Uma Ferramenta para Apoio à Rastreabilidade de Software

Eduardo Kreuch[1,3], Jean Carlo Rossa Hauck[2], Alessandra Zoucas[3]

[1]Specto Tecnologia.
São José/SC – Brasil

[2] Departamento Informática e Estatística - Universidade Federal de Santa Catarina.
Florianópolis, Santa Catarina, Brasil

[3]Univali – Universidade do Vale do Itajaí – Campus Florianópolis.
Florianópolis/SC – Brasil

{eduardo.kreuch@specto.com.br, jeanhauck@inf.ufsc.br,
alessandrazoucas@gmail.com

Abstract. *Maintaining traceability between the various software development artifacts is often a difficult task and prone to failures, generating resistance from developers in maintaining this type of documentation. Tools that support this kind of activity are, in general, not designed for developers. This paper presents the development of an integration between an IDE and a modelling tool in order to facilitate the maintenance of vertical traceability. The developed plug-in is applied in a case study in a software development company and the observed results and lessons learned indicate that the plug-in facilitates the maintenance of vertical traceability.*

Resumo. *Manter a rastreabilidade entre os diversos artefatos no desenvolvimento de software é, muitas vezes, uma tarefa difícil e sujeita a falhas, gerando resistências por parte dos desenvolvedores na manutenção desse tipo de documentação. Ferramentas que apoiam essa atividade em geral não são voltadas especificamente para os desenvolvedores. Este artigo apresenta o desenvolvimento de uma integração entre uma IDE e uma ferramenta de modelagem para facilitar a manutenção da rastreabilidade vertical. O plug-in desenvolvido é aplicado em um estudo de caso em uma empresa de desenvolvimento de software e os resultados observados e lições aprendidas apontam indícios de que o plug-in facilita a manutenção da rastreabilidade vertical.*

1. Introdução

Um dos grandes desafios das organizações de software é manter o controle sobre os inúmeros artefatos gerados durante o processo de desenvolvimento de software e relacioná-los entre si. Este relacionamento entre os artefatos é importante, pois ajuda a identificar quais artefatos poderão sofrer impacto em termos de esforço, tempo ou custo, como resposta a mudanças solicitadas pelos clientes.

O gerenciamento dessas relações existentes entre os artefatos resultantes do processo de desenvolvimento de software deve ser baseado em informações atualizadas e precisas, com documentos que registram ou fornecem tais informações e sejam fáceis de ser rastreados [Beizer 1990]. A esse conjunto de informações dá-se o nome de rastreabilidade de software [IEEE 2014].

Dois tipos básicos de rastreabilidade de software são tipicamente utilizados: a rastreabilidade horizontal e a rastreabilidade vertical. A rastreabilidade horizontal preocupa-se com as relações existentes entre artefatos de mesmo tipo ou de mesmo nível de abstração, enquanto a rastreabilidade vertical registra as relações existentes entre artefatos de tipos diferentes ou as relações entre as partes que compõem um mesmo artefato [SOFTEX 2013].

Uma vez que abrange o controle de um grande número de associações entre artefatos, tipos de informações e pessoas envolvidas [Ramesh e Jarke 2001], a implantação e manutenção da rastreabilidade tende a ser complexa e cara [Cleland-Huang et al. 2003]. Além disso, a manutenção da rastreabilidade depende dos membros da equipe que podem se mostrar relutantes em manter a documentação atualizada [Richardson e Green 2004]. Ainda assim, é recomendável que os próprios membros da equipe mantenham a rastreabilidade dos artefatos sob sua responsabilidade [Tortora et al 2007].

A ferramenta Enterprise Architect[1] (EA) suporta modelagem em notação UML2 [OMG 2011] e cobre os principais aspectos do ciclo de vida do desenvolvimento de uma aplicação, oferecendo suporte ao controle de mudanças e rastreabilidade, em especial através do da visão: Matriz de Rastreabilidade. A Matriz de Rastreabilidade do EA é muito semelhante ao modelo de referência cruzada [Evans 1989], o qual mantém os relacionamentos através de uma tabela, na qual as linhas indicam um tipo de artefato e as colunas outro, permitindo documentar relacionamentos entre elas nos cruzamentos entre linhas e colunas. No entanto, o uso dessa matriz em projetos de grande porte, onde milhares de artefatos (especialmente códigos-fonte) estão envolvidos, tem sido percebida na prática como complexa e de difícil manutenção.

