COMPUTING CONFERENCE, EDOC, 10., 2006, Hong Kong, China. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2006. p.355–366.

ANTONIOL, G.; CIMITILE, A.; CASAZZA, G. Traceability Recovery by Modeling Programmer Behavior. In: WORKING CONFERENCE ON REVERSE ENGINEERING, WCRE, 7., 2000, Brisbane, Australia. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2000. p.240.

ANTONIOL, G. et al. On feature traceability in object oriented programs. In: INTERNATIONAL WORKSHOP ON TRACEABILITY IN EMERGING FORMS OF SOFTWARE ENGINEERING, TEFSE, 3., 2005, Long Beach, CA. **Proceedings...** New York: ACM Press, 2005. p.73–78.

ARKLEY, P.; RIDDLE, S. Overcoming the Traceability Benefit Problem. In: IEEE INTERNATIONAL CONFERENCE ON REQUIREMENTS ENGINEERING, RE, 13., 2005, Washington, DC, USA. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2005. p.385–389.

BASILI, V. R.; CALDIERA, G.; ROMBACH, H. D. The Goal Question Metric Approach. In: **Encyclopedia of Software Engineering**. [S.l.]: Wiley, 1994.

BERG, K. van den; CONEJERO, J. M.; HERNÁNDEZ, J. Analysis of crosscutting across software development phases based on traceability. In: INTERNATIONAL WORKSHOP ON EARLY ASPECTS AT ICSE, EA, 2006. **Proceedings...** New York: ACM Press, 2006. p.43–50.

BIANCHI, A.; VISAGGIO, G.; FASOLINO, A. R. An Exploratory Case Study of the Maintenance Effectiveness of Traceability Models. In: INTERNATIONAL WORKSHOP ON PROGRAM COMPREHENSION, IPWC, 8., 2000, Limenck, Ireland. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2000. p.149.

CLELAND-HUANG, J. Toward improved traceability of non-functional requirements. In: INTERNATIONAL WORKSHOP ON TRACEABILITY IN EMERGING FORMS OF SOFTWARE ENGINEERING, TEFSE, 3., 2005, Long Beach, CA. **Proceedings...** New York: ACM Press, 2005. p.14–19.

CLELAND-HUANG, J. Requirements Traceability - When and How does it Deliver more than it Costs? In: IEEE INTERNATIONAL REQUIREMENTS ENGINEERING CONFERENCE, RE, 14., 2006, Minneapolis. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2006. p.323.

CLELAND-HUANG, J. Just Enough Requirements Traceability. In: ANNUAL INTERNATIONAL COMPUTER SOFTWARE AND APPLICATIONS CONFERENCE, COMPSAC, 30., 2006, Chicago, Illinois. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2006. p.41–42.

CLELAND-HUANG, J.; CHANG, C. K.; CHRISTENSEN, M. Event-Based Traceability for Managing Evolutionary Change. **IEEE Trans. Softw. Eng.**, Piscataway, NJ, USA, v.29, n.9, p.796–810, 2003.

CLELAND-HUANG, J. et al. Automating Speculative Queries through Event-Based Requirements Traceability. In: IEEE JOINT INTERNATIONAL CONFERENCE ON REQUIREMENTS ENGINEERING, RE, 2002, Essen, Germany. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2002. p.289–298.

CLELAND-HUANG, J. et al. Goal-centric traceability for managing non-functional requirements. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, ICSE, 27., 2005. **Proceedings...** [S.l.: s.n.], 2005. p.362–371.

CLELAND-HUANG, J. et al. **Grand challenges in traceability**. Montreal, Canada: Center of Excellence for Traceability, 2006. (COET-GCT-06-01-0.9).

CLELAND-HUANG, J.; ZEMONT, G.; LUKASIK, W. A Heterogeneous Solution for Improving the Return on Investment of Requirements Traceability. In: REQUIREMENTS ENGINEERING CONFERENCE, RE, 12., 2004, Kyoto, Japan. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2004. p.230–239.

DALGAARD, P. **Introductory Statistics with R**. [S.l.]: Springer, 2002. 288p.

DALGAARD, P. et al. **R**: a language and environment for statistical computing. Vienna, Austria: R Foundation for Statistical Computing, 2008.

DANEVA, M. Lessons Learnt from Five Years of Experience in ERP Requirements Engineering. In: IEEE INTERNATIONAL CONFERENCE ON REQUIREMENTS ENGINEERING, RE, 11., 2003, Monterey Bay, CA. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2003. p.45.

DELGADO, S. Next-Generation Techniques for Tracking Design Requirements Coverage in Automatic Test Software Development. In: IEEE SYSTEMS READINESS TECHNOLOGY CONFERENCE, AUTOTESTCON, 2006. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2006. p.806–812.

DÖMGES, R.; POHL, K. Adapting traceability environments to project-specific needs. **Communications of the ACM**, New York:, v.41, n.12, p.54–62, 1998.

EGYED, A. A scenario-driven approach to traceability. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, ICSE, 23., 2001, Washington, DC, USA. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2001. p.123–132.

EGYED, A. Automated abstraction of class diagrams. **ACM Trans. Softw. Eng. Methodol.**, New York:, v.11, n.4, p.449–491, 2002.

EGYED, A. A Scenario-Driven Approach to Trace Dependency Analysis. **IEEE Trans. Softw. Eng.**, Piscataway, NJ, USA, v.29, n.2003, p.116–132, 2003.

EGYED, A. et al. Determining the cost-quality trade-off for automated software traceability. In: IEEE/ACM INTERNATIONAL CONFERENCE ON AUTOMATED SOFTWARE ENGINEERING, ASE, 20., 2005. **Proceedings...** New York: ACM Press, 2005. p.360–363.

