UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Raquel Nitsche dos Santos

TÉCNICA INDUTIVA PARA OBTENÇÃO DE RASTROS ENTRE ARTEFATOS DE SOFTWARE

Florianópolis

2010

Raquel Nitsche dos Santos

TÉCNICA INDUTIVA PARA OBTENÇÃO DE RASTROS ENTRE ARTEFATOS DE SOFTWARE

Dissertação submetida ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Santa Catarina para a obtenção do Grau Mestre em Ciência da Computação
Orientador: Prof. Dr. Raul Sidnei Wazlawick.

Florianópolis

2010

Catalogação na fonte pela Biblioteca Universitária da
Universidade Federal de Santa Catarina

S237t Santos, Raquel Nitsche dos
Técnica indutiva para obtenção de rastros entre Artefatos de software [dissertação] / Raquel Nitsche dos Santos ; orientador, Raul Sidnei Wazlawick. - Florianópolis, SC, 2010.
108 p.: il., grafs., tabs.

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico. Programa de Pós-Graduação em Ciência da Computação.

Inclui referências

1. Informática. 2. Ciência da computação. 3. Rastreabilidade. 4. Projeto de sistemas - Análise. 5. Software - Desenvolvimento. I. Wazlawick, Raul Sidnei. II. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Ciência da Computação. III. Título.

CDU 681

TÉCNICA INDUTIVA PARA OBTENÇÃO DE RASTROS ENTRE ARTEFATOS DE SOFTWARE

Esta Dissertação foi julgada adequada para obtenção do Título de Mestre em Ciência da Computação, e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

Florianópolis, 01 de julho de 2010.

_____
Prof. Mário Dantas, Dr.
Coordenador do Curso

Banca Examinadora:

_____
Prof. Raul Sidnei Wazlawick, Dr.
Orientador
Universidade Federal de Santa Catarina

_____
Guilherme Horta Travassos, Dr.
Universidade Federal do Rio de Janeiro

_____
Patrícia Vilain, Dra.
Universidade Federal de Santa Catarina

_____
Frank Augusto Siqueira, Dr.
Universidade Federal de Santa Catarina

AGRADECIMENTOS

Agradeço ao Prof. Raul, orientador desta dissertação, por todo empenho, sabedoria e compreensão. Ao meu marido Venicius, pelo incentivo e contribuição na execução dos primeiros experimentos. A meus pais Terezinha e Romeu, irmãos Rodrigo e Robson, pelo apoio e incentivo. Agradeço, enfim, a todos aqueles que gentilmente contribuíram para o desenvolvimento desta dissertação.

Que ninguém se engane, só se consegue simplicidade através de muito trabalho.

Clarisse Lispector

RESUMO

A tarefa de manter a qualidade e consistência de artefatos ao longo de um projeto de software pode ser mais efetiva com a utilização do conceito de rastreabilidade. Porém, a criação de rastros consistentes ao longo de um projeto é uma tarefa tão complexa que muitas vezes é deixada de lado. Técnicas como a recuperação automática e a matriz de rastreabilidade apresentam limitações. Este trabalho propõe e avalia uma abordagem que consiste em permitir a criação de rastros de forma indutiva ao longo do desenvolvimento dos artefatos do projeto. Estudos com uma ferramenta CASE mostraram resultados animadores indicando que a técnica pode melhorar significativamente a produtividade. A produtividade é avaliada através do número de rastros corretamente criados pelo desenvolvedor, dentro de um mesmo limite de tempo, quando comparada a técnicas tradicionais.

Palavras-chave: rastreabilidade, rastros, artefatos de software, modelagem de software, análise e projeto de sistemas.

ABSTRACT

Maintaining quality and consistency of artifacts through a software project can be more effective with the use of traceability. However, creating consistent traceability relationships can be a task so complex that it is often ignored or minimized. Techniques such as automatic discovery and traceability matrix have known limitations. This thesis examines an alternative that consists in allowing the creation of traceability relationships inductively during the software development process. Experiments with a CASE tool showed encouraging results indicating that the technique can significantly improve productivity.

Key-words: traceability, software artifacts, software modeling, systems analysis and design.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

CASE - *Computer-Aided Software Engineering*
LSI - *Latent Semantic Indexing*
EA - *Enterprise Architect*™

SUMÁRIO

1 INTRODUÇÃO

Para que o desenvolvedor possa saber quais elementos afetam e são afetados por outros, ele pode utilizar o conceito de rastreabilidade, que consiste em uma maneira de associar elementos indicando um rastro entre eles.

A finalidade principal do mapeamento de rastros é possibilitar que a consistência entre os elementos seja mantida ao longo do projeto. Ao realizar uma mudança em um determinado elemento deve-se levar em consideração que o elemento que está associado a ele por meio do rastro possivelmente também deva ser alterado.

Em projetos que não se utilizam de rastreabilidade muitas vezes o elemento afetado é desconsiderado, devido a dificuldade em identificá-lo, pois seria necessário revisar praticamente todos os elementos do projeto, o que pode incluir centenas de elementos. A existência do rastro facilita a localização do elemento afetado. Sem ele há uma grande chance de que elementos afetados sejam esquecidos, inserindo inconsistências no projeto.

A rastreabilidade entre elementos de artefatos permite acompanhar a vida de um artefato durante o ciclo de vida do software (GOTEL e FINKELSTEIN, 1994). A rastreabilidade auxilia na compreensão dos relacionamentos entre os artefatos (PALMER, 1997), na análise de impacto e no reuso de software, proporcionando ao desenvolvedor uma importante visão para o processo de desenvolvimento e evolução do software. Ela é reconhecida como um importante fator para obtenção da qualidade no processo de desenvolvimento, bem como para um gerenciamento de projeto eficiente (NEUMULLER e GRUNBACHER, 2006). Processos de melhoria da qualidade de software como o CMMI (especialmente no nível 3), ISO 15504 e MPS-BR estabelecem que práticas básicas de rastreabilidade devem ser seguidas.

As principais técnicas existentes para a criação de rastros são a matriz de rastreabilidade e a recuperação automática. Ambas são importantes, porém possuem limitações que dificultam o uso efetivo da rastreabilidade na prática.

De acordo com CLELAND-HUANG, ZEMONT e LUKASIK (2004) a matriz de rastreabilidade assume um tamanho que rapidamente se torna não gerenciável, uma vez que o número de elementos tende a aumentar significativamente, tornando a utilização da matriz complexa.

O empenho dos desenvolvedores é retardado pela carência de suporte por parte das ferramentas, o que causa uma percepção de que o custo despendido para manter a matriz é muito alto, em comparação aos seus benefícios (CLELAND-HUANG, 2006b).

Técnicas de recuperação automática não recuperam todos os relacionamentos corretos sem também recuperar muitos falsos, o que acarreta trabalho extra para o desenvolvedor no descarte destes relacionamentos (SETTIMI et all, 2004). A tarefa de descarte pode ser muito trabalhosa, pois o desenvolvedor pode gastar mais tempo para descartar rastros falsos do que rastreando os corretos. Ainda que seja possível melhorar o desempenho destas técnicas elas estão longe de ser uma solução viável para o problema (LUCIA et al, 2006a).

Mesmo que a rastreabilidade seja requerida em grande parte das aplicações de segurança crítica, e faça parte de diversas iniciativas de melhoria de processo de software, as organizações ainda buscam uma maneira de implementá-la de forma que traga uma boa relação custo-benefício (CLELAND-HUANG, 2006b).

Apesar de sua reconhecida importância, não é satisfatório o suporte para a criação de rastros em ferramentas CASE contemporâneas, especificamente aquelas que permitem a construção de artefatos UML. O principal defeito destas ferramentas é a falta de um suporte automático ou semi-automático para a criação de links de rastreabilidade, o que torna sua utilização cansativa e custosa (OLIVETO et al., 2007).

Os rastros entre elementos de artefatos não são identificadas automaticamente nas ferramentas CASE porque a inclusão de novos elementos nos diagramas usualmente é feita sem que sua origem seja explicitada, os elementos são criados de forma individual, geralmente a partir de toolboxes. Desta forma, a criação de rastros é uma tarefa deixada para depois, ou seja, somente depois de inserir o novo elemento no diagrama o usuário da ferramenta CASE indica quais rastros se aplicam àquele elemento.

Esta dissertação[1] mostra que é possível automatizar a criação de rastros sob a hipótese de que a inserção de novos elementos nos artefatos não consiste simplesmente em criar um novo elemento no diagrama, mas em uma ação que, em muitos casos, tem uma origem bem definida a partir de algum outro elemento. Por exemplo, uma classe pode estar sendo inserida no modelo conceitual devido à existência de

[1] Foi realizada a publicação de um artigo sobre o tema desta dissertação, que pode ser visualizado no Anexo A.

um caso de uso que a menciona. Ou ainda, um caso de uso pode estar sendo inserido no diagrama em função de um ou mais requisitos que lhe dão origem.

Assim, a idéia examinada nesta dissertação é de que a inserção de elementos nos artefatos seja indutiva. Ou seja, com exceção dos elementos iniciais (base) a inserção de um elemento em um artefato deverá ocorrer a partir de outro elemento, o qual consiste em sua origem.

Esta dissertação apresenta, então, uma abordagem semi-automática para a criação da rastreabilidade, que objetiva tornar sua utilização menos custosa que a técnica tradicional, almejando seu uso efetivo nas organizações.

1.1 OBJETIVOS

Esta seção descreve o objetivo geral e os objetivos específicos da dissertação.

1.1.1 Objetivo Geral

O objetivo principal deste trabalho consiste em definir e avaliar preliminarmente (em situação de laboratório) a técnica indutiva, que permite que a criação de rastros aconteça ao longo do desenvolvimento dos artefatos do projeto. Desta forma, ao ser inserida no processo de desenvolvimento dos artefatos, o mapeamento de rastros é automatizado.

Convém caracterizar que este objetivo difere de outras técnicas de detecção de rastreabilidade como os trabalhos de LUCIA et al. (2007) e JIANG et al. (2007). Isto significa que a automatização da rastreabilidade de que trata este trabalho não acontece depois que os artefatos estão prontos e sim durante o processo de desenvolvimento destes.

Por outro lado ainda, a técnica difere também da abordagem de introdução manual dos rastros na matriz de rastreabilidade, pois com a técnica proposta grande parte destes rastros é criada automaticamente pela ferramenta CASE, sem a necessidade da realização de uma operação manual específica, como normalmente é feito nestas ferramentas.

1.1.2 Objetivos Específicos

Os objetivos específicos deste trabalho são:

a) Identificar e detalhar o conjunto das operações que afetam os rastros em um projeto de software.
b) Possibilitar a utilização da técnica indutiva em uma ferramenta CASE de largo uso comercial.
c) Mostrar que o uso das operações indutivas pode implicar em ganho de produtividade na criação de rastros.

1.2 MÉTODO DE TRABALHO

Inicialmente foram analisados os diferentes tipos de artefatos e seus respectivos rastros, os quais podem ser mapeados no decorrer do processo de desenvolvimento de software. Durante esta análise foi identificado que cada artefato é composto por um conjunto de elementos e os rastros são mapeados por meio do relacionamento entre dois ou mais elementos, seja relacionamento entre elementos do mesmo artefato ou relacionamento entre elementos pertencentes a artefatos diferentes.

Ao avaliar a forma com que os elementos afetam e são afetados por outros foi possível delinear o conjunto de operações que, ao serem executadas, impactam nos rastros. Este conjunto foi desenvolvido com o

objetivo de proporcionar ao desenvolvedor um suporte completo no mapeamento dos rastros de seu projeto.

A fim de possibilitar a utilização prática da técnica indutiva foi desenvolvido um *plugin* para a ferramenta CASE Enterprise Architect™ (EA). Todas as operações propostas foram disponibilizadas nesta ferramenta.

Um estudo quantitativo foi realizado com o intuito de efetuar um comparativo entre a técnica indutiva e a técnica tradicional no mapeamento da rastreabilidade. Foram utilizados gráficos e testes de hipóteses para avaliação dos resultados.

1.3 JUSTIFICATIVA

As principais técnicas (matriz e recuperação automática) possuem limitações que impedem o uso efetivo da rastreabilidade na prática. Por outro lado as organizações buscam uma maneira de utilizá-la de forma que tragam um bom custo-benefício (CLELAND-HUANG, 2006b).

O suporte para a rastreabilidade em ferramentas CASE contemporâneas não é satisfatório. O principal defeito destas ferramentas é a falta de um suporte automático ou semi-automático para a criação de links de rastreabilidade, o que torna sua utilização cansativa e custosa (OLIVETO et al., 2007).

Os rastros entre elementos de artefatos não são identificadas automaticamente nas ferramentas CASE porque a inclusão de novos elementos nos diagramas usualmente é feita sem que sua origem ou causa seja explicitada, os elementos são criados de forma individual, geralmente a partir de toolboxes. Desta forma, a criação da rastreabilidade é uma tarefa deixada para depois, ou seja, após inserir o novo elemento no diagrama o usuário da ferramenta CASE deverá indicar quais as rastros se aplicam àquele elemento.

Esta dissertação mostra que é possível automatizar a criação de rastros sob a hipótese de que a inserção de novos elementos nos artefatos não consiste simplesmente em criar um novo elemento no diagrama, mas em uma ação que, em muitos casos, tem uma origem bem definida a partir de algum outro elemento.

1.5 RESULTADOS ESPERADOS

As principais técnicas (matriz e recuperação automática) possuem limitações que impedem o uso efetivo da rastreabilidade na prática. Por outro lado as organizações buscam uma maneira de utilizá-la de forma que tragam um bom custo-benefício (CLELAND-HUANG, 2006b).

O suporte para a rastreabilidade em ferramentas CASE contemporâneas não é satisfatório. O principal defeito destas ferramentas é a falta de um suporte automático ou semi-automático para a criação de links de rastreabilidade, o que torna sua utilização cansativa e custosa (OLIVETO et al., 2007).

Os rastros entre elementos de artefatos não são identificadas automaticamente nas ferramentas CASE porque a inclusão de novos elementos nos diagramas usualmente é feita sem que sua origem ou causa seja explicitada, os elementos são criados de forma individual, geralmente a partir de toolboxes. Desta forma, a criação da rastreabilidade é uma tarefa deixada para depois, ou seja, após inserir o novo elemento no diagrama o usuário da ferramenta CASE deverá indicar quais as rastros se aplicam àquele elemento.

Esta dissertação mostra que é possível automatizar a criação de rastros sob a hipótese de que a inserção de novos elementos nos artefatos não consiste simplesmente em criar um novo elemento no diagrama, mas em uma ação que, em muitos casos, tem uma origem bem definida a partir de algum outro elemento.

1.6 LIMITAÇÕES

O escopo principal desta dissertação compreende melhorias para o processo de criação de rastros. A manutenção é abordada em alguns momentos, porém não faz parte do objetivo principal deste trabalho.

A técnica indutiva proposta é eficaz apenas em sistemas cuja documentação ainda vai ser produzida, ou seja, quando se tem a oportunidade de utilizar a técnica desde o início do projeto. Outra alternativa de utilização é na manutenção de sistemas onde as rastros estejam atualizadas. A técnica não detecta rastros que deveriam existir entre elementos e que por alguma razão não foram incluídos. Para este

tipo de situação, ela pode ser complementada, por exemplo, com a aplicação previa de uma técnica de recuperação automática, como a LSI (*Latent Semantic Indexing*).

O componente disponibilizado pela Sparx Systems (SPARX SYSTEMS, 2009) possui limitações que restringem o desenvolvimento de *plugins*, no que se refere à usabilidade final do mesmo. Algumas funcionalidades tiveram que ser adaptadas a partir do planejamento inicial, para se adequar ao funcionamento do componente.

1.7 ESTRUTURA DA DISSERTAÇÃO

O capítulo 2 REVISÃO BIBLIOGRÁFICA, contempla conceitos de rastreabilidade e o funcionamento geral das principais técnicas da área. O capítulo 3 RASTREABILIDADE INDUTIVA, apresenta em detalhes as principais características da técnica indutiva, incluindo as operações que afetam os rastros. O capítulo 4 PLUGIN PARA A TÉCNICA INDUTIVA contém detalhes do funcionamento do *plugin* desenvolvido. O capítulo 5 ESTUDO E RESULTADOS, possui a descrição e resultados do estudo realizado. No capítulo 6 CONCLUSÕES E TRABALHOS FUTUROS, estão expostas as conclusões finais da dissertação. O capítulo 7 REFERÊNCIAS BIBLIOGRÁFICAS, contém as referências bibliográficas utilizadas na dissertação.

2 REVISÃO BIBLIOGRÁFICA

Neste capítulo são apresentados os conceitos básicos relacionados à rastreabilidade, bem como as principais técnicas da área (estado da arte).

Matriz de rastreabilidade e recuperação automática de rastreabilidade são as principais técnicas da área. Estas tratam a rastreabilidade como uma atividade à parte da criação dos elementos nos artefatos, ou seja, primeiro são criados os elementos, depois são definidos os rastros.

2.1 RASTREABILIDADE

A rastreabilidade possibilita o entendimento das relações de dependência entre elementos nos diferentes artefatos, tais como requisitos, casos de uso, classes, telas, diagramas de seqüência, etc. Para que o desenvolvedor possa saber quais elementos afetam e são afetados por outros, ele pode utilizar o conceito de rastreabilidade, que consiste em uma maneira de associar elementos indicando um rastro entre eles.

A finalidade principal do mapeamento de rastros é possibilitar que a consistência entre os elementos seja mantida ao longo do projeto. Ao realizar uma mudança em um determinado elemento deve-se levar em consideração que o elemento que está associado a ele por meio do rastro possivelmente também deva ser alterado. A existência deste relacionamento facilita a localização do elemento afetado, sem ele há uma grande chance de que este seja esquecido e se torne inconsistente.

GOTEL e FINKELSTEIN (1994) afirmam que a rastreabilidade, entre elementos de artefatos, permite acompanhar a vida de um elemento durante o ciclo de vida do software.

PALMER (1997) destaca que a rastreabilidade auxilia na compreensão dos relacionamentos entre os artefatos de software, na análise de impacto e no reuso de software, proporcionando ao

desenvolvedor uma importante visão para o processo de desenvolvimento e evolução de software.

Ela é reconhecida como um importante fator para obtenção da qualidade no processo de desenvolvimento, bem como para um gerenciamento de projeto eficiente (NEUMULLER e GRUNBACHER, 2006). Processos de melhoria da qualidade de software como o CMMI, ISO 15504 e MPS-BR estabelecem que práticas básicas de rastreabilidade devem ser seguidas.

A Figura 1 contém uma representação gráfica da relação origem-destino entre os elementos, indicada por meio de rastros. Levando-se em consideração o rastro existente entre o elemento o_1 e o elemento d_1, observa-se que o o_1 é o elemento-origem, ou ponto de partida, do relacionamento e d_1 é o elemento-destino. O sentido da seta do rastro indica quem é a origem e quem é o destino, ou ainda, qual elemento foi inserido primeiro no diagrama, levando em consideração rastro específico.

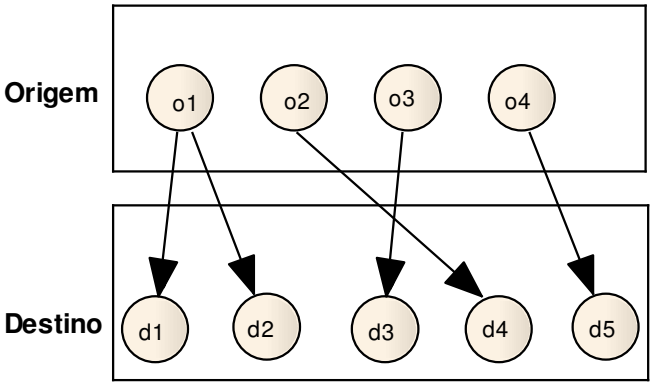

Figura 1- Representação gráfica da origem-destino dos rastros.

Gotel e Finkelstein (1994) afirmam que a rastreabilidade entre elementos de artefatos de software permite acompanhar a vida de um elemento durante o ciclo de vida do software em ambas as direções: para frente e para trás. Para exemplificar a rastreabilidade para frente, pode-se dizer que ao partir de um requisito é possível chegar ao código-fonte correspondente. Já na rastreabilidade para trás, ao partir do código-fonte pode-se chegar ao requisito relacionado.

Ao realizar uma análise de impacto, tomando como base o elemento o_1, os elementos d_1 e d_2 seriam identificados como possíveis

de sofrem impacto. Já ao tomar como base o elemento d_1, somente o_1 seria identificado. Portanto o sentido da seta indica a forma com que os elementos foram criados (originados), que, de uma forma geral, não interfere na identificação dos elementos impactados. Por exemplo, um rastro entre um caso de uso e uma classe pode indicar que a classe (elemento-destino) foi originada partir do caso de uso (elemento-origem). Caso ocorra uma alteração no caso de uso a classe possivelmente deverá sofrer alteração. Já no caso de uma alteração na classe o caso de uso também pode ser impactado. Este recurso é utilizado para auxiliar a manutenção da consistência das informações entre os elementos nos artefatos de software.

A seguir encontram-se descritos os principais conceitos da área de rastreabilidade: artefato, elemento e rastro de rastreabilidade.

