UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO
DEPARTAMENTO DE ENG. DE COMPUTAÇÃO E AUTOMAÇÃO INDUSTRIAL

Uma ferramenta para suporte à documentação e rastreabilidade da informação de um processo de teste de software.

Dissertação de Mestrado

Autor: **Jorge Luiz da Cruz**
Orientador: **Prof. Dr. Mario Jino**
Co-orientador: **Prof. Dr. Adalberto Nobiato Crespo**

Dissertação submetida à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas, como parte dos requisitos para obtenção do Título de Mestre em Engenharia Elétrica na área de Engenharia de Computação.

Campinas, agosto de 2009

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA E ARQUITETURA - BAE - UNICAMP

Cruz, Jorge Luiz da
C889f Uma ferramenta para suporte à documentação e rastreabilidade da informação de um processo de teste de software / Jorge Luiz da Cruz. --Campinas, SP: [s.n.], 2009.

Orientadores: Mario Jino, Adalberto Nobiato Crespo.
Dissertação de Mestrado - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Software - testes. 2. Documentação. 3. Rastreabilidade. 4. Software - Documentação. I. Jino, Mario. II. Crespo, Adalberto Nobiato. III. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. IV. Título.

Título em Inglês: A tool to support documentation and traceability of information elements in a software testing process
Palavras-chave em Inglês: Software testing, Documentation, Traceability, Documentation Software
Área de concentração: Engenharia de Computação
Titulação: Mestre em Engenharia Elétrica
Banca examinadora: José Mário De Martino, Marcelo Fantinato
Data da defesa: 24/08/2009
Programa de Pós Graduação: Engenharia Elétrica

COMISSÃO JULGADORA - TESE DE MESTRADO

Candidato: Jorge Luiz da Cruz

Data da Defesa: 24 de agosto de 2009

Título da Tese: "Uma Ferramenta para Suporte à Documentação e Rastreabilidade da Informação de um Processo de Teste de Software"

Prof. Dr. Mario Jino (Presidente): _____

Prof. Dr. Marcelo Fantinato: _____

Prof. Dr. José Mario De Martino: _____

Agradecimentos

À Deus, por me permitir estar aqui hoje e pela indicação dos caminhos a seguir;

Aos meus pais, João e Maria, que sempre me apoiaram e incentivaram na busca por novos conhecimentos e experiências e me deram condições de chegar aonde cheguei e ir além... Amo-os muito! Vocês são os melhores pais do mundo!

À minha querida esposa Elisa Maris, que ao longo deste desafio esteve sempre ao meu lado me incentivando nos momentos difíceis e apoiando com muito amor, carinho e paciência. Te amo muito!

À minha filha Laura, recém chegada neste cenário!

Ao meu sogro e minha sogra, novos pais que a vida me trouxe; sempre por perto em todos os momentos.

À família, minha fonte primordial de inspiração!

À professora Juliana Herbert, minha orientadora na Graduação, por ter acreditado em mim e dado o "empurrãozinho" que faltava para minha mudança para São Paulo;

Aos verdadeiros amigos, pelas horas de prazerosa companhia e proveitosa conversa, que de alguma forma me ajudaram na elaboração do conteúdo deste trabalho, com palavras de apoio e dicas valiosas.

Aos meus amigos e colegas de trabalho no CTI (Antigo CenPRA), que me acolheram e se tornaram uma nova família; em especial ao Grupo de Teste de Software.

Aos professores Mario Jino e Adalberto Crespo, respectivos orientador e co-orientador, pela exigência e paciência, que permitiram a concretização deste trabalho;

Aos amigos Miguel Argollo Jr., José Gonzaga de Souza Jr., Francisco Formoso Primo, Angelina Oliveira, Dalton, Rachel e Paulo Sarli, por sua amizade e orientação informal.

A todos vocês, meu carinho e gratidão.

"Aprender é descobrir aquilo que você já sabe.
Fazer é demonstrar que você o sabe.
Ensinar é lembrar aos outros que eles sabem tanto quanto você."
Richard Bach

"Qualquer idéia poderosa é de todo fascinante e de todo inútil até resolvermos usá-la."
Richard Bach

Resumo

Software possui um papel fundamental em inúmeras aplicações e a qualidade de produtos de software é crucial na competitividade das empresas produtoras de software, que dedicam esforço crescente na busca por boa qualidade nos seus produtos. Neste contexto, o teste é reconhecido como um processo fundamental para alcançar este objetivo; entretanto, testar software sistematicamente não é uma tarefa trivial. Para que o teste tenha efetividade e eficácia, além dos aspectos relacionados à avaliação do software, ele deve fazer uso de documentação de boa qualidade: toda a informação registrada deve estar atualizada e consistente; além disso, informação associada deve ser rastreável. É proposto um modelo de rastreabilidade para a informação contida na documentação do processo de teste; o modelo de dados desenvolvido dá suporte tanto à documentação baseada na norma IEEE Std 829-1998, como à rastreabilidade de toda informação associada; um protótipo de ferramenta foi desenvolvido para implementar o modelo de dados e o modelo de rastreabilidade.

Abstract

Software plays a key role in many applications and quality of software products is crucial in the competitivity of software development companies, which are increasingly putting effort in the quest for good quality in their products. In this context, testing is recognized as a key process to achieve this goal; however, systematic software testing is not an easy activity. For testing to have effectiveness and efficacy, in addition to product evaluation aspects, it must make use of good quality documentation: all the recorded information must be up to date and consistent; also, associated information must be traceable. A traceability model is proposed for the information contained in the documentation of the testing process; the data model developed gives support to documentation based on the standard IEEE Std 829-1998, as well as to traceability of all associated information; a prototype tool was developed to implement the data model and the traceability model.

Sumário

Lista de Figuras

Lista de Tabelas

Trabalhos afins publicados pelo autor

Cruz, J. L., Jino, M., A. N. Crespo, Argollo Júnior, M. T. (2008), "PROMETEU - A Tool to Support Documents Generation and Traceability in the Test Process", 01/2008, Jornada Iberoamericana de Ingeniería del Software e Ingeniería del Conocimiento (JIISIC),Vol. CD-ROM, pp.133-138, Guayaquil, Equador.

Cruz, J. L., Jino, M., A. N. Crespo, Argollo Júnior, M. T. (2006), "Suporte Automatizado à Rastreabilidade em um Processo de Teste de Software Baseado em Documentação", 06/2006, V Simpósio Brasileiro de Qualidade de Software - SBQS 2006, Vol. 1, pp.1-8, Vila Velha, ES, Brasil.

Capítulo 1

Introdução

1.1 Motivação e relevância

A cada dia que passa as aplicações de software[1] aumentam sua influência na execução das atividades humanas. Essa influência é de caráter global e ocorre de tal forma que aproximadamente 40% da população mundial sofreria impactos negativos caso alguns sistemas de software deixassem de funcionar ou apresentassem falhas críticas (Reed, 2000). Dessa forma, melhorar e assegurar a qualidade dos produtos de software tornou-se uma exigência constante por parte dos envolvidos na sua produção, utilização e comercialização: um software de boa qualidade (usando como parâmetro sua especificação de requisitos) fornece resultados corretos quando alimentado com dados válidos e identifica corretamente dados de entrada inválidos (Crespo et al., 2004). O teste é reconhecido como um processo[2] fundamental para obtenção de um produto de software de qualidade (Pressman, 2002; Chrissis et al., 2003; Pfleeger, 2004).

Testar um software é complexo e dispendioso pois, além da participação de diversos profissionais com diferentes expectativas e formações, um projeto de teste é composto por um conjunto de atividades complexas e de longa duração, que envolvem tanto questões técnicas como gerenciais (Beizer, 1990; Craig e Jaskiel, 2002; Pfleeger, 2004). As questões gerenciais representam, na maioria das vezes, um aspecto-chave para o sucesso de um projeto de teste (Craig e Jaskiel, 2002; Pressman, 2002; Neto e Travassos, 2006). No entanto, falhas no planejamento, organização e acompanhamento das atividades de teste ainda são comuns, gerando conseqüências negativas como (Tassey, 2002; Craig e Jaskiel, 2002; Black, 2003; Pfleeger, 2004):

[1] Ver a definição no Apêndice B – Glossário e Abreviaturas.
[2] Ver a definição no Apêndice B – Glossário e Abreviaturas.

- Ausência de controle do projeto[3] e incapacidade de relatar rapidamente o estado atual (p.ex.: informar como e quais testes foram realizados, quais foram os resultados obtidos até o momento, quantidade de defeitos existentes e em correção) sempre que necessário;
- Alocação inadequada de recursos, podendo acarretar aumento nos custos, estouro de prazo decorrente, por exemplo, de excesso de tempo e esforço em atividades menos críticas;
- Incapacidade de repetir os testes já executados, devido à ausência de registros;
- Software entregue com grande número de defeitos em função de, por exemplo, teste incompleto e inadequado de funcionalidades importantes ou da ausência de testes de algumas funcionalidades;
- Dificuldade para prever a abrangência dos impactos das mudanças nos artefatos[4] relacionados ao teste.

Um aspecto fundamental para o planejamento e gerenciamento adequado do teste é uma documentação[5] de boa qualidade, composta por elementos de informação[6] atualizados, precisos e consistentes entre si (Pressman, 2002; Pfleeger, 2004). "Talvez a pior documentação seja aquela que é bem estruturada e formatada, mas incorreta ou desatualizada, levando seu usuário a tomar decisões baseadas em suposições incorretas e ou incompletas" (Lewis, 2000).

Manter a qualidade da documentação do teste não é trivial, uma vez que a cada etapa da execução do processo uma quantidade considerável de dados pode ser registrada e, também, atualizada; ou seja, os documentos existentes podem ser alterados diversas vezes, com o agravante de que uma modificação em um documento pode causar impacto em vários outros, sendo grande a chance de deixar diversos deles desatualizados e inconsistentes em função de revisões incompletas (Pressman, 2002; Sommerville, 2003; Pfleeger, 2004). Sendo assim, uma empresa desenvolvedora de software deve definir o que será documentado, como será documentado e, também, deve definir de que maneira manterá atualizados e consistentes os elementos de informação que compõem essa documentação (Crespo et al., 2002; Crespo et al., 2004).

[3] Ver a definição no Apêndice B – Glossário e Abreviaturas.
[4] Ver a definição no Apêndice B – Glossário e Abreviaturas.
[5] Ver a definição no Apêndice B – Glossário e Abreviaturas.
[6] Ver a definição no Apêndice B – Glossário e Abreviaturas.

Definir o que deve ser documentado não é simples, mas existe uma norma internacional para documentação de um processo de teste, a Norma IEEE Std 829-1998, que pode ser utilizada como guia (IEEE, 1998b); esta norma é a base para as atividades do Grupo de Teste de Software do CTI[7], contexto no qual se originou este trabalho (Crespo et al., 2002; Crespo et al., 2004; Crespo e Jino, 2005). Para manter a integridade dos elementos de informação registrados, entre outras coisas, é necessário estabelecer *links* de rastreabilidade entre eles. Tanto a documentação como a rastreabilidade demandam o manuseio de uma quantidade enorme de dados, tornando necessário um suporte automatizado adequado.

1.2 Objetivo

Considerando o exposto anteriormente, os objetivos deste trabalho são:

1. Desenvolver um modelo de rastreabilidade entre os elementos de informação que compõem a documentação do processo de teste;
2. Desenvolver um modelo de dados que dê suporte tanto à documentação segundo a norma IEEE Std 829-1998 como à rastreabilidade;
3. Desenvolver um protótipo de ferramenta para implementar o modelo de dados e o modelo de rastreabilidade.

1.3 Estrutura da Dissertação

Este trabalho está organizado da seguinte maneira:

O Capítulo 1 apresenta a introdução ao assunto, buscando contextualizá-lo dentro de um mercado exigente por software de boa qualidade. Também são apresentadas a motivação e a relevância do trabalho, seus objetivos e a organização do texto.

O Capítulo 2 aborda o teste de software, destacando algumas definições e termos relacionados, além de motivações para testar um software. O foco do capítulo são os aspectos gerenciais de um processo de teste, sendo apresentados os níveis de teste, vários tipos de teste, além de técnicas e critérios que direcionam a definição do que e como testar; além disso, são destacadas questões envolvidas na tomada de decisão para aquisição, implantação e utilização de ferramentas de apoio ao teste.

[7] CTI – Centro de Tecnologia da Informação Renato Archer (antigo CenPRA). Página na Internet: http://www.cti.gov.br.

O Capítulo 3 enfoca a documentação no processo de software, alguns conceitos relacionados, a visão negativa que ainda paira sobre a geração e manutenção da documentação, algumas diretrizes para obtenção de documentos de boa qualidade, um modelo simplificado para representação de documentos e o papel da documentação no teste de software. Em relação ao teste, são destacados os possíveis benefícios da documentação para o processo de teste, o nível apropriado de documentação e a necessidade de padronização. Quanto à padronização da documentação do teste é apresentada a Norma IEEE Std 829-1998. É também discutida a necessidade de suporte automatizado à documentação do teste.

O Capítulo 4 aborda a rastreabilidade, algumas definições e conceitos básicos sobre o tema, e sua relação com um processo de software. Alguns modelos de rastreabilidade são apresentados resumidamente; os conceitos existentes nestes modelos servem de base para a definição de um modelo de rastreabilidade específico para a documentação de um processo de teste. Também são discutidas diversas vantagens da rastreabilidade para um processo de teste de software.

O Capítulo 5 apresenta o modelo de rastreabilidade e o modelo de dados usados pelo protótipo PROMETEU. O modelo de rastreabilidade está baseado na unificação dos conceitos dos modelos de rastreabilidade apresentados no Capítulo 4, juntamente com os artefatos e relacionamentos propostos na Norma IEEE Std 829-1998. O modelo de dados foi projetado para o armazenamento e manipulação das informações de rastreabilidade de acordo com o modelo de rastreabilidade definido. É feita uma breve análise de cada um dos modelos, destacando os aspectos atuais e possíveis pontos que podem ser melhorados.

O Capítulo 6 apresenta PROMETEU, o protótipo desenvolvido com o propósito de validar as idéias apresentadas na dissertação. São descritas as decisões de implementação, as características da ferramenta, além dos conceitos e a estratégia de rastreabilidade adotados.

O Capítulo 7 apresenta as considerações finais sobre o trabalho realizado, sendo apresentadas as contribuições do trabalho, as possíveis melhorias que podem ser implementadas no protótipo e indicações de trabalhos futuros.

O Apêndice A apresenta um exemplo de utilização do protótipo para ilustrar suas funcionalidades.

O Apêndice B apresenta a definição de alguns termos e siglas utilizados nesta dissertação.

Capítulo 2

Teste de Software

Devido à relevância cada vez maior do software na execução de diversas atividades humanas, tanto os envolvidos na sua produção como aqueles envolvidos na sua comercialização e utilização têm se preocupado cada vez mais em melhorar e assegurar sua qualidade. Apesar do conceito de qualidade não ser definido apenas com base nos fatores confiabilidade e funcionalidade, dificilmente um produto de software será considerado portador de boa qualidade enquanto apresentar falhas durante a execução das funções para as quais foi projetado (Herbert, 2000).

Alguns exemplos de falhas durante a execução de um software, com conseqüências extremamente danosas, são (Garfinkel, 2005; Huckle, 2008): a) a explosão do foguete Ariane 5, em 1996; b) a queda de uma aeronave militar V-22 Osprey do US Marine Corps, em dezembro de 2000, que resultou na morte dos quatro tripulantes; c) o cálculo errôneo da dosagem necessária de radiação a vários pacientes, em novembro de 2000, no Instituto Nacional do Câncer na cidade do Panamá, que resultou na morte de oito pessoas e danos significativos à saúde em outras 20.

Este capítulo fornece uma visão conceitual abrangente sobre o teste, um dos processos fundamentais para a garantia da qualidade de um software (Pressman, 2002; Pfleeger, 2004; Chrissis et al., 2003). O capítulo está estruturado da seguinte maneira: a Seção 2.1 apresenta definições de teste de software; a Seção 2.2 apresenta a relevância dos aspectos gerenciais em um processo de teste; a Seção 2.3 aborda questões relacionadas à definição do que deve ser testado; a Seção 2.4 enfoca como organizar as atividades do teste; a Seção 2.5 enfoca diversas questões relacionadas ao suporte automatizado às atividades de um processo de teste; a Seção 2.6 apresenta a síntese do capítulo.

2.1 Definições

Algumas definições de teste de software são:

- É o processo de executar um programa ou sistema com a intenção de encontrar defeitos (Myers, 1979);
- É o processo de operar um sistema ou componente sob condições controladas, observando e registrando os resultados, para avaliar algum aspecto do sistema ou componente (IEEE, 1991);
- É o processo de analisar um item de software para detectar diferenças entre os resultados obtidos e os resultados esperados e avaliar as funcionalidades do item de software (IEEE, 1998b);
- É um processo cujas atividades são executadas em paralelo ao processo de desenvolvimento, utilizando artefatos específicos para medir e melhorar a qualidade do software sendo testado (Craig e Jaskiel, 2002);
- É um mecanismo de controle para fornecer informações à gerência sobre os riscos associados à liberação de um produto de software (Perry, 2006). Um risco pode ser entendido como a probabilidade de que eventos indesejáveis ocorram.

Considerando essas definições, pode-se entender que testar um software é o processo de executá-lo de maneira controlada com o objetivo de encontrar defeitos, avaliar se o mesmo se comporta conforme as funcionalidades especificadas e, também, estimar os riscos de sua liberação para utilização.

Um conceito comum às definições apresentadas é o termo “processo”. Um processo de teste é um conjunto de passos parcialmente ordenados constituídos por atividades, métodos e práticas, usados para testar um software; esse conjunto de passos é detalhado de acordo com o ramo de atividade da empresa e do produto que será testado (Crespo e Jino, 2005). Neste contexto o elo entre as atividades, métodos e práticas de um processo de teste são os dados registrados nos documentos, na medida em que esses dados indicam, por exemplo, quais atividades devem ser realizadas, em qual ordem, por quais pessoas e qual o estado atual da execução dessas atividades; ou seja, a documentação assume papel de grande importância e deve permanecer completa, atualizada e consistente ao longo de um projeto de forma a fornecer informações úteis e relevantes à gerência do teste.

2.2 Aspectos gerenciais

Um processo de teste é complexo, composto por diversas atividades que são executadas por vários profissionais ao longo de todo o processo de desenvolvimento de software. Essas atividades podem consumir de 30% a 50% dos recursos de um projeto; caso o contexto de utilização do software seja extremamente crítico, o custo pode ultrapassar 50% (Beizer, 1990; Pressman, 2002; Pfleeger, 2004). Além disso, estudos indicam que o prejuízo anual de uma infra-estrutura inadequada (p.ex., falta de pessoas e alocação insuficiente de tempo) para o teste de software é estimado entre 22 e 60 bilhões de dólares (Tassey, 2002). Apesar de não representar uma medida muito precisa, esses dados oferecem uma idéia das conseqüências negativas de testes que sejam mal planejados, projetados e executados.

Desta forma, as questões gerenciais assumem um papel fundamental para que o processo de teste seja capaz de fornecer evidências confiáveis sobre os riscos de liberar um produto de software (Herbert, 2000; Craig e Jaskiel, 2002; Pfleeger, 2004; Perry, 2006). Negligenciar as questões gerenciais faz com que o teste seja conduzido de maneira não-sistemática e sem objetivos bem definidos. Falhas no planejamento e gerenciamento podem ocasionar conseqüências negativas como (Marciniak, 1994; Craig e Jaskiel, 2002; Black, 2003; Perry, 2006): a) teste incompleto ou inadequado, com liberação de produtos de má qualidade contendo grande número de defeitos; b) alocação equivocada de recursos, ocasionando consumo exagerado de tempo e esforço na execução das atividades do teste; c) perda do controle sobre o que foi testado e sobre qual o estado do teste; d) falhas na coordenação e comunicação entre os diversos atores[8]; e) retrabalho por não aproveitamento das experiências de projetos anteriores; e f) perda de credibilidade junto ao cliente.

O gerenciamento do teste envolve definir o que deve ser testado, organizar e acompanhar a execução das atividades do processo, além de decidir pelo uso, ou não, de ferramentas para apoiar a execução dessas atividades (Crespo e Jino, 2005). Dessa forma, percebe-se que o teste necessita de uma grande quantidade de dados para seu planejamento, projeto, execução e acompanhamento, o que torna a documentação de boa qualidade (baseada em elementos de informação atualizados, precisos e consistentes entre si) um aspecto crítico para o teste adequado de um software. Dito de outra forma: elementos de informação desatualizados ou inconsistentes

[8] Ver a definição no Apêndice B – Glossário e Abreviaturas.

implicam, entre outras coisas, testes incompletos ou inapropriados no software (Herbert, 2000; Crespo et al., 2002; Pressman, 2002; Pfleeger, 2004).