EGYED, A. et al. A value-based approach for understanding cost-benefit tradeoffs during automated software traceability, TEFSE, 3. In: INTERNATIONAL WORKSHOP ON TRACEABILITY IN EMERGING FORMS OF SOFTWARE ENGINEERING, 2005, Long Beach, CA. **Proceedings...** New York: ACM Press, 2005. p.2–7.

EGYED, A.; GRUNBACHER, P. Automating Requirements Traceability: beyond the record & replay paradigm. In: IEEE INTERNATIONAL CONFERENCE ON AUTOMATED SOFTWARE ENGINEERING, ASE, 17., 2002, Edinburgh, UK. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2002. p.163–171.

ESPINOZA, A.; ALARCON, P. P.; GARBAJOSA, J. Analyzing and Systematizing Current Traceability Schemas. **sew**, Los Alamitos, CA, USA, v.0, p.21–32, 2006.

FLETCHER, J.; CLELAND-HUANG, J. Softgoal Traceability Patterns. In: INTERNATIONAL SYMPOSIUM ON SOFTWARE RELIABILITY ENGINEERING, ISSRE, 17., 2006, Raleigh, North Carolina. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2006. p.363–374.

HAYES, J. H.; DEKHTYAR, A. Humans in the traceability loop: can't live with 'em, can't live without 'em. In: INTERNATIONAL WORKSHOP ON TRACEABILITY IN EMERGING FORMS OF SOFTWARE ENGINEERING, TEFSE, 3., 2005. **Proceedings...** New York: ACM Press, 2005. p.20–23.

HEINDL, M.; BIFFL, S. A case study on value-based requirements tracing. In: EUROPEAN SOFTWARE ENGINEERING CONFERENCE HELD JOINTLY WITH SIGSOFT INTERNATIONAL SYMPOSIUM ON FOUNDATIONS OF SOFTWARE ENGINEERING, ESEC, 10., 2005, Lisbon, Portugal. **Proceedings...** New York: ACM Press, 2005. p.60–69.

HEINDL, M.; BIFFL, S. Risk management with enhanced tracing of requirements rationale in highly distributed projects. In: INTERNATIONAL WORKSHOP ON GLOBAL SOFTWARE DEVELOPMENT FOR THE PRACTITIONER, GSD, 2006, Shanghai, China. **Proceedings...** New York: ACM Press, 2006. p.20–26.

JARKE, M. Requirements tracing. **Communications of the ACM**, New York, v.41, n.12, p.32–36, 1998.

JIANG, H. yi et al. Traceability Link Evolution Management with Incremental Latent Semantic Indexing. In: INTERNATIONAL COMPUTER SOFTWARE AND APPLICATIONS CONFERENCE, COMPSAC, 31., 2007, Beijing. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2007.

KELLEHER, J.; SIMONSSON, M. Utilizing use case classes for requirement and traceability modeling. In: IASTED INTERNATIONAL CONFERENCE ON MODELLING AND SIMULATION, MS, 17., 2006, Montreal, Canada. **Proceedings...** Anaheim, CA: ACTA Press, 2006. p.617–625.

KITCHENHAM, B.; PICKARD, L.; PFLEEGER, S. L. Case Studies for Method and Tool Evaluation. **IEEE Softw.**, Los Alamitos, CA, USA, v.12, n.4, p.52–62, 1995.

LORMANS, M.; DEURSEN, A. V. Can LSI help Reconstructing Requirements Traceability in Design and Test? In: EUROPEAN CONFERENCE ON SOFTWARE MAINTENANCE AND REENGINEERING, CSMR, 2006, Washington, DC, USA. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2006. p.47–56.

LORMANS, M.; DEURSEN, A. van. Reconstructing requirements coverage views from design and test using traceability recovery via LSI. In: INTERNATIONAL WORKSHOP ON TRACEABILITY IN EMERGING FORMS OF SOFTWARE ENGINEERING, TEFSE, 3., 2005, Long Beach, California. **Proceedings...** New York: ACM Press, 2005. p.37–42.

LUCIA, A. D. et al. Can Information Retrieval Techniques Effectively Support Traceability Link Recovery? In: IEEE INTERNATIONAL CONFERENCE ON PROGRAM COMPREHENSION,ICPC, 14., 2006, Washington, DC, USA. **Proceedings...** [S.l.: s.n.], 2006. p.307–316.

LUCIA, A. D. et al. Improving Comprehensibility of Source Code via Traceability Information: a controlled experiment. In: IEEE INTERNATIONAL CONFERENCE ON PROGRAM COMPREHENSION, ICPC, 14., 2006, Athens, Greece. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2006. p.317–326.

LUCIA, A. D.; OLIVETO, R.; SGUEGLIA, P. Incremental Approach and User Feedbacks: a silver bullet for traceability recovery. In: INTERNATIONAL CONFERENCE ON SOFTWARE MAINTENANCE, ICSM, 22., 2006, Philadelphia, Pennsylvania. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2006. p.299–309.

MAINDONALD, J.; BRAUN, J. **Data Analysis and Graphics Using R**. Cambridge: Cambridge University Press, 2003. 362p.

MALETIC, J. I.; COLLARD, M. L.; SIMOES, B. An XML based approach to support the evolution of model-to-model traceability links. In: INTERNATIONAL WORKSHOP ON TRACEABILITY IN EMERGING FORMS OF SOFTWARE ENGINEERING, TEFSE, 3., 2005, Long Beach, CA. **Proceedings...** New York: ACM Press, 2005. p.67–72.

MARCUS, A.; XIE, X.; POSHYVANYK, D. When and how to visualize traceability links? In: INTERNATIONAL WORKSHOP ON TRACEABILITY IN EMERGING FORMS OF SOFTWARE ENGINEERING, TEFSE, 3., 2005. **Proceedings...** New York: ACM Press, 2005. p.56–61.