Artefato é um conjunto de informações produzido durante o processo de desenvolvimento de software, podendo ser um diagrama de requisitos, de casos de uso, de classes, de seqüência, etc. Um artefato a é definido como sendo um conjunto de elementos. Um sistema de software pode então ser modelado por um conjunto de artefatos $A = \{a_1, a_2, ... a_n\}$ cada qual contendo um conjunto de elementos, ou seja, $A = \{ \{e_{1,1}, e_{1,2}, ...\}, \{e_{2,1}, e_{2,2}, ...\}, ... \{e_{n,1}, e_{n,2}, ...\} \}$. A Figura 2 mostra a representação gráfica de alguns artefatos de software.

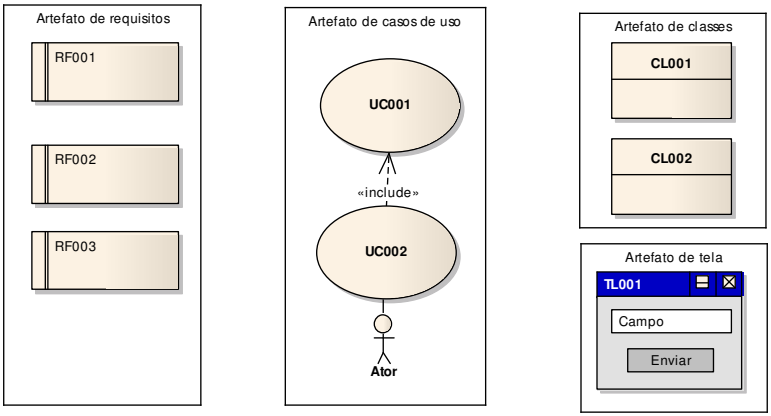

Figura 2 – Exemplos de artefatos de software.

Elemento é uma unidade de informação que compõe um artefato, por exemplo, um dos requisitos inseridos no diagrama de requisitos ou uma classe de um diagrama de classes. O universo de todos os

elementos possíveis é denotado por *E*, já um elemento específico é denotado por *e*.

As eventuais associações entre elementos de um artefato (composição, generalização, associação simples, etc.) são também consideradas elementos dos artefatos. Faz-se exceção apenas aos rastros, definidos a seguir, que são considerados externos aos artefatos, não sendo, portanto, elementos destes.

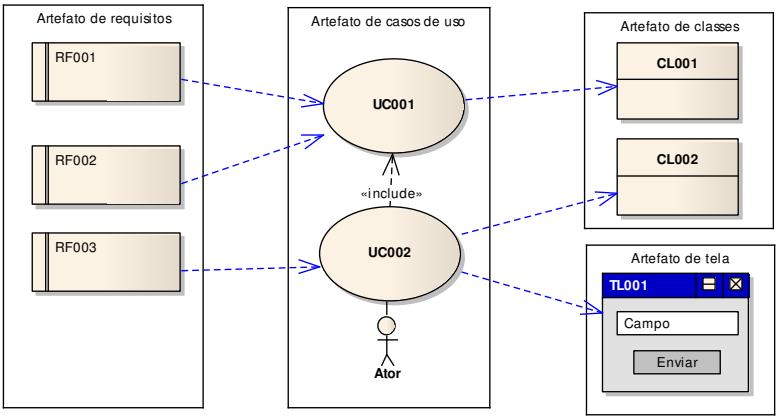

Figura 3 – Exemplo do mapeamento de rastros entre elementos.

Rastro é uma associação explícita entre elementos de artefatos, a Figura 3 apresenta um exemplo de rastros entre elementos. A finalidade principal do mapeamento de rastros é possibilitar que a consistência entre os elementos seja mantida ao longo do projeto. Por exemplo, ao realizar uma mudança no UC001 deve-se levar em consideração que o elemento que está associado a ele por meio do rastro, CL001, possivelmente também deva ser alterado. A existência deste relacionamento facilita a localização dos elementos afetados, sem ele há uma grande chance de que a devida alteração não seja realizada, tornando os elementos inconsistentes entre si.

O rastro se dá entre elementos mesmo que um elemento esteja presente em um ou mais artefatos, neste caso seus rastros são independentes do artefato onde ele apareça.

2.2 MATRIZ DE RASTREABILIDADE

A matriz de rastreabilidade geralmente é utilizada para representar os relacionamentos entre requisitos e demais artefatos de software. Em sua forma mais simples ela se manifesta em tabelas com linhas e colunas, nas quais os elementos de um projeto são relacionados aos requisitos que os satisfazem (ALMEIDA, ECK e IACOB, 2006).

Os rastros são, geralmente, estabelecidos manualmente pelo relacionamento explícito entre dois artefatos (FLETCHER e CLELAND-HUANG, 2006), e esta ainda é a forma como os rastros são criados atualmente nas organizações (CLELAND-HUANG, 2006b). A matriz de rastreabilidade é a técnica mais atendida pelas ferramentas CASE que suportam a rastreabilidade.

De acordo com CLELAND-HUANG, ZEMONT e LUKASIK (2004) a matriz de rastreabilidade assume um tamanho que rapidamente se torna não gerenciável, uma vez que o número de elementos tende a aumentar significativamente, tornando a utilização da matriz complexa. O empenho dos desenvolvedores é retardado pela carência de suporte por parte das ferramentas, o que causa uma percepção de que o custo despendido para manter a matriz de rastreabilidade é muito alto, em comparação aos seus benefícios (CLELAND-HUANG, 2006b).

O custo elevado é proveniente de diferentes causas: (1) dificuldade na criação dos rastros de forma automática ou semi-automática; (2) grande número de artefatos criados durante o processo de desenvolvimento de software e (3) carência de características como corretude e completude dos rastros (CLELAND-HUANG, 2006a).

A Figura 4 apresenta um exemplo de matriz de rastreabilidade, com o relacionamento entre casos de uso e classes de análise.

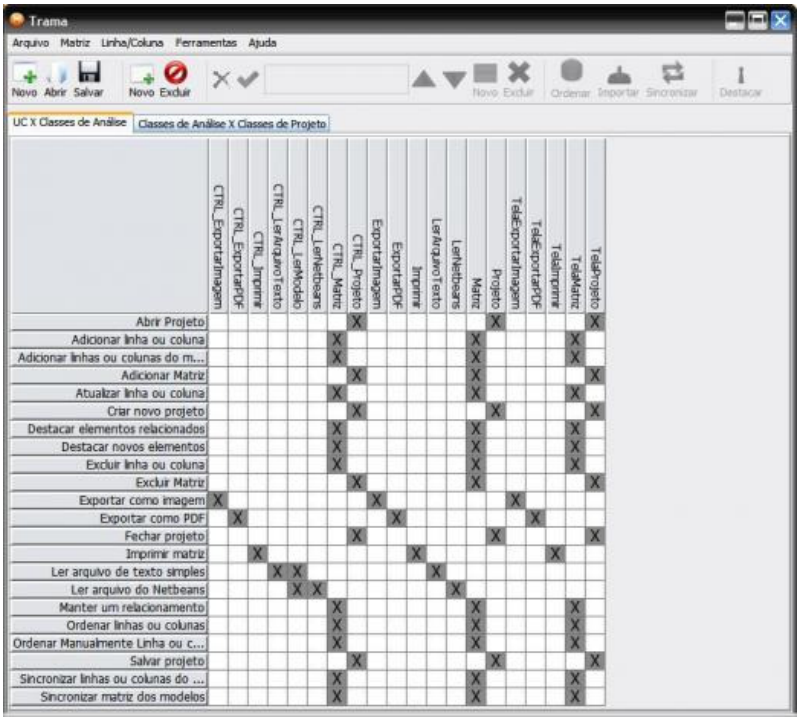

Figura 4- Matriz de rastreabilidade.

Fonte: http://www.testexpert.com.br/files/google%20TRAMA.jpg

2.3 RECUPERAÇÃO AUTOMÁTICA DE RASTREABILIDADE

Recentemente, técnicas de recuperação automática de rastros vêm sendo pesquisadas na tentativa de encontrar uma alternativa para o problema da custosa e complexa definição de rastros (ANTONIOL, CIMITILE e CASAZZA, 2000; EGYED, BIFFL, HEINDL e GRÜNBACHER, 2005; LORMANS e VAN DEURSEN, 2006;).

A recuperação automática procura identificar rastros baseando-se na similaridade entre o texto contido nos artefatos. Um exemplo de técnica de recuperação é a LSI (*Latent Semantic Indexing*) (ANTONIOL

et al, 2002; MARCUS e MALETIC, J. I., 2003; SETTIMI et al, 2004; LUCIA et al, 2007).

No entanto, segundo alguns autores (MALETIC et al, 2005; JIANG et al, 2007; LUCIA et al, 2007), ainda existem muitos desafios que precisam ser superados. Um dos problemas é que as técnicas de recuperação 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 automático torna-se ineficiente (LUCIA et al, 2006b), pois muitos links possíveis deixam de ser detectados e falsos links podem ser detectados. Ressalta-se que estas técnicas de recuperação não são completamente automáticas, pois o usuário deve interagir com o sistema para decidir sobre a aceitação ou rejeição dos rastros recuperados.

Durante a realização das pesquisas para esta dissertação não foram identificadas ferramentas CASE de escala industrial que suportassem a recuperação automática da rastreabilidade, mas apenas ferramentas em trabalhos de pesquisa (Murta et al, 2008; Aldrich et al 2002). De acordo com CLELAND-HUANG (2006b) a recuperação automática permanece apenas como uma solução em potencial para o problema da rastreabilidade.

Técnicas de recuperação automática não recuperam todos os relacionamentos corretos sem também recuperar muitos falsos, o que acarreta trabalho extra para o desenvolvedor no descarte destes relacionamentos (SETTIMI et all, 2004). A tarefa de descarte pode ser muito trabalhosa, pois o desenvolvedor pode gastar mais tempo para descartar rastros falsos do que rastreando os corretos. Ainda que seja possível melhorar o desempenho destas técnicas elas estão longe de ser uma solução viável para o problema (LUCIA et al, 2006a).

No estudo realizado por LUCIA et al (2006a) foram avaliados relacionamentos entre casos de uso e classes, com o intuito de capturar todos os relacionamentos corretos, foi preciso avaliar 1013 relacionamentos, sendo que somente 93 eram corretos.

Outro problema para as técnicas de recuperação automática está nas diferentes formas de escrita entre os artefatos, ou diferentes níveis de abstração. Por exemplo, para um sistema de controle bancário é criado requisito “O sistema deve controlar o limite de crédito da conta bancária”, a classe correspondente é escrita de forma resumida “ControleLimCredito”, estas diferenças na forma de escrita dificultam o cálculo automatizado da similaridade entre os mesmos.

Diferentes trabalhos sobre gerenciamento de rastros entre artefatos de software foram realizados. Um dos pontos em que estes

trabalhos diferem refere-se aos tipos de artefatos entre os quais a rastreabilidade é aplicada. Os trabalhos de MARCUS, MALETIC (2003) e ANTONIOL et al (2002) aplicam a rastreabilidade entre o código fonte e um texto livre utilizando as técnicas LSI (Latent Semantic Indexing), que recupera as rastros baseando-se na similaridade entre o texto contido nos artefatos de software. Nestes trabalhos o LSI é utilizado para encontrar a similaridade entre o código fonte e um texto livre, como especificação de requisitos, manual usuário, log de erros, ou relatório de ajustes de manutenção.

A criação de rastreabilidade entre a arquitetura e o código fonte (Aldrich et al 2002; Murta et al, 2008), é realizada através da ferramenta ArchTrace, que atualiza continuamente os rastros entre o código fonte e a arquitetura e também realiza controle de versões. ArchTrace trabalha apenas com dois artefatos: arquitetura e código-fonte.

O mapeamento de rastreabilidade também pode ser feito entre vários artefatos de software (SETTIMI et all, 2004; LUCIA et al., 2007) tais como: requisito funcional, diagrama de classes, diagrama de seqüência, caso de uso e código fonte, utilizando também a técnica LSI.

Pesquisas recentes apontam que há um ganho significativo de produtividade com a utilização da técnica LSI. No entanto, segundo alguns autores (JIANG et al, 2007; LUCIA et al, 2007; MALETIC et al, 2005), ainda existem muitos desafios que precisam ser superados. Um problema ainda não solucionado é a detecção de um grande número de links falsos (falso positivo), necessitando da intervenção humana na validação.

O capítulo seguinte aborda detalhes das características da técnica de rastreabilidade proposta nesta dissertação, a técnica indutiva.

3 RASTREABILIDADE INDUTIVA

Este capítulo apresenta detalhes sobre o funcionamento da técnica indutiva, seus conceitos fundamentais, e uma classificação das operações que ao serem executadas sobre elementos de artefatos afetam os rastros de rastreabilidade.

3.1 TÉCNICA INDUTIVA

O processo de desenvolvimento de software inclui a criação de artefatos e seus elementos nos diferentes graus de abstração. Então, na prática, o processo de desenvolvimento, do ponto de vista das atividades de um desenvolvedor, pode ser entendido como uma seqüência de ações que visam criar elementos nos diferentes artefatos. A proposta é que estas ações sejam realizadas de forma mais produtiva com a utilização da técnica indutiva.

A palavra "indutiva" é utilizada nesta dissertação com a finalidade de indicar o ato de induzir ou criar um elemento a partir de outro. Segundo FERREIRA (2008), induzir também pode significar inferir, causar, inspirar. Portanto, um novo elemento a ser criado utilizando a técnica indutiva deve, preferencialmente, ser inferido a partir de um elemento já existe no projeto, desta forma, além de criar um novo elemento, também o rastro é criado.

Diferentemente da técnica tradicional (matriz de rastreabilidade), na técnica indutiva a criação dos rastros está inserida no processo de construção dos elementos nos artefatos, ou seja, os rastros são criados como uma conseqüência da criação dos elementos nos diferentes artefatos e não como uma atividade extra. A criação de um novo elemento, sempre que for o caso, deve ser feita a partir de outro elemento, que consiste em sua origem. A este ato dá-se o nome de *derivação*.

A técnica é chamada semi-automática, pois há a interação do desenvolvedor durante o processo de criação de novos elementos e para

indicar a origem dos elementos. Mesmo as técnicas de recuperação automática não são totalmente automatizadas, pois faz-se necessário que o desenvolvedor realize o descarte dos links falsos, encontrados no processo de recuperação.

O processo de funcionamento da técnica indutiva constitui-se basicamente de dois sub-processos: (1) a criação de elementos base e (2) a criação de elementos derivados.

O elemento base é criado de forma individual no projeto, ou seja, sem a explicitação de rastros com outros elementos, uma vez que no momento de sua concepção não se tem a informação de quais são os elementos relacionados. Por exemplo, um elemento de requisito é criado de forma individual no artefato de requisitos, pois sua origem é externa, foi originado a partir da especificação repassada em reuniões com cliente e não há nenhum elemento registrado no projeto que esteja relaciondo ao requisito recém criado no projeto.

Somente é possível criar elementos derivados depois que existir pelo menos um elemento base. Fora esta restrição, elementos derivados ou elementos base podem ser criados sempre que necessário em qualquer ordem e combinação de acordo com os ditames do processo de modelagem utilizado pelo desenvolver.

A derivação pressupõe que a inserção de novos elementos nos artefatos não consiste simplesmente em criar um novo elemento no diagrama, mas em uma ação que, em diversas situações, tem uma origem bem definida a partir de algum outro elemento. Por exemplo, uma classe pode estar sendo inserida no modelo conceitual devido à existência de um caso de uso que a menciona. Ou ainda, um caso de uso pode estar sendo inserido no diagrama em função de um ou mais requisitos que lhe dão origem.

Assim a criação de novos elementos acontece de forma indutiva, com exceção dos elementos base, a criação de um novo elemento deve sempre acontecer a partir de outro, que consiste em sua origem. Desta forma a criação dos rastros acontece de forma automatizada, como uma conseqüência do processo de criação de novos elementos.

Com a finalidade de atender as diferentes necessidades dos projetos de software a técnica proposta atende os dois tipos de granularidade, alta e baixa. A granularidade do elemento é dinâmica e pode ser definida pelo desenvolvedor, em uma granularidade baixa o elemento pode ser uma palavra ou fragmento da descrição de um caso de uso, já em uma granularidade alta o elemento pode ser complexo como uma classe, um caso de uso ou uma associação.

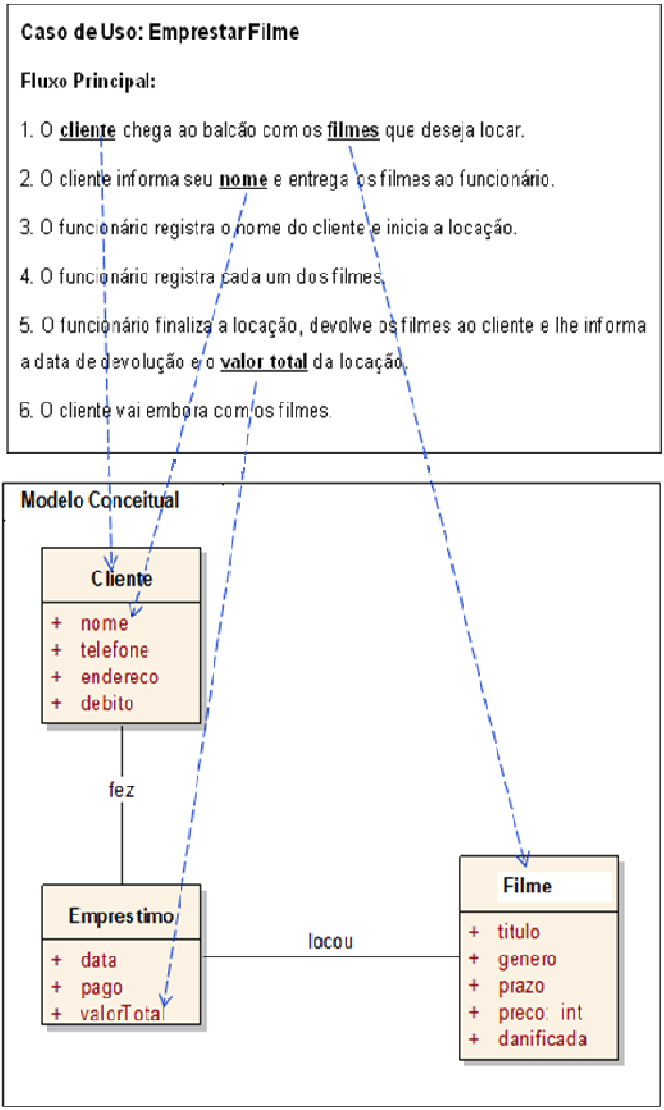

Figura 5 –Rastros entre elementos de granularidade baixa.

A Figura 5 apresenta um exemplo do mapeamento de rastros entre elementos de granularidade baixa (ou fina), na qual se observa que a partir de um caso de uso expandido foi dada origem a especificação de classes no modelo conceitual. A palavra cliente, no primeiro passo do

caso de uso, deu origem a classe Cliente no modelo conceitual, também a palavra nome no segundo passo do caso de uso deu origem ao atributo nome no modelo conceitual.

Na Figura 6 é apresentado um exemplo do mapeamento de rastros entre elementos de granularidade alta, na qual a partir de casos de uso de alto nível foram derivadas classes no artefato de classes. O caso de uso Emprestar Filme deu origem às classes Emprestimo, Cliente e Filme, já o caso de uso Reservar Filme deu origem às classes Cliente e Filme. Para a granularidade alta considera-se apenas o nome do caso de uso, já a expansão em fluxo principal e alternativos não faz parte desta e sim da granularidade baixa.

Outra característica prevista na técnica indutiva é multiplicidade de rastros entre elementos, um determinado elemento pode possuir diversos rastros, desde que sejam para elementos diferentes, ou seja, considerando inicialmente apenas dois elementos, estes podem possuir apenas um rasto de ligação entre os eles, mas cada um pode estar associado a vários outros, caracterizando um relacionamentos 1 x n (um para n). A Figura 6 apresenta um exemplo de múltiplos relacionamentos entre elementos, pode-se observar que o caso de uso Emprestar Filme possui rastro para três outros elementos, as classes Emprestimo, Cliente e Filme, porém há apenas um relacionamentos com cada um deles (1 x n).

Observa-se que a classe Cliente foi originada a partir de dois casos de uso Emprestar Filme e Reservar Filme, a técnica permite que dois ou mais elementos sejam definidos ao mesmo tempo como origem de outro elemento, bastando aplicar a operação de derivação somente uma vez.

A técnica não obriga o desenvolvedor a utilizar a derivação, mas se ele a usar os rastros serão criados de forma automatizada pela ferramenta CASE.

No trabalho desenvolvido por CARNEIRO, SANT´ANNA e MENDONÇA (2010) é apresentada uma ferramenta que disponibiliza múltiplas perfectivas na visualização de artefatos de software, proporcionando um auxílio importante na manutenção de software, porém este trabalho é focado na visualização dos rastros e não na criação dos mesmos. Este recurso pode ser utilizado, por exemplo, após a criação dos rastros com a utilização da técnica indutiva, como auxílio para a manutenção dos elementos nos artefatos de software.