Questões relacionadas à documentação no processo de teste são abordadas mais detalhadamente no próximo capítulo; as seções a seguir abordam as questões relacionadas aos aspectos gerenciais do teste, oferecendo uma visão geral sobre o processo.

2.3 Definição do que testar

O planejamento das atividades de teste deve estar baseado, inicialmente, nos requisitos (Craig e Jaskiel, 2002; Crespo e Jino, 2005). Requisitos são as características e condições que devem ser satisfeitas até o final do desenvolvimento do produto (Robertson e Robertson, 1999; Leffingwell e Widrig, 2000; Cockburn, 2005).

Os requisitos podem ser divididos em dois grandes grupos (Sommerville, 2003; Leffingwell e Widrig 2000): a) Características ou requisitos do usuário; b) Requisitos de sistema ou de software. As características são descrições genéricas, em alto nível e na visão do usuário, das funções do sistema e das restrições sob as quais deve operar. Os requisitos do software representam as funções e as restrições de operação do sistema, em um formato mais técnico, e podem ser divididos em funcionais e não funcionais (Sommerville, 2003; Pfleeger, 2004).

Os requisitos representam os critérios de aceitação de um software e direcionam tanto a definição do que deve ser testado como o projeto dos testes a serem realizados; entretanto, como é impossível testar completamente um software, deve ser feita uma análise de risco para definir quais requisitos serão testados e qual a intensidade do teste para cada um (Black, 2003; Perry, 2006). A qualidade dessa análise depende de documentação atualizada e consistente dos requisitos.

2.4 Organização das atividades

Idealmente, o planejamento e a definição das atividades de um processo de teste são feitos de acordo com um processo previamente caracterizado e institucionalizado (Herbert, 2000; Crespo e Jino, 2005).

Um processo de teste pode ser organizado seguindo uma divisão em níveis e tipos de teste, onde para cada nível e tipo são usadas técnicas e critérios que direcionam a criação dos casos de teste que serão executados. Um caso de teste pode ser entendido como sendo um artefato que

especifica um conjunto de dados de entrada para executar um programa, os passos necessários para essa execução e o resultado esperado dessa execução (Pfleeger, 2004). A quantidade necessária de casos de teste para testar um sistema pode variar de algumas centenas até milhares, dependendo do porte e da criticidade do sistema.

A Figura 2.1 representa a relação entre Níveis, Tipos, Técnicas e Critérios de Teste (Crespo e Jino, 2005).

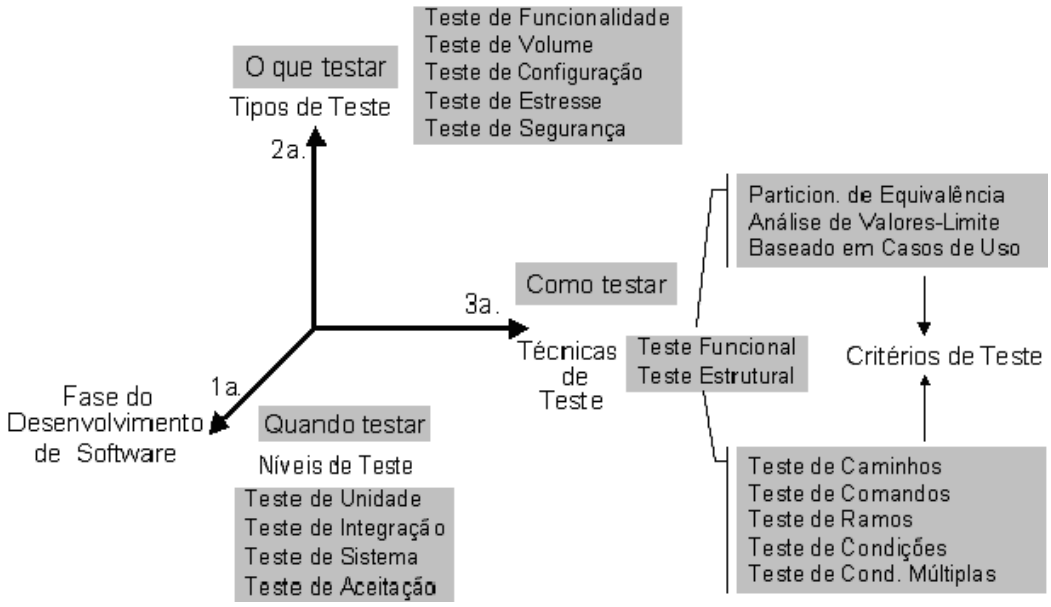

Figura 2.1: Relação entre Níveis, Tipos, Técnicas e Critérios de Teste.

Cada nível de teste (Unidade, Integração, Sistema e Aceitação) corresponde a uma fase do ciclo de vida do software; para cada nível de teste existe um conjunto de atividades e tipos de teste específicos associados (IEEE, 1998b; Craig e Jaskiel, 2002; Crespo e Jino, 2005). Esta abordagem para o planejamento do teste está baseada no Modelo V (Figura 2.2), segundo o qual as atividades do teste são planejadas, executadas e controladas em consonância com o desenvolvimento do software: para cada fase do desenvolvimento existe um nível de teste e um conjunto de atividades de teste correspondentes (Craig e Jaskiel, 2002; Black, 2003; Crespo e Jino, 2005). Por exemplo, o nivel de teste "Teste de Sistema" está associado à fase do desenvolvimento "Especificação do Sistema"; para este nível de teste utiliza-se a técnica "Teste Funcional", com os critérios "particionamento de equivalência", "análise de valores-limite" e "baseado em casos de uso" para gerar testes do tipo "funcionalidade".

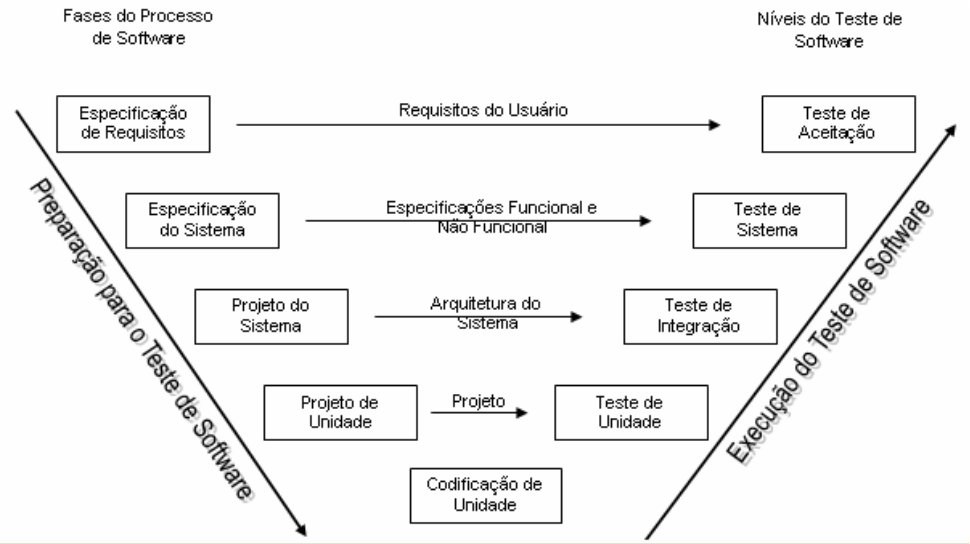

Figura 2.2: Modelo V de Teste de Software (Crespo e Jino, 2005).

A seguir são descritos brevemente os níveis, tipos e técnicas de teste. Essa descrição engloba diversos aspectos relacionados ao planejamento de um projeto de teste e evidencia a necessidade da existência de documentos atualizados e consistentes ao longo de um projeto dada a grande quantidade de dados utilizados pelos atores para suporte à execução das atividades de um projeto de teste.

2.4.1 Níveis de Teste

Teste de Unidade

Uma unidade pode ser definida como a menor parte de um software criada por um programador, armazenada em um único arquivo em disco (Copeland, 2004). Neste nível, o esforço de teste é focalizado nas unidades que compõem o software geralmente utilizando critérios da técnica caixa-branca (Pressman, 2002). Os testes de unidade geralmente são executados pelos próprios programadores, no ambiente de desenvolvimento. As técnicas e critérios de teste são explicados mais adiante neste capítulo.

Teste de Integração

Neste nível, testa-se como as partes de um sistema funcionam em conjunto com o objetivo de identificar erros nas interfaces entre as unidades tais como passagens errôneas de parâmetros e chamadas errôneas de módulos (Craig e Jaskiel, 2002; Pressman, 2002). Os testes de integração são projetados e executados no próprio ambiente de desenvolvimento. Possíveis fontes de

informação para projetar casos de teste de integração são os artefatos de *design*[9] (Craig e Jaskiel, 2002).

Teste de Sistema

Conjunto de testes cujo objetivo é comprovar se os requisitos do software foram corretamente implementados, de acordo com critérios de aceitação previamente definidos, após todos os subsistemas terem sido integrados em um único sistema de software. Os testes deste nível geralmente são projetados e executados na empresa que desenvolveu o software, em um ambiente que procura simular o ambiente real de operação (Craig e Jaskiel, 2002; Copeland, 2004; Pfleeger, 2004). Possíveis fontes de informação são os requisitos do software, *designs* de alto nível, os atores envolvidos na definição dos requisitos e dos *designs*, além de outros documentos relevantes de acordo com a natureza do software sendo desenvolvido (Craig e Jaskiel, 2002).

Teste de Aceitação

Este nível também é baseado nos requisitos do software, com o software totalmente integrado, mas idealmente é realizado pelo próprio usuário, no ambiente real de operação do software (Craig e Jaskiel, 2002; Copeland, 2004; Pfleeger, 2004). Possíveis fontes de informação para projetar os testes de aceitação são os requisitos do software, os atores envolvidos na definição dos requisitos, documentos de planejamento das atividades de teste, além de outros documentos relevantes de acordo com a natureza do software em desenvolvimento (Craig e Jaskiel, 2002).

Para cada nível de teste são projetados, documentados e executados diversos tipos de teste para validar diferentes aspectos do software. Tipificar os testes auxilia o planejamento e acompanhamento do estado atual dos testes projetados.

2.4.2 Tipos de Teste

Alguns exemplos de tipos de teste são apresentados a seguir (Pfleeger, 2004):

- Funcionalidade: teste das funções do sistema, baseado na especificação dos requisitos funcionais;
- Volume: verifica se o software atende aos requisitos quando submetido a grandes quantidades de dados. Por exemplo: verifica se a manipulação das estruturas de dados

[9] Ver a definição no Apêndice B – Glossário e Abreviaturas.

é feita da maneira esperada quando os dados alcançam seu tamanho máximo e, também, quando a quantidade dos dados é superior ao esperado;

- Configuração: testa o produto desenvolvido em diversas configurações de software e hardware, para garantir que ele funcione de acordo com o especificado em cada uma dessas configurações;
- Estresse: exercita o software ao máximo em relação ao esperado, de acordo com as especificações, e um pouco além desse limite. Por exemplo: a) exercitar o desempenho do sistema quando a quantidade máxima especificada de usuários e dispositivos estiverem conectados a ele; b) exercitar o software em condições de pouca memória disponível; b) exercitar o software simultaneamente com vários outros softwares;
- Segurança: testa a capacidade do software de manter a integridade e confidencialidade dos dados (p.ex., capacidade para resistir a possíveis ataques para roubar senhas).

2.4.3 Teste de Regressão

Um conceito que não pode ser caracterizado como nível de teste e nem como tipo de teste é o teste de regressão. Teste de regressão consiste em repetir um conjunto de testes já executados (que podem ser de vários tipos e abranger vários níveis), com o objetivo de assegurar que o comportamento do software permanece inalterado após a implementação de uma ou mais modificações (Pressman, 2002; Kruchten, 2003; Pfleeger, 2004).

2.4.4 Técnicas de Teste

O projeto dos casos de teste para os diversos níveis e tipos de teste é baseado em técnicas de teste, que direcionam o *design* de cada caso de teste de acordo com o enfoque de cada técnica.

Uma técnica de teste direciona a escolha de um ou mais critérios associados para geração de casos de teste. Um critério de teste determina um conjunto específico de elementos requeridos que deve ser exercitado no teste de um determinado aspecto do software; critérios sistematizam o *design* de casos de teste, aumentando a probabilidade de revelar defeitos antes da liberação do produto, aumentando sua confiabilidade (Rocha et al., 2001; Crespo e Jino, 2005).

A aplicação de uma técnica não exclui a aplicação de outra: é desejável que sejam aplicadas de forma complementar a fim de constituir uma abordagem de teste mais completa e eficaz,

visando aumentar as chances de detecção de defeitos com uma utilização otimizada de recursos (Rocha et al., 2001; Pfleeger, 2004; Crespo e Jino, 2005). Além disso, o sucesso na aplicação de técnicas de teste para a geração de casos de teste depende da qualidade da documentação do processo.

As técnicas amplamente usadas na indústria de desenvolvimento de software têm foco funcional e estrutural (Crespo e Jino, 2005).

Teste Funcional (Caixa-Preta)

O teste funcional direciona o *design* dos testes do ponto de vista do usuário sem considerar a estrutura interna do código-fonte. Esta técnica utiliza informações das especificações dos requisitos para projetar os casos de teste. Alguns tipos de defeitos que podem ser descobertos são: a) funções incorretas; b) funções não implementadas; c) erros de interfaces entre funções; d) erros de inicialização e finalização do software. Exemplos de critérios utilizados são: particionamento de equivalência, análise de valores-limite e baseado em casos de uso (Crespo e Jino, 2005).

Teste Estrutural (Caixa-Branca)

Esta técnica depende da disponibilidade de acesso ao código-fonte do software em teste. O objetivo é usar as informações das estruturas internas do software de forma projetar casos de teste para revelar defeitos do tipo: a) comandos incorretos; b) comandos ausentes; c) variáveis não definidas; d) erros de inicialização e finalização de *loops*. Exemplos de critérios utilizados são: teste de caminhos, teste de comandos, teste de ramos, teste de condições e teste de condições múltiplas (Crespo e Jino, 2005).

A seção a seguir discorre brevemente sobre alguns aspectos relacionados à automação em um processo de teste; são apresentados potenciais benefícios além de uma classificação para ferramentas de suporte às atividades do teste. Dessa maneira será possível contextualizar o protótipo desenvolvido em relação a outras ferramentas existentes.

2.5 Suporte à automação das atividades de um processo de teste

Considerando a complexidade de um processo de teste, é altamente recomendável utilizar ferramentas que forneçam algum nível de suporte automatizado à execução das suas inúmeras atividades (Beizer, 1990; Dustin, 2003; Pfleeger, 2004).

2.5.1 Potenciais benefícios da automação

Alguns possíveis benefícios advindos da utilização de ferramentas em um processo de teste são (Beizer, 1990; Mosley, 1993; Lewis, 2000; Dustin, 2003):

- Execução de tarefas em velocidade muito maior do que a de ações humanas equivalentes;
- Precisão na execução repetitiva de tarefas bem definidas;
- Maior facilidade na reutilização de informações geradas previamente;
- Geração de relatórios de acompanhamento com maior facilidade e rapidez;
- Aumento na confiabilidade dos resultados obtidos ao longo da execução do processo;
- Maior facilidade na manutenção dos artefatos do teste.

Os benefícios citados, quando obtidos, levam a uma economia de recursos em termos de tempo e dinheiro; no entanto, para aumentar as chances de obtenção dos benefícios da automação, algumas questões devem ser consideradas:

- Já existe um processo de teste definido? Caso não exista, provavelmente o teste é mal realizado e improdutivo. Introduzir uma ferramenta neste contexto somente reforçará o caos existente sem melhorar os resultados obtidos; se sim, as ferramentas a serem adquiridas são compatíveis com o processo de teste existente? (Kit, 1995; Craig e Jaskiel, 2002);
- O planejamento e o projeto de teste já são realizados? Ferramentas não eliminam tarefas essencialmente humanas como planejar e projetar o teste (Kit, 1995; Lewis, 2000; Dustin, 2003);
- Os objetivos a serem atingidos com a automação foram claramente analisados e mapeados nas características das ferramentas a serem adotadas? (Kit, 1995; Lewis, 2000; Dustin, 2003);
- Existem ferramentas disponíveis que sejam adequadas à natureza do sistema a ser testado? O custo da adoção da ferramenta é vantajoso para a organização? A ferramenta é compatível com o ambiente operacional necessário para o sistema sendo testado? (Dustin, 2003);
- Como será feito o treinamento para utilização da ferramenta? Deve ser previsto tempo suficiente para familiarizar os testadores com a ferramenta para poder extrair todo seu

potencial. Esse tempo pode variar de dias a meses, dependendo da complexidade da ferramenta (Beizer, 1990).

Mesmo que sejam obtidas respostas positivas para todas as questões apresentadas, convém ressaltar que não existe uma ferramenta que trate de todos os aspectos de um processo de teste, nem que seja compatível com todos os sistemas operacionais e com todas as ferramentas utilizadas para o desenvolvimento de um software (Dustin, 2003). Além disso, ferramentas não reduzem imediatamente os custos associados ao teste; pelo contrário, incialmente o tempo necessário ao teste pode até mesmo aumentar, considerando a curva de aprendizado e a necessidade de manutenção do ambiente de operação da ferramenta (Beizer, 1990; Dustin, 2003).

Existe uma grande variedade de ferramentas de apoio ao teste, cada uma oferecendo variados graus de suporte às atividades do processo. Apesar de essas ferramentas poderem ser classificadas de várias formas, nenhuma dessas classificações constitui uma categorização que seja mais aceita do que outras (Beizer, 1990; Kit, 1995; Lewis, 2000; Dustin, 2003). A classificação apresentada na seção a seguir é apenas uma entre várias possíveis.

2.5.2 Classificação das ferramentas de apoio às atividades do teste

Ferramentas de teste podem ser classificadas de acordo com os seguintes critérios (Mosley, 1993; Kit, 1995):

- Através das tarefas ou atividades de teste nas quais ela tem maior uso: verificação do código-fonte, planejamento do teste, gerência da execução do teste, documentação, execução automática da aplicação, etc;
- Através da função específica executada pela ferramenta: capture/replay, medição da cobertura do código-fonte, geração automática de dados de teste, etc;
- Através das áreas genéricas de aplicação: gerenciamento do teste, análise de cobertura do código, automação da execução dos testes, etc. Neste caso, existe uma ou mais ferramentas com funcionalidades que atendem à área em questão.

Neste trabalho, é adotada a categorização de acordo com áreas genéricas de aplicação e as ferramentas são divididas em quatro categorias: uso geral, geração automática de dados de teste, automação da execução dos testes e suporte ao gerenciamento dos testes.

Ferramentas de uso geral

Seu foco principal não é o teste mas podem auxiliar indiretamente na execução e no controle das atividades do processo. Exemplos desses tipos de ferramentas são processadores de texto e planilhas eletrônicas.

Ferramentas geradoras de dados de teste

Geram arquivos e os atualizam com dados de teste para programas que estão sendo testados. Esses arquivos podem ser utilizados por testadores em testes manuais ou por ferramentas que automatizam a execução dos testes (Pressman, 2002).

Ferramentas de automação da execução dos testes

Executam um conjunto de testes automaticamente após a programação ou gravação das ações que devem ser executadas. A seguir são apresentadas algumas categorias e exemplos:

- Teste de unidade: os testes de uma unidade são programados com o auxílio de *frameworks* específicos de acordo com a linguagem de programação; posteriormente esses programas são compilados e os testes são executados automaticamente. Exemplos: JUnit (linguagem Java), NUnit (ambiente Microsoft .NET), CppUnit (linguagem C++);
- Análise de cobertura do código: com base no código-fonte da aplicação, a ferramenta indica o quanto da estrutura de um módulo ou unidade (cobertura de comandos e condições, por exemplo) foi exercitada por um conjunto de testes (Lewis, 2000; Dustin, 2003; Pfleeger, 2004); exemplos de ferramentas deste tipo são NCover e CoverageEye.Net, JCover, Testwell CTC++ e CoverageMeter.
- Programação de *scripts*: por meio de pequenos programas é possível automatizar a execução de um software ou *site* web, passar valores de teste e comparar os resultados obtidos (Lewis, 2000; Dustin, 2003);
- *Capture/replay* ou *record/playback*: são ferramentas que geram *scripts* de teste em uma linguagem específica a partir da execução do software por parte do testador. Posteriormente, a ferramenta lê os *scripts*, executa o aplicativo com base nas ações e dados de entrada gravados, e compara os resultados obtidos com os resultados esperados (Beizer, 1990; Kit, 1995; Pfleeger, 2004).

Ferramentas de suporte ao gerenciamento do processo

Auxiliam no controle, coordenação e acompanhamento das fases de planejamento, projeto, execução e registro dos resultados da execução dos testes. Dependendo do porte, uma ferramenta deste tipo pode oferecer um ou mais dos recursos a seguir:

- Documentação em diversos níveis do processo de teste (Kit, 1995; Pressman, 2002);
- Gerência da execução de casos de teste (Kit, 1995);
- Rastreabilidade entre dois ou mais tipos de artefatos relacionados a um processo de teste, como requisitos e casos de teste (Lewis, 2000; Pressman, 2002; Dustin, 2003);
- Gerência dos defeitos (*bug tracking*) descobertos como conseqüência da execução dos casos de teste (Kit, 1995; Kaner, 2001).