Nesse sentido, o presente trabalho propõe o desenvolvimento e aplicação de um plug-in que possa ser instalado na IDE (*Integrated Development Environment*) utilizada pelos desenvolvedores de forma a facilitar a manutenção da rastreabilidade vertical para os códigos-fonte.

O restante deste artigo está estruturado da seguinte forma: a Seção 2 apresenta a abordagem metodológica utilizada; a Seção 3 descreve o desenvolvimento do Plug-in; na Seção 4 o estudo de caso do uso do plug-in é apresentado, e; na Seção 5 as principais conclusões são apresentadas.

[1] Enterprise Architect é marca registrada da Sparx Systems: http://www.sparxsystems.com

2. Abordagem metodológica

Por se tratar de uma pesquisa aplicada, do ponto de vista de sua natureza [Gil 1999], esse trabalho utiliza estudo de caso na sua abordagem metodológica de pesquisa. Um estudo de caso, segundo [Runeson e Host 2009] é "a investigação de fenômenos contemporâneos em seu contexto".

Assim, esse trabalho foi realizado seguindo os passos apresentados na Figura 1, tomando por base [Runeson e Host 2009]. Cada um dos passos é brevemente explicado na sequência.

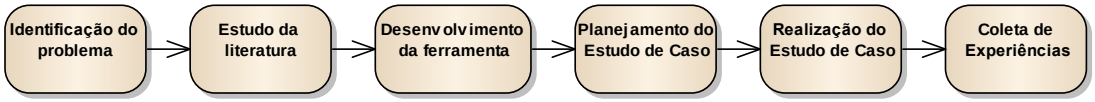

Figura 1: Abordagem metodológica

Identificação do Problema

Durante a implementação de melhoria de processos em uma empresa de desenvolvimento de software da grande Florianópolis[2] para obtenção do nível F do modelo MR-MPS-SW, foram observadas pelos autores deste trabalho, dificuldades na implementação da rastreabilidade vertical necessária a diversos processos do nível F, como Gerência de Requisitos e Gerência de Configuração. À medida que os projetos foram evoluindo, maior esforço foi sendo exigido dos envolvidos no sentido de manter a matriz de rastreabilidade atualizada, gerando insatisfação e resistência da equipe.

Estudo da Literatura

Como forma de entender melhor o problema, foi realizado um estudo da literatura, buscando alternativas que pudessem resolver o problema. Esse estudo também teve a intenção de contextualizar todos os autores, pois nem todos possuíam experiência na aplicação dos conceitos relativos à Gerência de Configuração de Software.

Desenvolvimento da Ferramenta

A partir desse entendimento, como a empresa já utilizava há bastante tempo a ferramenta EA e a IDE Eclipse[3], a solução proposta foi a de desenvolver um plug-in para o Eclipse que pudesse facilitar a manutenção da rastreabilidade vertical entre código-fonte e scripts de banco de dados, por exemplo, e os casos de uso relacionados. Um resumo de como foi desenvolvido o plug-in é apresentado na seção 3 deste artigo.

Planejamento, Execução do Estudo de Caso e Coleta de Experiências

Após o desenvolvimento do plug-in para o Eclipse, este foi aplicado na forma de um estudo de caso na empresa, com o objetivo de obter uma avaliação da utilidade do plug-in e se ele realmente reduzia a dificuldade de se manter a rastreabilidade vertical a

[2] O nome da empresa é preservado para atendimento do critério de submissão anônima.

[3] https://www.eclipse.org/

partir do código-fonte. O planejamento e a realização do estudo de caso são descritos na seção 4 deste artigo.