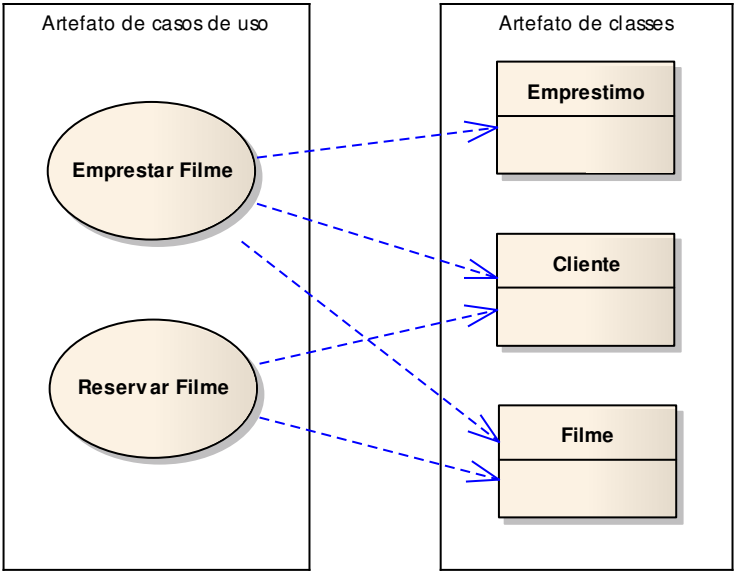

Figura 6 - Rastros entre elementos de granularidade alta.

3.2. OPERAÇÕES QUE AFETAM OS RASTROS

Nesta seção são apresentadas as operações que ao serem aplicadas durante o processo de desenvolvimento de software afetam os rastros. Aqui serão classificados diferentes tipos de operações sobre elementos nos artefatos e suas conseqüências para os rastros. O objetivo deste conjunto de operações é permitir que um sistema automatizado de criação de rastros tenha parâmetros que possam ser seguidos para manter esses rastros consistentes ao longo do tempo.

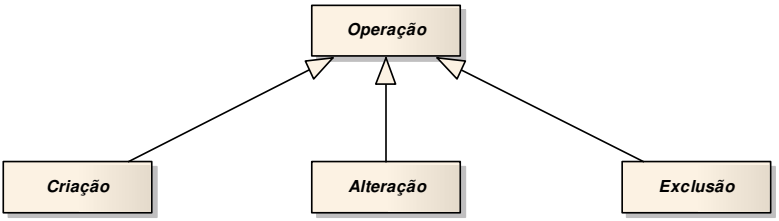

Figura 7- Operações elementares que afetam as rastros

Inicialmente as operações poderiam ser classificadas em três tipos elementares: *criação*, *alteração* e *exclusão* de elemento ou rastro, estas são apresentadas na Figura 7. Porém, em função da intenção do desenvolvedor, ainda pode-se sub-classificar essas operações, conforme poder ser visto na Figura 8.

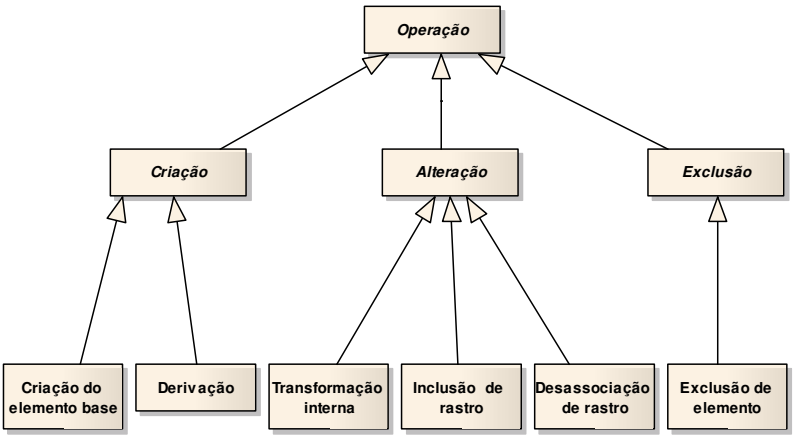

Figura 8- Operações sobre elementos de artefatos que afetam as rastros.

Primeiramente, a criação pode ser de dois subtipos: *criação de elemento-base* e *derivação*. A derivação consiste em criar um elemento a partir de outro, criando também os rastros de rastreabilidade entre os mesmos.

Existem duas maneiras de se modificar um elemento: transformação interna e a modificação de seus rastros, as quais podem ser incluídos ou desassociados.

A exclusão de um elemento é executada por meio da operação de *exclusão de elemento*, que além de destruir o elemento elimina também seus rastros.

As folhas da árvore representadas na Figura 8 são as operações que serão consideradas para fim de estudo neste trabalho. As demais operações são abstratas e aparecem no diagrama unicamente para indicar sua classificação.

A fim de demonstrar o efeito da execução dessas seis operações sobre os elementos e rastros, define-se um cenário inicial constituído por um conjunto de seis elementos e cinco rastros, conforme pode ser visto na Figura 9. As operações apresentadas nas subseções seguintes e seus efeitos sobre os rastros são demonstradas a partir deste cenário inicial.

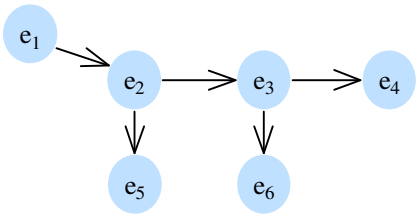

Figura 9- Cenário inicial de elementos e rastros.

Neste diagrama não são representados os artefatos aos quais os elementos pertencem porque esta informação não é necessária para a compreensão das operações em si.

As operações apresentadas nas subseções seguintes cobrem todas as possibilidades de operações sobre elementos, visto que são oriundas de subdivisões a partir das três operações fundamentais: criar, modificar e excluir (Figura 8). Possivelmente outras subdivisões ou combinações de operações ainda são possíveis, mas para fins deste trabalho, esta classificação já é suficiente, pois permite compreender a operação de derivação e suas congêneres no contexto das atividades possíveis sobre artefatos.

3.2.1 Criação de Elemento-Base

A operação de criação de elemento-base consiste em inserir um elemento em um artefato sem nenhuma origem que possa ser encontrada em outros elementos; sua causa é externa. Neste caso, conforme mostra a Figura 10, nenhum novo rastro é criado.

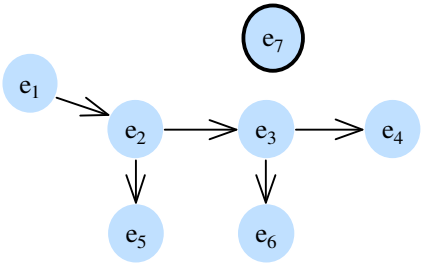

Figura 10- Criação de um elemento-base (e_7) no cenário inicial da Figura 9.

Exemplo: um requisito é criado no diagrama de requisitos a partir de entrevistas que o desenvolvedor realizou com o cliente. Nenhum elemento até o momento referenciava esse requisito ou suas conseqüências. Então ele pode ser inserido como um elemento-base.

3.2.2 Derivação

Quando um elemento é criado em função de outro, seja no mesmo artefato ou em artefatos diferentes, o elemento original é denominado *elemento-origem* e o elemento originado *elemento-destino*. É criado um rastro de rastreabilidade entre o elemento-origem e o elemento-destino no sentido do elemento-origem para o elemento-destino (Figura 11). Podem existir um ou mais elementos-origem para um mesmo elemento-destino.

Esta operação é uma das mais promissoras, é a única que executa duas ações em uma única operação, sendo elas: a criação de um novo elemento e a criação do rastro de associação com o elemento-origem.

Exemplo: um caso de uso (elemento-destino) é criado em função de um requisito funcional (elemento-origem). Um rastro é criado automaticamente entre o requisito e o caso de uso.

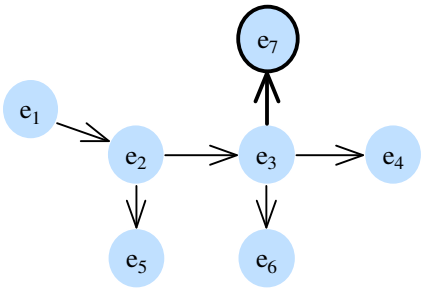

Figura 11- Derivação de um elemento-destino (e_7) a partir de um elemento-origem (e_3) no cenário inicial da Figura 9.

3.2.3 Transformação Interna

Um elemento qualquer pode ser modificado em seu conteúdo interno, ou seja, transformado. Este caso é particularmente interessante porque a modificação interna do elemento poderá, dependendo de seu teor, provocar alterações nos rastros. Mais ainda, a modificação interna de um elemento faz com que outros elementos que têm rastro de rastreabilidade de ou para ele tenham que ser revisados também, para que a informação nos artefatos seja mantida consistente.

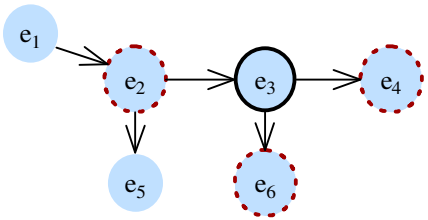

Figura 12- Transformação interna de um elemento (e_3). Os elementos tracejados precisam ser revisados.

Na Figura 12 é mostrada a transformação interna de um elemento (e_3) com associação para e_4 e e_6 e associação a partir de e_2. Os efeitos da transformação interna de e_3 nos rastros dependem de sua natureza. Os rastros existentes exigirão que os elementos e_2, e_4 e e_6 sejam revisados para que se verifique se continuam consistentes com a modificação realizada em e_3. O processo de revisão é recursivo. Ou seja, se, em função da revisão em e_3 também e_2 for modificado para manter consistência, então e_5 e e_1 também precisarão ser revisados.

O sentido da seta indica quem é o elemento-origem e quem é o elemento-destino do rastro. Este não possui interferência na identificação dos elementos afetados devido a uma alteração, por esta razão que ao modificar e_3 tanto e_2 quanto e_4 devem ser revisados.

Exemplo: um requisito como "O sistema deve permitir o cadastro de turmas" poderia, em algum momento, ser revisado e reescrito como "O sistema deve permitir o cadastro de turmas somente quando houver um professor disponível para ministrar disciplina para a mesma". Neste caso, quaisquer artefatos relacionados por rastreabilidade a este requisito, como um caso de uso ou uma classe, deveriam ser revistos para verificar se atendem a esta versão transformada do requisito.

3.2.4 Inclusão de Rastro

Dois elementos que já existam nos artefatos podem ser identificados como elemento-origem e elemento-destino independentemente de que o elemento-destino tenha sido criado pelo processo de derivação.

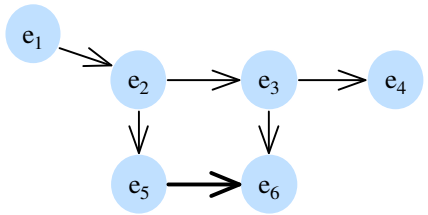

Figura 13- Criação de rastro entre um elemento-destino (e_6) e um elemento-origem pré-existente (e_5) no cenário inicial da Figura 9.

O estabelecimento do rastro de rastreabilidade entre dois elementos existentes é, portanto, uma operação possível. Isso pode acontecer tanto motivado por uma falha de observação do desenvolvedor que poderia criar um suposto elemento-base que já tivesse um elemento-origem em algum artefato, quanto pela possibilidade de haver mais de um elemento-origem para um mesmo elemento-destino (Figura 13).

Exemplo: Um requisito foi identificado como elemento-origem de uma classe. Posteriormente, percebe-se que outro requisito também contribui para a definição dessa classe, sendo também elemento-origem dela. Assim, a classe em questão terá dois elementos-origem. Um efetivamente originou a criação da classe, o outro foi posteriormente relacionado a ela. Mas para efeito do rastro de rastreabilidade, não há distinção de ordem ou importância entre os dois elementos-origem.

3.2.5 Desassociação de Rastro

A operação de destruição de rastro de rastreabilidade consiste simplesmente em remover um rastro entre dois elementos (Figura 14). É possível que seja efetuada em função de um rastro criado por engano ou ainda devido a alguma transformação interna dos elementos, que faça com que o rastro em questão não tenha mais sentido.

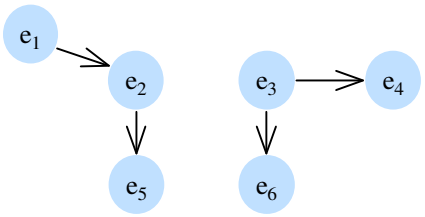

Figura 14- O rastro de rastreabilidade entre e_2 e e_3 é removido.

Exemplo: quando um caso de uso é revisado e se percebe que um requisito não está mais relacionado com ele, pode-se, então, eliminar o rastro entre os elementos.

3.2.6 Exclusão de Elemento

Quando um elemento é simplesmente removido ou destruído, deve-se revisar todos os elementos ligados a ele por rastreabilidade, seja para cima, seja para baixo. Para cima porque a remoção do elemento pode ter sido originada por uma decisão que afeta os elementos de origem. A revisão para baixo deve ser feita especialmente no caso de elementos que ficam órfãos, e também nos demais elementos originalmente associados ao elemento destruído, porque podem ter sido afetados pela eliminação deste.

Deve haver certo cuidado porque a destruição de um elemento pode significar a efetiva eliminação de alguma característica do sistema por algum motivo. Não se trata de remover uma versão de um elemento para substituir por outra versão, ou remover um elemento para incluir suas características em outro, o que corresponderia a transformação interna.

Na Figura 15, a destruição de e_3 faz com que e_4 e e_6 fiquem órfãos e por isso eles precisam ser revisados. O elemento e_2 também precisa ser revisado, pois este elemento deu origem a e_3. Se houve motivo para destruir e_3 é possível que e_2 necessite de uma transformação interna para refletir o fato de que e_3 foi eliminado.

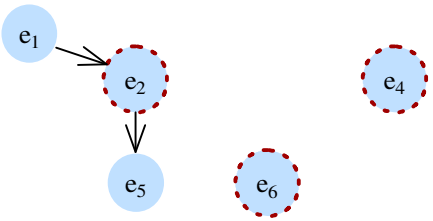

Figura 15- O elemento e_3 é destruído juntamente com suas rastros.

Exemplo: um requisito é removido do diagrama, pois não faz mais parte do escopo do projeto. Quaisquer elementos cuja criação foi devida a este requisito devem ser revisados. Possivelmente alguns deles, especialmente os órfãos, como e_4 e e_6 na Figura 15 poderão ser definitivamente removidos também.

4 PLUGIN PARA A TÉCNICA INDUTIVA

A fim de permitir a utilização prática da técnica indutiva foi desenvolvido um *plugin*[2] de rastreabilidade para a ferramenta CASE Enterprise Architect™ (EA) (SPARX SYSTEMS, 2009). Esta ferramenta foi escolhida por possuir uma ampla documentação no que se refere ao desenvolvimento de plugins, a serem utilizados junto à mesma.

A seguir serão apresentados detalhes do plugin, também será demonstrada utilização o EA a fim de cumprir cada uma das atividades do processo de funcionamento do *plugin*.

4.1 PROCESSO DE FUNCIONAMENTO DO *PLUGIN*

O processo de funcionamento do *plugin* na ferramenta CASE EA pode ser visualizado na Figura 16, ele é constituído por dois sub-processos: (1) a criação de elementos base e (2) a criação de elementos derivados. Como já dito anteriormente, somente é possível criar elementos derivados depois que existir pelo menos um elemento base. Fora esta restrição, elementos derivados ou elementos base podem ser criados sempre que necessário em qualquer ordem e combinação de acordo com o processo de modelagem utilizado.

[2] O plugin foi implementado por Diogo D. Fonseca dos Santos, aluno do curso de graduação em Ciência da Computação, como parte de seu trabalho de conclusão de curso.

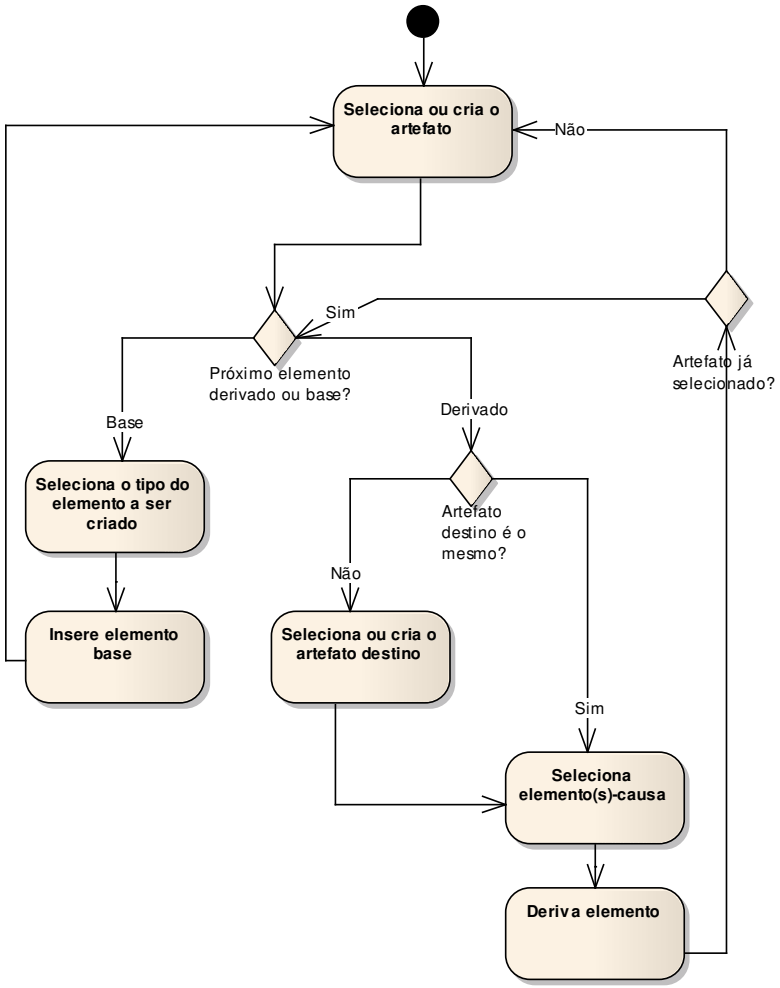

Figura 16- Processo de funcionamento do *plugin* (diagrama de atividade padrão UML).

O processo de funcionamento do *plugin* apresentado na Figura 16 mostra a criação de elementos base e derivados nos diferentes artefatos, este foi desenhado de forma simplificada para detalhar apenas a criação dos elementos e artefatos.

Para demonstrar a utilização do *plugin* são criados requisitos, casos de uso e classes, utilizado o domínio de negócios de um hotel. Foi

criado um novo projeto no EA, a sua estrutura de pacotes pode ser observada na Figura 17.

Cada uma das atividades do processo será demonstrada através da utilização do *plugin*. A primeira atividade a ser realizada é selecionar o artefato no qual se deseja trabalhar, ou ainda se este ainda não tiver sido criado deve ser realizada a criação do mesmo. Na Figura 13 o artefato de requisitos foi selecionado.

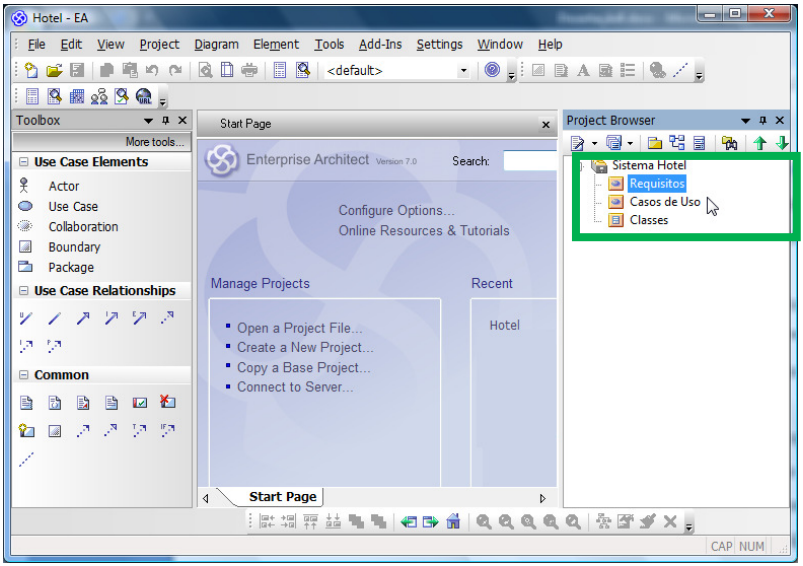

Figura 17- Estrutura de pacotes do projeto de hotel.

Uma vez que o artefato onde se deseja trabalhar foi selecionado é necessário definir qual é o próximo elemento que dever ser criado, se base ou derivado. Para criar um elemento base deve-se selecionar o elemento na toolbox da ferramenta em questão e posteriormente inserir o mesmo no diagrama. A Figura 18 apresenta dois elementos base já criados, são os requisitos: “RF001- O sistema deve permitir a reserva de quartos” e “RF002- O sistema deve permitir a realização de checkin”.

Os elementos base geralmente são criados no mesmo artefato de seu tipo, por exemplo, elementos de requisitos são criados no artefato de requisitos, elementos de caso de uso são criados no artefato de caso de uso. Porém o elemento base pode ser criado em qualquer artefato, independente de seu tipo.

Os elementos base devem ser definidos, pois darão origem a todos, ou quase todos, elementos nos diferentes artefatos da documentação do sistema. Por exemplo, os requisitos poderiam ser os elementos base. A partir destes são derivados casos de usos e protótipos, dos quais são derivadas classes de análise, das quais são derivados diagramas de seqüência, e assim por diante.