O protótipo desenvolvido como parte deste trabalho é uma ferramenta para suporte ao gerenciamento do processo, que oferece os quatro recursos apresentados para este tipo de ferramenta. As características e os princípios que norteiam seu projeto e funcionamento são abordados nos Capítulos 5 e 6; no Apêndice A é apresentado um exemplo de sua utilização.

2.6 Síntese do Capítulo

Neste capítulo foi apresentada uma visão abrangente de teste de software, um processo fundamental para a garantia da qualidade de um produto de software: quando um produto não é testado adequadamente, aumenta a probabilidade de ocorrência de falhas na sua utilização e, em alguns casos, falhas extremamente graves que podem até mesmo colocar vidas humanas em perigo.

O teste pode ser entendido como sendo o processo de executar um software de maneira controlada e com o objetivo de encontrar defeitos, avaliar se ele se comporta conforme as funcionalidades especificadas e, também, estimar os riscos de sua liberação para utilização.

Um processo de teste é um conjunto complexo de atividades relacionadas a todas as fases do ciclo de vida do software. Assim, é altamente recomendável utilizar ferramentas de software como um dos meios de obter melhores resultados. Existe uma grande variedade de ferramentas de apoio ao teste, que enfocam diversos aspectos do processo e oferecem variados graus de apoio. No entanto, é necessário ressaltar que a utilização de uma ou mais ferramentas não representa uma solução mágica para os desafios relacionados ao teste: ferramentas compõem somente uma

parte de uma estratégia que, primeiramente, deve se concentrar na implantação de um processo mínimo de teste.

Parte da complexidade de um processo de teste está relacionada aos aspectos gerenciais, que envolvem: a) definir o que deve ser testado, b) decidir como organizar as atividades e tarefas, c) definir uma estratégia de automação, d) empregar meios para coordenar a interação entre os diversos atores, cada uma com diferentes responsabilidades e expectativas, e d) decidir o que deve ser documentado e em que quantidade, de forma a facilitar o planejamento e acompanhamento do teste sem gerar burocracia desnecessária.

Planejar, registrar e acompanhar a execução das atividades de teste implica documentar e manipular uma grande quantidade de elementos de informação, que devem ser mantidos atualizados e consistentes para permitir a tomada de decisões. Neste contexto, uma ferramenta específica para facilitar a geração, manipulação e manutenção da integridade e consistência dos documentos pode auxiliar de maneira relevante o teste em todas as suas etapas (planejamento, projeto, registro e execução dos testes).

O protótipo descrito neste trabalho é uma ferramenta do tipo "suporte ao gerenciamento do processo", com recursos de suporte à documentação, à gerência da execução dos casos de teste, à rastreabilidade entre os artefatos e à gerência dos defeitos descobertos. Suas características e os princípios que norteiam seu projeto e funcionamento são abordados nos Capítulos 5 e 6 e no Apêndice A é apresentado um exemplo de sua utilização.

Aspectos relacionados à documentação, sua importância e seu papel em um processo de teste são abordados no próximo capítulo.

Capítulo 3

Documentação no Processo de Teste de Software

A documentação é de grande importância para a qualidade e o sucesso de um projeto, uma vez que os documentos são a única maneira tangível para registrar as informações que representam um software, além de fornecer visibilidade ao seu processo de construção (Pressman, 2002; Sommerville, 2003; Pfleeger, 2004). Normas e modelos para melhoria de processo também destacam a documentação como fator relevante para a melhoria e garantia da qualidade do software (Paulk, 1993; ISO 1995; Chrissis et al., 2003).

A utilidade da documentação de um projeto é diretamente proporcional à sua consistência e atualidade (reflete o momento atual do projeto) (Visconti e Cook, 2002); consistência pode ser entendida como a uniformidade, padronização e ausência de contradições entre os elementos de informação registrados em diferentes tipos de documentos de um projeto (IEEE, 1990). Caso os dados contidos nos documentos sejam imprecisos ou incompletos, decisões equivocadas podem ser tomadas ocasionando grandes prejuízos ao projeto (Rocha et al., 2001; Visconti e Cook, 2002; Perry, 2006).

Este capítulo apresenta alguns aspectos relacionados à documentação e sua relação com o processo de teste de software, e está dividido nas seguintes seções: a Seção 3.1 apresenta os conceitos de documentação, artefato e documento usados ao longo deste trabalho; a Seção 3.2 apresenta alguns motivos que alimentam a visão negativa sobre documentação; a Seção 3.3 apresenta diretrizes para obtenção de documentação de boa qualidade; a Seção 3.4 apresenta a documentação no contexto do teste de software e baseada em uma norma internacional; e a Seção 3.5 apresenta a síntese do capítulo.

3.1 Definições

A seguir são apresentados os conceitos de documentação, artefato e documento:

- Documentação de um projeto de software é formada pelo conjunto de artefatos que armazenam diversos elementos de informação gerados, mantidos e atualizados pela execução das várias atividades do processo (Pressman, 2002; Sommerville, 2003; Pfleeger, 2004). Exemplos de artefatos: documentos de planejamento e gerenciamento como diagramas (casos de uso, classes, etc.) e descrições e comentários em códigos-fonte;
- Artefato é uma informação que é produzida, modificada ou usada por um processo de desenvolvimento de software; pode ser um modelo, um elemento de modelo, um documento ou até mesmo o próprio software (Kruchten, 2003); alguns exemplos de artefatos são: a) um *template* (modelo) de um documento, b) um documento preenchido com base em um *template*, c) um plano de teste, d) uma especificação de caso de teste, e) um arquivo de código-fonte;
- Documento é uma coleção de informações que são representadas em papel ou em uma mídia que representa um papel; exemplos: documentos de processadores de texto, planilhas, agendas, gráficos de Gantt, páginas da Web e apresentações em *slides* (Kruchten, 2003).

3.2 Visão negativa sobre documentação

Apesar de sua importância, a documentação pode variar de desatualizada a inexistente em um grande número de empresas desenvolvedoras de software. Isso se deve em grande parte ao mito segundo o qual documentar é desperdício de tempo e de recursos (Lewis, 2000; Pressman, 2002; Sommerville, 2003). Essa visão negativa é cultivada por fatores como:

- Grande quantidade de elementos de informação armazenados em documentos estáticos ao invés de documentos dinâmicos. Documentos estáticos são arquivos com conteúdo sem vínculos explícitos com outros documentos, o que dificulta manter as informações atualizadas e consistentes além de limitar os recursos de busca de informações e de geração de relatórios. Exemplos são documentos gerados com processadores de texto ou planilhas eletrônicas. Documentos dinâmicos têm seus dados armazenados sob controle de alguma ferramenta específica geralmente baseada

em um gerenciador de banco de dados, o que facilita: a) definir e rastrear relações entre elementos de informação de documentos diferentes; b) manter a documentação atualizada e consistente; c) gerar relatórios padronizados ou parametrizados (DiMaggio, 2001);

- Desconhecimento de modelos para documentação, dificultando definir o que deve ser documentado e em qual quantidade. Isso acaba ocasionando problemas como documentação em excesso (quantidade tão grande de dados que se torna muito difícil mantê-la ou encontrar as informações necessárias) ou insuficiente (não é útil, uma vez que faltam dados) (Visconti e Cook, 2002; Crespo e Jino, 2005);
- Falta de procedimentos para registro e manutenção dos elementos de informação pois geralmente as atividades relacionadas à documentação não são consideradas como parte do processo de software (Visconti e Cook, 2002; Sommerville, 2003; Crespo e Jino, 2005);
- Inexistência de suporte automatizado adequado, o que dificulta gerenciar a grande quantidade de documentos de diferentes tipos e formatos gerados em um processo (Sommerville, 2003; Crespo e Jino, 2005); manter artefatos em ferramentas, ao invés de papel, possibilita que modificações sejam mais facilmente propagadas entre os artefatos relacionados (DiMaggio, 2001; Kruchten, 2003).

O seguinte questionamento ilustra a importância da documentação e serve de contraponto à visão negativa sobre a necessidade de existência da mesma: quanto custará, no futuro, para uma empresa, um conjunto de documentos com dados desatualizados e/ou inconsistentes quando, por exemplo, após algum tempo for necessária uma manutenção em um ou mais produtos e o planejamento e a execução dessa manutenção forem baseados em uma documentação desatualizada? (Pressman, 2002; Wiegers, 2003).

Seguir diretrizes pode facilitar a geração de uma documentação de boa qualidade, além de contribuir para sua manutenção ao longo de um projeto (Gelperin, 1982; Pressman, 2002; Sommerville, 2003).

3.3 Diretrizes para obtenção de documentação de boa qualidade

Algumas diretrizes para obtenção de documentação de boa qualidade são (Gelperin, 1982; Pressman, 2002; Sommerville, 2003):

- Adotar padrões e modelos, pois eles fornecem uma estrutura e aparência padronizada aos documentos, propiciando: a) estabelecer um entendimento comum sobre o significado e a importância dos dados registrados, b) produzir e localizar os dados desejados com maior rapidez e facilidade, c) minimizar a possibilidade de elementos de informação relevantes não serem registrados;
- Revisar e modificar os modelos regularmente, para refletir eventuais mudanças tecnológicas, na política da empresa ou em decorrência do aprendizado ao longo dos projetos;
- Considerar a documentação como um processo, e planejar a integração das suas atividades com as de outros processos de software;
- Avaliar aspectos de conteúdo como completude, atualização, consistência, formato, rapidez de acesso e disponibilidade dos documentos;
- Utilizar ferramentas de software para apoiar as atividades de documentação.

3.4 Documentação e o processo de teste de software

Documentação de boa qualidade é um dos aspectos fundamentais para a eficácia e eficiência de um processo de teste (Herbert, 2000; Crespo et al., 2002; Sommerville, 2003; Pfleeger, 2004). Alguns possíveis benefícios são (Gelperin, 1982; Copeland, 2004; Perry, 2006):

- Aperfeiçoamento do processo: a análise dos dados registrados possibilita, por exemplo: a) avaliar ações que poderiam ter sido evitadas e outras que deveriam ter sido executadas, de acordo com os resultados obtidos e documentados; b) determinar possíveis causas para os defeitos descobertos, avaliar de que maneira poderiam ter sido evitados e propor ações corretivas;
- Melhoria no gerenciamento: dados consistentes e atualizados auxiliam no planejamento e controle do processo, fornecendo informações como: quantos casos de teste foram executados, quantos falharam, quantos ainda devem ser executados, quantas atividades estão dentro do prazo e quantas estão atrasadas, etc.;
- Melhoria no *design* dos testes: uma quantidade maior de elementos de informação disponíveis sobre os requisitos que devem ser testados aumenta a compreensão sobre o que deve ser testado e possibilita um *design* mais minucioso dos casos de teste;

- Melhoria na comunicação e coordenação entre os atores: documentação atualizada aumenta a visibilidade e o entendimento do processo, além de favorecer o compartilhamento do conhecimento produzido e facilitar análises dos resultados obtidos na execução das atividades;
- Melhoria na qualidade da própria documentação: com o conhecimento adquirido e registrado ao longo de diversos projetos pode-se aumentar ou diminuir o nível de detalhamento dos documentos, incluindo ou excluindo elementos de informação;
- Suporte à auditagem por equipes independentes de especialistas e avaliadores externos à organização para comprovar quais testes foram realizados e quais foram os resultados obtidos;
- Melhoria na produtividade por meio de, por exemplo: a) capacidade de repetir os testes realizados previamente, uma vez que estão documentados em nível suficiente de detalhes para permitir que qualquer membro da equipe os execute; b) redução do retrabalho, em virtude do reaproveitamento dos esforços e conhecimentos gerados previamente (exemplo: evitar duplicação de casos de teste para detectar o mesmo tipo de defeito); c) guia e orientação a novos membros da equipe, reduzindo a dependência do conhecimento e da experiência de indivíduos específicos.

Contudo, obter os benefícios da documentação em um processo de teste não é trivial: deve ser feito um estudo detalhado e cuidadoso sobre o que deve ser documentado; deve ser decidido qual o nível apropriado de documentação de maneira a evitar que se documente em excesso ou menos que o necessário; e, também, deve ser considerado como manter a integridade dos dados registrados à medida que as atividades do processo são executadas (Fewster e Graham, 2000; Black, 2003; Crespo e Jino, 2005). Muitas empresas decidem não documentar o processo e justificam a decisão reafirmando muitos dos mitos apresentados anteriormente (Crespo et al. 2002, 2004). Parte da solução deste problema reside em seguir uma das diretrizes apresentadas, que sugere a utilização de um padrão para registro dos elementos de informação de um processo.

Considerando o contexto de um processo de teste de software, existe a norma internacional IEEE Std 829, que pode ser empregada como referência e guia para documentação (IEEE, 1998b; Craig e Jaskiel, 2002; Crespo et al., 2002; Crespo e Jino, 2005). Essa norma é apresentada resumidamente a seguir.

3.4.1 A Norma IEEE Std 829-1998

A norma descreve um conjunto de oito documentos para documentação de um processo de teste, abordando as atividades de planejamento, projeto (*design*) e registro da execução do teste. A Figura 3.1 ilustra os documentos e os relacionamentos entre eles.

O documento "Plano de Teste" aborda a tarefa de planejamento do teste.

Os três documentos a seguir abordam a tarefa de design dos testes: "Especificação de Projeto (*Design*) de Teste", "Especificação de Caso de Teste" e "Especificação de Procedimento de Teste".

O registro das atividades de teste é feito nos seguintes documentos: "Diário de Teste", "Relatório de Incidente de Teste", "Relatório-Resumo de Teste" e "Relatório de Encaminhamento de Item de Teste".

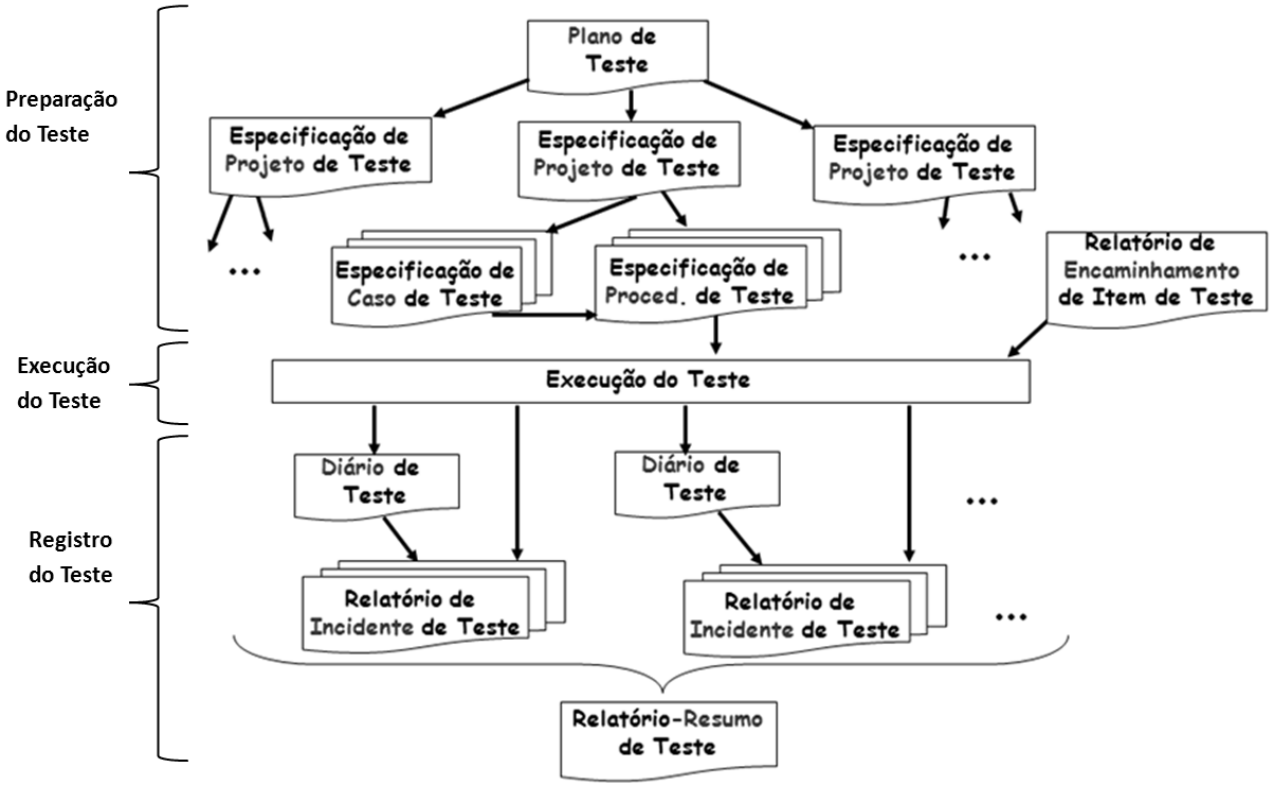

Figura 3.1: Vínculos entre os documentos de teste de acordo com a norma IEEE Std 829-1998 (Crespo et al., 2004).

As próximas seções apresentam resumidamente cada documento.

Plano de Teste (*Test Plan*)

Neste documento são registradas informações relativas ao planejamento para execução do teste, incluindo os itens e as funcionalidades a serem testados, a abordagem que será utilizada, o cronograma das atividades, os recursos necessários, as pessoas responsáveis por cada atividade, a definição do ambiente operacional para a execução dos testes e os riscos associados ao projeto sendo planejado. O Plano de Teste deve ser encarado como um guia para cada projeto de teste a ser realizado; ele possui a seguinte estrutura:

a) Identificador do Plano de Teste;
b) Introdução;
c) Itens de Teste;
d) Funcionalidades e Características do Software que Devem ser Testadas;
e) Funcionalidades e Características do Software que Não Devem ser Testadas;
f) Abordagem;
g) Critérios de Aprovação/Reprovação de Itens;
h) Critérios de Suspensão e Requisitos para a Retomada do Teste;
i) Produtos do Teste;
j) Tarefas de Teste;
k) Requisitos de Ambiente;
l) Responsabilidades;
m) Equipe e Treinamento Necessários;
n) Cronograma;
o) Riscos e Contingências;
p) Apêndices;
q) Aprovações.

Especificação de Projeto de Teste (*Test Design Specification*)

Este documento registra as informações que refinam a abordagem apresentada no Plano de Teste: as funcionalidades e características a serem testadas pelo projeto são detalhadas e associadas a casos de teste e procedimentos de teste; também são identificados os critérios de aprovação do teste. Este documento possui a seguinte estrutura:

a) Identificador da Especificação de Projeto de Teste;

b) Funcionalidades e Características Tratadas;
c) Refinamentos da Abordagem de Teste;
d) Identificação dos Casos de Teste Associados;
e) Critérios de Aprovação/Reprovação.

Especificação de Procedimento de Teste (*Test Procedure Specification*)

Este documento contém informações sobre a descrição dos passos para executar um determinado conjunto de casos de teste ou, de um modo mais geral, os passos usados para analisar um item de software com o objetivo de avaliar um conjunto de funcionalidades e características. Este documento possui a seguinte estrutura:

a) Identificação da Especificação de Procedimento de Teste;
b) Propósito do Procedimento de Teste;
c) Requisitos Especiais;
d) Passos do Procedimento de Teste
    - Registro de Teste
    - Preparação do Procedimento
    - Início do Procedimento
    - Procedimento de Teste
    - Medição de Teste
    - Suspensão de Teste
    - Retomada de Teste
    - Parada de Teste
    - Encerramento de Teste
    - Contingências

Especificação de Caso de Teste (*Test Case Specification*)

Este documento registra itens de informação que caracterizam um caso de teste, como dados de entrada, resultados esperados, ações e condições gerais para a execução do teste. A norma justifica a separação deste documento em relação ao documento Especificação de Projeto de Teste tanto para permitir seu uso em mais de um processo de teste como para permitir sua reutilização nos testes de outros produtos. Uma Especificação de Caso de Teste possui a seguinte estrutura:

a) Identificador da Especificação de Caso de Teste;
b) Itens de Testes;
c) Especificações de Entrada;

d) Especificações de Saída;

e) Requisitos de Ambiente de Teste;

f) Requisitos de Procedimentos Especiais;

g) Dependências entre Casos de Teste.

Diário de Teste (*Test Log*)

Apresenta registros cronológicos dos detalhes relevantes relacionados com a execução dos testes, e possui a seguinte estrutura:

a) Identificador do Diário de Teste;

b) Descrição do Teste;

c) Registros de Atividades e Eventos.