MURTA, L. G. P.; HOEK, A. van der; WERNER, C. M. L. ArchTrace: policy-based support for managing evolving architecture-to-implementation traceability links. In: IEEE INTERNATIONAL CONFERENCE ON AUTOMATED SOFTWARE ENGINEERING, ASE, 21., 2006, Washington, DC, USA. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2006. p.135–144.

NASLAVSKY, L. et al. Using scenarios to support traceability. In: INTERNATIONAL WORKSHOP ON TRACEABILITY IN EMERGING FORMS OF SOFTWARE ENGINEERING, TEFSE, 3., 2005. **Proceedings...** New York: ACM Press, 2005. p.25–30.

NEUMULLER, C.; GRUNBACHER, P. Automating Software Traceability in Very Small Companies: a case study and lessons learned. In: INTERNATIONAL CONFERENCE ON AUTOMATED SOFTWARE ENGINEERING, ASE, 21., 2006, Washington, DC, USA. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2006. p.145–156.

NOLL, R. P.; RIBEIRO, M. B. Ontological Traceability over the Unified Process. In: IEEE INTERNATIONAL CONFERENCE AND WORKSHOPS ON THE ENGINEERING OF COMPUTER-BASED SYSTEMS, ECBS, 14., 2007, Washington, DC, USA. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2007. p.249–255.

PAECH, B.; KERKOW, D. Non-Functional Requirements Engineering - Quality is essential. In: INTERNATIONAL WORKSHOP ON REQUIREMENTS ENGINEERING: FOUNDATION FOR SOFTWARE QUALITY, REFSQ, 2004, London, UK. **Proceedings...** Berlin: Springer-Verlag, 2004. p.27–40.

PENTA, M. D.; GRADARA, S.; ANTONIOL, G. Traceability Recovery in RAD Software Systems. In: INTERNATIONAL WORKSHOP ON PROGRAM COMPREHENSION, IWPC, 10., 2002, Washington, DC, USA. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2002. p.207.

PFLEEGER, S. L.; ATLEE, J. M. **Experimentation in software engineering**: an introduction. Norwell, MA, USA: Prentice Hall, 2005.

RAMESH, B. Factors influencing requirements traceability practice. **Communications of the ACM**, New York:, v.41, n.12, p.37–44, 1998.

RAMESH, B.; JARKE, M. Toward Reference Models for Requirements Traceability. **IEEE Trans. Softw. Eng.**, Piscataway, NJ, USA, v.27, n.1, p.58–93, 2001.

RIEBISCH, M.; HUBNER, M. Traceability-Driven Model Refinement for Test Case Generation. In: IEEE INTERNATIONAL CONFERENCE AND WORKSHOPS ON THE ENGINEERING OF COMPUTER-BASED SYSTEMS, ECBS, 12., 2005, Greenbelt, Maryland. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2005. p.113–120.

SALEM, A. Improving Software Quality through Requirements Traceability Models. In: IEEE INTERNATIONAL CONFERENCE ON COMPUTER SYSTEMS AND APPLICATIONS, 2006, Washington, DC, USA. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2006. p.1159–1162.

SETTIMI, R. et al. Supporting Software Evolution through Dynamically Retrieving Traces to UML Artifacts. In: INTERNATIONAL WORKSHOP ON PRINCIPLES OF SOFTWARE EVOLUTION, IWPSE, 7., 2004, Kyoto, Japan. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2004. p.49–54.

SJOBERG, D. I. K. et al. Conducting Realistic Experiments in Software Engineering. In: INTERNATIONAL SYMPOSIUM ON EMPIRICAL SOFTWARE ENGINEERING, ISESE, 2002, Nara, Japan. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 2002. p.17–26.

SPANOUDAKIS, G. Plausible and adaptive requirement traceability structures. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING AND KNOWLEDGE ENGINEERING, SEKE, 14., 2002, New York:. **Proceedings...** ACM Press, 2002. p.135–142.

STIREWALT, R. E. K.; DENG, M.; CHENG, B. H. C. UML formalization is a traceability problem. In: INTERNATIONAL WORKSHOP ON TRACEABILITY IN EMERGING FORMS OF SOFTWARE ENGINEERING, TEFSE, 3., 2005, Long Beach, CA. **Proceedings...** New York: ACM Press, 2005. p.31–36.

TRYGGESETH, E.; NYTRO, O. Dynamic Traceability Links Supported by a System Architecture Description. In: INTERNATIONAL CONFERENCE ON SOFTWARE MAINTENANCE, ICSM, 1997, Bari, Italy. **Proceedings...** Los Alamitos, CA: IEEE Computer Society, 1997. p.180–187.

VERZANI, J. **Using R for Introductory Statistics**. Boca Raton, FL: Chapman & Hall/CRC, 2005.

WOHLIN, C. et al. **Experimentation in software engineering**: an introduction. New York:: Kluwer Academic Publishers, 2000.

ANEXO A PROTÓTIPO PARA VALIDAÇÃO DO MODELO

Com o intuito de aplicar o modelo proposto, e facilitar a validação do mesmo, foi implementado o protótipo ReMan (Requirements Manager). Este permite gerenciar requisitos e código fonte, implementando o modelo de rastreabilidade através de entidades.

Ao iniciar o protótipo ReMan, é apresentada a tela inicial, conforme a figura A.1.