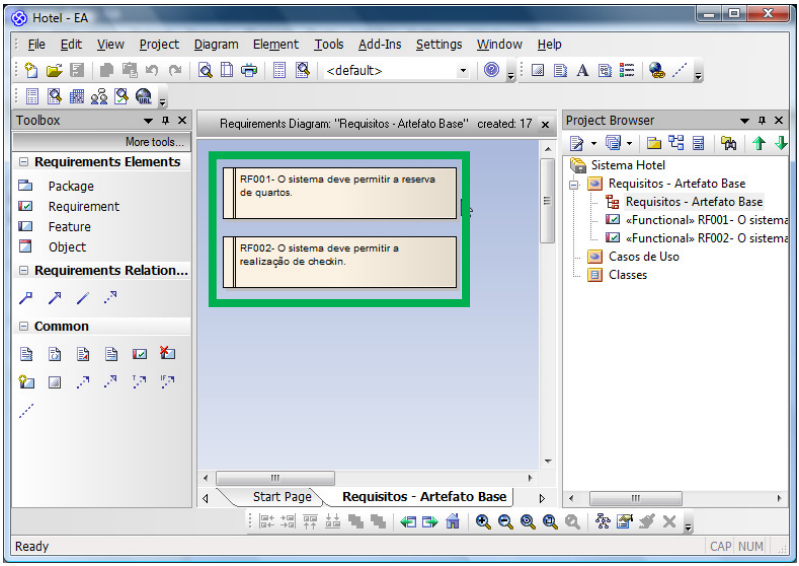

Figura 18- Criação de elementos no artefato base.

Já em relação a criação dos elementos derivados é necessário verificar se o artefato de destino do elemento a ser derivado é o mesmo que o desenvolvedor já está utilizando, caso não seja o artefato atual este deve ser selecionado. Isso só é necessário quando o elemento a ser derivado pertence a um artefato de tipo diferente do elemento de origem do rastro (elemento-origem), por exemplo, derivar uma classe a partir um caso de uso. Para derivar elementos dentro do mesmo artefato, artefato corrente, não é necessário selecionar o artefato de destino.

A seguir será demonstrada a derivação de um caso de uso a partir de um requisito, uma vez que o artefato de destino (caso de uso) é diferente do artefato de origem (requisito) deve-se selecionar o artefato de destino. Na Figura 19 foi selecionado o diagrama de caso de uso como artefato de destino.

A atividade seguinte consiste em selecionar o elemento de origem do rastro de rastreabilidade. Pode-se, inclusive, selecionar dois ou mais elementos como elemento-origem em uma derivação, pois um elemento pode ter várias origens. Por exemplo, selecionar dois requisitos para dar origem a um caso de uso. Na Figura 20 o requisito “RF001- O sistema deve permitir a reserva de quartos” foi selecionado a fim de ser o elemento-origem do elemento a ser derivado.

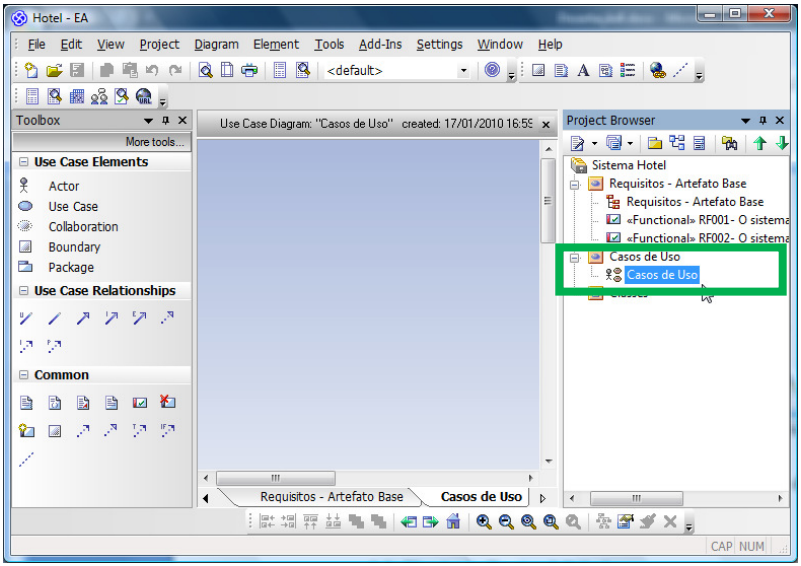

Figura 19- Seleção do artefato de destino.

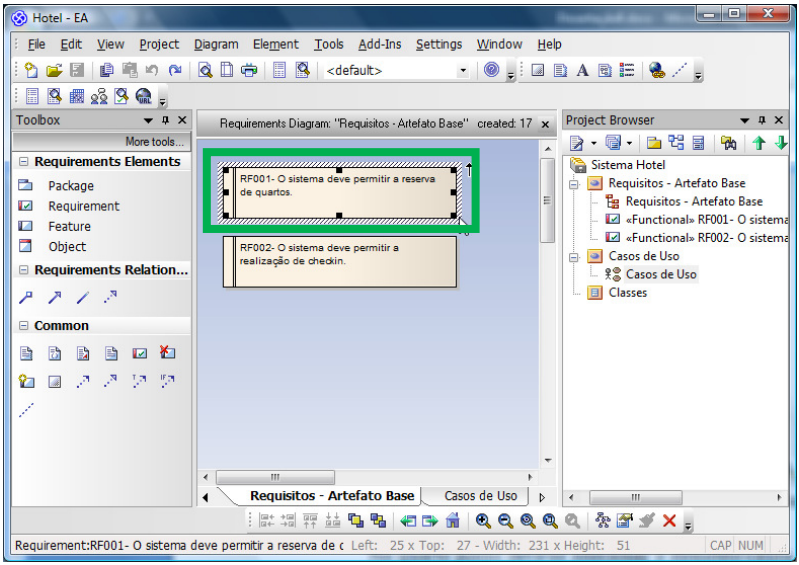

Figura 20- Seleção do elemento-origem.

Para finalizar o processo de derivação deve-se aplicar a derivação sobre o elemento-origem, para dar origem ao elemento-destino. O *plugin* de rastreabilidade cria, juntamente com elemento-destino, o rastro de rastreabilidade.

Para realizar esta derivação basta clicar com o botão direito sobre o elemento-origem (RF001) e selecionar a opção "Derivar Caso de Uso", conforme apresentado na Figura 21.

Pode-se notar que o sistema destacou a opção "[> Derivar Caso de Uso <]" com a utilização dos caracteres "[> <]", isto acontece, pois o sistema identifica o tipo de artefato selecionado como destino da operação de rastreabilidade, na atividade de seleção do artefato de destino foi selecionado o diagrama de caso de uso, o *plugin* destaca a opção como forma de auxílio para o usuário. Caso o artefato de destino fosse uma classe o sistema destacaria a opção "[> Derivar Classe <]".

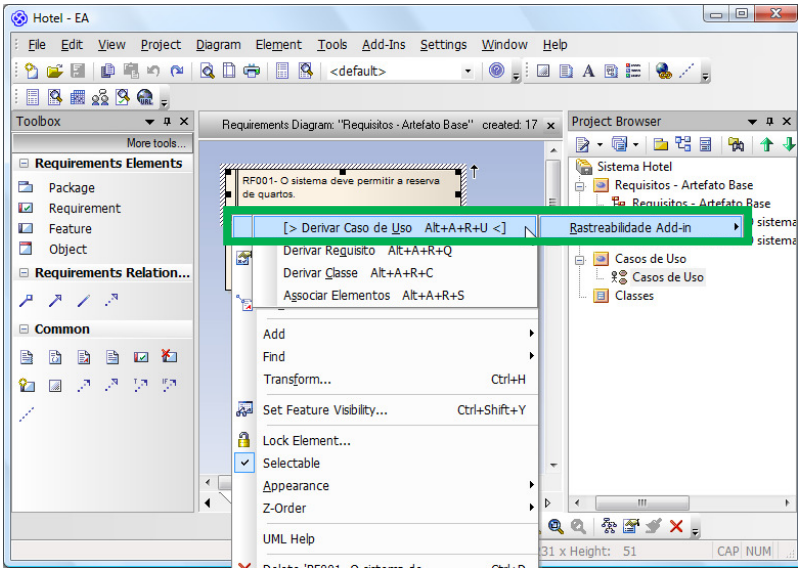

Figura 21- Derivação do elemento-destino.

Assim que a operação de derivação é acionada pelo usuário o sistema automaticamente abre o artefato de destino, cria o elemento derivado e apresenta o mesmo no artefato, neste exemplo foi criado um caso de uso no artefato de caso de uso, como pode ser visto na Figura 22. Juntamente com o novo elemento é criado o rastro de rastreabilidade.

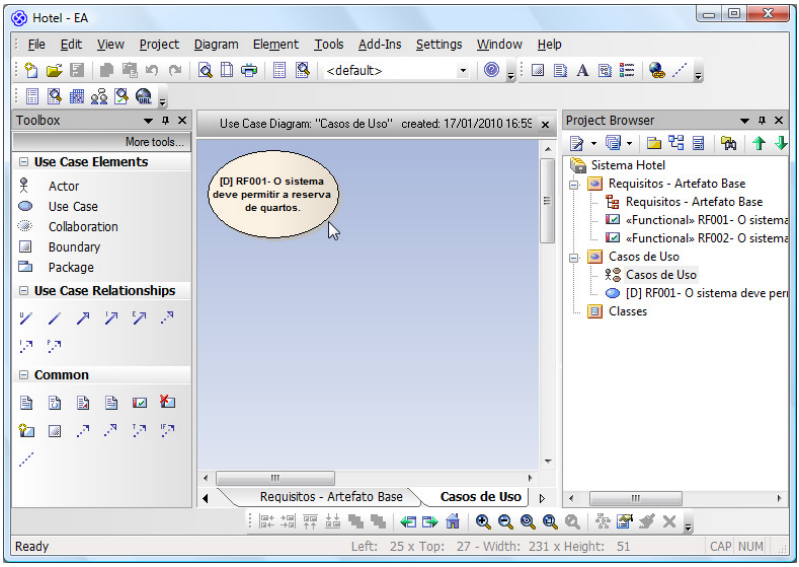

Figura 22- Resultado da operação de derivação.

A técnica não obriga o desenvolvedor a utilizar a derivação, mas se ele a usar os rastros serão criados de forma automatizada pela ferramenta CASE.

No processo indutivo apresentado na Figura 16 não há nodo final, pois não existe necessariamente um último passo na sua execução. Isso pode ser entendido pelo fato de que os elementos e artefatos construídos estão sempre abertos para receber mudanças, especialmente no processo de manutenção ou evolução do software.

4.2 CARACTERÍSTICAS ADICIONAIS DO PLUGIN

Esta seção possui a descrição de características adicionais ao processo de funcionamento do *plugin*, tais como a identificação de elementos derivados e as diferentes formas de visualização dos rastros no EA.

O início do nome do elemento possui os caracteres “[D]” para indicar que se trata de um elemento que foi criado a partir da derivação de outro elemento. Como forma de auxílio o novo elemento é criado

com mesmo nome de seu elemento-origem, a partir deste momento pode-se alterar este nome, como o exemplo da Figura 23.

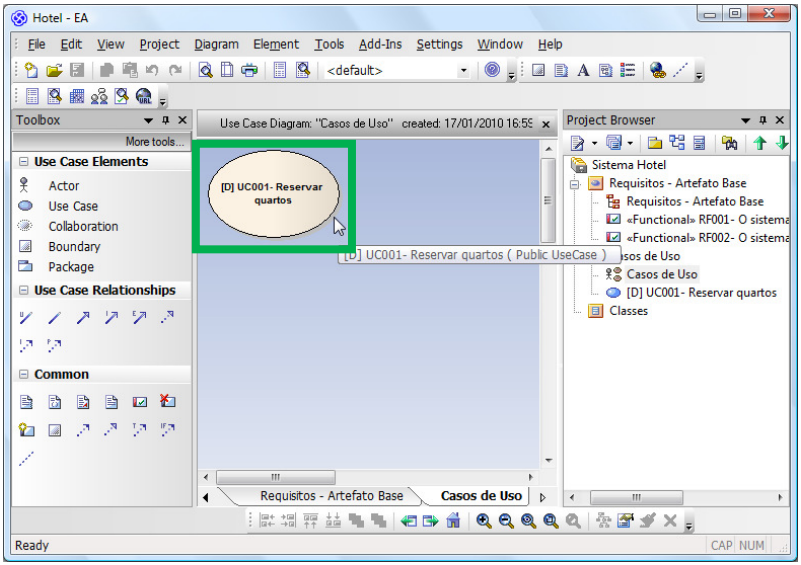

Figura 23- Alteração do nome do elemento-destino.

A relação de rastreabilidade, criada automaticamente entre o elemento-origem e o elemento-destino, pode ser visualizada principalmente de duas formas: por meio da funcionalidade *Hierarchy* e colocando os elementos relacionados por rastreabilidade no mesmo diagrama.

Ao acionar a funcionalidade *Hierarchy* esta é apresentada na parte inferior da tela, conforme a Figura 24. Ela apresenta de forma hierárquica as dependências entre os elementos. Ao selecionar o caso de uso recém criado "UC001- Reservar quartos" é apresentado na área *Hierarchy* o requisito "RF001- O sistema deve permitir a reserva de quartos", indicando que os elementos encontram-se associados por rastreabilidade (Figura 24). Esta é uma forma simples e rápida de ter acesso às rastros.

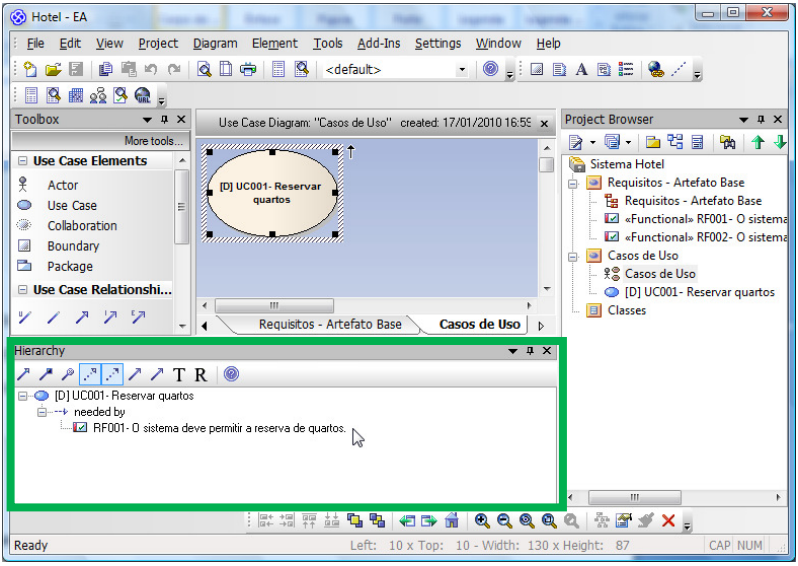

Figura 24- Visualização da relação de rastreabilidade - funcionalidade *Hierarchy*.

Outra forma de visualizar as rastros é colocando os elementos origem e destino no mesmo artefato. Ao arrastar o requisito RF001 para o diagrama de caso de uso o EA apresenta a relação de forma visual (**Erro! Fonte de referência não encontrada.**), com uma linha indicando a relação de dependência entre os dois elementos.

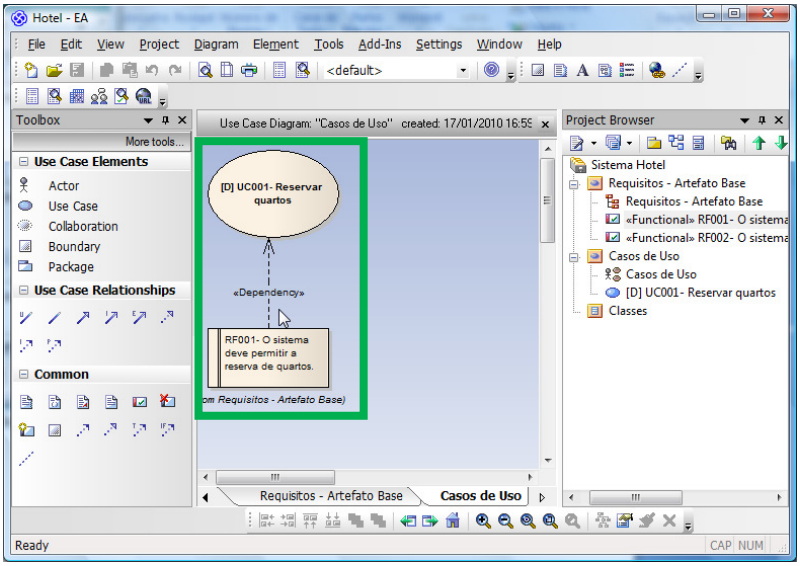

Figura 25- Visualização de relação de rastreabilidade - elementos origem e destino no mesmo artefato.

A derivação apresentada anteriormente foi realizada a partir de apenas um elemento-origem, é possível realizar a derivação a partir de dois ou mais elementos, para demonstrar esta funcionalidade foi realizada a derivação de mais um caso de uso, "UC002- Realizar checkin". Também foi realizada a derivação de uma classe denominada "Quarto" a partir dos casos de uso UC001 e UC002. O resultado final é apresentado na Figura 26, na qual é possível observar a classe derivada e os elementos associados por rastreabilidade, apresentados na área *Hierarchy*. Pode-se notar que a classe recém derivada está associada diretamente aos casos de uso UC001 e UC002 ao mesmo tempo. Também é possível observar quais são os requisitos que deram origem aos casos de uso. Esta funcionalidade é uma importante aliada na análise de toda hierarquia das rastros, tendo como base um determinado elemento.

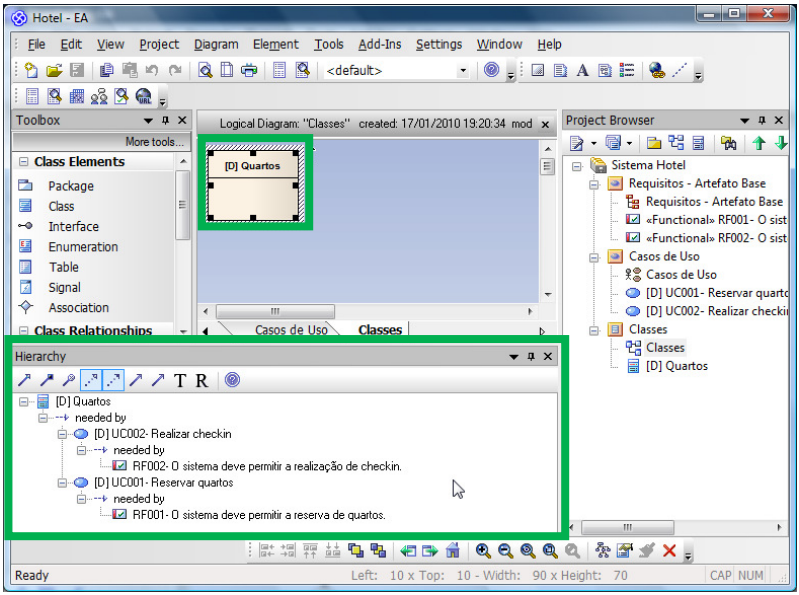

Figura 26- Resultado da derivação a partir de dois elementos.

Outra funcionalidade importante existente no *plugin* é a associação de elementos. Para demonstrar esta funcionalidade foi criado um novo requisito "RF003- O sistema deve permitir a realização de checkout", a partir deste requisito foi derivado um caso de uso "UC003- Realizar checkout". Este novo caso de uso também deve possuir uma relação de rastreabilidade com a classe "Quartos", derivada anteriormente a partir dos casos de uso UC001 e UC002. Ocorre que não é conveniente derivar novamente esta classe, pois haverá duplicação e as rastros ficarão incorretas. Neste caso é apropriado utilizar a funcionalidade de associação de elementos, para utilizá-la basta selecionar o elemento-destino (Quartos), selecionar o elemento-origem (UC003), clicar com o botão direito e escolher a opção "Associar Elementos", conforme Figura 27, desta forma foi criada uma relação entre o caso de uso UC003 e a classe Quartos.

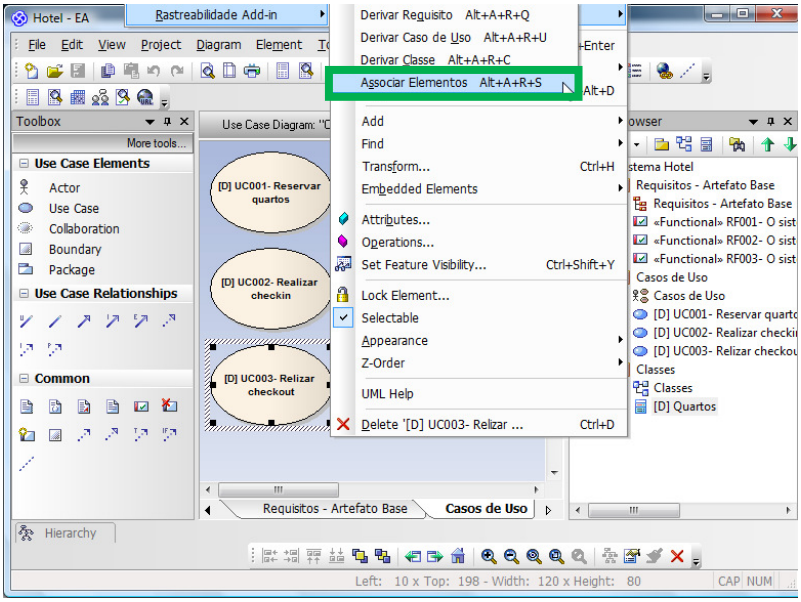

Figura 27- Associação de elementos.