3. Desenvolvimento do Plug-in

Conforme citado, a solução proposta para tentar reduzir as dificuldades na manutenção da rastreabilidade vertical consiste em integrar as ferramentas EA e Eclipse, possibilitando a geração semi-automatizada de ligações de rastreabilidade entre casos de uso documentados no EA e o código-fonte escrito no Eclipse. O plug-in desenvolvido e o seu código-fonte podem ser baixados em: https://github.com/grupospecto/eatplugin.

Após a análise de algumas possibilidades de integração entre as ferramentas, foi decidido que a integração seria realizada por meio do banco de dados, pois o EA possibilita o armazenamento dos modelos dessa forma, sendo que na empresa é utilizado o banco de dados Oracle[4] para esse fim. Assim, o primeiro passo necessário foi o estudo do modelo de dados do EA, no sentido de entender como os objetos modelados na ferramenta (os *UseCases*, por exemplo) são armazenados no banco de dados e em quais entidades as informações de rastreabilidade são registradas. Essa etapa consumiu a maior parte do esforço de análise e modelagem da solução.

Vencida essa etapa, a integração foi desenvolvida como um plug-in do Eclipse, com o objetivo de facilitar o acesso do desenvolvedor à funcionalidade. A partir do Eclipse o desenvolvedor pode invocar um menu de contexto que exibe uma lista de casos de uso de projetos existentes na base de dados EA.

A Figura 2 ilustra o ambiente de desenvolvimento do Eclipse, já com a opção "Matriz de Rastreabilidade" do *plug-in* desenvolvido. Nesse cenário, o programador está escrevendo uma classe no código-fonte do sistema que irá satisfazer – ou realizar segundo a UML – um ou mais casos de uso do projeto, já previamente documentados no EA.

O uso do *plug-in* funciona da seguinte forma: Durante a codificação, o programador deseja criar uma nova associação entre a classe com a qual está trabalhando e um caso de uso existente. Nesse momento ele clica com o botão direito do mouse sobre a classe desejada e seleciona a opção Matriz de Rastreabilidade, conforme ilustrado na Figura 2.

[4] Oracle é banco de dados de propriedade da *Oracle Corporation*: http://www.oracle.com/

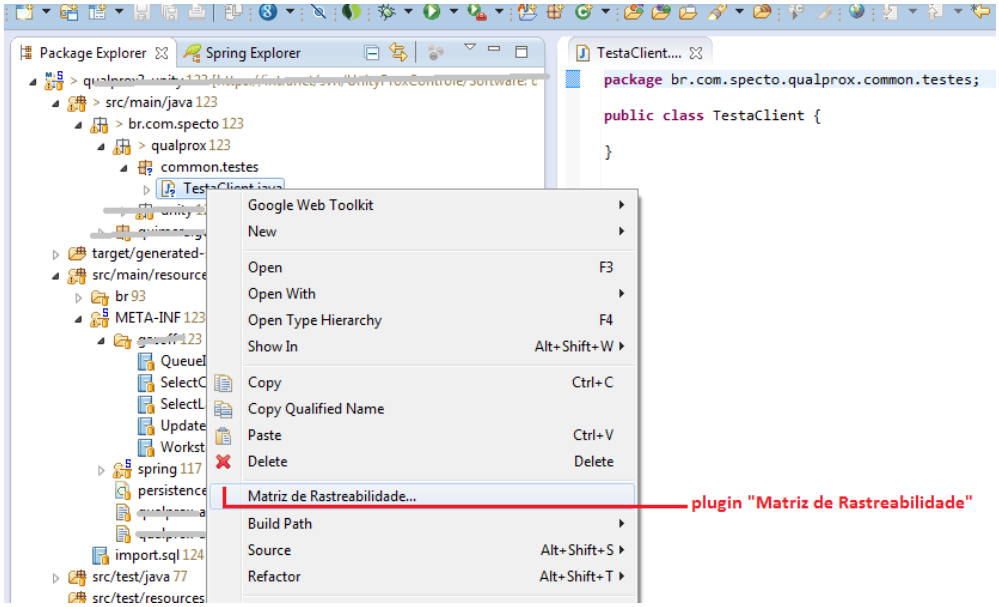

Figura 2. Menu disponibilizado pelo *plug-in* "Matriz de Rastreabilidade"

O *plug-in* neste momento conecta com o banco de dados do EA, e por meio de um comando SQL, lista todos os casos de uso cadastrados, já pré-selecionando os casos de uso que estavam relacionados àquela classe, permitindo ao programador a seleção de um ou mais deles, conforme mostra a Figura 3.