- Descrição da Execução
- Resultados do Procedimento
- Informação de Ambiente
- Eventos não Esperados
- Identificadores de Relatórios de Incidentes

Relatório de Incidente de Teste (*Test Incident Report*)

Documenta qualquer evento que ocorra durante as atividades de teste e que requeira análise posterior, e possui a seguinte estrutura:

a) Identificador do Relatório de Incidente de Teste;

b) Resumo do Incidente;

c) Descrição do Incidente;

- Entradas do teste;
- Resultados esperados;
- Resultados obtidos;
- Anomalias;
- Data e hora;
- Passo do procedimento de teste;
- Ambiente de teste;
- Tentativas de repetição;
- Testadores;
- Observadores do teste.

d) Impacto do Incidente.

Relatório-Resumo de Teste (*Test Summary Report*)

Apresenta de forma resumida os resultados das atividades de teste associadas com uma ou mais especificações de projeto de teste, e contém a seguinte estrutura:

a) Identificador do Relatório-Resumo de Teste;
b) Resumo da avaliação dos Itens de Teste;
c) Desvios das especificações;
d) Avaliação de Abrangência do Teste;
e) Resumo de Resultados;
f) Avaliação dos Itens de Teste;
g) Resumo de Atividades.
h) Aprovações

Relatório de Encaminhamento de Item de Teste (*Test Item Transmittal Report*)

Identifica os itens de software encaminhados para teste no caso de existirem equipes distintas para tarefas de desenvolvimento e de teste. Contém a seguinte estrutura:

a) Identificador do relatório;
b) Itens enviados;
c) Localização dos itens;
d) Estado atual;
e) Aprovações.

3.4.2 Considerações sobre a utilização da norma IEEE Std 829-1998

A norma é um guia para definição dos elementos de informação a serem documentados em um processo de teste; entretanto, ela não deve ser aplicada de maneira literal como é apresentada: sua utilização requer um estudo prévio do contexto onde ela será aplicada para que as devidas adaptações sejam feitas (Crespo et al., 2002; Crespo et al., 2004; Crespo e Jino, 2005).

A norma define o formato e os elementos de informação necessários para cada documento, sem determinar quais documentos e quais elementos de informação devem ser utilizados em uma dada situação. Os oito documentos podem ser utilizados como proposto pela norma ou com seus elementos de informação combinados em um número menor de documentos. Definir os documentos a serem utilizados, assim como os elementos de informação a registrar, depende das

características do produto em teste e do contexto de cada empresa em particular: variáveis como a natureza das aplicações, o tamanho e o escopo dos projetos, e a compatibilidade com as metodologias de projeto em uso têm grande impacto nessa decisão (Black, 2003; Crespo et al., 2004). Por exemplo: em um projeto no qual esteja sendo desenvolvido um software para controle de um grande avião para transporte de passageiros, provavelmente os oito documentos são o mínimo a utilizar dada a natureza crítica do projeto e a necessidade de rigor nos testes (Phillips, 1998). A maioria das empresas, contudo, pode adaptar esses oito documentos de acordo com a própria necessidade e contexto: um arranjo possível é Plano de Teste, Especificações de Procedimento de Teste e Relatório-Resumo de Teste. Estes três documentos registram o esforço do teste que se pretende realizar, os testes realizados e os incidentes ocorridos de modo a orientar sua correção (Phillips, 1998; Crespo et al., 2004).

A utilização da norma não é isenta de críticas e advertências (Kaner, 2001). Entretanto, a experiência do Grupo de Teste de Software do CTI no trabalho com pequenas empresas mostra que ela pode ser empregada com sucesso (Crespo et al., 2002; Crespo et al., 2004; Crespo e Jino, 2005).

3.4.3 Situação atual da norma IEEE Std 829

A norma possui uma nova versão, lançada em julho de 2008. A versão de 1998 é conhecida como "IEEE Std 829-1998 Standard for Software Test Documentation" e a versão de 2008 foi renomeada para "IEEE Std 829-2008 Standard for Software and System Test Documentation" (IEEE, 2008).

A mudança da versão de 2008 em relação à de 1998 é, fundamentalmente, a seguinte: na primeira versão existe somente a dimensão dos documentos do teste e, na versão atual, além da dimensão dos documentos existe a dimensão que trata das atividades de um processo de teste. Assim, o componente adicional desta nova versão da norma é a visão do teste como um processo, que pode visar somente um software como, também, sistemas baseados em software (p.ex.: sistemas embarcados).

Em relação à documentação, o princípio da versão de 1998 permanece inalterado na versão de 2008: a norma indica os documentos necessários para um processo de teste, mas cabe a cada empresa analisar seu contexto e o do produto sendo testado, para definir quais documentos, ou conjuntos de elementos de informação de cada um deles, deverão ser usados para documentar as

atividades do teste. Como este trabalho é fundamentado na versão de 1998 e com a premissa do teste sendo baseado em documentação (sem considerar a dimensão do processo), são feitas observações relativas somente à organização dos documentos da versão 2008 e sua equivalência com os documentos da versão de 1998. A dimensão do processo é referenciada no capítulo que descreve o protótipo, quando o modelo de rastreabilidade da ferramenta é apresentado e analisado, e na conclusão, onde são indicados trabalhos futuros. A versão atual do protótipo está preparada para gerar documentos de acordo com a versão de 1998; as modificações necessárias para atender à versão de 2008 fazem parte dos trabalhos futuros decorrentes desta dissertação.

Os documentos propostos na versão de 2008, e sua relação com seus pares da versão de 1998, são brevemente apresentados a seguir.

Plano-Mestre de Teste (*Master Test Plan*)

Este documento não existe na versão de 1998. Ele tem o propósito de registrar o planejamento geral do teste, fornecer uma visão geral para a gerência do teste e ser o documento que integra os planos de teste específicos de outros níveis (p.ex.: Plano de Teste de Unidade, Plano de Teste de Integração, Plano de Teste de Sistema). Sua estrutura é apresentada a seguir.

1. Introdução
    - 1.1. Identificador do documento
    - 1.2. Escopo
    - 1.3. Referências
    - 1.4. Visão geral do sistema e das características
    - 1.5. Visão geral do teste
        - 1.5.1 Organização
        - 1.5.2 Cronograma-mestre do teste
        - 1.5.3 Organização dos níveis de integridade
        - 1.5.4 Visão geral dos recursos
        - 1.5.5 Responsabilidades
        - 1.5.6 Ferramentas, técnicas, métodos e métricas
2. Detalhamento
    - 2.1. Processo de teste e definição dos níveis de teste
        - 2.1.1 Processo: Gerenciamento
        - 2.1.2 Processo: Aquisição
        - 2.1.3 Processo: Fornecimento
        - 2.1.4 Processo: Desenvolvimento
        - 2.1.5 Processo: Operação
        - 2.1.6 Processo: Manutenção
    - 2.2. Requisitos para documentação do teste
    - 2.3. Requisitos para administração do teste

2.4. Requisitos para geração de relatórios
3. Geral
3.1. Glossário
3.2. Histórico e procedimentos de modificação do documento

Relatório de Encaminhamento de Item de Teste (*Test Item Transmittal Report*)

Na versão de 2008, não existe mais este documento; seus elementos de informação agora fazem parte do documento "*Level Test Plan*".

Plano de Teste para nível de teste (*Level Test Plan*)

1. Introdução
1.1. Identificador do documento
1.2. Escopo
1.3. Referências
1.4. Nível (posicionamento global)
1.5. Classes de teste e condições globais de teste
2. Detalhamento
2.1 Itens de teste e seus identificadores
2.2 Matriz de Rastreabilidade do Teste
2.3 Características do Software que Devem ser Testadas;
2.4 Características do Software que Não Devem ser Testadas;
2.5 Abordagem
2.6 Critérios de Aprovação/Reprovação de Itens
2.7 Critérios de suspensão e requisitos para reinício
2.8 Artefatos resultantes
3. Gerenciamento do Teste
3.1 Atividades e tarefas planejadas; progresso do teste
3.2 Ambiente e infra-estrutura necessários
3.3 Responsabilidades e autorizações
3.4 Interfaces entre os atores
3.5 Recursos e sua alocação
3.6 Treinamento
3.7 Cronograma, estimativas e custos
3.8 Riscos e contingências
4. Geral
4.1 Procedimentos para Garantia da Qualidade
4.2 Métricas
4.3 Cobertura do teste
4.4. Glossário
4.5. Histórico e procedimentos de modificação do documento

Este documento é equivalente, em termos de estrutura, ao documento Plano de Teste (*Test Plan*) da versão de 1998. A palavra "Level" do título é substituída pelo nível de teste ao qual o documento se refere (p.ex.: *Integration Test Plan*, *System Test Plan*, e *Acceptance Test Plan*). Geralmente os projetos possuem diferentes níveis de teste, cada um com recursos, ambientes e métodos diferentes, o que implica planejamento específico para cada nível.

Em relação à versão de 1998, novos elementos de informação incluídos na estrutura deste documento são: a) indicação do nível do plano e sua relação com outros níveis; b) utilização de uma matriz de rastreabilidade; c) interfaces entre os atores. A seção "Aprovações" (versão de 1998) foi alterada para "Responsabilidades e autorizações".

Design de teste para nível de teste (_Level Test Design_)

Neste documento é refinada a abordagem do teste, descrita no Plano de Teste, e são identificadas as características que serão testadas por este *design* e casos de teste associados. Em relação à versão de 1998, novos elementos de informação incluídos na estrutura deste documento são: a) escopo e referências a documentos; b) artefatos resultantes da elaboração do documento; c) glossário; d) histórico e procedimentos de modificação do documento.

1. Introdução
    - 1.1. Identificador do documento
    - 1.2. Escopo
    - 1.3. Referências
2. Detalhamento do *Design* do Teste
    - 2.1. Características a serem testadas
    - 2.2. Refinamentos da abordagem
    - 2.3. Identificação do teste
    - 2.4. Critérios de aprovação/reprovação da Característica
    - 2.5 Artefatos resultantes
3. Geral
    - 3.1. Glossário
    - 3.2. Histórico e procedimentos de modificação do documento

Casos de teste para nível de teste (_Level Test Case_)

Em relação à versão de 1998, novos elementos de informação incluídos na estrutura deste documento são: a) escopo e referências a documentos; b) descrição do contexto; c) notação de identificação – algum esquema específico de identificação dos casos de teste, d) objetivo, e) glossário; f) histórico e procedimentos de modificação do documento. A seção "Itens de Teste Associados" foi removida.

1. Introdução (uma vez por documento)
    1.1. Identificador do documento
    1.2. Escopo
    1.3. Referências
    1.4. Contexto
    1.5. Notação de identificação (*Notation for description*)
2. Detalhamento (uma vez por caso de teste)
    2.1. Identificador do caso de teste
    2.2. Objetivo
    2.3. Entradas
    2.4. Resultados esperados
    2.5. Necessidades de ambiente
    2.6. Requisitos especiais de procedimento
    2.7. Dependências entre casos de teste
3. Geral (uma vez por documento)
    3.1. Glossário
    3.2. Histórico e procedimentos de modificação do documento

Procedimentos de teste para nível de teste (*Level Test Procedure*)

Os novos elementos de informação incluídos na estrutura deste documento são: a) escopo e referências a documentos; b) relacionamento com outros procedimentos; c) glossário; d) histórico e procedimentos de modificação do documento. A seção "Requisitos especiais" foi alterada para "Entradas, saídas e requisitos especiais".

1. Introdução
    1.1. Identificador do documento
    1.2. Escopo
    1.3. Referências
    1.4. Relacionamentos com outros procedimentos
2. Detalhamento
    2.1. Entradas, saídas e requisitos especiais
    2.2. Descrição ordenada dos passos para executar os casos de teste
        Registro do teste
        Preparação
        Início
        Procedimento
        Medição
        Suspensão
        Re-início
        Parada
        Encerramento do teste
        Contingências
3. Geral
    3.1. Glossário

3.2. Histórico e procedimentos de modificação do documento

Diário de teste para nível de teste (*Level Test Log*)

Os novos elementos de informação incluídos na estrutura deste documento são: a) escopo e referências a documentos; c) glossário.

1. Introdução
    1.1. Identificador do documento
    1.2. Escopo
    1.3. Referências
2. Detalhamento
    2.1. Descrição
    2.2. Atividades e eventos
        Descrição da execução
        Resultados do procedimento
        Informações do ambiente de execução
        Eventos anômalos
        Identificadores de relatos de incidentes
3. Geral
    3.1. Glossário

Relatório de Anomalia (*Anomaly Report*)

Os novos elementos de informação incluídos na estrutura deste documento são: a) escopo e referências a documentos; b) avaliação da urgência; c) descrição da ação corretiva; d) estado atual da anomalia; e) conclusões e recomendações; f) histórico e procedimentos de modificação do documento. A seção "Descrição do incidente" foi renomeada para "Descrição da anomalia" e sua subseção "Anomalias" foi renomeada para "Resultados inesperados".

1. Introdução
    1.1. Identificador do documento
    1.2. Escopo
    1.3. Referências
2. Detalhamento
    2.1. Resumo
    2.2. Data de descoberta da anomalia
    2.3. Contexto
    2.4. Descrição da anomalia
        Entradas
        Resultados esperados
        Resultados obtidos
        Resultados inesperados
        Passos do procedimento

Ambiente de execução
Tentativas de repetição
Testadores
Observadores

2.5. Impacto
2.6. Avaliação da urgência (pelo gerador da anomalia)
2.7. Descrição da ação corretiva
2.8. Estado atual da anomalia
2.9. Conclusões e recomendações

3. Geral
3.1. Histórico e procedimentos de modificação do documento

Relatório-Resumo de Teste (*Test Summary Report*)

Este documento não existe mais na versão de 2008. Em seu lugar foram criados três novos documentos: *Level Interim Test Status Report*, *Level Test Report* e *Master Test Report*. Estes três documentos são apresentados a seguir.

Relatório de Situação do Nível de Teste (*Level Interim Test Status Report*)

O objetivo é apresentar de forma condensada os resultados das atividades de teste em andamento e, opcionalmente, prover avaliações e recomendações baseadas nos resultados apresentados. A palavra "Nível" no título refere-se ao nível de teste para o qual o relatório foi confeccionado. O detalhamento do relatório pode variar conforme as características da empresa e do projeto.

1. Introdução
1.1. Identificador do documento
1.2. Escopo
1.3. Referências
2. Detalhamento
2.1. Resumo do estado atual do teste
2.2. Mudanças no planejamento
2.3. Métricas relativas ao estado atual
3. Geral
3.1. Histórico e procedimentos de modificação do documento

Relatório de Nível de Teste (*Level Test Report*)

O objetivo é apresentar de forma condensada os resultados finais das atividades de teste e prover avaliações e recomendações baseadas nos resultados apresentados. A palavra "Nível" no título refere-se ao nível de teste para o qual o relatório foi confeccionado. Dependendo do porte do projeto, é possível existir somente um relatório que aglutina os resultados de todos os níveis de teste. O nível de detalhamento pode variar conforme as características da empresa e do projeto.

1. Introdução
    - 1.1. Identificador do documento
    - 1.2. Escopo
    - 1.3. Referências
2. Detalhamento
    - 2.1. Visão geral dos resultados do nível de teste
    - 2.2. Detalhamento dos resultados dos testes
    - 2.2. Justificativas para as decisões
    - 2.4. Conclusões e recomendações
3. Geral
    - 3.1. Glossário
    - 3.2. Histórico e procedimentos de modificação do documento

Relatório-Mestre de Teste (*Master Test Report*)

O objetivo é apresentar de forma condensada os resultados finais das atividades de teste de todos os Relatórios de Nível de Teste e prover avaliações e recomendações baseadas nos resultados apresentados. Para todo Plano-Mestre de Teste que for criado existe a necessidade de um Relatório-Mestre de Teste para descrever os resultados da implementação do plano.

1. Introdução
    - 1.1. Identificador do documento
    - 1.2. Escopo
    - 1.3. Referências
2. Detalhamento
    - 2.1. Visão geral consolidada de todos os resultados dos testes
    - 2.2. Justificativas para as decisões
    - 2.3. Conclusões e recomendações
3. Geral
    - 3.1. Glossário
    - 3.2. Histórico e procedimentos de modificação do documento

A seguir são apresentadas algumas considerações relativas à necessidade de suporte automatizado para as atividades de documentação do teste.

3.4.4 Suporte automatizado à documentação em um processo de teste

A documentação de um processo de teste é formada por uma quantidade muito grande de elementos de informação; além disso, os vínculos que devem ser registrados e mantidos entre esses elementos de informação demandam a manipulação de uma quantidade considerável de dados. Assim, se as tarefas relacionadas à documentação forem executadas de maneira exclusivamente manual, elas se tornam improdutivas e muito propensas a enganos e esquecimentos (Pressman, 2002; Sommerville, 2003); caso não exista um suporte automatizado adequado, a documentação ficará rapidamente desatualizada, incompleta e inconsistente (DiMaggio, 2001; Sommerville, 2003). Por exemplo, no momento de uma modificação em um artefato, todos os elementos de informação, em todos os documentos associados, devem ser facilmente localizados, revisados e, se necessário, atualizados; além disso, os atores relacionados com cada documento devem ser localizados e estar cientes de eventuais modificações. Se todas essas tarefas forem conduzidas de maneira manual, documentos podem não ser revisados e atores importantes podem deixar de ser consultados.

Neste contexto, ferramentas para produção e manutenção de documentos representam um importante apoio, pois auxiliam a manter a documentação consistente e atualizada facilitando a coordenação entre as partes envolvidas e a propagação das modificações a todos os elementos de informação relacionados (Pressman, 2002; Sommerville, 2003; Pfleeger, 2004). O fato de a ferramenta ser baseada em alguma norma ou modelo significa atender uma das diretrizes para produção de documentação de boa qualidade. Basear-se em uma norma também significa fazer uso de uma solução que abrange um contexto amplo e não somente um contexto específico, justamente pelo fato da norma ser gerada por um grupo de especialistas com diferentes experiências em diferentes contextos. PROMETEU, que é apresentado no Capítulo 6, é um protótipo de ferramenta baseada na norma IEEE Std 829-1998, que oferece suporte à documentação de um processo de teste com rastreabilidade entre os elementos de informação que compõem a documentação.

3.5 Síntese do Capítulo

Apesar da importância da documentação para o sucesso de um projeto de software, em um grande número de organizações desenvolvedoras de software ela ainda varia de desatualizada a inexistente. Isso pode ser atribuído, em grande parte, à existência de uma cultura negativa em relação à documentação, segundo a qual documentar é perda de tempo e recursos. Essa cultura é realimentada, geralmente, em função da não aplicação de algumas diretrizes para obtenção de documentação de boa qualidade, diretrizes como: utilizar padrões e modelos, revisar e modificar os modelos regularmente, considerar a documentação como um dos processos do desenvolvimento de software, avaliar constantemente diversos aspectos relacionados ao seu conteúdo e utilizar suporte automatizado adequado, priorizando documentos dinâmicos.

A visão negativa supracitada também existe em relação à necessidade de documentar o processo de teste. No entanto, sabe-se que uma documentação de boa qualidade é um aspecto crítico para o sucesso de um processo de teste de software, com benefícios que incluem: a) melhoria no gerenciamento do processo; b) melhoria na comunicação e coordenação entre os atores; c) melhoria no *design* dos casos de teste; d) preservação do conhecimento produzido ao longo de diversos projetos; e) redução do retrabalho, pela capacidade de repetir os testes realizados previamente e reaproveitar testes previamente projetados; e f) melhoria do próprio processo.

No entanto, para obter estes benefícios, deve ser feito um planejamento cuidadoso do que deve ser documentado, para diminuir riscos como documentar demais ou documentar de maneira insuficiente. Nesse sentido, a norma IEEE Std 829, resultado do conhecimento de diversos especialistas de diversos contextos ao redor do mundo, serve como guia confiável na definição de um padrão de documentos para o processo de teste de qualquer empresa. Ressalta-se, porém, que a utilização da norma requer um estudo prévio dos seus elementos para uma correta adequação ao contexto onde será empregada.

Este capítulo também apresentou as duas versões da Norma IEEE Std 829: a versão de 1998, denominada “IEEE Std 829-1998 Standard for Software Test Documentation” e a versão de 2008, denominada “IEEE Std 829-2008 Standard for Software and System Test Documentation”. A mudança da versão de 2008 em relação à de 1998 é, fundamentalmente, a seguinte: na versão de 1998 existia somente a dimensão dos documentos do teste e, na versão atual, além da dimensão dos documentos existe a dimensão que trata das atividades de um

processo de teste. Foram apresentadas as duas versões da norma (a de 1998 e a de 2008), as estruturas dos documentos de cada versão e a equivalência entre os documentos das duas versões.

Este trabalho é fundamentado na versão de 1998, nos trabalhos do Grupo de Teste do CTI e na premissa do processo de teste ser baseado em documentação. Assim, a versão atual do protótipo está preparada para gerar documentos de acordo com a versão de 1998; as modificações necessárias para atender à versão de 2008 (tanto em relação à dimensão do processo como em relação à nova estrutura e relacionamento dos documentos) fazem parte dos trabalhos futuros decorrentes desta dissertação.