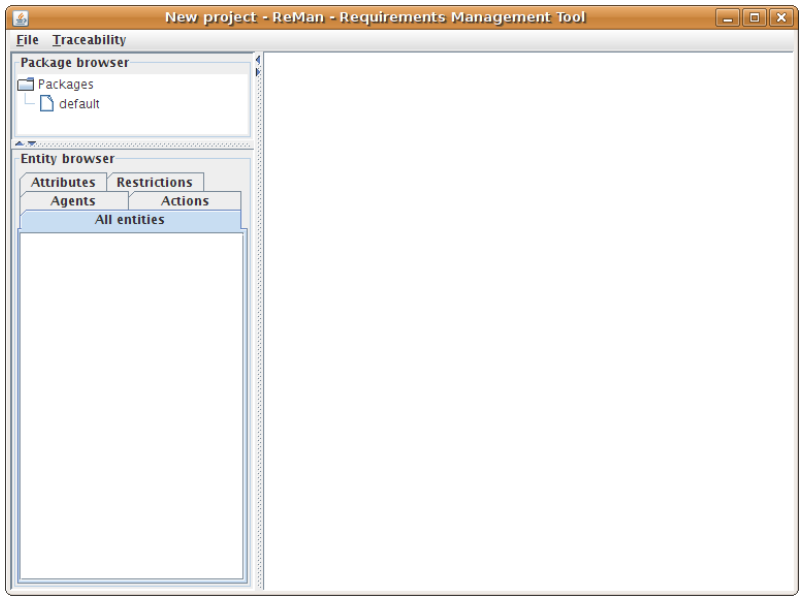


Figura A.1: Tela inicial do protótipo

Inicialmente, é necessário abrir ou criar novos arquivos de requisitos e de código fonte, que são os únicos artefatos que o protótipo consegue rastrear. A figura A.2 exibe a forma pela qual cria-se um arquivo requisito.

Ao criar um novo requisito, é apresentada a tela de edição do mesmo, conforme a figura A.3, permitindo a descrição resumida e a completa do requisito. Note que cada campo da tela possui uma barra de ferramentas. Esta contém vários botões para identificação de agentes, atributos, ações, restrições e entidades sem classificação (botões AG, ATT, ACT, RES e ENT, respectivamente). O botão CLR remove a área de interesse que identifica quaisquer das entidade citadas.

Ao selecionar parte do texto, considerado uma área de interesse deve dar origem à uma entidade, e clicar no botão correspondente à entidade que deseja-se criar, é

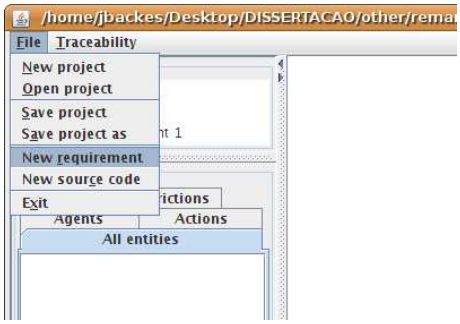


Figura A.2: Criando um novo requisito

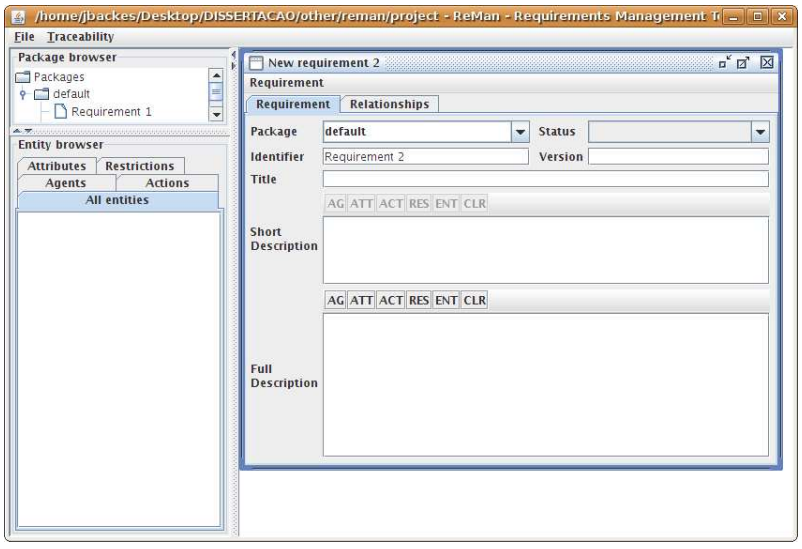


Figura A.3: Editando o requisito

apresentada a tela de definição de entidades, conforme a figura A.4: define-se um nome para a entidade (no caso do exemplo, um agente com nome “Agente 2”). Basta clicar em “create” para criar a entidade e voltar ao texto. Caso área de interesse deva ser relacionada a uma entidade já existente, basta selecioná-la na lista e clicar no botão “select”.

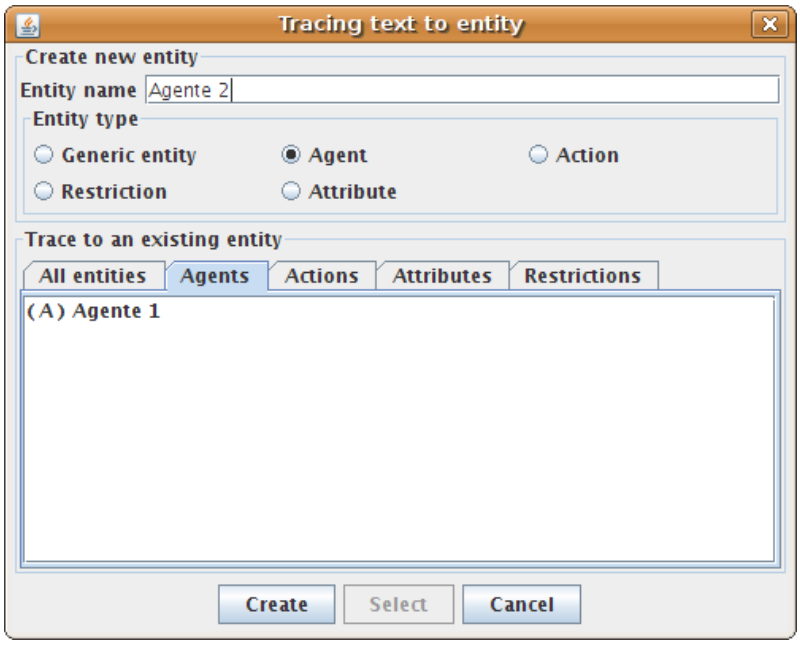


Figura A.4: Tela de definição de entidade

Ao criar/selecionar uma entidade, esta estará realçada no texto, de acordo com o tipo selecionado, conforme a figura A.5.