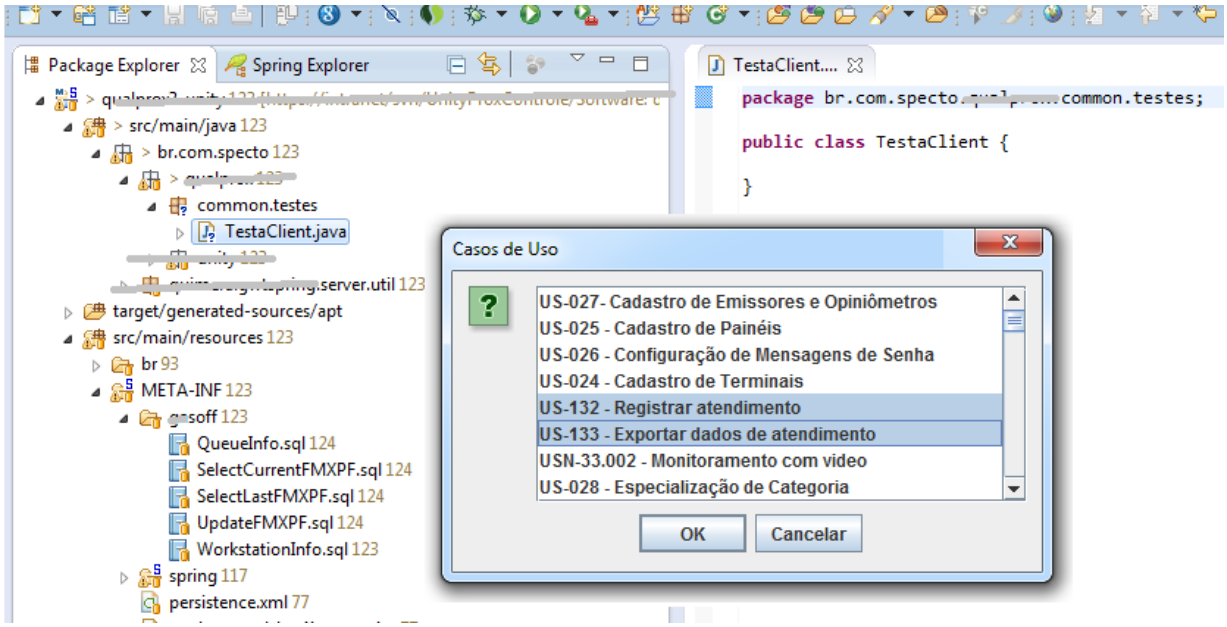

Figura 3. Caixa de diálogo para seleção de casos de uso

Ao confirmar a ação, o *plug-in* efetua a ligação entre a classe e os casos de uso selecionados. Nesse momento ocorre a integração com o EA, onde é gravado o link de rastreabilidade no banco de dados, por meio de comandos SQL, em duas etapas:

a) Primeiramente, apaga-se qualquer ligação pré-existente de casos de uso com a classe selecionada:

```
DELETE FROM TABELA_CONEXAO[5] WHERE ARTEFATO_ORIGEM = <classe-selecionada>;
```

b) Em seguida, dentro de uma estrutura de laço, para cada caso de uso selecionado, cria-se uma ligação da classe sendo editada para o caso de uso:

```
INSERT INTO TABELA_CONEXAO (..., ARTEFATO_ORIGEM, ARTEFATO_DESTINO, ...) VALUES (..., <classe-selecionada>, <caso-de-uso-selecionado>);
```

Uma vez que a operação no banco de dados do EA tenha sido bem-sucedida, o *plug-in* termina por escrever um comentário de código, no estilo JavaDoc[6], na classe sendo editada. Este comentário possui uma *tag* de identificação recuperável: "`<UseCases>`", que contém uma lista dos códigos de casos de uso para os quais foram criadas ligações na Matriz de Rastreabilidade do Enterprise Architect. A Figura 4 mostra o resultado da anotação na classe.