Documentar um processo de teste envolve manipular uma grande quantidade de elementos de informação, geralmente com relacionamentos muito fortes entre si. Além disso, grande parte da documentação será alterada durante a execução das atividades do processo. Como a qualidade e utilidade da documentação são diretamente proporcionais à sua completude, consistência e atualidade, torna-se necessária a existência de um mecanismo que contribua para manter essas características entre os diversos elementos de informação que compõem a documentação. Esse mecanismo é a rastreabilidade, que é abordada no próximo capítulo.

Capítulo 4

Rastreabilidade

Rastreabilidade é uma propriedade imprescindível em um processo de software para obtenção de um produto de boa qualidade (Domges e Pohl, 1998; Leffingwell e Widrig, 2000). Um exemplo marcante é o Departamento de Defesa dos EUA, para o qual a rastreabilidade é uma propriedade obrigatória em todos os contratos de desenvolvimento de software (Watkins e Neal, 1994; Ramesh e Jarke, 2001). Normas e modelos para melhoria de processo também destacam a rastreabilidade como um dos mecanismos necessários para se obter software de boa qualidade (Paulk et al., 1993; IEEE, 1998a; Ahern et al., 2003).

O tema rastreabilidade é apresentado neste capítulo nas seguintes seções: a Seção 4.1 apresenta definições de rastreabilidade; a Seção 4.2 apresenta motivações para sua utilização; a Seção 4.3 apresenta diretrizes para o planejamento e implantação da rastreabilidade; a Seção 4.4 apresenta modelos de rastreabilidade; a Seção 4.5 apresenta questões relacionadas ao armazenamento e visualização dos *links* de rastreabilidade; a Seção 4.6 apresenta a rastreabilidade no contexto do teste de software; a Seção 4.7 destaca a necessidade de suporte automatizado à rastreabilidade; e a Seção 4.8 apresenta a síntese do capítulo.

4.1 Definições

Algumas definições de rastreabilidade são:

- Habilidade de descrever e acompanhar a evolução de um requisito ao longo do seu ciclo de vida, tanto na direção das suas fontes (pessoas, idéias, artefatos) como na direção dos artefatos que o sucedem (p.ex.: elementos do *design*, documentos do teste) (Gotel e Finkelstein, 1995);

- Conjunto de informações de diversos tipos que deve ser mantido sobre as dependências entre requisitos, as motivações para sua existência e os artefatos que implementam esses requisitos (Kotonya e Sommerville, 1998);
- Propriedade de um sistema que possibilita definir e manter associações e ligações lógicas entre um requisito e outros artefatos do sistema, como requisitos de vários tipos, regras de negócio, componentes de *design*, códigos-fonte, casos de teste (Wiegers, 2003);
- Capacidade de rastrear um elemento do projeto (que passa a ser um item de rastreabilidade quando for rastreado) a outros elementos correlatos, especialmente aqueles relacionados aos requisitos. Exemplos de itens de rastreabilidade são requisitos e seus diferentes tipos, *designs*, artefatos do processo de teste e documentos de suporte ao usuário final (Kruchten, 2003);
- Grau de relacionamento que pode ser estabelecido entre dois ou mais artefatos do processo de desenvolvimento, especialmente artefatos que, entre si, possuam relacionamento do tipo predecessor-sucessor ou mestre-subordinado, como o relacionamento existente entre um requisito e um ou mais artefatos do *design* que o implementam (IEEE, 1990).

A rastreabilidade de artefatos não se restringe somente aos requisitos; em maior ou menor grau deve existir ao longo de todo o processo de software, entre os artefatos gerados ao longo de todas as fases do desenvolvimento de um produto de software (Paulk et al., 1993; Ahern et al., 2003). Neste trabalho, a rastreabilidade é tratada com foco nos artefatos de um processo de teste de software.

4.2 Motivações para utilização da rastreabilidade

Algumas motivações para implantar rastreabilidade em um processo são:

a) Maior exatidão na análise do impacto de modificações: associações (*links*) de rastreabilidade possibilitam descobrir rapidamente todos os artefatos relacionados a um artefato que será modificado, além dos atores relacionados, sendo possível estimar com maior exatidão o custo da modificação (Leffingwell e Widrig, 2000; Wiegers, 2003);

b) Melhor controle do processo: *links* de rastreabilidade possibilitam, entre outras coisas, descobrir se artefatos necessários ao processo ainda não foram criados (por exemplo, um requisito foi especificado, mas não possui nenhum caso de teste associado) ou se foram criados artefatos desnecessários (por exemplo, foram criados casos de teste para requisitos que não existem na documentação) (Wiegers, 2003);
c) Manutenção da integridade dos elementos de informação do processo: *links* de rastreabilidade entre os atores, os artefatos e seus elementos de informação contribuem significativamente para manter a consistência e atualidade da documentação de um processo (Wiegers, 2003). Quando, por exemplo, um requisito for modificado é possível descobrir rapidamente todos os artefatos associados, como outros requisitos, *designs*, especificações de casos de testes, etc.; dessa maneira é possível fazer uma revisão em todos os artefatos de forma a refletir as modificações feitas;
d) Obtenção de certificação: rastreabilidade é uma propriedade obrigatória em um processo de software de acordo com diversas normas de qualidade de software (Paulk et al., 1993; Ahern et al., 2003; Wiegers, 2003).

4.3 Diretrizes para implantação da rastreabilidade e sua utilização

Implantar rastreabilidade não é trivial (Gotel e Finkelstein, 1995; Ramesh e Jarke, 2001). A escolha de quais elementos de informação serão rastreados e quais *links* de rastreabilidade serão definidos e mantidos deve considerar o contexto da empresa, as características de seus projetos e os tipos de usuários que se beneficiarão das informações de rastreabilidade (Gotel e Finkelstein, 1995; Domges e Pohl, 1998; Ramesh e Jarke, 2001).

Convém ressaltar que, apesar de ser possível estabelecer *links* entre todos os artefatos de um processo, isto não é necessário pois, quanto maior o número de elementos de informação a rastrear, maior é a complexidade e custo para implantar e manter a rastreabilidade. Ou seja, rastrear uma quantidade exagerada de dados pode ser tão prejudicial como rastrear uma quantidade insuficiente (Spence e Probasco, 1998; Domges e Pohl, 1998; Wiegers, 2003). Além disso, para a maioria dos projetos obtém-se até 80% dos benefícios da rastreabilidade rastreando até 20% dos *links* possíveis (Kotonya e Sommerville, 1998; Wiegers, 2003). E, também,

geralmente os participantes de um projeto não dispõem de tempo para coletar dados que não tenham um objetivo claro de utilização (Leffingwell e Widrig, 2000).

Para obter os benefícios advindos da implantação da rastreabilidade é necessário organizar, armazenar e gerenciar uma grande quantidade de diversos tipos de artefatos. Nesse sentido, existem modelos que prescrevem orientações e diretrizes sobre o que deve ser rastreado, em qual quantidade e quais relacionamentos devem ser estabelecidos.

4.4 Modelos de Rastreabilidade

Modelos de rastreabilidade sugerem elementos de informação que devem ser rastreados e os tipos de relacionamentos que devem ser definidos entre eles. Conhecer diversos modelos de rastreabilidade possibilita unificar e adequar seus conceitos em um modelo unificado que seja adequado para um determinado contexto; no caso deste trabalho, o contexto é representado pelo processo de teste de software. Alguns modelos de rastreabilidade encontrados na literatura são apresentados a seguir.

4.4.1 Ramesh e Jarke

Os conceitos-chave deste modelo são: entidade de rastreabilidade, *link* de rastreabilidade e tipo de usuário (Ramesh e Jarke, 2001).

Uma entidade de rastreabilidade representa os elementos de informação a serem rastreados; são representadas no meta-modelo (Figura 4.1) dentro de retângulos e com letras maiúsculas. As três principais entidades para as quais a rastreabilidade deve ser mantida são: FONTE/ORIGEM (artefatos físicos onde as informações a serem rastreadas são documentadas e mantidas), STAKEHOLDER (ator em um projeto de software) e ARTEFATO (informações de entrada para atividades e informações resultantes da execução de atividades de um processo de software, como requisitos, justificativas para decisões do projeto, *designs*, etc.). A Figura 4.1 representa os elementos essenciais da rastreabilidade neste modelo. *Links* de rastreabilidade representam as associações entre as entidades e recebem rótulos que identificam as relações entre as entidades. Exemplos são: ASSOCIADO A, TEM PAPEL NO, GERENCIA e DOCUMENTA. Exemplo: um relacionamento entre os objetos SUPOSIÇÃO e REQUISITO pode ser representado por um *link* ASSOCIADO A. Convém ressaltar que outros rótulos podem ser definidos, de acordo com a necessidade.

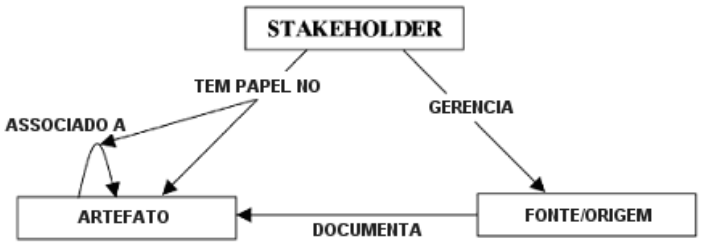

Figura 4.1: Meta-modelo de rastreabilidade (Ramesh e Jarke, 2001).

O meta-modelo representado na Figura 4.1 pode ser detalhado em maior ou menor grau em função dos tipos de usuários da rastreabilidade, que são classificados em *low-end* e *high-end*; cada tipo de usuário possui uma percepção diferente da utilidade da rastreabilidade, além de possuir necessidades diferentes de tipos e quantidade de informações de rastreabilidade.

Usuários *low-end*

Usuários *low-end* utilizam a rastreabilidade para controlar um número pequeno de entidades e *links* de rastreabilidade, restringindo-se a decomposição e alocação básica dos requisitos a outros artefatos do processo como apoio a uma verificação básica de conformidade e controle simples de modificações. Para esse tipo de usuário, a rastreabilidade é percebida simplesmente como obrigação imposta pelos patrocinadores do projeto.

Usuários *high-end*

Os usuários *high-end* percebem a rastreabilidade como um componente importante no desenvolvimento de um produto, que aumenta a probabilidade de produzir um software de boa qualidade que satisfaz todos os requisitos do cliente e que seja de fácil manutenção. Para este tipo de usuário, a rastreabilidade deve fazer parte de todo o ciclo de vida de um software, ao longo das dimensões de produto e de processo; conseqüentemente demanda uma maior quantidade e diversidade de dados. Ao invés de um modelo único como o existente para usuários *low-end*, o meta-modelo é detalhado em quatro modelos complementares: a) Gerenciamento dos Requisitos, b) Alocação a Artefatos do *Design*, c) Verificação da Qualidade e d) Gerenciamento de Justificativas e Suposições (motivações para decisões e ações) originadas pelos atores de um projeto. Esses quatro modelos representam um esquema de rastreabilidade mais detalhado e complexo, com uma quantidade maior de entidades e *links* de rastreabilidade.

O meta-modelo, tanto com foco em usuário *low-end* como com foco em usuário *high-end*, objetiva capturar as seguintes dimensões de informações de rastreabilidade:

- O quê – se refere a uma entidade de rastreabilidade (o que está sendo rastreado); tanto a entidade em si (ex.: especificação de caso de teste) como atributos da entidade que devam ser rastreados (testador, data da execução, resultado obtido);
- Quem – atores do processo e artefatos relacionados;
- Onde – quais as origens ou fontes, incluindo sua localização, dos dados que originaram o artefato, representando o elo entre o artefato e as razões básicas para sua proposição e existência. Exemplo: atores ou documentos;
- Como – qual a tecnologia utilizada e qual o formato de representação dos dados (somente texto? Combinação de texto e gráficos?);
- Por quê (*rationale*) – registro do motivo ou justificativa para uma tomada de decisão ou ação (p.ex.: descrição do motivo da criação, modificação ou evolução de um determinado artefato). O registro deste tipo de informação melhora a comunicação entre os atores e aumenta a compreensão sobre as decisões tomadas ao longo da execução de um processo. O termo em inglês *rationale* é usado muitas vezes como sinônimo e representa o conjunto de atributos que permite rastrear as decisões tomadas ao longo do processo, seus motivos, além de hipóteses e observações resultantes;
- Quando – data e hora em que um elemento de informação foi registrado ou modificado.

4.4.2 Gotel e Finkelstein

Os conceitos principais neste modelo são: a) pré e pós-rastreabilidade, b) direção ou sentido da rastreabilidade: *backward* e *forward*, c) estruturas de contribuição (Gotel e Finkelstein, 1994; Gotel e Finkelstein, 1995).

Pré-rastreabilidade se refere à atenção que deve ser dada a diversos aspectos relacionados a um requisito antes da sua inclusão em uma Especificação de Requisitos. São representadas por *links* do tipo *backward*. Pós-rastreabilidade se refere a questões relacionadas a um requisito após sua inclusão em uma Especificação de Requisitos. São representadas por *links* do tipo *forward*.

Uma associação *backward* (rastreabilidade para trás) liga um requisito às suas fontes, permitindo rastrear a origem do requisito: registros de suposições, justificativas, atores e outros artefatos que forneceram informações que contribuíram para a definição do requisito. Uma

associação *forward* (rastreabilidade para frente) liga um requisito a artefatos geralmente criados posteriormente, possibilitando rastrear quais artefatos existem em função de um determinado requisito.

Estruturas de contribuição se referem aos vários atores que influenciam na descoberta, especificação e posterior evolução dos requisitos. Sobre os atores podem ser registrados diversos elementos de informação, como: papéis desempenhados em um projeto, relações com outros atores, tipos de contribuições (hipóteses, justificativas para decisões, área de conhecimento, etc.) ao projeto e aos requisitos, entre outros conforme o contexto da organização e os objetivos a serem atingidos.

4.4.3 Domges e Pohl

Neste modelo são destacadas quatro categorias de informação que devem ser rastreadas (Domges e Pohl, 1998):

- *Links* bidirecionais: representam as associações entre dois ou mais artefatos nas duas direções; por exemplo: do artefato A até o artefato B, e do artefato B até o artefato A. Os tipos de artefatos a serem rastreados são requisitos, requisitos derivados e outros artefatos de alguma maneira relacionados aos requisitos;
- Estruturas de contribuição: representam os atores, seus papéis e suas contribuições na criação ou manutenção de um artefato;
- Motivações das decisões de processo: representam as hipóteses e justificativas que fundamentaram as decisões tomadas ao longo de um projeto. Entre outros benefícios, ocorre uma melhoria na gerência de modificações através da redução das chances de serem negligenciadas importantes considerações e motivações;
- Dados do processo: representam as tarefas realizadas, recursos consumidos e métricas relacionadas às tarefas e aos recursos. Um benefício típico é a melhoria no controle de um projeto ao se visualizar mais claramente o progresso da execução das atividades.

4.4.4 Pfleeger e Bohner

Neste modelo os autores propõem os conceitos de rastreabilidade vertical e rastreabilidade horizontal, para indicar os relacionamentos entre artefatos de tipos distintos e entre artefatos do mesmo tipo (Pfleeger e Bohner, 1990; Pfleeger, 2004).

Rastreabilidade vertical representa as relações existentes entre artefatos do mesmo tipo, gerados na mesma fase de um processo de software (exemplo: interdependências entre diversos requisitos) ou entre as partes que compõem um artefato (exemplo: as seções que compõem um documento). Rastreabilidade horizontal representa as associações entre tipos distintos de artefatos, geralmente originados em fases diferentes do ciclo de vida do software (exemplo: definição de *links* entre um requisito e os casos de teste que o validam).

A Figura 4.2 exemplifica a rastreabilidade horizontal e vertical: os retângulos agrupam artefatos de mesmo tipo (cada artefato é representado por um círculo); as setas sólidas dentro dos retângulos representam a rastreabilidade vertical, enquanto as setas tracejadas entre os retângulos representam a rastreabilidade horizontal.

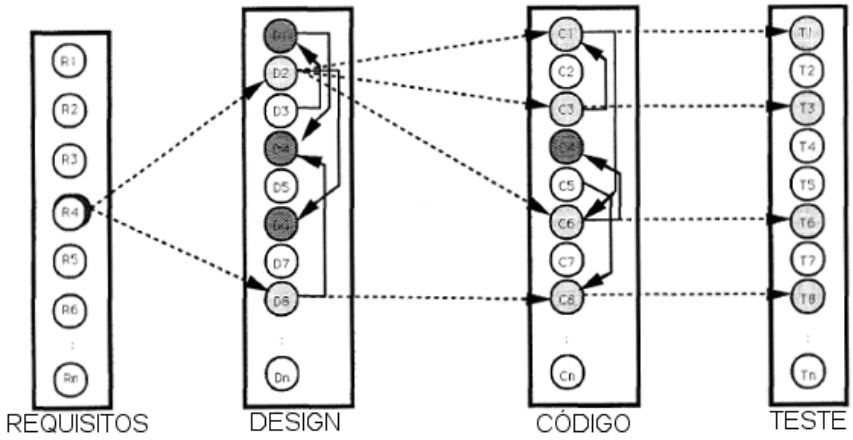

Figura 4.2: Rastreabilidade horizontal e vertical (Pfleeger e Bohner, 1990).

Os autores propõem a utilização de grafos para armazenar os *links* de rastreabilidade entre os artefatos.

4.4.5 RUP (Rational Unified Process)

O RUP é um processo de software criado pela Rational Software Corporation (http://www.rational.com/), adquirida pela IBM (http://www.ibm.com/). Os conceitos-chave apresentados neste modelo são: item de rastreabilidade e relacionamento de rastreabilidade (Kruchten, 2003).

Um item de rastreabilidade representa qualquer elemento de um processo que precise ser rastreado de forma explícita a partir de outro elemento do processo de software. Exemplos de

itens são: regras de negócio, casos de uso, uma seção de caso de uso, propostas de modificação, casos de teste. Cada item de rastreabilidade possui atributos que podem ou não serem rastreados.

Um relacionamento de rastreabilidade é um vínculo entre dois ou mais itens. Esse vínculo pode ser bidirecional ou em um único sentido e seu estado deve ser indicado. Por exemplo: um vínculo é dito "SUSPEITO" quando um dos itens associados for modificado e esta modificação não for revisada e o estado do vínculo não for atualizado; ou seja, a modificação pode ter gerado inconsistência e/ou desatualização na documentação.

A Figura 4.3 representa de maneira simplificada os itens e relacionamentos de rastreabilidade de um processo de software, com foco nos requisitos e no seu relacionamento com outros tipos de artefatos. Os requisitos do software são compostos pelos seguintes tipos de artefato: "Regra de Negócio", "Documento de Visão", "Solicitação de Envolvido", "Modelo de Casos de Uso" e "Especificação Suplementar". A partir dos requisitos do software é feita a vinculação com artefatos do *Design*, com artefatos do Teste e com a documentação de treinamento e do usuário final.

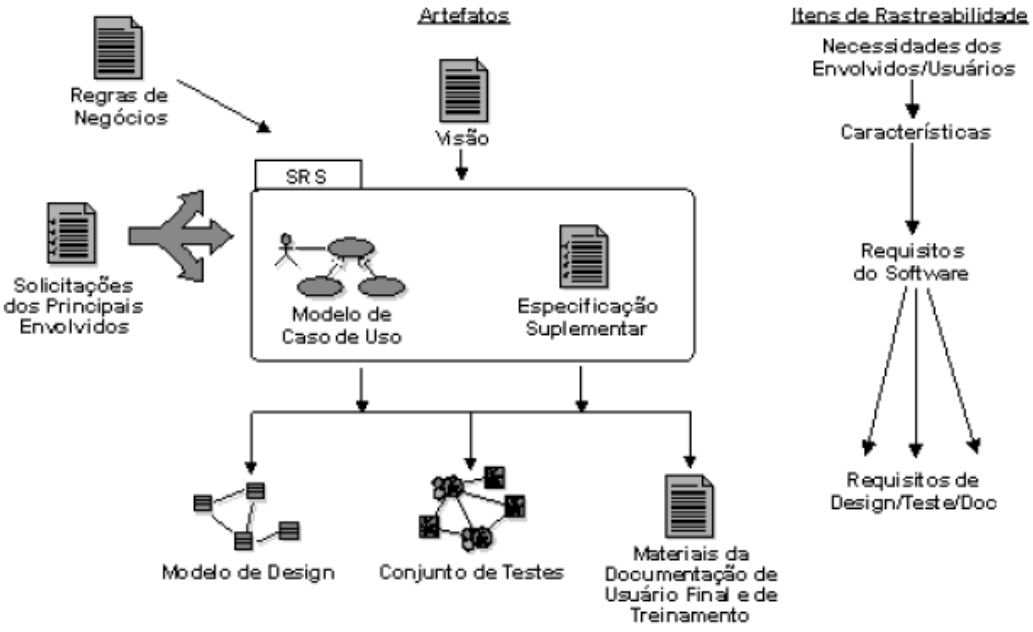

Figura 4.3: Rastreabilidade com foco nos requisitos (Kruchten, 2003).