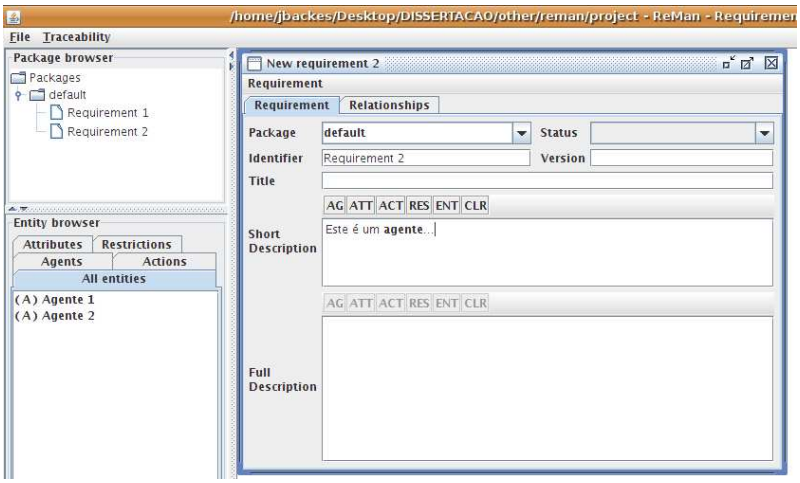


Figura A.5: Agente identificado nos requisitos

Agentes são apresentados em negrito, atributos em itálico, restrições são sublinhadas, ações ficam com o fundo amarelo, e entidades sem classificação têm letra na cor azul, conforme a figura A.6.

Para editar uma entidade, basta clicar com o botão direito sobre a área de interesse. Aparecerá um menu com as entidades identificadas nesta região. As

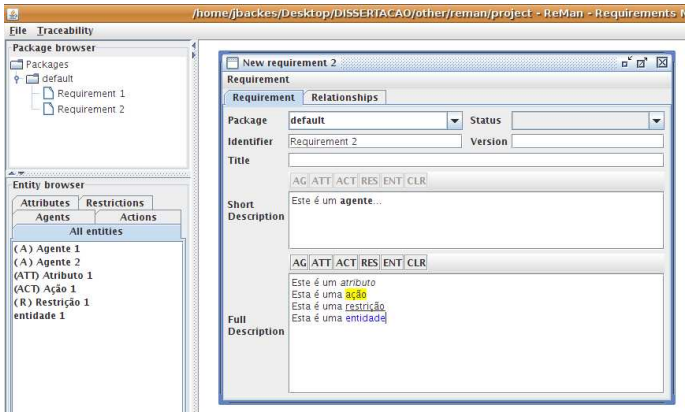


Figura A.6: Todas as entidades identificadas nos requisitos

opções "edit" e "remove" aparecerão, permitindo remover a área interesse de cada entidade que se encontra no texto, ou editar uma das entidades, conforme a figura A.7.

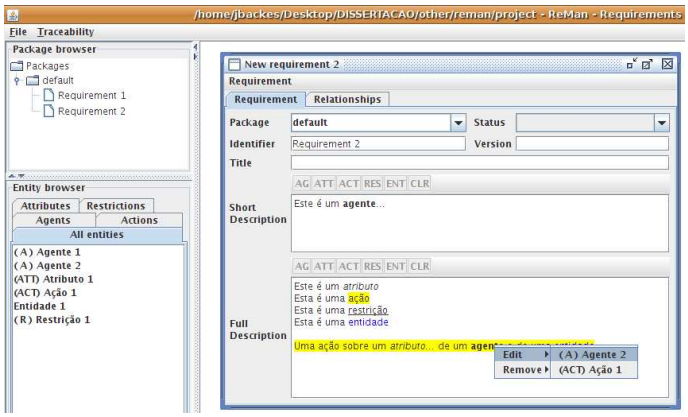


Figura A.7: Menu de edição das entidades localizadas no texto

Ao selecionar uma entidade, a tela de propriedades desta é apresentada. No caso do exemplo, são apresentadas as propriedades do agente 2. Na aba "Agent Component", pode-se definir quais são os atributos, ações e restrições do agente, conforme figrua A.8. Basta selecionar as entidades existentes na lista de baixo, e clicar no botão select para relacionar uma entidade à outra (atributo 1 é um atributo do agente, e ação 1 é uma ação do agente. Restrição 1 não está relacionada ao agente).

A tela edição é diferente para cada entidade. No caso de ações e restrições, por exemplo, identifica-se as entidades afetadas pelas, bem como as entidades que originam cada uma, conforme demonstra a figura A.9.

Na aba "properties", é possível associar propriedades ao agente, conforme apresenta a figura A.10. Esta tela é idêntica para todas as entidades.

Na aba "traces", define-se elos de rastreabilidade entre as entidades, conforme a figura A.11.

Ao clicar no botão "create new trace", pode-se criar relações entre as entidades, conforme a figura A.12. É apresenta a tela para seleção das entidades. Basta selecionar as entidades que devem ser relacionadas ao agente "agente 2", e clicar em "create".