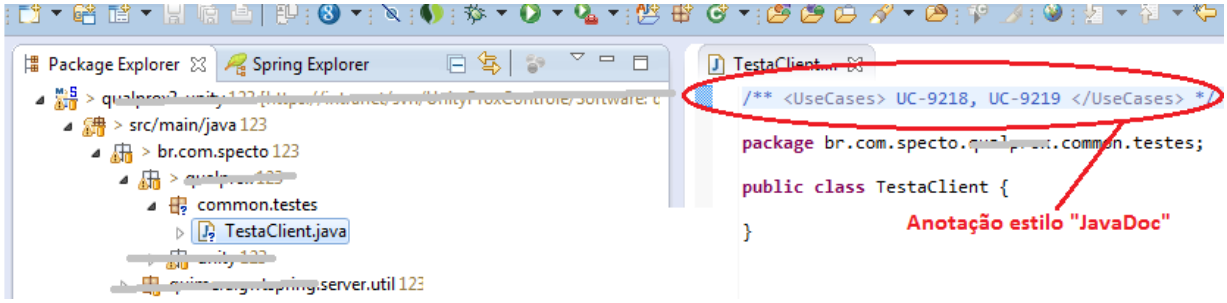

Figura 4. Comentário na classe, garantindo a rastreabilidade bidirecional.

O resultado pode ser visto na Matriz de Rastreabilidade do EA (Figura 5).

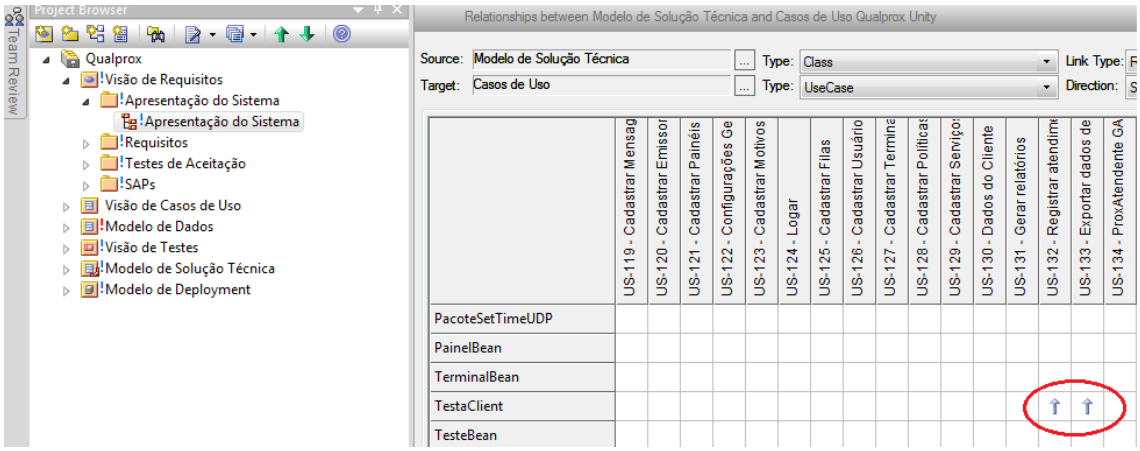

Figura 5. Matriz de Rastreabilidade ligando a classe "TestaClient" aos casos de uso "US-132" e "US-133", selecionados a partir do *plug-in*, no Eclipse

[5] O nome real da tabela é alterado para preservar o modelo de dados do EA nesta publicação.

[6] http://www.oracle.com/technetwork/java/javase/documentation/javadoc-137458.html

Este *plug-in* foi instalado no ambiente de desenvolvimento da empresa Specto Tecnologia[7], onde são utilizados há vários anos as ferramentas EA e Eclipse. A Seção a seguinte apresenta o estudo de caso de utilização do plug-in desenvolvido.