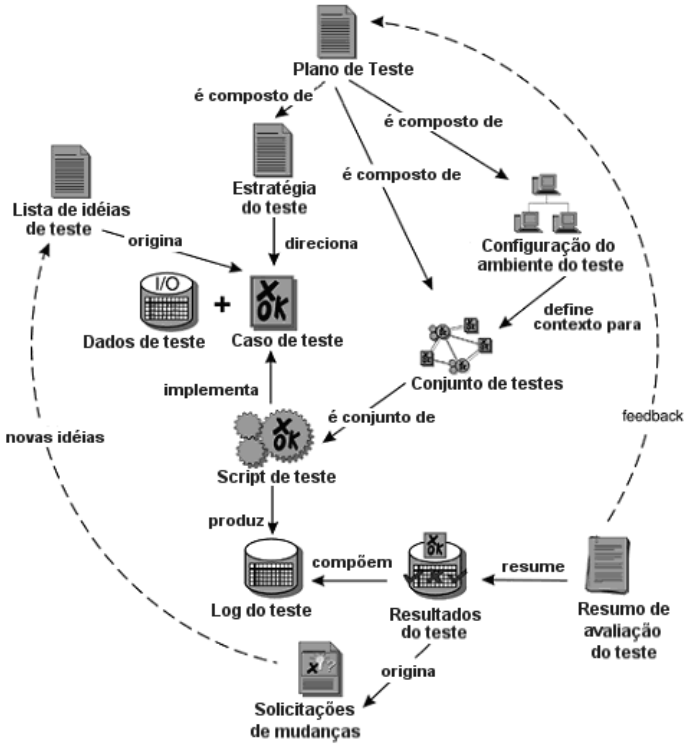

Figura 4.4: Artefatos de teste e os relacionamentos entre si (Kruchten, 2003).

A Figura 4.4 representa a rastreabilidade sob ponto de vista específico do processo de teste: podem ser vistos diversos tipos de documentos de teste e o relacionamento entre eles. Neste modelo estão contemplados o "Plano de Teste" (Plano de Teste), a "Especificação de Projeto de Teste" (Estratégia de Teste), a "Especificação de Procedimento de Teste" (Script de Teste), a "Especificação de Casos de Teste" (Casos de Teste), e o "Relatório-Resumo de Teste" (Resumo de Avaliação do Teste). Considerando a norma IEEE Std 829-1998: a) faltam os documentos "Diário de Teste" e "Relatório de Incidente de Teste"; b) os seguintes tipos de artefatos não fazem parte da norma: "Lista de Idéias de Teste" e "Solicitação de Mudanças".

4.4.6 Davis

Neste modelo se destacam quatro tipos de *links*, que possibilitam estabelecer vínculos bidirecionais entre os artefatos (Jarke, 1998 apud Davis, 1990):

- *Backward from requirements*: vincula um requisito às fontes de informação que o originaram;
- *Forward from requirements*: vincula os requisitos aos *designs* e outros tipos de artefatos que os implementam;

- *Backward to requirements*: vincula um artefato que implementa um requisito até o requisito.
- *Forward to requirements*: vincula as fontes de informações aos requisitos delas originados.

4.4.7 Leffingwell e Widrig

Este modelo destaca os seguintes conceitos: relacionamento de rastreabilidade, *link* de rastreabilidade e semântica de um *link* (Leffingwell e Widrig, 2000).

Um relacionamento de rastreabilidade representa algum tipo de relação entre dois artefatos. Essa relação é representada por um *link* de rastreabilidade, que pode ser do tipo "*traced-to*" ou "*traced-from*".

Um *link* do tipo *traced-to* indica, geralmente, quais artefatos criados em fases posteriores implementam, detalham ou dependem de alguma maneira do artefato que originou o *link*. Um *link* do tipo *traced-from* liga um artefato a outro que, de alguma maneira, origina ou justifica a existência desse artefato; ou seja, liga o artefato a suas fontes (documentos, indivíduos ou grupos). Implementar esses dois tipos de *links* possibilita a definição de vínculos bidirecionais entre os artefatos.

A Figura 4.5 representa uma situação entre dois artefatos, onde Artefato1 é implementado ou detalhado por Artefato2 e uma fonte, ou origem, de Artefato2 é Artefato1. Exemplo: considerando a existência de um conjunto de casos de teste para validar um requisito A especificado, pode-se dizer que esses casos de teste são "*traced-from*" do requisito A. E o requisito A é "*traced-to*" aos casos de teste.

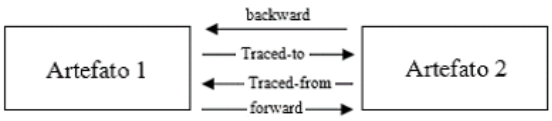

Figura 4.5: Precedência entre artefatos.

Semântica

Para melhorar a compreensão sobre o motivo da sua utilização de um determinado *link* de rastreabilidade, pode-se utilizar um ou mais atributos para descrevê-lo (Leffingwell e Widrig, 2000; Ramesh e Jarke, 2001; Cleland-Huang et al., 2003). Na Figura 4.6 é apresentada uma situação na qual um artefato do tipo "característica" é detalhado por artefatos do tipo "requisito

funcional", que por sua vez estão associados a artefatos do tipo "caso de teste" e "*design*". O significado dos *links*, neste caso, é detalhado por um único atributo: o rótulo do *link*. A associação entre requisitos e casos de teste é identificada como "testado por", o vínculo entre requisitos é caracterizada como "dependente de" e a associação entre requisitos e artefatos do *design* é identificada pelo rótulo "implementado por". Também se pode dizer, por exemplo, que um requisito possui uma relação "*traced-to*" até o conjunto de casos de teste.

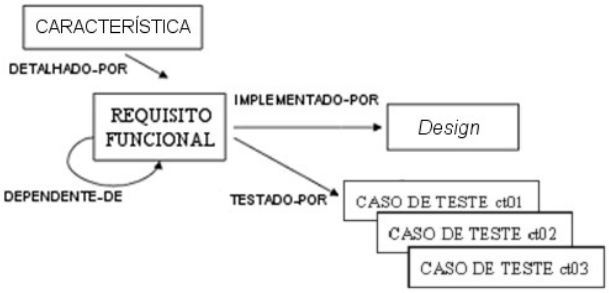

Figura 4.6: Semântica dos *links* de rastreabilidade.

A visibilidade das informações de rastreabilidade pode ser na forma de "árvore" ou na forma de matriz. O *status* das associações entre os artefatos deve ser indicado de alguma maneira; por exemplo: em um *link* onde um dos objetos envolvidos em um relacionamento de rastreabilidade foi modificado, mas ainda não foram realizadas todas as revisões e modificações necessárias, deve ser indicada uma mensagem de alerta como "SUSPEITO", "DESATUALIZADO" ou outra mensagem similar; caso a associação esteja atualizada, a mensagem pode ser "OK", e caso não existam associações, esta situação também deve ser indicada.

4.4.8 Wiegers

Os conceitos apresentados neste modelo são: elemento do sistema, *link* de rastreabilidade e atores (Wiegers, 2003):

- Elementos do sistema incluem requisitos de vários tipos, regras de negócio, *designs*, códigos-fonte, casos de teste, etc.;
- *Links* de rastreabilidade definem relacionamentos (interconexões e dependências) entre os diversos elementos do sistema; os *links* são derivados do modelo de Davis e a cardinalidade entre eles pode ser de um-para-um (p.ex. um requisito é validado por um caso de teste), um-para-vários (p.ex. um requisito é validado por mais de um caso

de teste) e vários-para-vários (p.ex. um requisito é validado por diversos casos de teste e um caso de teste pode ser usado na validação de mais de um requisito). Geralmente os relacionamentos entre elementos do sistema são organizados e visualizados na forma de uma matriz de rastreabilidade. O estado dos *links* também deve ser apresentado, conforme visto no modelo de Leffingwell e Widrig;

- Atores são agentes do processo sobre os quais devem ser mantidas informações que permitam rastrear suas contribuições.

A Figura 4.7 representa um exemplo de rastreabilidade entre diversos elementos do sistema, com foco nos artefatos da fase de Requisitos do Sistema. É possível observar diversos atributos que detalham o significado dos *links* existentes, como na associação entre "Requisito Funcional" e "Teste de Sistema" onde existe o rótulo "é validado por".

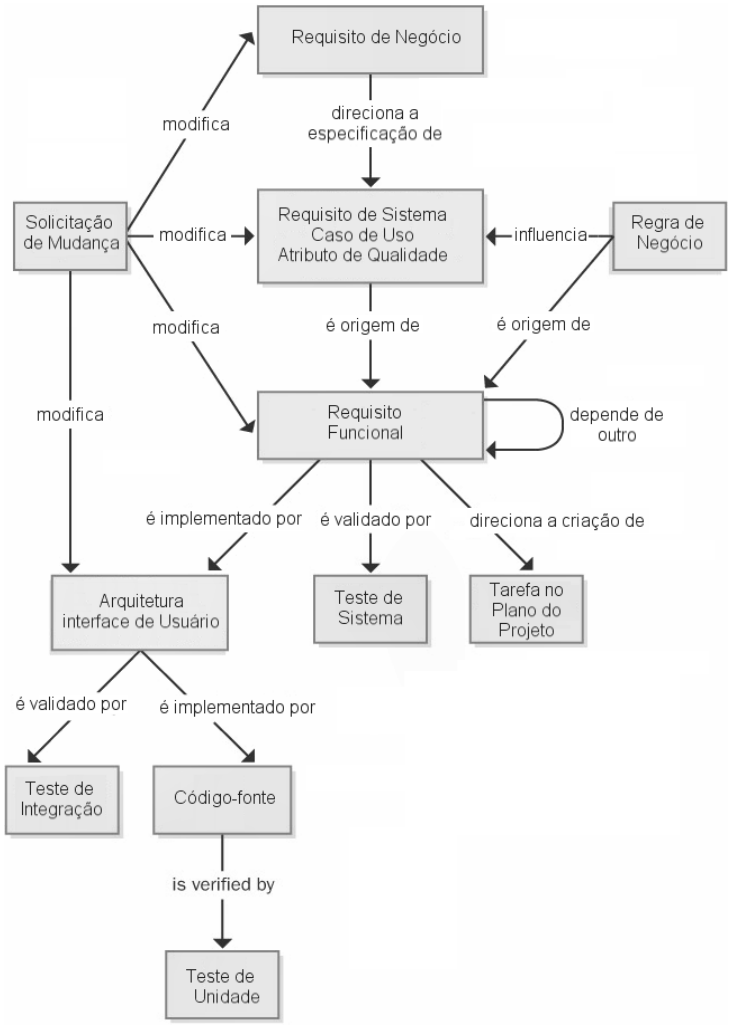

Figura 4.7: Exemplo de rastreabilidade em um projeto (Wiegers, 2003).

4.4.9 Gills

O modelo de Gills aborda exclusivamente o processo de teste. O autor identifica cinco artefatos de teste relevantes: *problem report*, *test*, *test case*, *test suite* e *test log*, vinculando-os aos requisitos e *designs* (Gills, 2005). Além disso, apresenta possíveis relacionamentos destes artefatos com artefatos típicos de um processo de desenvolvimento, conforme Figura 4.8.

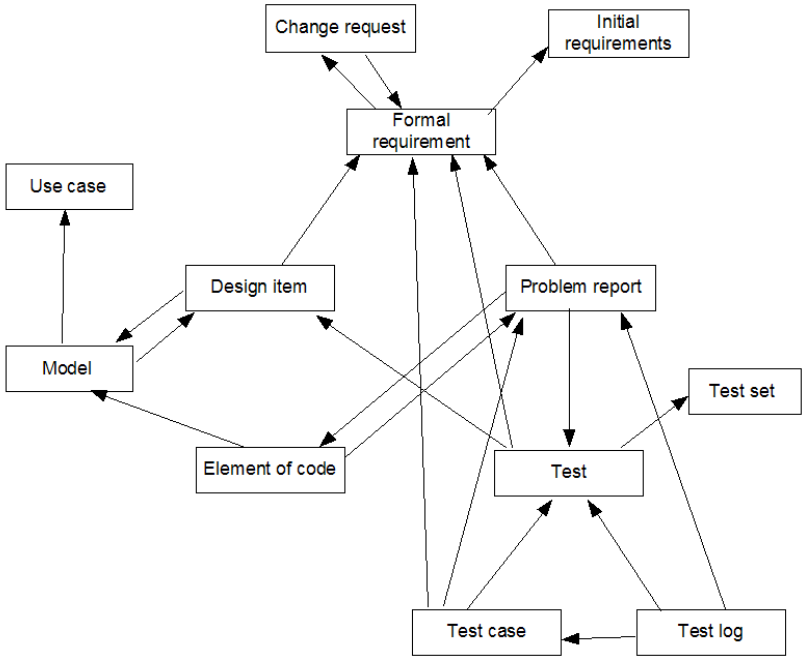

Figura 4.8: Itens e relações de rastreabilidade típicos de um processo de teste (Gills, 2005).

Alguns artefatos deste modelo podem ser mapeados diretamente com artefatos propostos na norma IEEE Std 829-1998: "*test log*" com "diário de teste", "*problem report*" com "relatório de incidente", "*test case*" com "especificação de caso de teste" e "*test suite*" com "Especificação de Procedimento de Teste". Já os documentos "Plano de Teste" e "Especificação de Projeto de Teste" não são previstos, além de ser visível a ausência de vínculos bidirecionais entre os artefatos do teste, o que prejudica a recuperação de informações de rastreabilidade.

4.5 Armazenamento e visualização dos *links* de rastreabilidade

Esta seção apresenta algumas considerações sobre a forma de armazenar *links* de rastreabilidade e, também, sobre a forma de visualizar os vínculos de rastreabilidade existentes entre os artefatos.

Um *link* de rastreabilidade é a definição explícita de um relacionamento (interconexão ou dependência) entre dois itens de rastreabilidade (Cleland-Huang et al., 2003; Wiegers, 2003).

4.5.1 Armazenamento

Para armazenar as associações de rastreabilidade podem ser usados grafos (Pfleeger e Bohner, 1990), bancos de dados e *links* de hipertexto (Sametinger e Riebisch, 2002; Maletic et al., 2003).

4.5.2 Visualização

Para visualizar as associações de rastreabilidade podem ser usadas matrizes (Phillips, 1998; Craig e Jaskiel, 2002; Wiegers, 2003), *links* de hipertexto (Sametinger e Riebisch, 2002; Maletic et al., 2003), e de maneira hierárquica (formato de árvore) (Leffingwel e Widrig, 2000). Exemplos de artefatos e tipos de relacionamentos que podem ser definidos entre eles: a) necessidades do usuário com as características (*features*) do software; b) características com requisitos; c) requisitos com *designs*; d) requisitos com documentação do teste; e) elementos da implementação com documentação do teste; f) propostas de modificações; g) hipóteses e justificativas para decisões tomadas (Leffingwel e Widrig, 2000).

4.5.2.1 Matriz de rastreabilidade

As associações entre os artefatos são visualizadas na forma de linhas e colunas, podendo assumir diversos formatos, com diferentes arranjos entre os tipos de artefatos, dependendo do que deve ser visualizado (Phillips, 1998; Craig e Jaskiel, 2002; Wiegers 2003).

A Figura 4.9 representa um exemplo no qual a primeira coluna lista os requisitos do sistema, e as colunas restantes representam os casos de teste existentes. Cada "x" em uma célula representa uma associação entre um requisito e um caso de teste. Valores em branco em alguma coluna indicam que o requisito não será validado pelo caso de teste correspondente (Craig e Jaskiel, 2002).

REQUISITO	CT1	CT2	CT3	CT4	CT5	CT6	CTn
RQ-01	x	x				x	
RQ-02		x	x				x
RQ-03					x	x	
RQ-04							

Figura 4.9: Matriz Requisitos x Casos de Teste.

4.5.2.2 Árvore de rastreabilidade

As associações de rastreabilidade entre os artefatos são mostradas de maneira hierárquica. A Figura 4.10 mostra um exemplo de árvore de rastreabilidade no protótipo PROMETEU, onde é possível visualizar as associações entre Plano de Teste, Especificação de Projeto de Teste, Especificação de Procedimento de Teste e Especificações de Casos de Teste.

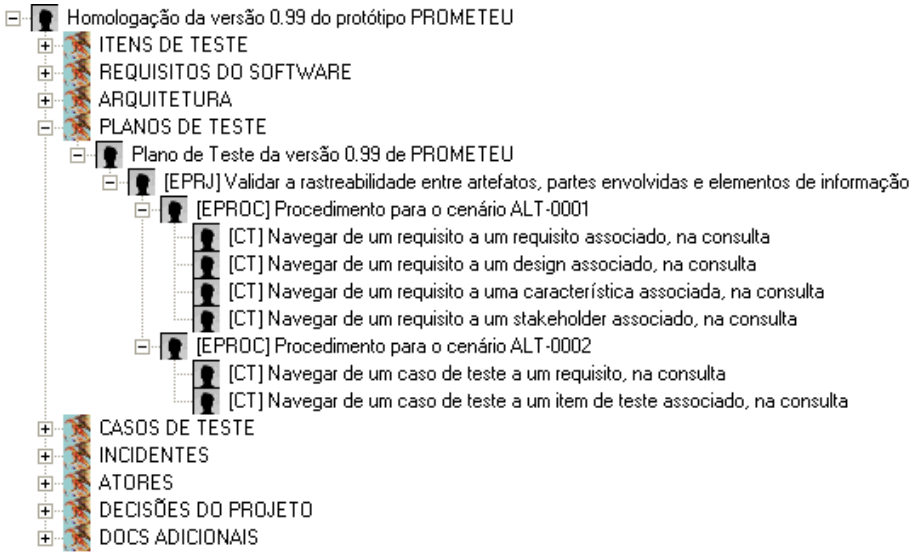

Figura 4.10: Exemplo de árvore de rastreabilidade na PROMETEU.

4.5.2.3 *Links* de hipertexto

Neste caso, os vínculos entre os artefatos podem ser representados em páginas HTML por meio dos *links* de hipertexto que a linguagem permite criar.

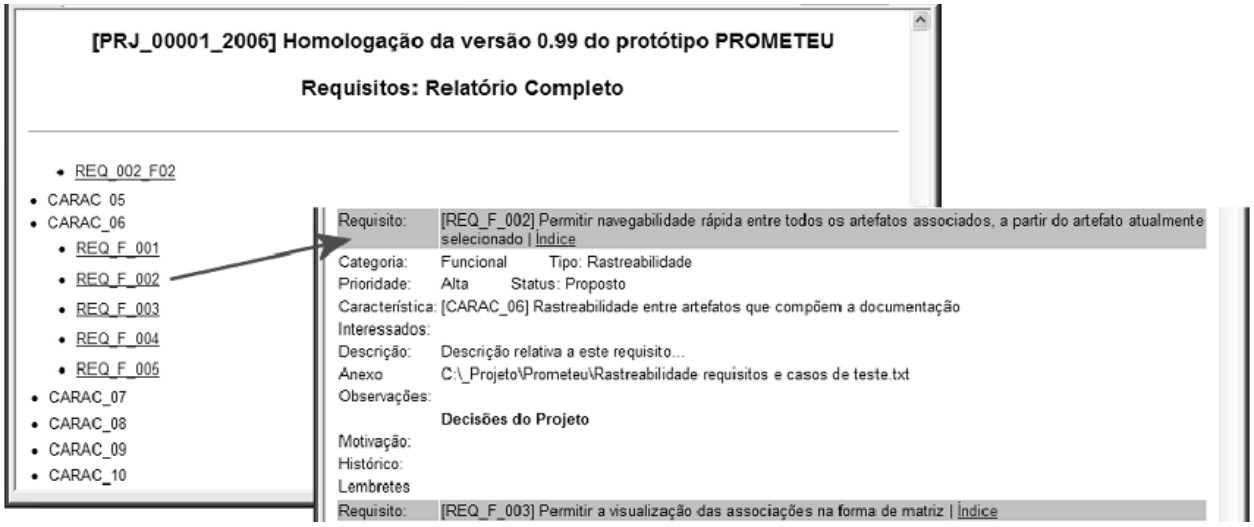

Figura 4.11: Matriz Requisitos x Casos de Teste.

A Figura 4.11 mostra um exemplo do uso de *links* de hipertexto para representar os vínculos em relação aos requisitos do software. Podem ser criados tantos *links* quanto forem necessários e podem ser usadas diversas páginas HTML, conforme a necessidade de visualização das informações de rastreabilidade. Em relação à figura, no trecho de página apresentado é possível visualizar os requisitos vinculados à Característica 'CARAC_06'.

Utilizar, em uma ferramenta, uma ou mais de uma das formas de visualização de informações de rastreabilidade apresentadas é uma decisão de projeto durante a construção da ferramenta.