Alternativamente os rastros criados por meio da técnica indutiva podem ser visualizadas na matriz de rastreabilidade existente no EA. A Figura 28 apresenta os rastros existentes entre os requisito e casos de uso criados anteriormente, referentes a um sistema para um hotel. Esta funcionalidade foi prevista na técnica indutiva a fim de proporcionar ao usuário maneira adicional de visualização dos relacionamentos entre os elementos de dois artefatos. Desta forma, além da visão de todos os relacionamentos de um elemento, por meio da funcionalidade *Hierarchy*, o usuário também pode ter a visão de todos os relacionamentos existentes entre dois artefatos, por meio da utilização da matriz.

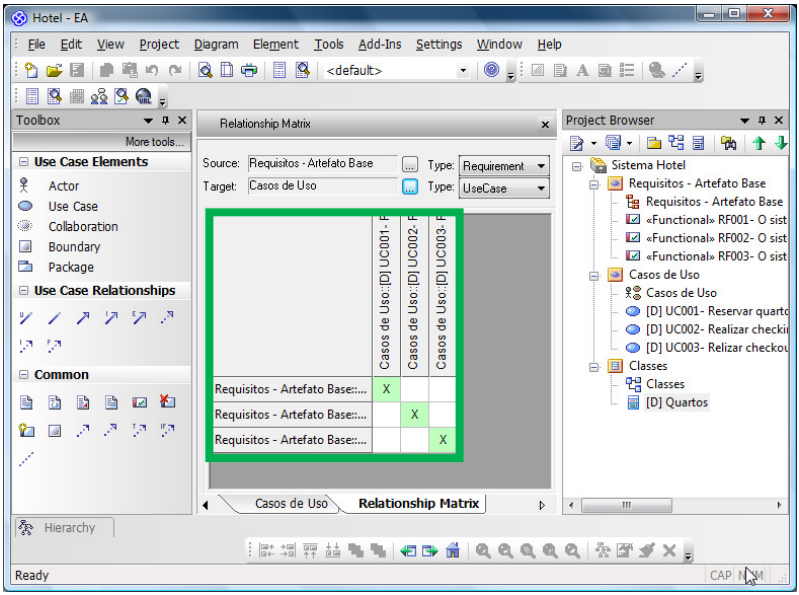

Figura 28- Visualizando os rastros na matriz de rastreabilidade.

O *plugin* de rastreabilidade prevê três formas de subordinação de um elemento: subordinação a um artefato, subordinação a um pacote e subordinação a um elemento.

A subordinação a um artefato já foi apresentada anteriormente (Figura 19 da página 55), onde um artefato de caso de uso foi selecionado como destino. Esta é a principal forma de subordinação, porém também há a possibilidade de utilização das outras duas formas.

Para subordinar um elemento a um pacote é preciso selecionar o pacote como destino da operação de derivação, conforme apresentado na Figura 29. A fim de demonstrar esta funcionalidade foi criado mais um requisito "RF004- O sistema deve permitir o registro do consumo do cliente no hotel". Ao realizar a derivação de um caso de uso, subordinando o mesmo a um pacote, este é criado dentro do pacote, observe o UC004 na Figura 30. A diferença entre subordinar um elemento a um artefato ou a um pacote é que no primeiro o elemento fica visível quando o artefato é aberto para visualização, no segundo o elemento apenas fica visível dentro da estrutura de pacotes do EA, como indicado na Figura 30.

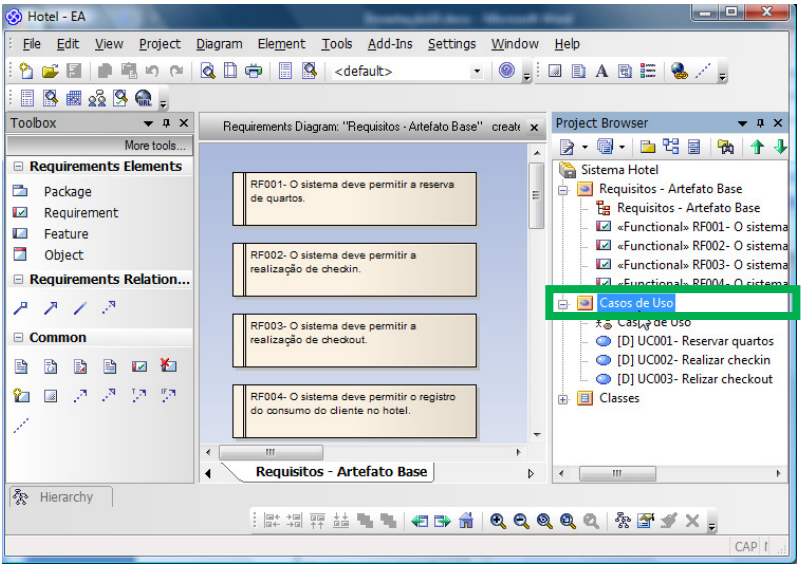

Figura 29- Seleção de um pacote como destino.

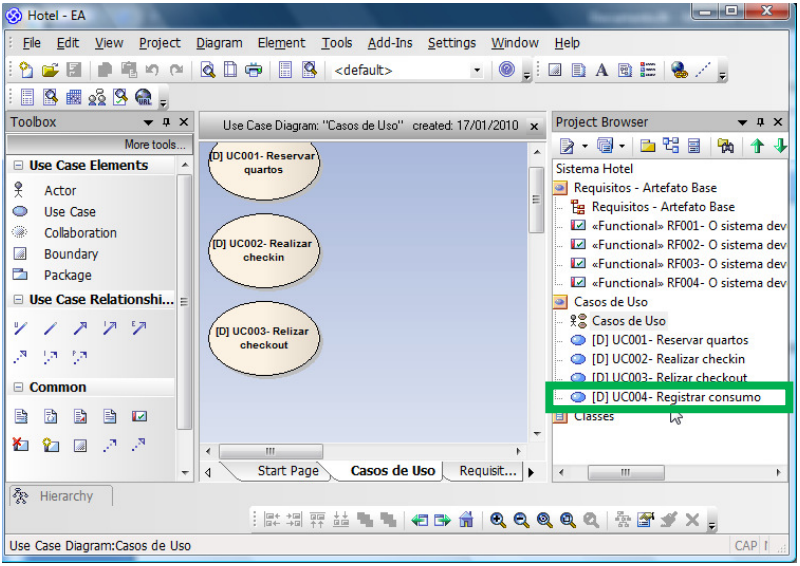

Figura 30- Resultado da suborninação a um pacote.

A terceira opção de subordinação é a subordinação a um elemento. Um novo requisito foi criado "RF005- O sistema deve

permitir que a reserva seja alterada", o caso de uso que será derivado a partir deste requisito estará subordinado ao "UC001- Reservar quartos", por esta razão é necessário selecionar este elemento como destino (Figura 31). Ao aplicar a derivação o elemento criado é posicionado em um nível abaixo da estrutura hierárquica do elemento selecionado como destino, conforme pode ser observado na Figura 32.

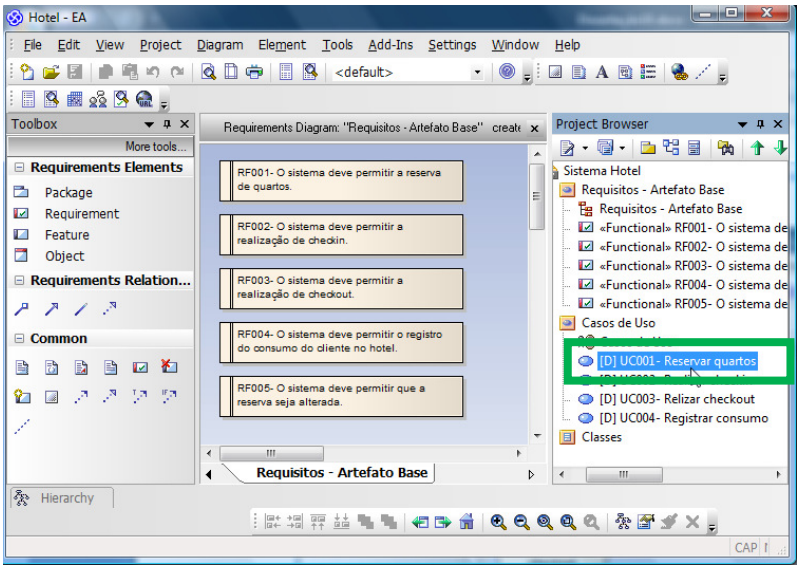

Figura 31- Seleção de um elemento como destino.

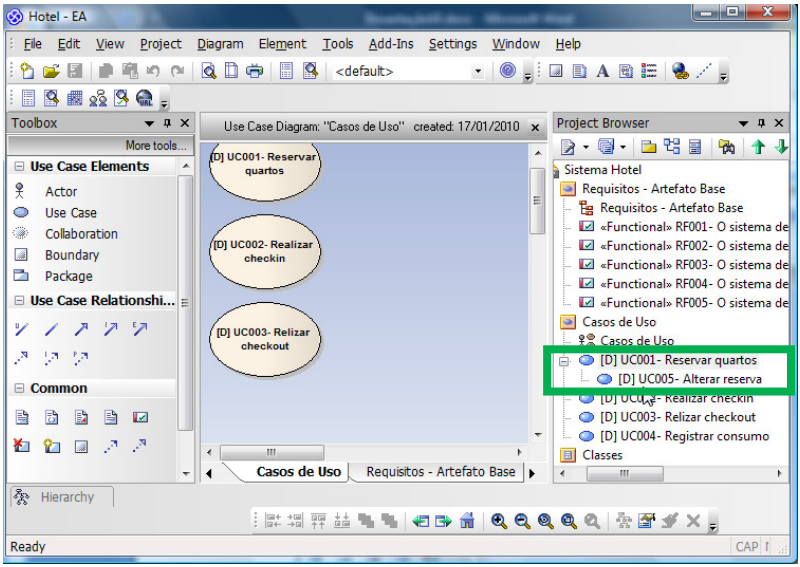

Figura 32- Elemento subordinado a outro elemento.

A intenção da técnica indutiva é reduzir o esforço do desenvolvedor no mapeamento da rastreabilidade, tornando a atividade menos penosa em comparação às técnicas atualmente utilizadas. Com isto espera-se que a resistência à utilização da rastreabilidade possa diminuir.

5 ESTUDO E RESULTADOS

Este capítulo apresenta os detalhes do estudo quantitativo que foi realizado com o intuito de realizar um comparativo entre a técnica indutiva e a técnica tradicional no mapeamento da rastreabilidade. Foram utilizados gráficos e testes de hipóteses para avaliação dos resultados.

5.1 PLANEJAMENTO DO ESTUDO

O objetivo do estudo consiste em comparar técnica indutiva com técnicas tradicionais, avaliando qual a mais adequada no sentido de aumentar a produtividade no mapeamento de rastros. A produtividade é medida através do número de rastros corretamente criados pelo desenvolvedor, comparando a execução das técnicas dentro de um mesmo limite de tempo.

As técnicas tradicionais utilizadas são ferramentas disponíveis na ferramenta CASE Enterprise Architect™ (EA) para definição de rastreabilidade *a posteriori*, ou seja, a matriz de relacionamentos e a janela de edição do elemento (criação direta de links).

A população alvo do estudo foram desenvolvedores, neste estudo participaram estudantes do terceiro ano do curso de graduação em Ciência da Computação.

Para a realização do estudo foi preparado um material detalhando os passos de sua execução com cada uma das técnicas: indutiva, matriz de relacionamentos, criação de links. O material referente a técnica indutiva pode ser visualizado no Anexo B.

Outro material preparado para o estudo foi um arquivo do EA contendo 80 requisitos, já criados no diagrama de requisitos do Enterprise Architect™, a fim de solicitar a criação de um caso de uso correspondente para cada requisito bem como os respectivos rastros.

A fim de evitar dúvidas em relação aos artefatos a serem trabalhados foi estruturado um arquivo do EA com os respectivos

pacotes e diagramas já criados: Requisitos e Casos de Uso, esta estrutura pode ser visualizada Figura 33, na qual o diagrama de requisitos encontra-se na área de visualização da ferramenta, com os requisitos já criados; e o diagrama de caso de uso, no qual os elementos deveriam ser criados pelos desenvolvedores, encontra-se na lateral direita da figura.

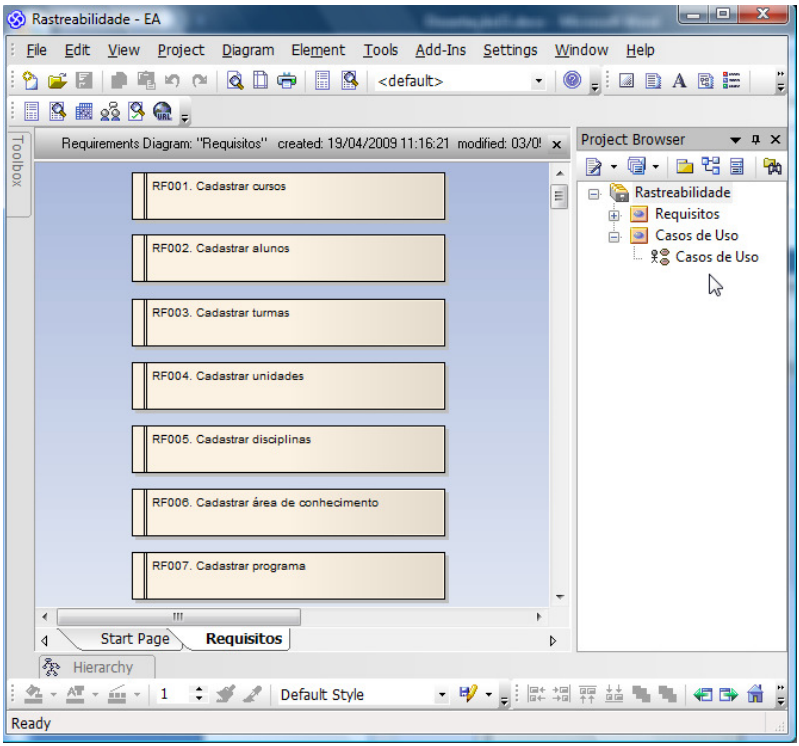

Figura 33- Estrutura de artefatos e elementos utilizada no estudo.

A fim de impedir que a definição de nomes interferisse no tempo despendido pelos participantes, os casos de uso poderiam ser criados com o mesmo nome dos requisitos, conforme apresentado no Quadro 1, o qual é uma amostra do quadro apresentado para os desenvolvedores no dia da realização do estudo. O quadro original contém 80 requisitos e casos de uso, como dito anteriormente.

Quadro 1- Amostra da tabela de correspondência entre requisitos e casos de uso.

Requisitos entregues	Casos de Uso sugeridos
RF001. Cadastrar cursos	Cadastrar cursos
RF002. Cadastrar alunos	Cadastrar alunos
RF003. Cadastrar turmas	Cadastrar turmas
RF004. Cadastrar unidades	Cadastrar unidades
RF005. Cadastrar disciplinas	Cadastrar disciplinas

Medidas foram tomadas a fim de produzir dados sem vícios, objetivando evitar possíveis interferências nos resultados. Essas medidas foram adotadas antes, durante e depois do estudo, conforme detalhado a seguir.

No material produzido para ser utilizado no estudo foi realizado um pré-teste com dois desenvolvedores com características semelhantes com os participantes do estudo. Foram realizados ajustes no material após a execução do pré-teste a fim de facilitar o entendimento e evitar dúvidas durante o estudo.

Antes da realização do estudo real os desenvolvedores receberam um arquivo de testes com os requisitos para praticarem e tirar dúvidas relacionadas ao funcionamento da técnica e do EA. Isto foi feito no intuito de nivelar o conhecimento dos mesmos.

A configuração (hardware e software) dos computadores do laboratório foi verificada, assegurando que todos possuíam as mesmas características. Pois diferenças nas velocidades dos computadores poderiam interferir no número de elementos criados.

A execução da tarefa de cada desenvolvedor foi gravada em vídeo com o software ScreenCam™ (SMARTGUYZ INCORPORATEC, 2009), com o objetivo de detectar anormalidades, tais como: o usuário realizar outra atividade além do estudo e ocorrência de falha no funcionamento dos softwares. Os desenvolvedores tinham conhecimento que seu exercício estava sendo gravado, os próprios iniciaram a execução em seus computadores.

Os desenvolvedores receberam um treinamento prévio referente a conceitos e funcionamento de rastreabilidade de software.

5.2 EXECUÇÃO DO ESTUDO

Quinze desenvolvedores foram avaliados no estudo, dos quais cinco usaram a técnica indutiva, cinco usaram a técnica criação de links e cinco usaram a técnica matriz de relacionamentos. O tempo disponibilizado para a tarefa foi de 10 minutos.

Cada desenvolvedor recebeu um material com instruções sobre como realizar o estudo com a técnica correspondente, o material referente a técnica indutiva pode ser visto no Anexo B.

Antes da realização do estudo real os desenvolvedores receberam um arquivo de testes com os requisitos a fim de praticar e tirar dúvidas relacionadas ao funcionamento da técnica e do EA, assim que não haviam mais dúvidas o experimento real foi iniciado. Cada desenvolvedor executou passos específicos da técnica recebida.

A seguir serão apresentados os principais passos executados no experimento para utilização de cada uma das técnicas: matriz de relacionamentos, criação de links e indutiva.

Na matriz de relacionamentos inicialmente devem ser criados os casos de uso, para criar um caso de uso deve-se selecionar o tipo de elemento na toolbox e inserir o elemento na área de visualização do diagrama, como pode ser observado na Figura 34.

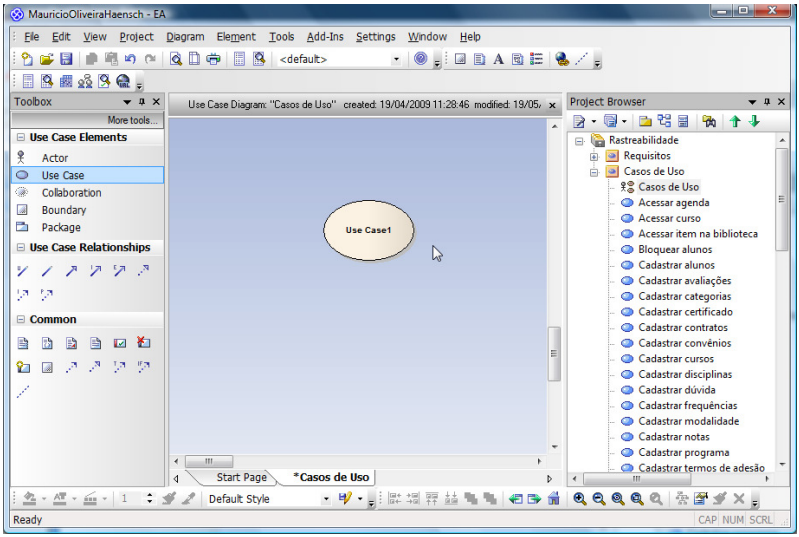

Figura 34- Criação do caso de uso.

Assim que um ou mais casos de usos foram criados (a critério do desenvolvedor) pode-se acessar a matriz, por meio do menu *Relationship Matriz*, apresentado na Figura 35. O sistema apresenta a matriz permitindo a seleção de dois artefatos, na Figura 36 foram selecionados os artefatos de requisitos e casos de uso. Os requisitos aparecem na coluna principal e os casos de uso na linha principal, para criar um rastro entre um requisito *x* e um caso de uso *y* deve-se encontrar a intersecção entre os mesmos na matriz e acionar a opção para a criação do rastro.

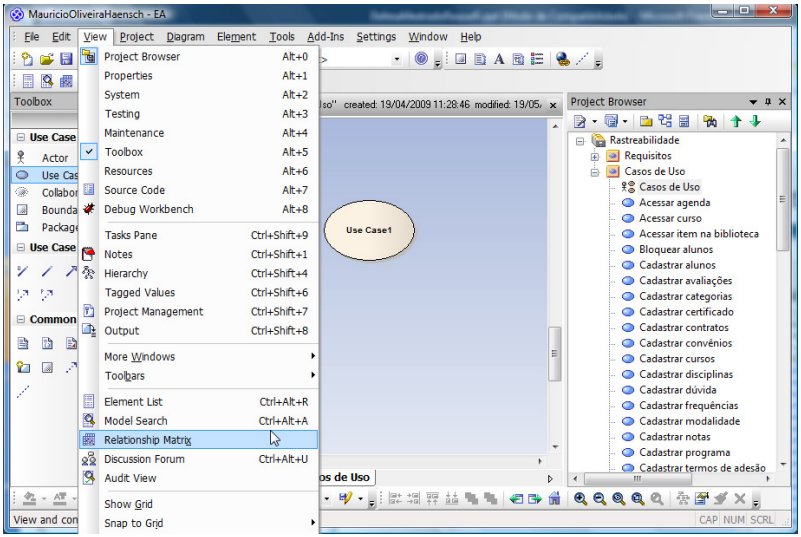

Figura 35- Menu de acesso a matriz.