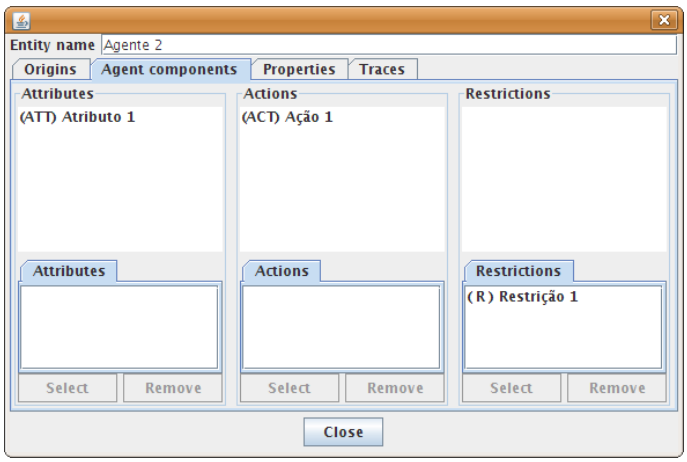


Figura A.8: Definindo componentes de um agente

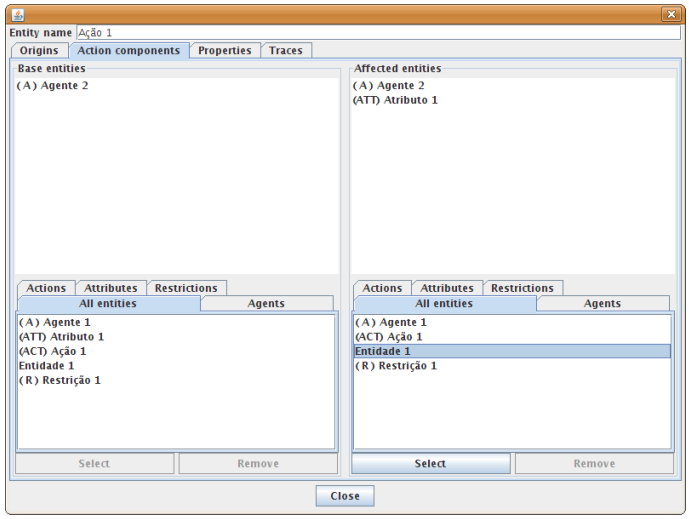


Figura A.9: Definindo componentes de uma ação

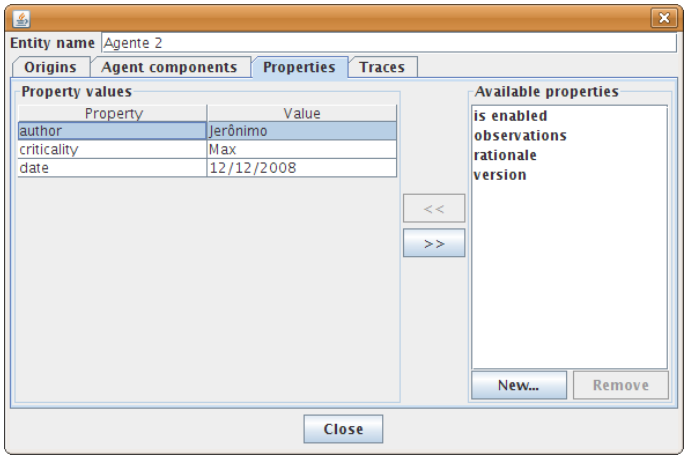


Figura A.10: Propriedades associadas ao agente

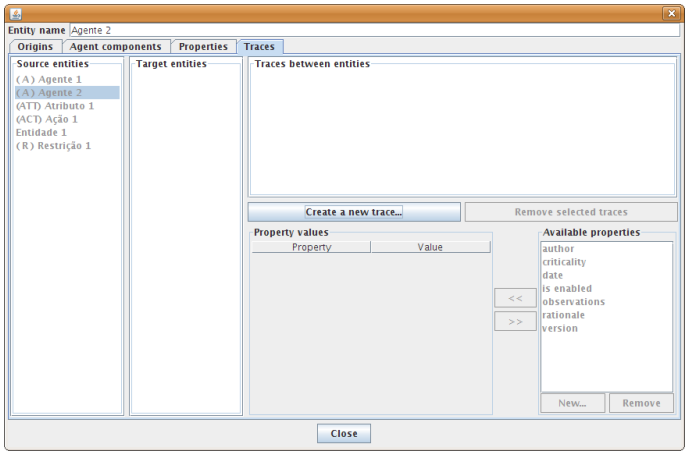


Figura A.11: Definindo relações entre entidades

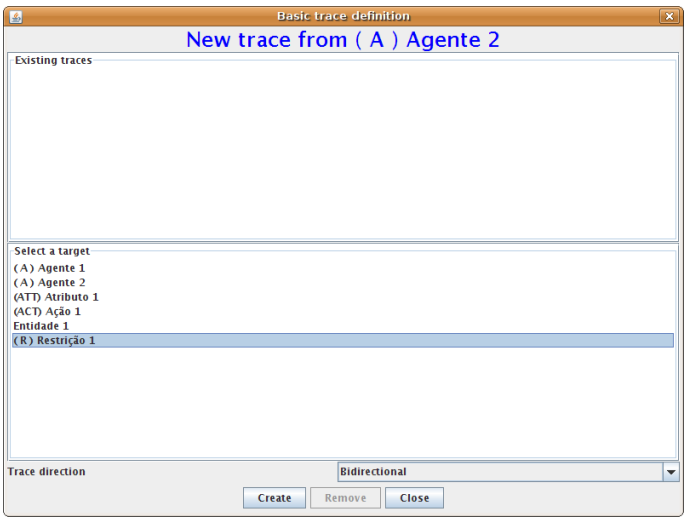


Figura A.12: Tela para criação de elos

As relações serão apresentadas na metade superior da tela, conforme a figura A.13.