4.6 Rastreabilidade e Teste de Software

Alguns trabalhos relacionados à rastreabilidade são destacados a seguir.

4.6.1 Trabalhos relacionados

Na maioria dos modelos de rastreabilidade pesquisados (Seção 4.4) a rastreabilidade é modelada com foco na análise e gerência dos requisitos; o teste, quando citado, é em alto nível de abstração e com foco nos casos de teste e no registro dos incidentes. Entretanto, estes dois tipos de artefatos não são os únicos existentes em um processo de teste. Dois modelos focam no processo de teste (Gills, 2005; Kruchten, 2003), mas ainda assim, sem detalhar quais elementos de informação devem ser registrados para cada tipo de artefato e sem detalhar as relações dos artefatos do teste com outros tipos de artefatos de um processo de software.

Neste trabalho, a rastreabilidade será tratada com foco no processo de teste de software, onde serão destacados aspectos como: a) possíveis benefícios da rastreabilidade para o teste; b) adaptação de modelos de rastreabilidade para o contexto do teste; c) suporte automatizado à rastreabilidade no contexto citado.

4.6.2 Aplicações da rastreabilidade em um processo de teste

O teste de um software é um processo caro que, em média, custa entre 40% e 50% do orçamento de um projeto; além disso, na maioria das vezes, possui prazo subestimado ou deliberadamente reduzido (Beizer, 1990; Pressman, 2002; Pfleeger, 2004). Considerando este contexto restritivo, a rastreabilidade é um mecanismo que pode contribuir de diversas maneiras para melhorar a qualidade do teste.

Teste baseado nos requisitos

O teste deve ser baseado na versão correta e atualizada dos requisitos especificados (Beizer, 1990; Craig e Jaskiel, 2002). *Links* de rastreabilidade entre os requisitos e documentos de teste possibilitam, entre outras coisas: a) garantir que todos os requisitos são considerados no planejamento do teste e que serão identificados aqueles que serão testados e quais não serão testados; b) garantir que cada requisito tenha um conjunto de casos de teste atribuído; c) visualizar a quantidade de casos de teste alocada a um requisito e analisar se existem casos de teste em excesso ou em quantidade insuficiente; d) evidenciar se todos os casos de teste documentados possuem requisitos associados. Caso existam casos de teste que não estejam associados a nenhum requisito, é provável que tenham sido projetados testes desnecessários.

Teste preventivo

O custo da remoção de um defeito aumenta exponencialmente à medida que ele é propagado das fases iniciais até as fases finais de um processo de software (Pressman, 2002; Sommerville, 2003). Projetar e associar casos de teste a um requisito tão logo este esteja especificado permite que o requisito seja revisado o mais cedo possível no processo: isso contribui para reduzir a ocorrência de requisitos vagos e mal especificados que geralmente provocam defeitos em alguma fase do processo. O objetivo do teste preventivo é evitar a inserção e propagação de defeitos ao longo do processo, ao invés de meramente erradicar esses defeitos após sua manifestação, com a intenção de economizar tempo e recursos (Beizer, 1990; Craig e Jaskiel, 2002; Wiegers, 2003).

Análise do impacto das modificações nos artefatos do teste

Através dos *links* de rastreabilidade é possível identificar e visualizar rapidamente quais artefatos e atores estão relacionados a uma solicitação de modificação, sendo possível estimar com maior exatidão o custo envolvido (Pfleeger, 2004). Uma situação deste tipo ocorre, por exemplo, quando uma especificação de requisito deve ser alterada, o que pode acarretar, entre outros impactos, a necessidade de: a) modificações em um grande número de casos de teste (manuais ou automatizados); b) alterações na configuração do ambiente para execução do teste; c) atualizações nos documentos de planejamento do teste. Como estes impactos podem prolongar o processo em poucos dias ou ainda mais, eles devem ser identificados, comunicados, analisados e quantificados antes de uma modificação ser aprovada.

Manutenção dos artefatos do processo de teste

O teste de um software é um processo dinâmico e sua documentação é composta por uma grande quantidade de artefatos. As várias modificações que ocorrem ao longo da execução das atividades do processo podem gerar artefatos com elementos de informação desatualizados e inconsistentes, o que aumenta o risco de decisões incorretas serem tomadas em função de dados que não correspondem à realidade atual do processo.

A existência de *links* de rastreabilidade entre os diversos artefatos do teste auxilia a manter a atualidade e consistência dos elementos de informação ao longo da execução do processo. Por exemplo: quando for necessário alterar o conteúdo de um artefato relacionado ao teste (requisitos, *designs*, planos, procedimentos de teste, casos de teste, registros de incidentes, etc.), os *links* de rastreabilidade indicam rapidamente quais artefatos devem ser revisados e, eventualmente, alterados para manter a boa qualidade da documentação.

Correção de defeitos do software

Links de rastreabilidade auxiliam na localização rápida do conjunto de atores, artefatos e atributos que caracterizam cada artefato relacionado a uma falha (manifestação de um defeito) que tenha sido registrada. Essas informações melhoram a compreensão dos motivos da ocorrência da falha (p.ex.: não-conformidade com as especificações, erros de projeto, violação de padrões, falha de comunicação com o cliente, etc.) (Dustin, 2003). Além disso, visualizar o conjunto de artefatos associados a uma falha pode facilitar a localização e correção do artefato onde reside o defeito (por exemplo: é possível concluir que o defeito não se encontra no código, mas sim em algum artefato do *design* ou dos requisitos). Após a execução da correção, através dos *links* de rastreabilidade é possível identificar e re-testar somente os artefatos relacionados ao defeito em questão.

Implementação de melhorias no software

No caso de melhorias, *links* de rastreabilidade facilitam identificar os impactos que modificações ou a inserção de novos requisitos trarão aos artefatos já existentes. Questões que podem ser respondidas são: artefatos existentes poderão ser reutilizados? Quantos, e quais, deverão ser modificados? Novos casos de teste deverão ser projetados?

Teste de regressão

Após uma modificação no software, fruto de uma correção ou implementação de uma melhoria, deve ser assegurado que o conjunto de funcionalidades do sistema continue funcionando corretamente. *Links* de rastreabilidade possibilitam identificar todos os atores e artefatos de alguma forma afetados por uma modificação, além de facilitar a seleção de todos os casos de teste relacionados que devam ser re-executados.

Como a correção de um defeito pode introduzir novos defeitos no software, as correções devem ser testadas para garantir que a correção foi feita e que novos defeitos não tenham sido introduzidos no software (Pfleeger, 2004). Já as melhorias devem ser testadas para garantir que não possuem defeitos e que, também, novos defeitos não tenham sido introduzidos.

Gerenciamento do processo de teste

Links de rastreabilidade contribuem para manter a integridade (atualidade, consistência e precisão) dos elementos de informação, o que auxilia a gerência na tomada de decisões relativas ao planejamento, *design* e acompanhamento do teste. Exemplos são: a) redução no risco de que sejam desperdiçados tempo e recursos no teste de funcionalidades em versões incorretas ou que não deveriam ser testadas; b) melhoria nas estimativas de prazo e custo relacionados a modificações através da identificação mais precisa de todos os artefatos e atores relacionados a uma modificação; c) facilidade na geração de relatórios sobre o *status* do teste como: (i) quais requisitos foram validados por quais casos de teste; (ii) dos casos de teste projetados, quantos já foram executados e quantos ainda faltam; (iii) atores envolvidos nos testes.

Auditagem por equipes independentes

Links de rastreabilidade auxiliam uma equipe externa ao projeto a comprovar rapidamente como o teste foi realizado, por quem, quais resultados foram obtidos e quais artefatos foram validados (Wiegers, 2003).

4.7 Suporte Automatizado à rastreabilidade

O suporte automatizado à rastreabilidade, na sua forma mais simples, inclui a manutenção dos *links* de rastreabilidade com o apoio de processadores de texto e planilhas eletrônicas; entretanto, como implantar e manter a rastreabilidade em um projeto demanda a manipulação de

um grande volume de dados, até mesmo para pequenos projetos é aconselhável a utilização de ferramentas especializadas (Leffingwell e Widrig, 2000; Kruchten, 2003; Wiegers, 2003).

Ferramentas de suporte à rastreabilidade possibilitam a criação de *links* de maneira dinâmica ou não-dinâmica (Cleland-Huang et al., 2003; Antoniol et al., 2002). Métodos dinâmicos visam capturar automaticamente associações de rastreabilidade entre diversos tipos de artefatos, com o mínimo de intervenção do usuário. Métodos não dinâmicos suportam rastreabilidade por meio da definição direta de *links* por parte do usuário ou por meio de *links* pré-definidos nas ferramentas; ferramentas nesta categoria são: ferramentas gerenciadoras de requisitos como IBM Rational Requisite-PRO (IBM, 2001) e ferramentas freeware como Testlink.

Também existem ferramentas acadêmicas com suporte à rastreabilidade; essas ferramentas seguem a mesma tendência dos modelos de rastreabilidade mostrados, que é a de focar nos artefatos relevantes para a gerência dos requisitos de um software (Pohl, 1996; Pinheiro e Goguen, 1996). PROMETEU, que é apresentada no Capítulo 6, é uma ferramenta acadêmica que oferece *links* pré-definidos de rastreabilidade para o usuário; neste capítulo também é feita uma comparação entre as funcionalidades da PROMETEU e as funcionalidades de outras ferramentas quanto aos seus recursos de rastreabilidade.

4.8 Síntese do Capítulo

Este Capítulo apresenta o tema rastreabilidade nas seguintes seções: Definições, Motivações para Utilização, Diretrizes para o Planejamento e Implantação, Modelos, Armazenamento e Visualização dos *Links,* Rastreabilidade no contexto do Teste de Software e Suporte Automatizado à Rastreabilidade.

Rastreabilidade pode ser entendida como uma propriedade de um sistema que torna possível estabelecer relações ou referências cruzadas entre dois ou mais artefatos de um processo de software, possibilitando recuperar informações como: a) quem criou um artefato e quando, b) localização e histórico de utilização de um artefato, c) quais os vínculos de um artefato com outros, d) qual a origem de um artefato, e) registro da contribuição dos atores para as decisões tomadas no projeto.

Motivações para implantação e utilização da rastreabilidade em um processo incluem maior exatidão na análise do impacto de modificações nos artefatos, obtenção de certificação, melhoria

no planejamento e controle da execução de um processo, além de redução no tempo e custos associados à manutenção da consistência e atualidade dos elementos de informação dos documentos. Considerando o teste de software, contribuições adicionais incluem suporte ao teste baseado nos requisitos e ao teste preventivo, aumento da eficiência e eficácia tanto no teste de regressão como na correção dos defeitos descobertos.

Implantar rastreabilidade não é trivial, pois envolve decidir o que deve ser rastreado levando em consideração o contexto da organização e os tipos de usuários. Além disso, rastreabilidade envolve armazenar e gerenciar uma grande quantidade de diversos tipos de elementos de informação. Sendo assim, foram apresentados nove modelos que prescrevem orientações e diretrizes sobre o que deve ser rastreado e quais *links* de rastreabilidade devem ser estabelecidos. Convém ressaltar que apesar de ser possível estabelecer *links* eventualmente entre todos os artefatos de um processo, a estratégia de rastreabilidade deve ser planejada com cuidado, pois quanto maior o número de elementos de informação a rastrear, maiores são a complexidade e custo para a implementação e gerenciamento da rastreabilidade.

Estabelecer e manter a rastreabilidade entre a enorme variedade de elementos de informação que compõem a documentação de um processo de teste é algo que envolve a manipulação de uma quantidade muito grande de dados. Em função disso, para que a implantação da rastreabilidade dê o retorno esperado, além da modelagem do que deve ser rastreado, é imprescindível a existência de um suporte automatizado adequado, que facilite a definição, manutenção e utilização dos *links* de rastreabilidade entre os diversos elementos de informação que compõem a documentação.

O próximo Capítulo apresenta o modelo de rastreabilidade e o modelo de dados usados pelo protótipo PROMETEU. Esses modelos são baseados na unificação dos conceitos dos modelos apresentados neste capítulo juntamente com os artefatos e relacionamentos propostos na norma IEEE Std 829-1998 e Modelo V (Figura 2.2). No Capítulo 6 são apresentadas questões relacionadas à implementaçãodo protótipo.

Capítulo 5

Modelos adotados pela PROMETEU

Este capítulo apresenta o modelo de rastreabilidade e o modelo de dados adotados pela PROMETEU, e está dividido nas seguintes seções: a Seção 5.1 apresenta o modelo de rastreabilidade usado no protótipo, baseado nos modelos apresentados no Capítulo 4; a Seção 5.2 apresenta o modelo de dados; e a Seção 5.3 apresenta a síntese do capítulo.

5.1 Modelo de rastreabilidade

Projetar uma ferramenta que possui entre suas características o recurso de rastreabilidade implica definir o modelo de rastreabilidade a ser utilizado. O modelo da PROMETEU é baseado nos modelos apresentados no Capítulo 4, na Norma IEEE Std 829-1998 e no Modelo V (Figura 2.2).

Em relação aos modelos apresentados no Capítulo 4, sete deles focam a fase dos requisitos e dois (RUP e Gills) abordam o teste com mais detalhes, indicando artefatos e associações entre eles. O estudo desses modelos permitiu extrair um conjunto comum de conceitos que foram utilizados na definição do modelo usado pela PROMETEU. Como o foco do modelo do protótipo é o processo de teste, as indicações prescritas nos modelos analisados foram complementadas com o uso da Norma IEEE Std 829-1998 e o Modelo V. A partir da norma é possível definir quais elementos de informação devem ser armazenados e rastreados com foco no processo de teste e quais relacionamentos de rastreabilidade devem ser estabelecidos e mantidos entre os artefatos do teste e artefatos de outras fases do processo de software. O Modelo V, por sua vez, reforça as relações existentes entre os diversos níveis de teste e as várias fases do processo de desenvolvimento, como a fase de definição dos requisitos e o nível de teste de sistema.

A Figura 5.1 ilustra o Modelo de Rastreabilidade da PROMETEU, com os principais itens de rastreabilidade e as relações entre eles, representadas por setas bidirecionais: dentro do retângulo tracejado estão os documentos e no exterior do retângulo estão os atores e as razões para as decisões tomadas em um projeto (esses dois itens de rastreabilidade estão na parte externa, pois estão associados a todos os artefatos que se encontram no interior do retângulo).

As setas que conectam dois tipos de artefatos indicam vínculo direto entre eles; entretanto, mesmo quando um artefato não possui uma associação direta com outro, é possível acessá-lo indiretamente a partir de qualquer um dos artefatos controlados pela ferramenta por meio dos mecanismos de consulta e navegabilidade entre os artefatos existentes no protótipo.

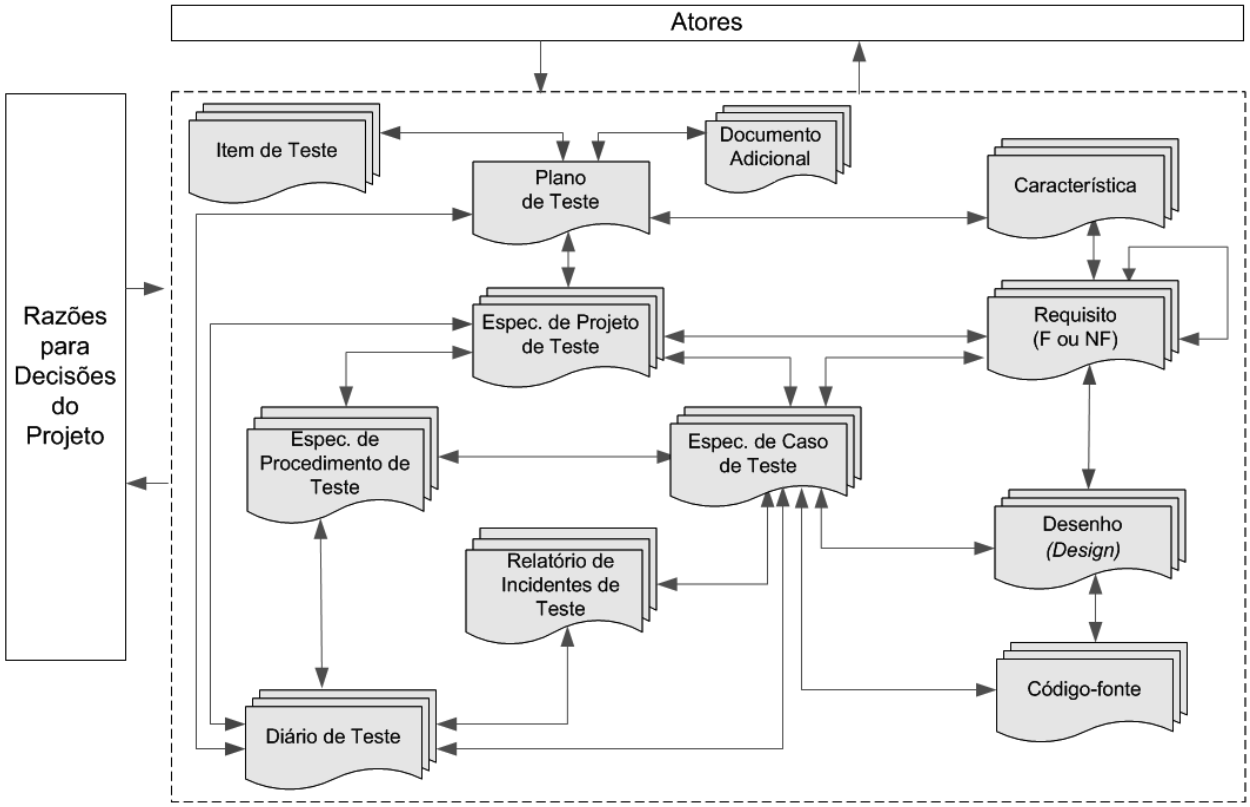

Figura 5.1: Modelo de rastreabilidade usado na PROMETEU.

O modelo de rastreabilidade implementado contempla os seguintes aspectos:

a) Tipos de usuários das informações de rastreabilidade;
b) Itens de rastreabilidade;
c) *Links* (associações) de rastreabilidade.

5.1.1 Tipos de usuários das informações de rastreabilidade

Os usuários da rastreabilidade podem variar entre os que necessitam somente de um nível básico de informações de rastreabilidade (usuários *low-end*) até os que necessitam de uma diversidade complexa de informações (usuários *high-end*) (modelo de Ramesh e Jarke). As exigências em relação ao suporte automatizado são diferentes para cada tipo: usuários *low-end* utilizam uma quantidade menor de informações de rastreabilidade e podem utilizar ferramentas

menos complexas, com *links* pré-definidos; usuários *high-end*, por utilizarem um número muito maior de informações de rastreabilidade, necessitam de ferramentas mais complexas, que contem com *links* pré-definidos além de oferecerem a possibilidade de definição tanto manual como automática de *links* de rastreabilidade.

PROMETEU possui foco em usuários *low-end* e disponibiliza vínculos pré-definidos entre os artefatos sob seu controle. O objetivo é interferir o mínimo possível na execução das atividades do teste, liberando os atores da responsabilidade de definir e manter os *links*, além de facilitar o aprendizado da ferramenta permitindo uma compreensão mais rápida da importância da rastreabilidade e de seus benefícios.

5.1.2 Itens de rastreabilidade

Itens de rastreabilidade representam elementos de informação que devem ser rastreados. Podem ser instâncias de um tipo de artefato (como um requisito do software ou um caso de teste), partes de um artefato (atributos ou elementos de informação que o compõem, como uma seção de um documento) ou, ainda, atores e suas contribuições (exemplos: registro de observações, ações sobre outros artefatos). Para que possa ser rastreado, um item de rastreabilidade deve possuir um identificador único no sistema (Craig e Jaskiel, 2002; Wiegers, 2003). Os itens de rastreabilidade definidos para a PROMETEU, baseados na norma IEEE Std 829-1998 e nos modelos apresentados no Capítulo 4, adotam como foco o processo de teste. Os itens de rastreabilidade, para cada projeto, são:

- Os atores;
- As razões para as decisões tomadas em um projeto;
- Os requisitos do software, que são organizados como Características, em um primeiro nível, e como Requisitos do tipo Funcional ou Não-Funcional, em um segundo nível;
- *Designs* do software, derivados a partir dos requisitos;
- Códigos-fonte;
- Documentos do teste de acordo com a Norma IEEE Std 829-1998;
- Documentação adicional; não detalhada na Norma IEEE Std 829-1998, mas que pode (na atual versão do protótipo) ser referenciada no Plano de Teste.

Os itens de rastreabilidade supracitados são descritos a seguir.

5.1.2.1 Atores

Quando existe a necessidade de rastrear informações sobre um ator, este se torna um item de rastreabilidade; os atores afetam a definição de artefatos, além de usar artefatos como apoio para execução das suas atividades; geralmente produzem e alteram artefatos ao executar suas atividades (modelo RUP e modelo de Wiegers). Na PROMETEU é possível: a) atribuir papéis a todos os participantes de um projeto de teste; b) para cada ator, registrar observações, notas e justificativas para decisões tomadas ao longo do processo (representam as 'estruturas de contribuição', conforme o modelo de Gotel e Finkelstein); e c) registrar os documentos com os quais o ator está relacionado.