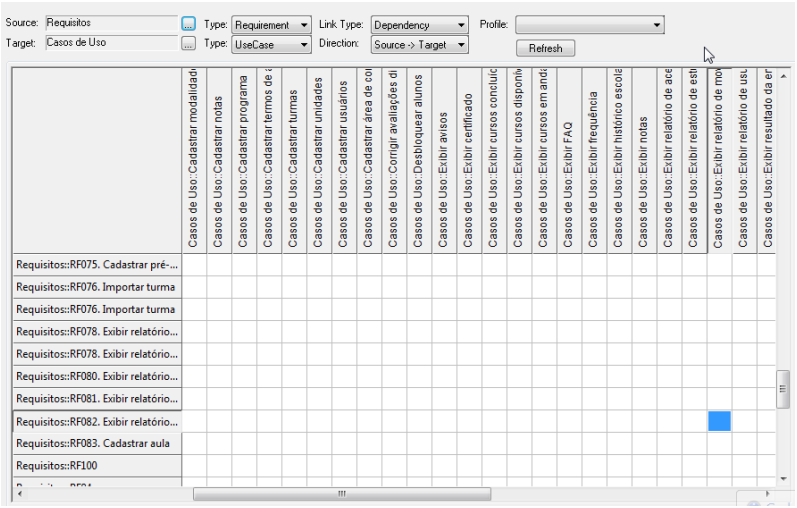

Figura 36- Matriz de rastreabilidade.

Para utilização da técnica criação de links, inicialmente o caso de uso deve ser criado, este passo é igual ao executado na técnica matriz de relacionamentos (Figura 34). Posteriormente deve-se selecionar o elemento na lateral direita da ferramenta, denominada *Project Browser* e acionar a opção *Create Link* (Figura 36). O sistema apresenta uma tela onde pode ser selecionado o tipo de artefato com o qual se deseja criar rastros. Na Figura 38 foi selecionado o artefato de requisitos, por este motivo são listados todos os requisitos já criados até o momento na ferramenta. Nesta etapa deve-se selecionar o requisito desejado e o rastro com o caso de uso é criado.

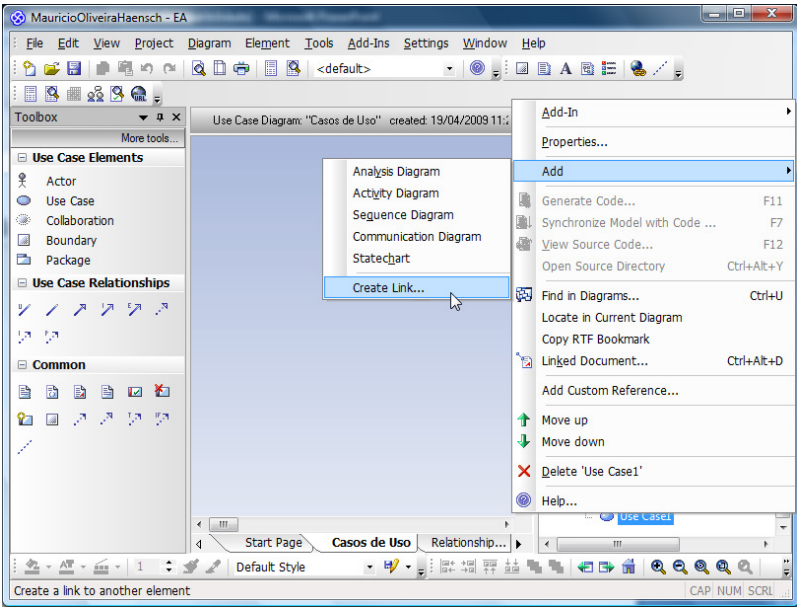

Figura 37- Criação de links.

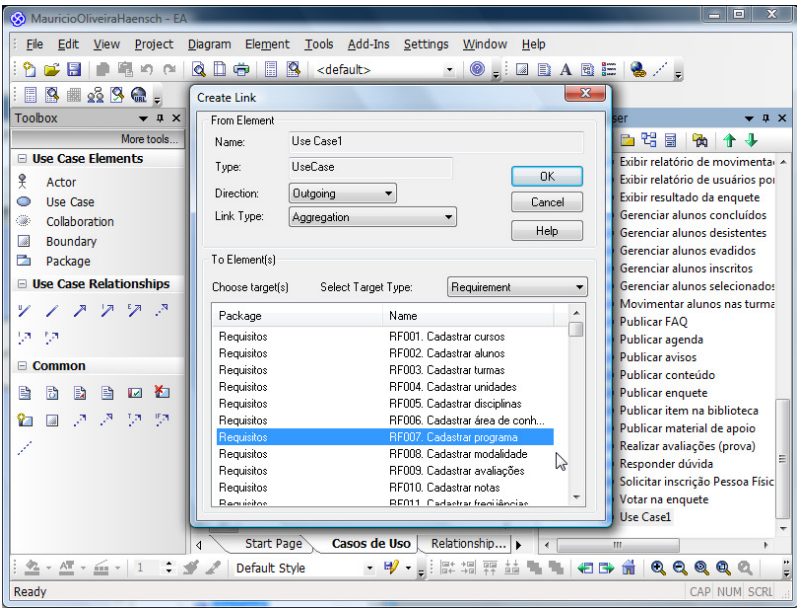

Figura 38- Lista de requisitos da técnica criação de links.

A título ilustrativo na Figura 39 é apresentado um diagrama de caso de uso, criado durante o estudo por um dos participantes.

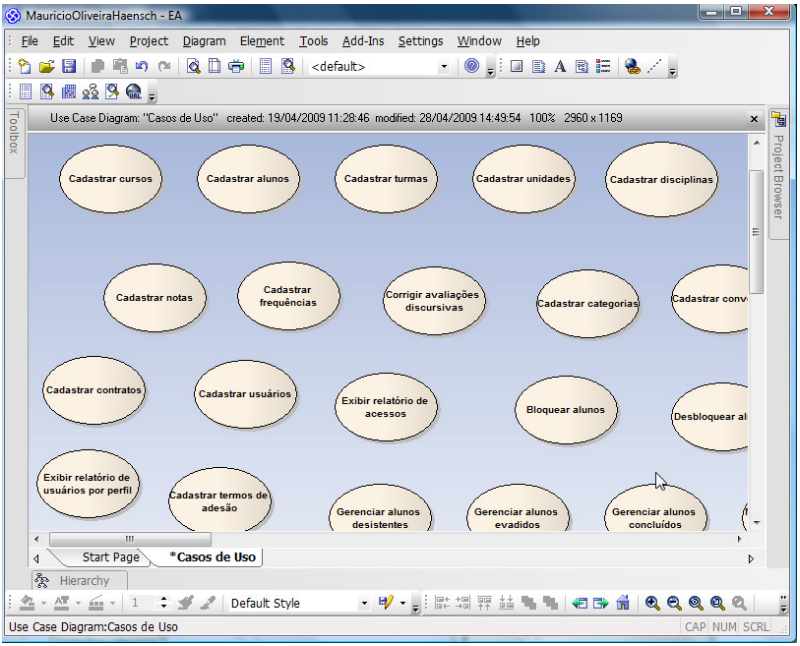

Figura 39- Casos de uso criados durante o estudo.

5.3 AVALIAÇÃO DO ESTUDO

Para avaliação dos resultados foram contabilizados, para cada um dos desenvolvedores, o número de rastros corretamente criados, durante a execução do estudo pelos desenvolvedores. Para as técnicas matriz e criação de links o tempo de digitação dos nomes dos casos de uso não foram contabilizados como tempo de execução.

O resultado é apresentado na Figura 40, observa-se que a média de rastros criados com a técnica indutiva (72,6) é significativamente maior do que a média obtida com a técnica matriz de relacionamentos (28,2) e a média obtida com a técnica de criação de links (23,6).

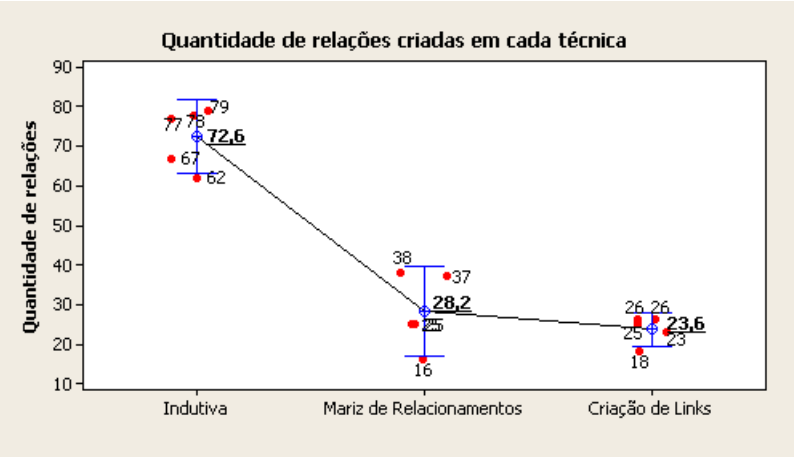

Figura 40- Rastros corretas criadas com cada técnica.

Um dos principais fatores que contribuem para a vantagem na quantidade de rastros criados na técnica indutiva, em comparação com a matriz de rastreabilidade, está no fato de que ao utilizar a matriz há um gasto significativo de tempo para encontrar os elementos a serem relacionados.

Para exemplificar esta situação supõe-se que o desenvolvedor deseja relacionar o requisito que se encontra na posição 100 da matriz com o caso de uso que se encontra na posição 200. Uma vez que o requisito já tenha sido localizado no diagrama de requisito, para criar um rastro é necessário acessar outra tela que contém a matriz, com uma listagem de requisitos na primeira coluna e a listagem de casos de uso na primeira linha. Faz-se necessário encontrar novamente o requisito, pois o formato de visualização entre as tela é diferente, primeiro o usuário visualiza os elementos dispostos graficamente no diagrama, depois ele deve encontrar este elemento em uma listagem de requisitos, muitas vezes, disposto em uma ordem diferente do diagrama. Depois é necessário localizar o caso de uso, e o mesmo problema no formato diferente na visualização se repete, acrescido pela necessidade de efetuar rolagem de tela devido a grande quantidade de elementos. Por último o desenvolvedor deve encontrar o ponto na matriz onde acontece a intersecção entre o requisito e o caso de uso, para finalmente acionar a funcionalidade que cria a rastro de rastreabilidade.

Com a utilizaçao da técnica indutiva este tempo de busca do elemento é economizado, uma vez que o usuário pode trabalhar em uma

única a área de vizualização dos elementos, visualizando os mesmos em um único formato e ordem.

Em função do tamanho da amostra, foram aplicados testes estatísticos de hipóteses para verificar se a diferença encontrada é significativa. Como referência estatística foi utilizado o livro de Barbetta et al, 2004. O teste de hipóteses foi aplicado primeiramente comparando a técnica indutiva com a matriz de relacionamentos. A hipótese H_0 (hipótese nula) é que em média o desempenho da técnica indutiva é igual ao da técnica matriz de relacionamentos. A hipótese H_1 (hipótese alternativa) é que em média a técnica indutiva possui maior desempenho que a técnica matriz de relacionamentos, para a criação e evolução de rastros.

Maior desempenho, neste trabalho, significa criar o maior número de rastros corretos. Rastros corretos são aqueles criados entre elementos que realmente possuem uma relação de dependência. Um dos erros que podem ocorrer no mapeamento da rastreabilidade é criar por engano uma relação entre elementos que não possuem dependência (relação falsa).

As amostras são independentes, pois as técnicas foram executadas com grupos diferentes. Neste caso aplica-se o teste *t* para amostras independentes. Ao aplicar o teste o resultado foi *8,27*. Com *8* graus de liberdade a probabilidade de significância foi $p=0,02$.

Considerando um nível de significância de *5%* ($a=0,05$) tem-se ($p<a$), o que significa que se deve rejeitar H_0 em favor de H_1. Conclui-se, portanto, com *95%* de confiança que a técnica indutiva tende a apresentar maior desempenho do que a técnica matriz de relacionamentos. O desempenho melhor da técnica indutiva indica que o esforço despendido para sua utilização pode ser significativamente menor em comparação à técnica matriz de relacionamentos.

O mesmo teste foi realizado comparando o desempenho da técnica indutiva com o da técnica criação de *links*, onde a probabilidade de significância resultou em $p=0$. Conclui-se com 95% de confiança que a técnica indutiva tende a apresentar maior desempenho do que a técnica criação de *links*.

O teste também foi aplicado entre a técnica matriz de relacionamentos e a técnica criação de links, avaliando se as duas técnicas possuíam desempenho distinto. O teste não detectou diferença entre estas duas técnicas.

6 CONCLUSÕES E TRABALHOS FUTUROS

6.1 CONCLUSÕES

As principais técnicas de rastreabilidade apresentam limitações que dificultam o uso efetivo da rastreabilidade na prática. A matriz de relacionamentos, técnica atualmente utilizada nas organizações, torna-se de difícil gerenciamento à medida que o número de artefatos aumenta. A recuperação automática, apesar de ser uma alternativa ou complemento para a matriz, pode detectar muitos relacionamentos falsos e durante as pesquisas realizadas não foram localizados registros de utilização desta técnica em ferramentas CASE comerciais. As organizações ainda buscam uma forma de criar as rastros que forneça um bom custo-benefício.

A principal vantagem da técnica indutiva comparada a matriz de relacionamentos está no fato de que o desenvolvedor pode criar os relacionamentos de rastreabilidade de forma integrada aos elementos, utilizando apenas o ambiente de criação dos elementos. Já na matriz de relacionamentos o desenvolvedor deve criar os elementos e depois criar as rastros, utilizando para isto dois ambientes, um para a criação de elementos, outro para a criação dos relacionamentos (matriz). A utilização de dois ambientes pode tornar mais trabalhoso e menos eficaz o mapeamento da rastreabilidade.

Dentre as principais contribuições do trabalho está a mudança da forma com que o usuário cria novos elementos nos artefatos, ao invés de criar elementos de forma independente entre si, a proposta é que eles sejam criados um a partir do outro, assim os rastros vão sendo mapeados a medida com que os elementos são definidos, evitando que esta definição seja deixada para depois ou que acabe perdida/esquecida durante o projeto.

O estudo realizado indicou que a técnica indutiva tende a apresentar uma maior produtividade em relação às técnicas tradicionais, por meio da criação de um maior número de rastros. O esforço despendido para sua utilização pode ser significativamente menor, tornando a atividade menos penosa, reduzindo o esforço do

desenvolvedor. Desta forma futuramente a técnica indutiva poderia ser utilizada nas organizações.

Em relação a técnicas de recuperação automática (como LSI), a técnica indutiva não apresenta as mesmas desvantagens, pois não procura identificar rastros a partir de artefatos existentes. Ao invés disso, a técnica indutiva procura criar as rastros no momento da criação dos próprios elementos, o que evita a formação de falsos rastros.

A técnica indutiva representa uma nova perspectiva para o mapeamento das rastros. O estudo realizado mostrou resultados animadores indicando que a técnica pode melhorar significativamente a produtividade. Com a ampla aplicação da técnica em ferramentas CASE pode-se alcançar o uso efetivo da rastreabilidade nas organizações. E como conseqüência disto, uma maior qualidade na documentação gerada nos projetos.

A técnica indutiva, por outro lado, propõe que a tarefa executada pelo desenvolvedor seja redefinida. Ao invés de simplesmente desenhar uma classe em um diagrama, o desenvolvedor deverá deixar claro o porquê desta classe, ou seja, qual o outro elemento que fez com que ele chegasse à conclusão de que tal classe seria necessária.

A técnica indutiva proposta é eficaz apenas em sistemas cuja documentação ainda vai ser produzida, onde se tem a oportunidade de utilizar a técnica desde o início. Ela não detecta rastros que deveriam existir entre elementos e que por alguma razão não foram incluídas. Estas são limitações da técnica proposta. Nestas situações, ela pode ser complementada, por exemplo, com a aplicação de recuperação automática.

6.2 TRABALHOS FUTUROS

Como trabalho futuro poderiam ser realizadas pesquisas no sentido de avaliar a opinião dos desenvolvedores sobre a utilização da técnica indutiva, bem como uma avaliação da aderência da técnica nas organizações.

Outra linha de pesquisa que poderia ser desenvolvida se refere a manutenção das rastros, no que diz respeito a sinalização dos elementos afetados por uma determinada alteração. Esta poderia ser uma nova funcionalidade do *plugin* de rastreabilidade.

7 REFERÊNCIAS BIBLIOGRÁFICAS

ALDRICH, J., CHAMBERS, C., NOTKIN, D. 2002. “ArchJava: connecting software architecture to implementation”. In: Proceedings of the 24th international Conference on Software Engineering (Orlando, Florida, May 19 - 25, 2002). ICSE '02. ACM, New York.

ALMEIDA, J. P., ECK, P. v., IACOB, M. (2006) “Requirements Traceability and Transformation Conformance in Model-Driven Development”. In Proceedings of the 10th IEEE international Enterprise Distributed Object Computing Conference. EDOC. IEEE Computer Society, Washington, DC, 355-366.

ANTONIOL, G., CANFORA, G., CASAZZA, G., DE LUCIA, A., MERLO, E. (2002) “Recovering Traceability Links between Code and Documentation”. IEEE Trans. Softw. Eng. 28, 10 (Oct. 2002), 970-983.

ANTONIOL, G., CIMITILE, A., CASAZZA, G. (2000) “Traceability Recovery by Modeling Programmer Behavior”. In Proceedings of the Seventh Working Conference on Reverse Engineering (Wcre'00), WCRE. IEEE Computer Society, Washington, DC, 240.

BARBETTA, P. A., REIS, M., BORNIA, A. C. (2004) “Estatística para Cursos de Engenharia; Informática”. São Paulo: Atlas.

CARNEIRO, G. F., SANT´ANNA, C. N., MENDONÇA, M. G. (2010) "On the Design of a Multi-Perspective Visualization Environment to Enhance Software Comprehension Activities". Proceedings of the VII Workshop on Modern Software Maintenance (WMSWM 2010), Co-located with SBQS, 2010. Belém, Pará – Brazil.

CLELAND-HUANG, J. (2006a) "Requirements Traceability - When and How does it Deliver more than it Costs?". In Proceedings of the 14th IEEE international Requirements Engineering Conference (September 11 - 15, 2006). RE. IEEE Computer Society, Washington, DC, 323.

CLELAND-HUANG, J. (2006b) "Just Enough Requirements Traceability". In Proceedings of the 30th Annual international Computer Software and Applications Conference - Volume 01 (September 17 - 21, 2006). COMPSAC. IEEE Computer Society, Washington, DC, 41-42.

CLELAND-HUANG, J., ZEMONT, G., LUKASIK, W. (2004) "A Heterogeneous Solution for Improving the Return on Investment of Requirements Traceability," Requirements Engineering, IEEE International Conference on, pp. 230-239, 12th IEEE International Requirements Engineering Conference (RE'04).

EGYED, A., BIFFL, S., HEINDL, M., GRÜNBACHER, P. (2005) Determining the cost-quality trade-off for automated software traceability. In Proceedings of the 20th IEEE/ACM international Conference on Automated Software Engineering. ASE '05. ACM, New York, NY, 360-363.

FERREIRA, A. B. H., ANJOS, M.; FERREIRA, M. "Dicionário Aurélio ilustrado". 1.ed. Curitiba: Ed. Positivo, 2008. 560p.

FLETCHER, J., CLELAND-HUANG, J. (2006) "Softgoal Traceability Patterns," Software Reliability Engineering, International Symposium on, pp. 363-374, 17th International Symposium on Software Reliability Engineering (ISSRE'06).

GOTEL, O.C.Z., FINKELSTEIN, C.W. (1994) “An analysis of the requirements traceability problem”, Requirements Engineering: Proceedings of the First International Conference, p. 94-101.

HEINDL, M., BIFFL, S. (2006) Risk management with enhanced tracing of requirements rationale in highly distributed projects”. In Proceedings of the 2006 international Workshop on Global Software Development For the Practitioner (Shanghai, China, May 23 - 23, 2006). GSD '06. ACM, New York, NY, 20-26.

JIANG, H., NGUYEN, T. N., CHANG, C. K., DONG, F. (2007) "Traceability Link Evolution Management with Incremental Latent Semantic Indexing", In: Proceedings of the 31st Annual International Computer Software and Applications Conference, IEEE Computer Society, USA, p.309-316.

LARMAN, C. (2007) “Utilizando UML e Padrões: Uma Introdução à Análise e ao Projeto Orientados a Objetos e ao Desenvolvimento Iterativo. Porto Alegre: Bookman.

LORMANS, M., VAN DEURSEN, A. (2006) Can LSI help Reconstructing Requirements Traceability in Design and Test?. In Proceedings of the Conference on Software Maintenance and Reengineering, CSMR. IEEE Computer Society, Washington, DC, 47-56.

LUCIA, A. D., FASANO, F., OLIVETO, R., TORTORA, G. (2007) "Recovering traceability links in software artifact management systems using information retrieval methods", ACM Trans. Softw. Eng. Methodol, USA, p.13.