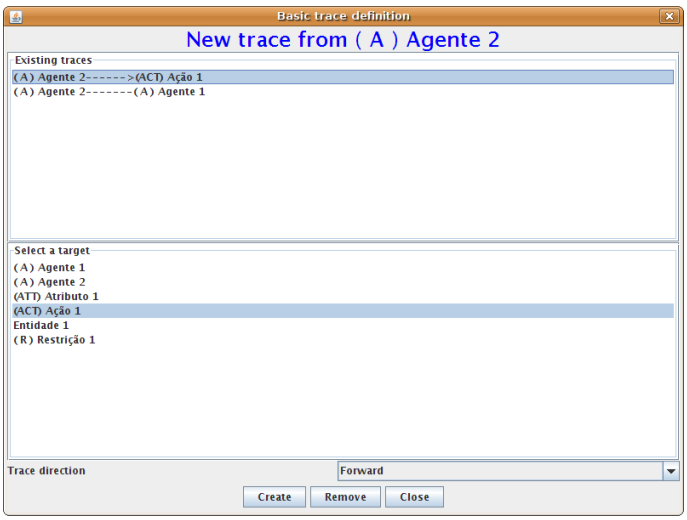


Figura A.13: Elos criados entre o agente 2 e outras entidades

Ao finalizar a criação de elos, volta-se para a tela anterior. Esta estará exibindo os elos existentes entre a entidade. Para cada elo pode-se adicionar propriedades com valores diversos, conforme apresenta a figura A.14.

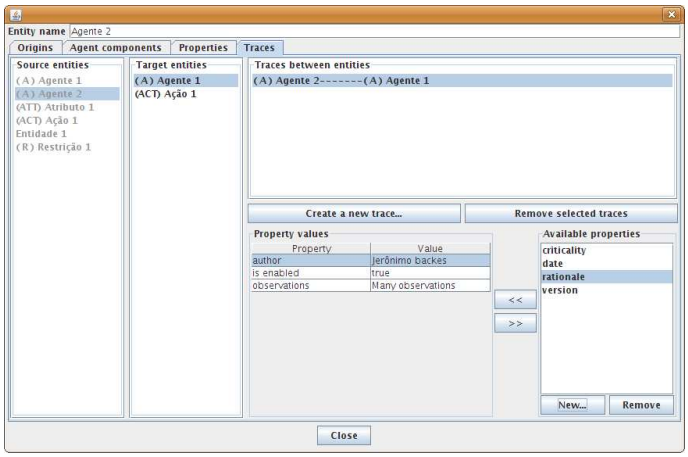


Figura A.14: Adicionando atributos nas relações entre entidades

Voltando à tela inicial, é possível abrir um arquivo de código fonte, e realizar o rastreamento de entidades identificadas nos requisitos, com as identificadas no código. A figura A.15 apresenta um trecho de código com entidades marcadas no texto.

Para visualizar as áreas de interesse que dão origem a cada entidade, editar a entidade, e acessar a aba “origins”. Serão apresentadas duas listas: uma com cada artefato que dá origem à entidade (à esquerda) e outra com as áreas de interesse identificadas no artefato selecionado (à direita). Caso nenhum artefato seja selecionado, todas as áreas de interesse de todo os artefatos serão apresentadas, conforme mostra a figura A.16.

Finalmente, para visualizar as relações entre os artefatos, basta clicar no menu “Traceability -> Find Relations” da tela principal. Será exibida a relação entre os

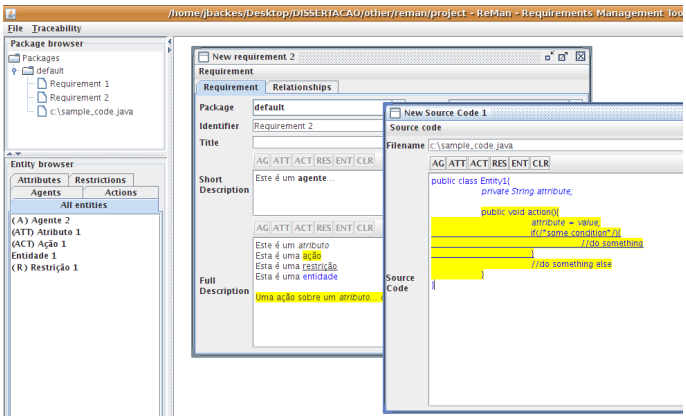


Figura A.15: Editando código fonte

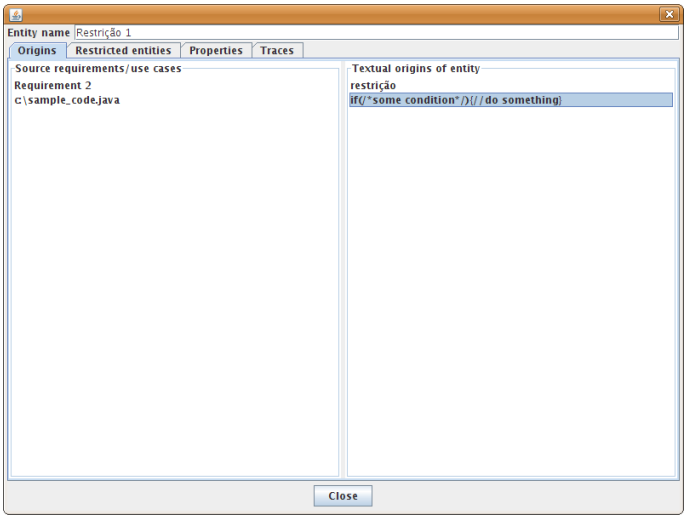


Figura A.16: Visualizando as áreas de interesse de uma entidade

artefatos, com base nas entidades originadas a partir de suas áreas de interesse, como demonstra a figura A.17.

Figura A.17: Identificando relações entre artefatos

Esta relação é meramente auxiliar, já que a edição de cada entidade trás as informações de rastreabiliade com maiores detalhes. Por tratar-se de um protótipo, não foram implementadas consultas complexas. Este foi criado única e exclusivamente para permitir a avaliação do modelo proposto, apresentada no capítulo 4.