4. Estudo de Caso

Contextualizado o estudo de caso, a empresa Specto está localizada na cidade de São José/SC e possui três divisões de produtos, sendo que a divisão DGA desenvolve soluções tecnológicas de software e hardware para gestão de atendimento, possuindo mais de 3500 soluções já implantadas no Brasil, tendo obtido o nível F do MR-MPS-SW em 2012. Os softwares são desenvolvidos utilizando linguagem de programação principalmente Java, tanto para ambientes Web quanto *Desktop*.

O estudo de caso realizado teve como objetivo principal, responder à pergunta: "O uso do plug-in desenvolvido reduz as dificuldades de manutenção da rastreabilidade vertical no contexto dos projetos de desenvolvimento Java da empresa?". O estudo foi aplicado no período de dois meses, envolvendo como população todos os desenvolvedores Java dos projetos atualmente em execução, autorizados pela gerência a participarem do estudo.

Dois projetos de software estavam em desenvolvimento no período do estudo e foram utilizados para aplicação, ambos em Java, utilizando frameworks diferentes, sendo um deles contendo também um módulo desenvolvido em Java para desktop. Como todos os desenvolvedores possuíam prática no uso de *plug-ins* do Eclipse, os desenvolvedores foram rapidamente treinamentos na instalação e uso do *plug-in* e foram liberados para uso nos projetos que estavam desenvolvendo.

O perfil dos desenvolvedores demonstra que a maioria deles (67%) possui curso superior, sendo que possuem, em média, três anos de experiência no desenvolvimento de software e todos possuíam conhecimento básico sobre o uso da rastreabilidade no desenvolvimento de software.

Após duas semanas de uso do *plug-in*, foi submetido aos desenvolvedores, como instrumento de coleta de dados, um questionário formado por questões fechadas, utilizando escala Likert [Chang 1993] e duas questões abertas, que procuravam avaliar:

- A possível facilidade de manutenção da rastreabilidade comparando-se com o uso somente do EA, definida em termos de facilidade de instalação e uso intuitivo;
- A percepção subjetiva dos possíveis benefícios do seu uso, tanto em termos do tempo de preenchimento da matriz de rastreabilidade, quanto da assertividade na execução desta tarefa;
- Questões abertas onde os participantes eram convidados a avaliar os pontos fortes e os pontos fracos do *plug-in*.

[7] http://www.specto.com.br/

Como resultado da aplicação do formulário, quase todos os desenvolvedores responderam que o uso do plug-in facilita a rastreabilidade vertical, conforme pode ser observado na Tabela 1.

Tabela 1. Resultado consolidado da avaliação do plug-in pelos desenvolvedores

Característica	Concordam (%)
É de fácil instalação	83
É de uso intuitivo	100
Economiza tempo	100
Reduz margem de erro humano	100

4.1 Lições Aprendidas

Após a execução do estudo de caso, foram identificadas as seguintes lições aprendidas tanto no desenvolvimento quanto no uso do plug-in:

- Rastreabilidade vertical até o código-fonte não é fácil: especialmente em equipes de projeto pequenas, manter a rastreabilidade vertical não é fácil, o que acaba levando esse tipo de processo a ser colocado de lado, gerando repetidas não-conformidades.
- Automatização é a solução: acostumados com o uso de ferramentas que automatizam boa parte dos processos manuais, especialmente de *deployment* de software, os desenvolvedores são normalmente resistentes à implantação de processos que aumentam a "burocracia". O uso de ferramentas que automatizem alguma parte desses processos é muito bem visto pelos desenvolvedores;
- Envolver os desenvolvedores na melhoria do processo: o envolvimento dos desenvolvedores em todas as etapas da melhoria de processos é fundamental. Quando foi implantado na empresa um processo alinhado ao MR-MPS-SW, a rastreabilidade vertical foi imposta à equipe, como necessidade para a avaliação oficial do MPS.BR, o que gerou muitas resistências. Com o desenvolvimento do plug-in sendo participativo e envolvendo diretamente a equipe de desenvolvimento, a recepção foi muito diferente, sendo a ideia abraçada logo de início;
- Compatibilidade com versões do Eclipse: vale relatar que houve problemas em dois casos, com a versão "Juno" do Eclipse, o que motiva uma revisão no sentido de tornar a solução compatível com todas as versões ou, pelo menos, com as mais recentes. Para os dois casos citados, os participantes optaram por não responder ao item, uma vez que "não foi possível emitir opinião", mas deixando observação por escrito como "ponto fraco", valendo o relato como sugestão de melhoria/correção de bug.