LUCIA, A., FASANO, F., OLIVETO, R., TORTORA, G. (2006a) Can Information Retrieval Techniques Effectively Support Traceability Link Recovery?. In Proceedings of the 14th IEEE international Conference on Program Comprehension. ICPC. IEEE Computer Society, Washington, DC, 307-316.

Lucia, A., Oliveto, R., Zurolo, F., and Di Penta, M. (2006b) Improving Comprehensibility of Source Code via Traceability Information: a Controlled Experiment. In Proceedings of the 14th IEEE international Conference on Program Comprehension. ICPC. IEEE Computer Society, Washington, DC, 317-326.

MALETIC, J. I., COLLARD, M. L., SIMOES, B. (2005) "An XML based approach to support the evolution of model-to-model traceability links", In: Proceedings of the 3rd international Workshop on Traceability in Emerging Forms of Software Engineering, ACM, USA.

MARCUS, A., MALETIC, J. I. 2003. "Recovering documentation-to-source-code traceability links using latent semantic indexing". In Proceedings of the 25th international Conference on Software Engineering (Portland, Oregon, May 03 - 10, 2003). IEEE Computer Society, Washington, DC, 125-135.

MURTA, L. G., HOEK, A., WERNER, C. M. 2008. "Continuous and automated evolution of architecture-to-implementation traceability links". Automated Software Eng. 15, 1 (Mar. 2008), 75-10.

NEUMULLER, C., GRUNBACHER, P. (2006) “Automating Software Traceability in Very Small Companies: A Case Study and Lessons Learne”, In: Proceedings of the 21st IEEE/ACM international Conference on Automated Software Engineering, IEEE Computer Society, USA, p. 145-156.

OLIVETO, R., ANTONIOL, G., MARCUS, A. e HAYES, J. (2007) "Software Artefact Traceability: the Never-Ending Challenge", Software Maintenance, IEEE International Conference.

PALMER, J. D. (1997) “Traceability”, In: Software Requirements Engineering, R.H. Thayer and M. Dorfman, p. 364-374.

SANTOS, Raquel Nitsche, WAZLAWICK, Raul Sidnei (2009) “Rastreabilidade Indutiva Aplicada a Artefatos de Software”. Proceedings 6th Experimental Software Engineering Latin American Workshop (November 11-13, 2009). ESELAW. UFSCar - São Carlos, São Paulo, Brazil, 52-61.

SETTIMI, R., CLELAND-HUANG, J., KHADRA, O. B., MODY, J., LUKASIK, W., DE PALMA, C. (2004) “Supporting Software Evolution through Dynamically Retrieving Traces to UML Artifacts”. In: Proceedings of the Principles of Software Evolution, 7th international Workshop. IWPSE. IEEE Computer Society, Washington, DC.

SETTIMI, R., CLELAND-HUANG, J., KHADRA, O. B., MODY, J., LUKASIK, W., DEPALMA, C. 2004. “Supporting Software Evolution through Dynamically Retrieving Traces to UML Artifacts”. In: Proceedings of the Principles of Software Evolution, 7th international Workshop (September 06 - 07, 2004). IWPSE. IEEE Computer Society, Washington, DC.

SMARTGUYZ INCORPORATEC (2009) "ScreenCam". Disponível em: http://www.smartguyz.com/index-2.html. Acesso em: 01 dez. 2009.

SPARX SYSTEMS (2009) "Enterprise Architect". Disponível em: http://www.sparxsystems.com/products/ea. Acesso em: 01 dez. 2009.

ANEXO A – Artigo Publicado

VI Experimental Software Engineering Latin American Workshop - ESELAW'09

Rastreabilidade Indutiva Aplicada a Artefatos de Software

Trabalho técnico

Raquel Nitsche dos Santos, Raul Sidnei Wazlawick

Departamento de Informática e Estatística – Programa de Pós-Graduação em Ciência da Computação – Universidade Federal de Santa Catarina (UFSC) – Cx. P. 476 –Florianópolis, SC – Brasil

{raquelnitsche, raul}@inf.ufsc.br

Abstract. *Maintaining quality and consistency of artifacts through a software project can be more effective with the use of traceability. However, creating consistent traceability relationships can be a task so complex that it is often ignored or minimized. Techniques such as automatic discovery and traceability matrix have known limitations. This paper examines an alternative that consists in allowing the creation of traceability relationships inductively during the software development process. Experiments with a CASE tool showed encouraging results indicating that the technique can significantly improve productivity.*

Resumo. *A tarefa de manter a qualidade e consistência de artefatos ao longo de um projeto de software pode ser mais efetiva com a utilização de rastreabilidade. Porém, a criação de rastros consistentes ao longo de um projeto é uma tarefa tão complexa que muitas vezes é deixada de lado. Técnicas como a recuperação automática e a matriz de rastreabilidade apresentam limitações. Este trabalho examina outra abordagem que consiste em permitir a criação de rastros indutivamente ao longo do desenvolvimento dos artefatos do projeto. Estudos com uma ferramenta CASE mostraram resultados animadores indicando que a técnica pode melhorar significativamente a produtividade.*

Introdução

Os artefatos de software são constituídos por diversos elementos. Por exemplo, o modelo conceitual é formado por classes, atributos e associações, enquanto o diagrama de caso de uso é formado por casos de uso, associações e

atores. Para que o desenvolvedor[3] possa saber quais elementos afetam e são afetados por outros, ele pode utilizar o conceito de *rastreabilidade*, que consiste em uma maneira de associar elementos indicando uma relação de causa-efeito entre eles. A rastreabilidade entre elementos de artefatos permite acompanhar a vida de um artefato durante o ciclo de vida do software [1].

A rastreabilidade auxilia na compreensão dos relacionamentos entre os artefatos [7], na análise de impacto e no reuso de software, proporcionando ao desenvolvedor uma importante visão para o processo de desenvolvimento e evolução do software. A rastreabilidade é reconhecida como um importante fator para obtenção da qualidade no processo de desenvolvimento, bem como para um gerenciamento de projeto eficiente [5]. Processos de melhoria da qualidade de software como o CMMI nível 3, ISO 15504 e MPS-BR estabelecem que práticas básicas de rastreabilidade devem ser seguidas.

As principais técnicas de rastreabilidade existentes são a matriz de rastreabilidade e a recuperação automática de rastreabilidade. Ambas são importantes, porém possuem limitações que dificultam o uso efetivo da rastreabilidade na prática.

De acordo com Cleland-Huang *et alii* [20] a matriz de rastreabilidade assume um tamanho que rapidamente se torna não gerenciável, uma vez que o número de relações tende a aumentar significativamente, tornando a utilização da matriz complexa. O empenho dos desenvolvedores é retardado pela carência de suporte por parte das ferramentas, o que causa uma percepção de que o custo despendido para manter a matriz de rastreabilidade é muito alto, em comparação aos seus benefícios [21].

Técnicas de recuperação automática não recuperam todos os relacionamentos corretos sem também recuperar muitos falsos, o que acarreta trabalho extra para o desenvolvedor no descarte destes relacionamentos [15]. A tarefa de descarte pode ser muito trabalhosa, pois o desenvolvedor pode gastar mais tempo para descartar relações falsas do que rastreando as corretas. Ainda que seja possível melhorar o desempenho destas técnicas elas estão longe de ser uma solução viável para o problema [22].

Mesmo que a rastreabilidade seja requerida em grande parte das aplicações de segurança crítica, e faça parte de diversas iniciativas de melhoria de processo de software, as organizações ainda buscam uma maneira de implementá-la de forma que traga um bom custo-benefício [21].

Apesar de sua reconhecida importância, não é satisfatório o suporte para a rastreabilidade em ferramentas CASE contemporâneas, especificamente aquelas que permitem a construção de artefatos UML. O principal defeito destas

[3] *Desenvolvedor* é entendido neste artigo como qualquer profissional (analista de sistemas, projetista, programador, etc.) que crie artefatos relacionados ao produto de software em desenvolvimento.

ferramentas é a falta de um suporte automático ou semi-automático para a criação e manutenção de *links* de rastreabilidade, o que torna sua utilização cansativa e custosa [6]. Esta carência é um dos fatores que promovem a baixa qualidade de sistemas [3].

As relações de causa-efeito entre elementos de artefatos não são identificadas automaticamente nas ferramentas CASE porque a inclusão de novos elementos nos diagramas usualmente é feita sem que sua origem ou causa seja explicitada, os elementos são criados de forma individual, geralmente a partir de *toolboxes*. Desta forma, a criação da rastreabilidade é uma tarefa deixada para depois, ou seja, após inserir o novo elemento no diagrama o usuário da ferramenta CASE deverá indicar quais as rastros se aplicam àquele elemento.

Este artigo mostra que é possível automatizar a criação de rastros sob a hipótese de que a inserção de novos elementos nos artefatos não consiste simplesmente em criar um novo elemento no diagrama, mas em uma ação que, em muitos casos, tem uma causa bem definida a partir de algum outro elemento. Por exemplo, uma classe pode estar sendo inserida no modelo conceitual devido à existência de um caso de uso que a menciona. Ou ainda, um caso de uso pode estar sendo inserido no diagrama em função de um ou mais requisitos que lhe dão origem. Assim, a idéia examinada neste artigo é de que a de inserção de elementos nos artefatos seja *indutiva*. Ou seja, com exceção dos elementos iniciais (base) a inserção de um elemento em um artefato deverá ocorrer a partir de outro elemento, o qual consiste em sua causa.

Este artigo apresenta, então, uma abordagem semi-automática para a criação da rastreabilidade[4], que objetiva tornar sua utilização menos custosa que a técnica tradicional, almejando seu uso efetivo nas organizações.

O restante do artigo apresenta na seção 2 trabalhos relacionados; na seção 3 definições de rastreabilidade; na seção 4 a rastreabilidade indutiva, incluindo sua implementação em uma ferramenta CASE; na seção 5 o estudo realizado e seus resultados; e na seção 6 as conclusões.

Trabalhos Relacionados

A matriz e a recuperação automática de rastreabilidade são as principais técnicas da área. Estas tratam a rastreabilidade como uma atividade à parte da criação dos elementos nos artefatos, ou seja, primeiro são criados os elementos, depois são definidas as relações.

Em sua forma mais simples a matriz de rastreabilidade se manifesta em tabelas com linhas e colunas, nas quais os elementos de um projeto são

[4] O escopo deste artigo compreende o processo de criação das rastros, já manutenção das relações não faz parte de seu intuito.

relacionados aos requisitos que os satisfazem [23]. As relações são, geralmente, estabelecidas pelo relacionamento explícito entre dois artefatos [24], e esta ainda é a forma como as relações são criadas atualmente nas organizações [21]. A matriz de rastreabilidade é a técnica mais atendida pelas ferramentas CASE que suportam a rastreabilidade.

Recentemente, técnicas de recuperação automática de rastros vêm sendo pesquisadas na tentativa de encontrar uma alternativa para o problema da custosa e complexa definição da rastreabilidade [16, 17, 18]. A recuperação automática procura identificar rastros baseando-se na similaridade entre o texto contido nos artefatos. Um exemplo de técnica de recuperação é a LSI (*Latent Semantic Indexing*) [3, 11, 12, 15]. No entanto, segundo alguns autores [2, 3, 4], ainda existem muitos desafios que precisam ser superados. Um dos problemas é que as técnicas de recuperação confiam na hipótese de que o uso correto de termos do domínio entre artefatos permite rastreá-los. Nos casos em quem isto não acontece, o processo de recuperação automático torna-se ineficiente [25], pois muitos *links* possíveis deixam de ser detectados e falsos *links* podem ser detectados. Estas técnicas de recuperação não são completamente automáticas, pois o usuário deve interagir com o sistema para decidir sobre a aceitação ou rejeição das relações recuperadas.

Durante a realização das pesquisas para este artigo não foram identificadas ferramentas CASE de escala industrial que suportassem a recuperação automática da rastreabilidade, mas apenas ferramentas em trabalhos de pesquisa [13, 14].

A técnica indutiva, aqui proposta, diferencia-se dos trabalhos citados em três aspectos: (1) ela não procura detectar ligações de rastreabilidade entre elementos preexistentes, mas propõe que a criação de novos elementos em artefatos seja feita de forma que a ligação de rastreabilidade seja criada pela explicitação da relação causa-efeito entre o elemento causador e o elemento criado, possibilitando assim um menor esforço no mapeamento das relações; (2) a técnica é definida de maneira totalmente genérica, ou seja, pode ser aplicada a quaisquer elementos e quaisquer artefatos, pois não analisa o conteúdo dos elementos ou seu significado, mas a forma de criação destes, o que possibilita o atendimento de diversos artefatos e não apenas de artefatos específicos e (3) a técnica já foi integrada a uma ferramenta CASE de largo uso comercial.

Rastreabilidade

Nesta seção são definidos os conceitos fundamentais para este trabalho: elemento, artefato e relação de rastreabilidade.

Um *elemento e* é uma unidade de informação que compõe um artefato. O universo de todos os elementos possíveis é denotado por *E*. Exemplos de elementos: um caso de uso, uma classe, um requisito, um protótipo de tela,

Um *artefato a* é definido como sendo um conjunto de elementos de E. Um sistema de software pode então ser modelado por um conjunto de artefatos $A = \{a_1, a_2, ... a_n\}$ cada qual contendo um conjunto de elementos, ou seja, $A = \{ \{e_{1,1}, e_{1,2}, ...\}, \{e_{2,1}, e_{2,2}, ...\}, ... \{e_{n,1}, e_{n,2}, ...\} \}$. As eventuais associações entre elementos de um artefato (composição, generalização, associação simples, etc.) são também consideradas elementos dos artefatos. Faz-se exceção apenas às rastros, definidas a seguir, que são consideradas externas aos artefatos, não sendo, portanto, elementos destes.

A *relação de rastreabilidade* $R \subseteq E \times E$ é uma relação acíclica e transitiva que estabelece relações entre elementos de artefatos. A relação de rastreabilidade se dá entre os elementos: mesmo que um elemento esteja presente em um ou mais artefatos, suas relações permanecem as mesmas.

Rastreabilidade Indutiva

O processo de desenvolvimento de software inclui a criação de artefatos e seus elementos nos diferentes graus de abstração. Então, na prática, o processo de desenvolvimento, do ponto de vista das atividades de um desenvolvedor, pode ser entendido como uma seqüência de ações que visam criar elementos nos diferentes artefatos.

Diferente da técnica tradicional, na técnica indutiva a criação das relações está inserida no processo de construção dos elementos nos artefatos, ou seja, as relações são criadas como uma conseqüência da criação dos elementos nos diferentes artefatos e não como uma atividade extra.

O processo se inicia com a criação do elemento base. A partir dele podem ser derivados os demais elementos nos diferentes artefatos. Por exemplo, os requisitos poderiam ser os elementos base. A partir destes são derivados casos de usos e protótipos, dos quais são derivadas classes, e assim por diante. A técnica não obriga o desenvolvedor a utilizar a derivação, mas se ele a usar as relações e rastreabilidade serão criadas automaticamente pela ferramenta CASE.

Os principais passos da técnica indutiva podem ser visualizados na Figura 1. No passo inicial o artefato de destino do elemento a ser derivado é selecionado. Isso só é necessário quando o elemento a ser derivado deve pertencer a um artefato diferente do elemento-causa. Para derivar elementos dentro do mesmo artefato não é necessário selecionar o artefato de destino.

O segundo passo consiste em selecionar o elemento-causa, elemento de origem da relação de rastreabilidade. Pode-se, inclusive, selecionar dois ou mais elementos como elemento-causa em uma derivação porque um elemento pode ter várias causas.

No terceiro passo é aplicada a derivação sobre o elemento-causa, para dar origem ao elemento-efeito. Juntamente com a criação do elemento-efeito é criada a relação de rastreabilidade.

Figura 1. Passos da rastreabilidade indutiva.

Um *plugin* foi desenvolvido na ferramenta CASE *Enterprise Architect*™ (EA) [8], para permitir a utilização da técnica indutiva. Na Figura 2 é apresentada sua interface, os itens destacados (1, 2 e 3) correspondem aos passos necessários para a criação da rastreabilidade apresentados na Figura 1. Neste exemplo são derivados casos de uso a partir de requisitos, porém a técnica indutiva pode permitir a criação de elementos em diversos outros artefatos. As relações criadas podem ser visualizadas por meio da matriz de rastreabilidade e também da funcionalidade *Hierarchy* do EA. *Hierarchy* apresenta de forma hierárquica as dependências entre os elementos.

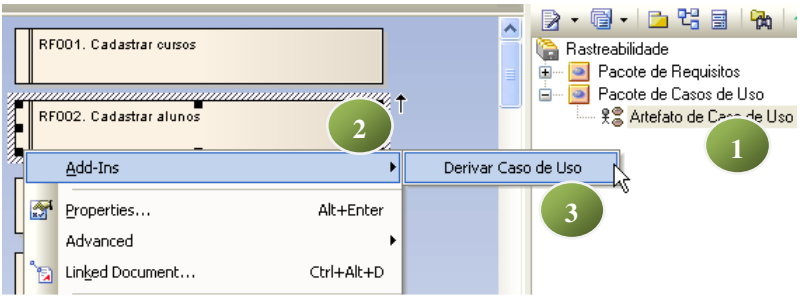

Figura 2. Interface do *plugin* de rastreabilidade.

A intenção da técnica indutiva é reduzir o esforço do desenvolvedor no mapeamento da rastreabilidade, tornando a atividade menos penosa em comparação às técnicas atualmente utilizadas. Com isto espera-se que a resistência à utilização da rastreabilidade possa diminuir.

Estudo

A fim de comparar o desempenho da técnica indutiva e da técnica tradicional realizou-se um estudo com três grupos, sendo que um grupo usou a técnica indutiva, e os outros dois usaram ferramentas disponíveis no EA para definição de rastreabilidade *a posteriori*, ou seja, a matriz de relacionamentos e a janela de edição do elemento (criação direta de links).

O estudo foi estruturado de tal forma que a subjetividade e reflexão proporcionassem pouca interferência nos resultados, uma vez que o objetivo deste consistiu em realizar uma análise quantitativa da técnica indutiva, analisando a quantidade de rastros que podem ser criadas com a técnica indutiva comparada às técnicas tradicionais no mesmo intervalo de tempo.

Para o estudo foram apresentados 80 requisitos no diagrama de

requisitos e se solicitou a cada grupo que criasse um caso de uso correspondente para cada requisito bem como as rastros. O tempo disponibilizado para a tarefa foi de 10 minutos.

Quinze desenvolvedores foram avaliados no estudo, dos quais cinco usaram a técnica indutiva, cinco usaram a técnica criação de links e cinco usaram a técnica matriz de relacionamentos. As técnicas foram atribuídas de forma aleatória. Cada um recebeu um material com instruções sobre como realizar o estudo com a técnica correspondente.

Medidas foram tomadas a fim de produzir dados sem vícios, objetivando evitar possíveis interferências nos resultados. Essas medidas foram adotadas antes, durante e depois do estudo, conforme detalhado a seguir.

Antes da realização do estudo real os desenvolvedores receberam um arquivo de testes com os requisitos para praticarem e tirar dúvidas relacionadas ao funcionamento da técnica e do EA. Isto foi feito no intuito de nivelar o conhecimento dos mesmos.

A configuração (hardware e software) dos computadores do laboratório foi verificada, assegurando que todos possuíam as mesmas características. Pois diferenças nas velocidades dos computadores poderiam interferir no número de elementos criados.

A execução da tarefa de cada desenvolvedor foi gravada em vídeo com o software *ScreenCam*™ [10], com o objetivo de detectar anormalidades, tais como: o usuário realizar outra atividade além do estudo e falha no funcionamento dos softwares.

Para avaliação dos resultados foram contabilizados, para cada um dos desenvolvedores, o número de rastros corretamente criadas, conforme mostrado na Figura 3. Nesta figura observa-se que a média de relações criada com a técnica indutiva (72,6) é significativamente maior do que a média obtida com a técnica matriz de relacionamentos (28,2) e a média obtida com a técnica de criação de links (23,6).

Em função do tamanho da amostra, foram aplicados testes estatísticos de hipóteses [9] para verificar se a diferença encontrada é significativa. O teste de hipóteses foi aplicado primeiramente comparando a técnica indutiva com a matriz de relacionamentos. A hipótese H_0 (hipótese nula) é que em média o desempenho da técnica indutiva é igual ao da técnica matriz de relacionamentos. A hipótese H_1 (hipótese alternativa) é que em média a técnica indutiva possui maior desempenho que a técnica matriz de relacionamentos, para a criação e evolução de rastros.

Maior desempenho, neste trabalho, significa criar o maior número de relações corretas. Relações corretas são aquelas criadas entre elementos que realmente possuem uma relação de dependência. Um erro comum no

mapeamento da rastreabilidade é criar por engano uma relação entre elementos que não possuem dependência (relação falsa).