ANEXO B TABELA DE DISTRIBUIÇÃO T

Graus de Liberdade	**Probabilidade, p a 0.05**
1	12.71
2	4.30
3	3.18
4	2.78
5	2.57
6	2.45
7	2.37
8	2.31
9	2.26
10	2.23
11	2.20
12	2.18
13	2.16
14	2.14
15	2.13
16	2.12
17	2.11
18	2.10
19	2.09
20	2.09
21	2.08
22	2.07
23	2.07
24	2.06
25	2.06
26	2.06
27	2.05
28	2.05
29	2.05
30	2.04
40	2.02
60	2.00
120	1.98
∞	1.96

ANEXO C ATIVIDADES PARA AVALIAÇÃO QUANTITATIVA DO RASTREAMENTO POR ENTIDADES

Avaliação Quantitativa

RASTREAMENTO POR ENTIDADES

Atividade #01: Identificar entidades nos requisitos
Registrar o horário de início da atividade.
• Abra o protótipo ReMan, e depois abra o projeto REQUIREMENT.rmf. • Os requsitos do sistema aparecem em Package Browser. Ao clicar duas vezes em um requisito do pacote, este é aberto. Abra os dois requisitos e identifique as entidades relevantes em cada itern do caso de uso descrito em "full description". • Identifique os componentes que fazem parte de cada entidade (ações de agentes, restrições e atributos de agentes; entidades afetadas por ações e restrições, etc), clicando com o botão direito sobre esta, e a seguir, selecionando "edit" e o nome da entidade na qual você está interessado. Em "agent components", basta selecionar as entidades já identificadas nos requisitos, e clicar em "select", para relacioná-las.
Registrar o horário de término da atividade.

Atividade #02: Identificar entidades no código-fonte
Registrar o horário de início da atividade.
• Abra o protótipo ReMan, e depois abra o projeto REQUIREMENT.rmf. • Os códigos fonte do sistema aparecem em Package Browser. Ao clicar duas vezes em um código do pacote, este é aberto. Abra os códigos fonte e identifique as entidades relevantes, que se relacionam aos requisitos implementados pelo código. • Novamente, identifique os componentes que fazem parte de cada entidade, da mesma forma corno foi feito na **Atividade #01**. • Salve o projeto com seu nome.
Registrar o horário de término da atividade.

Atividade #03: Alterar a estrutura de rastreabilidade.
Registrar o horário de início da atividade.
• Abra o protótipo ReMan, e depois abra o projeto com seu nome. • Apague três métodos do código fonte que foram rastreados para alguma entidade. • Transforme três métodos do código fonte, que foram rastreados para uma ou mais entidades, para um único método. • Atualize as relações, se necessário. • Identifique as relações entre os requisitos e o código fonte. • Salve o projeto.
Registrar o horário de término da atividade.

ANEXO D ATIVIDADES PARA AVALIAÇÃO QUANTITATIVA DO RASTREAMENTO POR MATRIZES

Avaliação Quantitativa

RASTREAMENTO POR MATRIZES

Atividade #01: Identificar relacionamentos entre os requisitos
Registrar o horário de início da atividade.
• Abra ao casos de uso em html, e então a planilha horizontal_traces. Nesta planilha, estão descritos os casos de uso, cada passo do fluxo principal, e os fluxos alternativos. • Identifique, com um "X", as relações entre os passos do caso de uso que você considera válidas.
Registrar o horário de término da atividade.

Atividade #02: Identificar relações entre os elementos dos requisitos e o código-fonte
Registrar o horário de início da atividade.
• Abra os arquivos de códigos fonte e a planilha vertical_traces. Nesta planilha, estão descritos em cada coluna, os casos de uso, cada passo do fluxo principal, e os fluxos alternativos. Em cada linha, estão identificadas as classes, seus atributos e métodos. • Identifique os elementos do código fonte que satisfazem os requisitos, marcando com um "X" na planlilha, o elemento do código fonte e o elemento do caso de uso que é satisfeito, ou possui alguma relação que você considera importante. • Salve a planilha com seu nome.
Registrar o horário de término da atividade.

Atividade #03: Alterar a matriz de rastreabilidade.
Registrar o horário de início da atividade.
• Abra a planilha com seu nome, os requisitos e o código fonte. • Apague três métodos do código fonte que foram rastreados a algum requisito • Transforme três métodos do código fonte, que foram rastreados a um ou mais requisitos, em um único método. • Atualize as relações, se necessário. • Identifique as relações entre os requisitos e o código fonte. • Salve a planilha.
Registrar o horário de término da atividade.

ANEXO E QUESTIONÁRIO DE AVALIAÇÃO QUALITATIVA DA PROPOSTA

Avaliação Qualitativa

RASTREAMENTO POR

( ) Entidades

( ) Matrizes

Avaliação Qualitativa:

(1) Muito Ruim	*(2) Ruim*	*(3) Regular*	*(4) Bom*	*(5) Muito bom*
	(Q1) Qual o grau de **usabilidade** da técnica no rastreamento dos artefatos?			
	(Q2) Qual o grau de **utilidade** da técnica no rastreamento dos artefatos?			
(1) Muito trabalhoso	*(2) Trabalhoso*	*(3) Aceitável*	*(4) Pouco esforço*	*(5) Praticamente nenhum*
	(Q3) Como você avalia o **esforço** necessário para **estabelecer** uma estrutura de rastreabildiade?			
	(Q4) Como você avalia o **esforço** necessário para **atualizar** a estrutura de rastreabilidade?			
(1) Em nenhum caso	*(2) Em poucos casos*	*(3) Em alguns casos*	*(4) Na maioria dos casos*	*(5) Em todos os casos*
	(Q5) Na sua opinião, a técnica **atende ao que se propõe**?			
	(Q6) Esta técnica é **viável** no processo de desenvolvimento da empresa?			

Quais são os pontos fracos que você identificou?

Quais são os pontos fortes que você identificou?