5. Conclusão

Este trabalho apresenta as dificuldades enfrentadas por uma equipe de desenvolvimento em manter a documentação rastreabilidade vertical entre os casos de uso e os códigos-fonte dos projetos mantidos em uma ferramenta de modelagem e documentação de software. Como solução, um plug-in para a IDE utilizada pela equipe é desenvolvido com intenção de facilitar a manutenção da rastreabilidade.

O plug-in é utilizado em um estudo de caso e os resultados e lições aprendidas coletados indicam que o seu uso facilita a manutenção da rastreabilidade vertical entre os casos de uso e os códigos-fonte.

Como trabalhos futuros, planeja-se expandir as funcionalidades do plug-in, pois nesta versão inicial, o programador seleciona casos de uso para uma classe de cada vez. Como melhoria seria importante também oferecer funcionalidades adicionais como: selecionar um determinado pacote e anotar todas as classes daquele pacote de uma única vez e listar as classes modificadas recentemente, ou ainda não atualizadas, e sugerir a associação a casos de uso.

Referências

BEIZER, B. (1990) Software Testing Techniques. Dreamtech Press, New York.

CHANG, L. (1993) Using Confirmatory Factor Analysis of Multitrait-multimethod Data to Assess the Psychometrical Equivalence of 4-point and 6-point Likert type Scales. Proceeding of the Annual Meeting of the National Council on Measurement in Education, ERIC, Atlanta.

CLELAND-HUANG, J.; CHANG, C. K.; CHRISTENSEN, M. (2003) Event-based traceability for managing evolutionary change. Software Engineering, IEEE Transactions on, v. 29, n. 9, p. 796-810.

EVANS, M. W. (1989) The Software Factory: a fourth generation software engineering environment. London: Wiley–Interscience, 332 p.

GIL, A. C. (1999) Métodos e técnicas de pesquisa social. 5. ed. São Paulo: Atlas.

IEEE, C. S. (2014) SWEBOK - Guide to the Software Engineering Body of Knowledge. California, IEEE Computer Society.

OMG (2011) Unified Modeling Language (OMG UML), Superstructure, V2.4.1. Technical Report formal/2011-08-06. OMG – Object Management Group.

RAMESH, B.; JARKE, M. (2001) Toward Reference Models for Requirements Traceability IEEE Transactions on Software Engineering, Vol. 27, No. 1, January 2001.

RICHARDSON, J.; GREEN, J. (2004) Automating Traceability for Generated Software Artifacts, pp.24-33, 19th IEEE International Conference on Automated Software Engineering (ASE'04).

RUNESON, P.; HÖST, M. (2009) Guidelines for Conducting and Reporting Case Study Research in Software Engineering. Empirical software engineering, v. 14, n. 2, p. 131-164.

SOFTEX (2013) Guia de Implementação – Parte 1: Fundamentação para Implementação do Nível F do MR-MPS-SW:2012. SOFTEX. Disponível em: <http://www.softex.br/wp-content/uploads/2013/07/MPS.BR_Guia_de_Implementacao_Parte_2_2013.pdf> Acessado em 11 set. 2013.

TORTORA, G.; DE LUCIA, A.; FASANO, F.; OLIVETO, R. (2007) Recovering traceability links in software artifact management systems using information retrieval methods. ACM Trans. Softw. Eng. Methodol. 2007, Vol. 16.