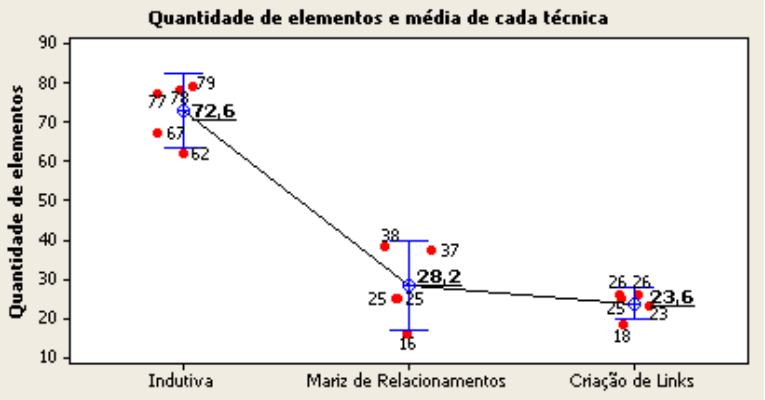

Figura 3. Rastros corretas criadas com cada técnica.

As amostras são independentes, pois as técnicas foram executadas com grupos diferentes. Neste caso aplica-se o teste *t* para amostras independentes. Ao aplicar o teste o resultado foi $8,27$. Com 8 graus de liberdade a probabilidade de significância foi $p=0,02$.

Considerando um nível de significância de 5% ($a=0,05$) tem-se ($p<a$), o que significa que se deve rejeitar H_0 em favor de H_1. Conclui-se, portanto, com 95% de confiança que a técnica indutiva tende a apresentar maior desempenho do que a técnica matriz de relacionamentos. O desempenho melhor da técnica indutiva indica que o esforço despendido para sua utilização pode ser significativamente menor em comparação à técnica matriz de relacionamentos.

O mesmo teste foi realizado comparando o desempenho da técnica indutiva com o da técnica criação de *links*, onde a probabilidade de significância resultou em $p=0$. Conclui-se com 95% de confiança que a técnica indutiva tende a apresentar maior desempenho do que a técnica criação de *links*.

O teste também foi aplicado entre a técnica matriz de relacionamentos e a técnica criação de links, avaliando se as duas técnicas possuíam desempenho distinto. O teste não detectou diferença entre estas duas técnicas.

Conclusão

As principais técnicas de rastreabilidade apresentam limitações. A matriz de relacionamentos, técnica atualmente utilizada nas organizações, torna-se de difícil gerenciamento à medida que o número de artefatos aumenta. A

recuperação automática, apesar de ser uma alternativa para a matriz, pode detectar muitos relacionamentos falsos e ainda não é amplamente utilizada por ferramentas CASE comerciais. As organizações ainda buscam uma forma de criar as rastros que forneça um bom custo-benefício.

A principal vantagem da técnica indutiva comparada a matriz de relacionamentos está no fato de que o desenvolvedor pode criar os relacionamentos de rastreabilidade de forma integrada aos elementos, utilizando apenas o ambiente de criação dos elementos. Já na matriz de relacionamentos o desenvolvedor deve criar os elementos e depois criar as rastros, utilizando para isto dois ambientes, um para a criação de elementos, outro para a criação dos relacionamentos (matriz). A utilização de dois ambientes pode tornar mais trabalhoso e menos eficaz o mapeamento da rastreabilidade.

O estudo realizado indicou que a técnica indutiva tende a apresentar uma maior produtividade em relação às técnicas tradicionais, por meio da criação de um maior número que rastros. O esforço despendido para sua utilização pode ser significativamente menor, tornando a atividade menos penosa, reduzindo o esforço do desenvolvedor. Desta forma futuramente a técnica indutiva poderia ser utilizada nas organizações.

Em relação à técnicas de recuperação automática (como LSI), a técnica indutiva não apresenta as mesmas desvantagens, pois não procura identificar relações a partir de artefatos existentes. Ao invés disso, a técnica indutiva procura criar as rastros no momento da criação dos próprios elementos, o que evita a formação de falsas rastros.

A técnica indutiva, por outro lado, propõe que a tarefa executada pelo desenvolvedor seja redefinida. Ao invés de simplesmente desenhar uma classe em um diagrama, o desenvolvedor deverá deixar claro o porquê desta classe, ou seja, qual o outro elemento que fez com que ele chegasse à conclusão de que tal classe seria necessária.

A técnica indutiva proposta é eficaz apenas em sistemas cuja documentação ainda vai ser produzida, onde se tem a oportunidade de utilizar a técnica desde o início. Ela não detecta relações que deveriam existir entre elementos e que por alguma razão não foram incluídas. Estas são limitações da técnica proposta. Nestas situações, ela pode ser complementada, por exemplo, com a aplicação de recuperação automática.

Referências Bibliográficas

1. Gotel, O.C.Z. e Finkelstein, C.W. (1994) “An analysis of the requirements traceability problem”, Requirements Engineering: Proceedings of the First International Conference, p. 94-101.

2. Jiang, H., Nguyen, T. N.; Chang, C. K. e Dong, F. (2007) "Traceability Link Evolution Management with Incremental Latent Semantic Indexing", In:

Proceedings of the 31st Annual International Computer Software and Applications Conference, IEEE Computer Society, USA, p.309-316.

3. Lucia, A. D., Fasano, F., Oliveto, R., e Tortora, G. (2007) “Recovering traceability links in software artifact management systems using information retrieval methods”, ACM Trans. Softw. Eng. Methodol, USA, p.13.

4. Maletic, J. I., Collard, M. L., e Simoes, B. (2005) “An XML based approach to support the evolution of model-to-model traceability links”, In: Proceedings of the 3rd international Workshop on Traceability in Emerging Forms of Software Engineering, ACM, USA.

5. Neumuller, C. e Grunbacher, P. (2006) “Automating Software Traceability in Very Small Companies: A Case Study and Lessons Learned”, In: Proceedings of the 21st IEEE/ACM international Conference on Automated Software Engineering, IEEE Computer Society, USA, p. 145-156.

6. Oliveto, R., Antoniol, G., Marcus, A. e Hayes, J. (2007) "Software Artefact Traceability: the Never-Ending Challenge", Software Maintenance, IEEE International Conference.

7. Palmer, J. D. (1997) “Traceability”, In: Software Requirements Engineering, R.H. Thayer and M. Dorfman, p. 364-374.

8. Sparx Systems. (2009) "Enterprise Architect". Disponível em: http://www.sparxsystems.com/products/ea. Acesso em: 01 mai. 2009.

9. Barbetta, P. A., Reis, M. M. e Bornia, A. C. “Estatística para Cursos de Engenharia e Informática”. São Paulo: Atlas, 2004.

10. SmartGuyz Incorporatec. (2009) "ScreenCam". Disponível em: http://www.smartguyz.com/index-2.html. Acesso em: 01 mai. 2009.

11. Marcus, A. and Maletic, J. I. (2003). “Recovering documentation-to-source-code traceability links using latent semantic indexing”. In Proceedings of the 25th international Conference on Software Engineering (Portland, Oregon, May 03 - 10, 2003). IEEE Computer Society, Washington, DC, 125-135.

12. Antoniol, G., Canfora, G., Casazza, G., De Lucia, A., and Merlo, E. (2002) “Recovering Traceability Links between Code and Documentation”. IEEE Trans. Softw. Eng. 28, 10 (Oct. 2002), 970-983.

13. Murta, L. G., Hoek, A., and Werner, C. M. (2008) “Continuous and automated evolution of architecture-to-implementation traceability links”. Automated Software Eng. 15, 1 (Mar. 2008), 75-10.

14. Aldrich, J., Chambers, C., and Notkin, D. (2002) “ArchJava: connecting software architecture to implementation”. In: Proceedings of the 24th international Conference on Software Engineering. ICSE '02. ACM, New York.

15. Settimi, R., Cleland-Huang, J., Khadra, O. B., Mody, J., Lukasik, W., and DePalma, C. (2004) “Supporting Software Evolution through Dynamically Retrieving Traces to UML Artifacts”. In: Proceedings of the Principles of Software Evolution, 7th international Workshop. IWPSE. IEEE Computer Society, Washington, DC.

16. Egyed, A., Biffl, S., Heindl, M., and Grünbacher, P. (2005) Determining the cost-quality trade-off for automated software traceability. In Proceedings of the 20th IEEE/ACM international Conference on Automated Software Engineering. ASE '05. ACM, New York, NY, 360-363.

17. Lormans, M. and Van Deursen, A. (2006) Can LSI help Reconstructing Requirements Traceability in Design and Test?. In Proceedings of the Conference on Software Maintenance and Reengineering, CSMR. IEEE Computer Society, Washington, DC, 47-56.

18. Antoniol, G., Cimitile, A., and Casazza, G. (2000) Traceability Recovery by Modeling Programmer Behavior. In Proceedings of the Seventh Working Conference on Reverse Engineering (Wcre'00), WCRE. IEEE Computer Society, Washington, DC, 240.

19. Egyed, A., Biffl, S., Heindl, M., and Grünbacher, P. (2005) Determining the cost-quality trade-off for automated software traceability. In Proceedings of the 20th IEEE/ACM international Conference on Automated Software Engineering ASE '05. ACM, New York, NY, 360-363.

20. Cleland-Huang, J., Zemont, G., Lukasik, W. (2004) "A Heterogeneous Solution for Improving the Return on Investment of Requirements Traceability," Requirements Engineering, IEEE International Conference on, pp. 230-239, 12th IEEE International Requirements Engineering Conference (RE'04).

21. Cleland-Huang, J. (2006) Just Enough Requirements Traceability. In Proceedings of the 30th Annual international Computer Software and Applications Conference - Volume 01 (September 17 - 21, 2006). COMPSAC. IEEE Computer Society, Washington, DC, 41-42.

22. De Lucia, A., Fasano, F., Oliveto, R., and Tortora, G. (2006) Can Information Retrieval Techniques Effectively Support Traceability Link Recovery?. In Proceedings of the 14th IEEE international Conference on Program Comprehension. ICPC. IEEE Computer Society, Washington, DC, 307-316.

23. Almeida, J. P., Eck, P. v., and Iacob, M. (2006) Requirements Traceability and Transformation Conformance in Model-Driven Development. In Proceedings of the 10th IEEE international Enterprise Distributed Object Computing Conference. EDOC. IEEE Computer Society, Washington, DC, 355-366.

24. Jesse Fletcher, Jane Cleland-Huang. (2006) "Softgoal Traceability Patterns," Software Reliability Engineering, International Symposium on, pp. 363-374, 17th International Symposium on Software Reliability Engineering (ISSRE'06).

25. De Lucia, A., Oliveto, R., Zurolo, F., and Di Penta, M. (2006) Improving Comprehensibility of Source Code via Traceability Information: a Controlled Experiment. In Proceedings of the 14th IEEE international Conference on Program Comprehension. ICPC. IEEE Computer Society, Washington, DC, 317-326.

ANEXO B –Material do Estudo - Técnica Indutiva

1. Técnica Indutiva

Dado um conjunto de requisitos e casos de uso a tarefa do estudo consiste em realizar a criação de rastros entre os mesmos, utilizando a ferramenta CASE Enterprise Architect (EA). As rastros devem ser criadas utilizando a **Técnica Indutiva**, que é uma das formas de realizar a criação de rastros.

As próximas seções contem informações sobre como o estudo será realizado. A seção 1.1 contém as informações gerais do estudo; a seção 1.2 apresenta a ferramenta CASE Enterprise Architect (EA); e a seção 1.3 apresenta a técnica.

1.1. Observações Gerais do Estudo

Para a realização do estudo devem ser observados alguns itens detalhados a seguir:

- O tempo máximo para a realização do estudo é que 10 minutos;
- Será utilizada a ferramenta CASE Enterprise Architect (EA), apresentada na seção 1.2.
- Com base nos requisitos, previamente cadastrados no EA, devem ser criados os casos de uso e as rastros correspondentes, utilizando os passos da técnica descrita na seção 1.3.
- A correspondência entre os requisitos e os casos de uso é apresentada na **Tabela 1**. Cada linha da tabela contém um requisito e um caso de uso que devem ser relacionados através de uma relação de rastreabilidade. No total são 80 requisitos e 80 casos de uso. Para cada requisito será criada uma relação de rastreabilidade com um caso de uso.

- No final do estudo somente serão considerados como válidos os casos de uso que possuírem uma relação de rastreabilidade com o seu requisito correspondente na Tabela 1.

A seguir, é apresentada a Tabela 1 que contém a correspondência entre os requisitos e casos de uso utilizados neste estudo.

Tabela 1- Correspondência entre requisitos e casos de uso

Requisitos	**Casos de Uso**
RF001. Cadastrar cursos	Cadastrar cursos
RF002. Cadastrar alunos	Cadastrar alunos
RF003. Cadastrar turmas	Cadastrar turmas
RF004. Cadastrar unidades	Cadastrar unidades
RF005. Cadastrar disciplinas	Cadastrar disciplinas
RF006. Cadastrar área de conhecimento	Cadastrar área de conhecimento
RF007. Cadastrar programa	Cadastrar programa
RF008. Cadastrar modalidade	Cadastrar modalidade
RF009. Cadastrar avaliações	Cadastrar avaliações
RF010. Cadastrar notas	Cadastrar notas
RF011. Cadastrar freqüências	Cadastrar freqüências
RF012. Corrigir avaliações discursivas	Corrigir avaliações discursivas
RF013. Cadastrar categorias	Cadastrar categorias
RF014. Cadastrar convênios	Cadastrar convênios
RF015. Cadastrar contratos	Cadastrar contratos
RF016. Cadastrar usuários	Cadastrar usuários
RF017. Exibir relatório de acessos	Exibir relatório de acessos
RF018. Bloquear alunos	Bloquear alunos
RF019. Desbloquear alunos	Desbloquear alunos
RF020. Exibir relatório de usuários por perfil	Exibir relatório de usuários por perfil

RF021. Cadastrar termos de adesão	Cadastrar termos de adesão
RF022. Gerenciar alunos inscritos	Gerenciar alunos inscritos
RF023. Gerenciar alunos selecionados	Gerenciar alunos selecionados
RF024. Gerenciar alunos desistentes	Gerenciar alunos desistentes
RF025. Gerenciar alunos evadidos	Gerenciar alunos evadidos
RF026. Gerenciar alunos concluídos	Gerenciar alunos concluídos
RF027. Movimentar alunos nas turmas	Movimentar alunos nas turmas
RF028. Exibir cursos em andamento	Exibir cursos em andamento
RF029. Exibir cursos concluídos	Exibir cursos concluídos
RF030. Exibir cursos disponíveis	Exibir cursos disponíveis
RF031. Acessar curso	Acessar curso
RF032. Realizar avaliação (prova)	Realizar avaliação (prova)
RF033. Exibir notas	Exibir notas
RF034. Exibir freqüência	Exibir freqüência
RF035. Cadastrar certificado	Cadastrar certificado
RF036. Exibir certificado	Exibir certificado
RF037. Exibir histórico escolar	Exibir histórico escolar
RF038. Exibir relatório de movimentação de estudantes na turma	Exibir relatório de movimentação de estudantes na turma
RF039. Exibir relatório de estudantes	Exibir relatório de estudantes
RF040. Cadastrar dúvida	Cadastrar dúvida
RF041. Responder dúvida	Responder dúvida
RF042. Publicar conteúdo	Publicar conteúdo
RF043. Publicar material de apoio	Publicar material de apoio

RF044. Publicar avisos	Publicar avisos
RF045. Exibir avisos	Exibir avisos
RF046. Publicar enquete	Publicar enquete
RF047. Votar na enquete	Votar na enquete
RF048. Exibir resultado da enquete	Exibir resultado da enquete
RF049. Publicar agenda	Publicar agenda
RF050. Acessar agenda	Acessar agenda
RF051. Publicar FAQ	Publicar FAQ
RF052. Exibir FAQ	Exibir FAQ
RF053. Publicar item na biblioteca	Publicar item na biblioteca
RF054. Acessar item na biblioteca	Acessar item na biblioteca
RF055. Solicitar inscrição Pessoa Física	Solicitar inscrição Pessoa Física
RF056. Solicitar inscrição Pessoa Jurídica	Solicitar inscrição Pessoa Jurídica
RF057. Autorizar matrícula	Autorizar matrícula
RF058. Confirmar matrícula	Confirmar matrícula
RF059. Gerar boleto Banco do Brasil	Gerar boleto Banco do Brasil
RF060. Gerar boleto Caixa Econômica Federal	Gerar boleto Caixa Econômica Federal
RF061. Gerar boleto Bradesco	Gerar boleto Bradesco
RF062. Gerar boleto Santander	Gerar boleto Santander
RF063. Gerar boleto HSBC	Gerar boleto HSBC
RF064. Cadastrar boleto	Cadastrar boleto
RF065. Gerar parcelas	Gerar parcelas
RF066. Realizar conciliação de boletos	Realizar conciliação de boletos
RF067. Relatório de parcelas pagas	Relatório de parcelas pagas
RF068. Relatório de parcelas pendentes	Relatório de parcelas pendentes

RF069. Gerar contrato	Gerar contrato
RF070. Configurar módulo preferencial	Configurar módulo preferencial
RF071. Exibir relatório de matrículas por período	Exibir relatório de matrículas por período
RF072. Exibir relatório de alunos por status	Exibir relatório de alunos por status
RF073. Exibir relatório de acompanhamento geral	Exibir relatório de acompanhamento geral
RF074. Exibir relatório de estatísticas de acesso	Exibir relatório de estatísticas de acesso
RF075. Cadastrar pré-requisitos	Cadastrar pré-requisitos
RF076. Importar turma	Importar turma
RF077. Importar usuários	Importar usuários
RF078. Exibir relatório de alunos inscritos	Exibir relatório de alunos inscritos
RF079. Exibir relatório de alunos selecionados	Exibir relatório de alunos selecionados
RF080. Exibir relatório de alunos desistentes	Exibir relatório de alunos desistentes

1.2. Enterprise Architect (EA)

O estudo será realizado na ferramenta CASE EA, neste estudo serão utilizadas, principalmente, três áreas do EA: *Toolbox*, *Diagram View* e *Project Browser*. Conforme pode ser observado na **Figura** .

Toolbox: é a área onde estão disponíveis os elementos do artefato selecionado. É utilizada para inserir um novo elemento na *Diagram View*.

Diagram View: é a área utilizada para visualizar e editar elementos do artefato selecionado. Neste caso serão visualizados o artefato de requisitos e o artefato de casos de uso.

Project Browser: é a área que apresenta a hierarquia entre pacotes, artefatos e elementos dos artefatos. Ao selecionar um artefato na *Project Browser* o mesmo será exibido na *Diagram View*.

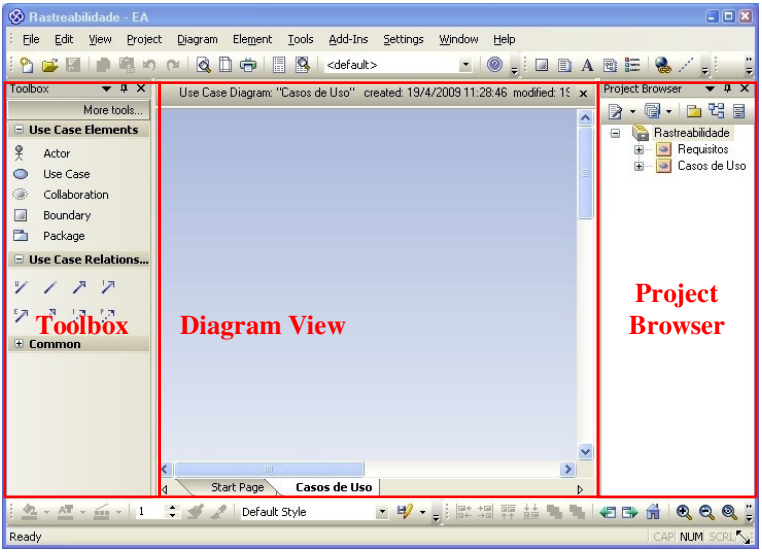

Figura 1- Áreas do EA: *Toolbox*, *Diagram View* e *Project Browser*

1.3. Técnica Indutiva

A técnica indutiva é um plugin desenvolvido para criar rastros no EA. Para criar casos de uso e rastros utilizando a técnica indutiva devem ser seguidos os passos detalhados a seguir:

1. Abrir o arquivo do EA 'Rastreabilidade.eap'
2. No *Project Browser* abrir o pacote 'Requisitos'.
3. Abrir o artefato 'Requisitos'.

4. No *Project Browser* fechar o pacote ‘Requisitos’ (para facilitar a localização do artefato que ‘Casos de Uso’ durante a realização do estudo).

[A partir deste ponto repetir os passos para cada requisito]

5. No *Project Browser* abrir o pacote ‘Casos de Uso’.
6. Selecionar o artefato ‘Casos de Uso’ (caso ainda não esteja selecionado).
7. Na área de *Diagram View* do artefato ‘Requisitos’ escolher um requisito e clicar com o botão direito sobre o mesmo, em seguida selecionar a opção ‘Add-Ins/Derivar Caso de Uso’.
8. O EA abre a área de *Diagram View* do artefato ‘Casos de Uso’ mostrando o caso de uso recém criado (nesta técnica os casos de uso são criação todos na mesma posição, os mesmo não precisam ser reposicionados). Neste passo a relação de rastreabilidade foi criada juntamente com o caso de uso.
9. Na parte inferior da área de *Diagram View* voltar para o artefato ‘Requisitos’, clicando na aba ‘Requisitos’.
10. Repetir os passos 5, 6, 7, 8 e 9 para cada um dos requisitos presentes no artefato ‘Requisitos’.