5.1.2.2 Razões para decisões do projeto

Uma "razão para decisão do projeto" é um item de rastreabilidade que torna possível o rastreamento das justificativas para determinadas decisões tomadas em relação a um artefato ao longo da execução das atividades de um projeto (modelos de Gotel e Finkelstein, de Domges e Pohl e de Ramesh e Jarke).

5.1.2.3 Requisitos do software

Na PROMETEU, os requisitos do software são representados, no nível mais alto, como características; e cada característica é detalhada por um ou mais requisitos do tipo Funcional (RF) ou Não-Funcional (RNF) (Robertson e Robertson, 1999; Leffingwell e Widrig, 2000; Cockburn 2005). A relação dos requisitos com o teste está explícita Na norma IEEE Std 829-1998, no Modelo V e nos modelos RUP, de Wiegers e de Gills.

Os requisitos funcionais representam o comportamento que um sistema deve apresentar diante de certas ações de seus usuários, ou seja, "o quê" o sistema faz; os requisitos não-funcionais são aqueles que não dizem respeito diretamente às funções do sistema, caracterizando restrições sobre os serviços e funções oferecidas pelo produto e/ou sistema (p.ex.: restrições de tempo, de confiabilidade, de facilidade de uso, entre outros) (Sommerville, 2003; Pfleeger, 2004). A descrição de cada requisito deve possuir detalhamento e informações suficientes para possibilitar a criação dos casos de teste para validá-lo (Pfleeger, 2004).

A Figura 5.2 mostra os vínculos existentes entre o artefato "Característica" e outros artefatos da documentação na PROMETEU. A Figura 5.3 mostra os vínculos entre o artefato

"Requisito" e outros artefatos. Os *links* são representados como setas bidirecionais permitindo navegar nos dois sentidos entre artefatos associados.

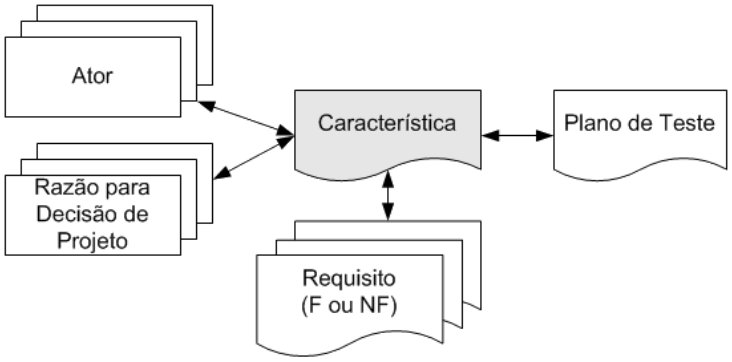

Figura 5.2: Vínculos entre uma "Característica" e outros artefatos.

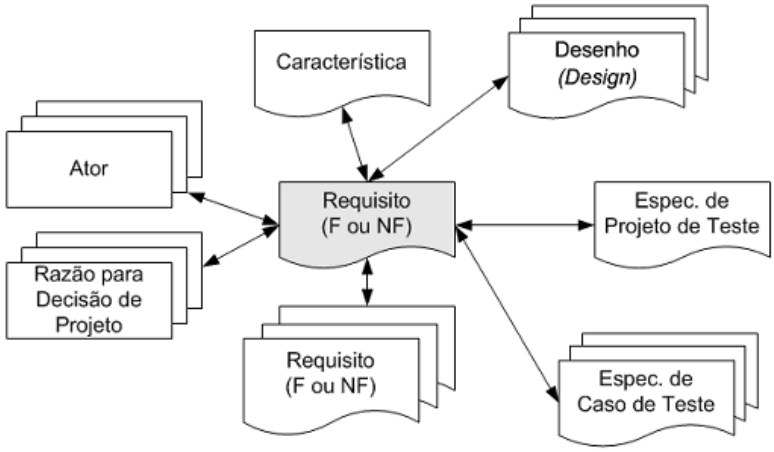

Figura 5.3: Vínculos entre um "Requisito" e outros artefatos.

5.1.2.4 *Design* (desenho) do software

Assim como os requisitos, os *designs* do software fornecem informações importantes para o planejamento do teste (Craig e Jaskiel, 2002). O Modelo V e os modelos do RUP, de Wiegers e de Gills também indicam o relacionamento entre os *designs* e os artefatos do teste. Os documentos da norma não sugerem a associação direta entre *designs*, códigos-fonte e casos de teste; entretanto, por meio da ferramenta é possível associar *designs* com casos de teste e *designs* com códigos-fonte. A Figura 5.4 mostra os vínculos existentes entre o artefato "Desenho" e outros artefatos da documentação na PROMETEU.

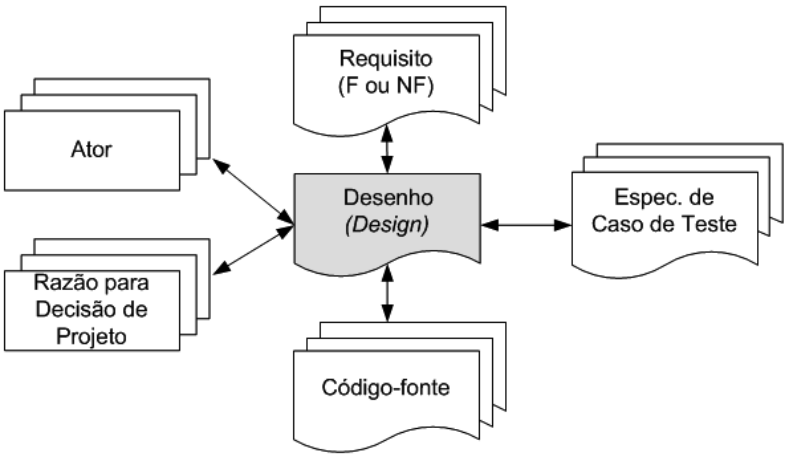

Figura 5.4: Vínculos entre um "*Design*" e outros artefatos.

5.1.2.5 Código-fonte do software

Os artefatos de código-fonte podem ser vinculados aos *designs* do software e também a um conjunto de casos de teste. Dessa forma é possível estabelecer um caminho que inicia nos atores que definiram os requisitos, passa pelos requisitos, *designs* relacionados aos requisitos, códigos-fonte associados aos *designs* e chega aos casos de teste projetados e executados para validá-los. O Modelo V e os modelos de Wiegers e de Gills também indicam o relacionamento entre os códigos-fonte e os artefatos do teste. A Figura 5.5 mostra os vínculos existentes entre o artefato "Código-fonte" e outros artefatos da documentação no protótipo.

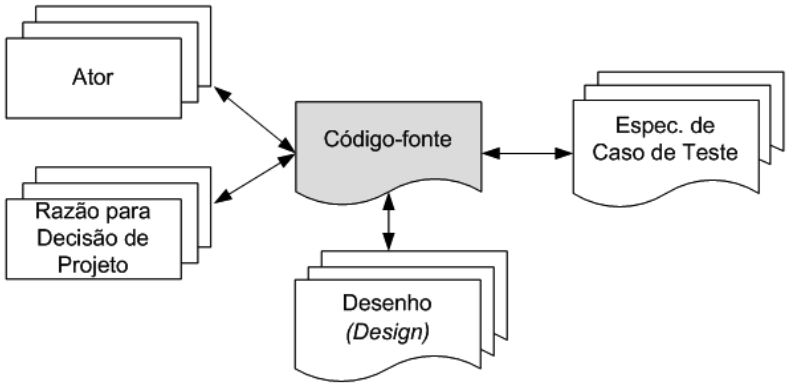

Figura 5.5: Vínculos de um "Código-fonte" com outros artefatos.

5.1.2.6 Documentação adicional

A documentação adicional é formada por documentos cujos elementos de informação não são controlados diretamente na ferramenta; entretanto, esses documentos podem ser referenciados

por documentos do teste - conforme indicado na Norma IEEE Std 829-1998 (na versão atual do protótipo, o Plano de Teste é o artefato que pode referenciar documentos adicionais). A Figura 5.6 mostra os vínculos existentes entre o artefato "Documento Adicional" e outros artefatos da documentação no protótipo.

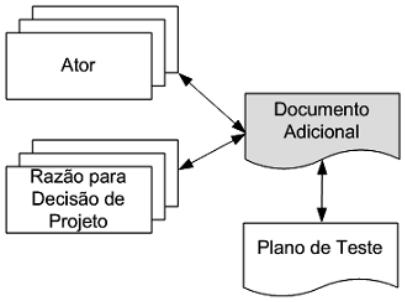

Figura 5.6: Vínculos de um "Documento Adicional" com outros artefatos.

5.1.2.7 Documentos do teste

Os documentos armazenados são: Plano de Teste, Especificação de Projeto de Teste, Especificação de Procedimento de Teste, Especificação de Caso de Teste, Relatório de Incidente de Teste, Diário de Teste e Relatório de Encaminhamento de Item de Teste. Quanto ao Relatório-Resumo do Teste, o protótipo oferece diversas opções de relatório que cumprem o papel deste tipo de documento. A Figura 5.7 ilustra os vínculos existentes entre um "Plano de Teste" e outros artefatos da documentação controlados no protótipo.

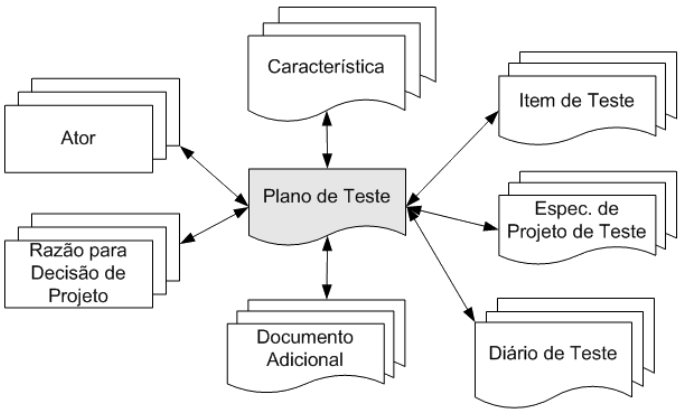

Figura 5.7: Vínculos de um "Plano de Teste" com outros artefatos.

Especificação de Projeto de Teste

A Figura 5.8 ilustra os vínculos entre uma "Especificação de Projeto de Teste" e outros artefatos controlados pela PROMETEU.

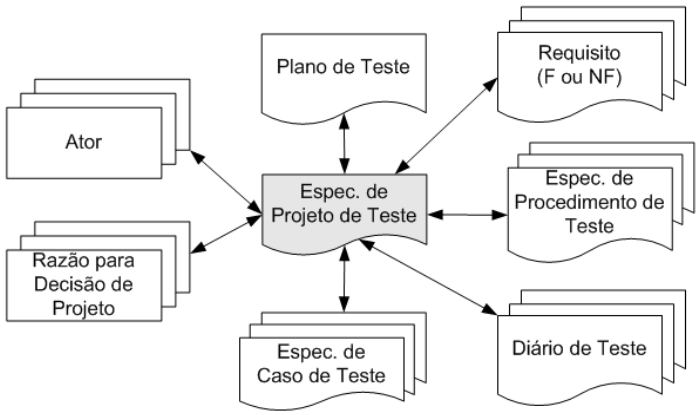

Figura 5.8: Vínculos de uma "Especificação de Projeto de Teste" com outros artefatos.

Especificação de Procedimento de Teste

A Figura 5.9 mostra os vínculos entre um artefato "Especificação de Procedimento de Teste" e outros artefatos.

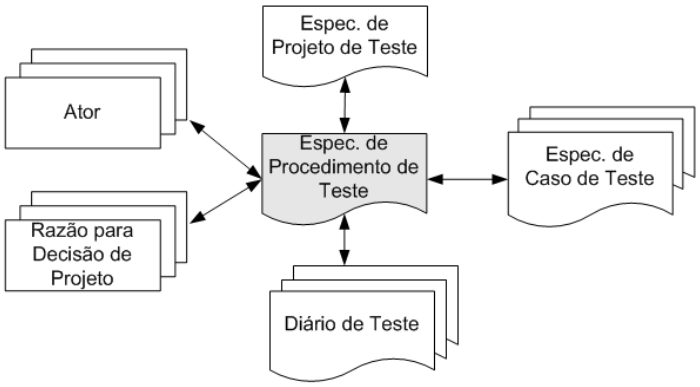

Figura 5.9: Vínculos de uma "Especificação de Procedimento de Teste" com outros artefatos.

Especificação de Caso de Teste

A Figura 5.10 ilustra os vínculos entre o artefato do tipo "Especificação de Caso de Teste" e outros artefatos.

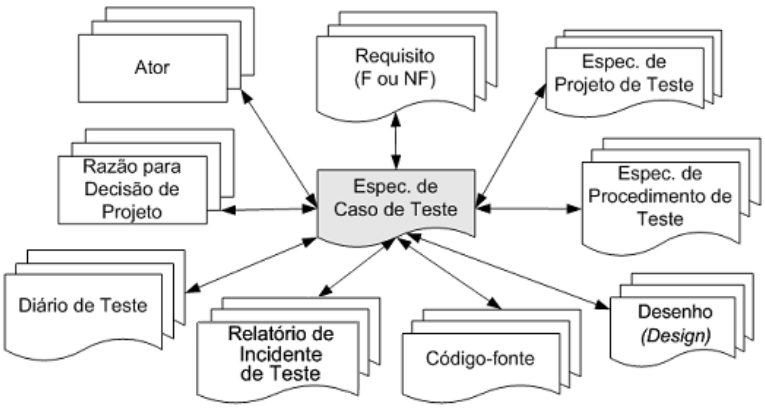

Figura 5.10: Vínculos entre uma "Especificação de Caso de Teste" e outros artefatos.

Diário de Teste

A Figura 5.11 ilustra os vínculos entre o artefato do tipo "Diário de Teste" e outros artefatos.

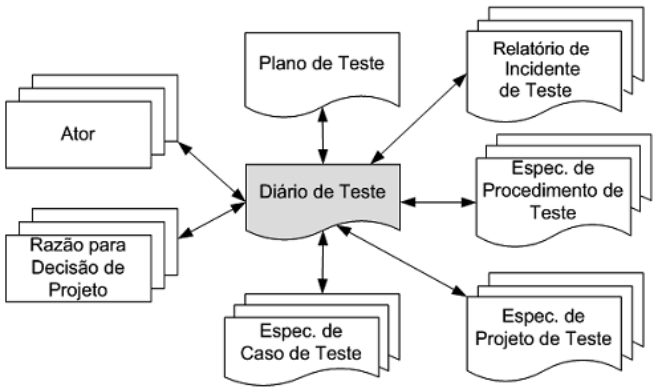

Figura 5.11: Vínculos entre um "Diário de Teste" e outros artefatos.

Relatório de Incidente de Teste

Um incidente de teste relaciona-se diretamente ao caso de teste que o gerou. Assim, a partir do caso de teste associado é possível consultar qualquer artefato relacionado ao incidente. A Figura 5.12 mostra os vínculos do artefato "Relatório de Incidente de Teste" com outros artefatos.

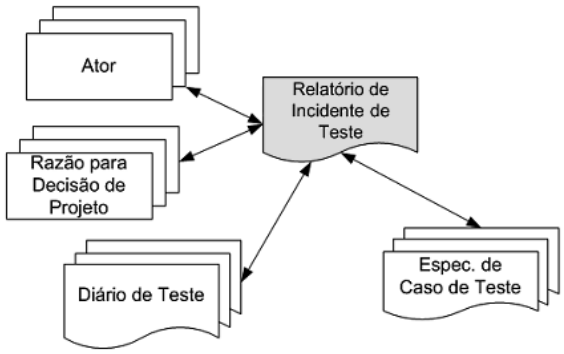

Figura 5.12: Vínculos entre um "Relatório de Incidente de Teste" e outros artefatos.

5.1.3 Associações (links) de rastreabilidade

Itens de rastreabilidade são vinculados uns aos outros por meio de *links* de rastreabilidade. Como o foco do protótipo é em usuários do tipo *low-end*, os *links* e as operações de consulta sobre eles são pré-determinados.

Armazenamento

No protótipo, os *links* são implementados como relacionamentos entre tuplas de tabelas em um sistema gerenciador de banco de dados relacional.

Tipo e precedência

PROMETEU possui *links* bidirecionais entre os diversos artefatos rastreados. O protótipo oferece suporte a rastreabilidade vertical e rastreabilidade horizontal entre artefatos, como proposto no modelo de Pfleeger e Bohner.

Na Figura 5.13 um exemplo de rastreabilidade horizontal são as relações entre um requisito (REQ03) e os casos de teste usados para validá-lo (CT003 e CT004); exemplos de rastreabilidade vertical são as relações que descrevem a interdependência entre os requisitos REQ03, REQ04 e REQ05 (REQ03 é detalhado por REQ04 e REQ05). Também são exemplos de rastreabilidade vertical as dependências entre os casos de teste da seção 'Casos de Teste' no documento "Especificação de Procedimento de Teste", onde o primeiro caso de teste a ser executado é CT003 e o último é CT006. Os exemplos supracitados podem ser implementados na PROMETEU.

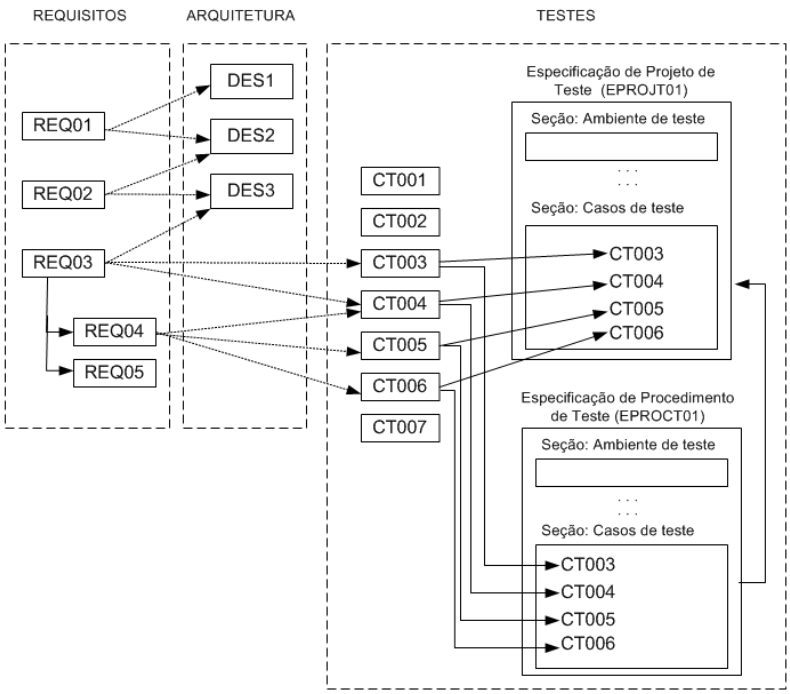

Figura 5.13: Rastreabilidade horizontal e vertical entre artefatos.

As rastreabilidades tanto vertical como horizontal também podem ser representadas por *links* bidirecionais do tipo *forward* e *backward*, também conhecidos como *traced-to* e *traced-from* (modelo de Gotel e Finkelstein, modelo de Davis e modelo de Leffingwell e Widrig). Por exemplo: analisando a Figura 5.13, CT004 é *traced-from* REQ03 e *traced-to* EPROCT01.

Semântica

O uso de atributos para caracterizar um *link* de rastreabilidade é citado no modelo de Ramesh e Jarke, no modelo do RUP, no modelo de Leffingwell e Widrig e no modelo de Wiegers. Utilizar atributos em *links* de rastreabilidade é uma decisão de projeto e depende dos recursos oferecidos pela ferramenta utilizada; na versão atual da PROMETEU não é possível caracterizar semanticamente um *link* de rastreabilidade.

5.1.4 Representação das associações

Na PROMETEU os *links* de rastreabilidade podem ser visualizados como matrizes e como árvores de rastreabilidade, conforme proposto nos modelos de Wiegers e de Leffingwell e Widrig.

5.1.5 Análise do modelo de rastreabilidade

Tendo em vista o contexto da origem do protótipo, o modelo atual foi projetado com foco no processo de teste e está centrado na rastreabilidade entre artefatos. Em relação ao teste de software, exclusivamente, o modelo atual pode ser melhorado das seguintes maneiras:

- Modelar as estruturas de contribuição do teste: estas estruturas se referem aos diversos tipos de contribuição por parte dos vários atores que influenciam na descoberta, especificação e posterior evolução dos artefatos do teste e à relação entre estes atores (modelo de Gotel e Finkelstein e modelo de Domges e Pohl);
- Modelar a dimensão do processo: representar as atividades e tarefas realizadas, recursos consumidos e métricas relacionadas às tarefas e recursos; um ponto de partida pode ser a versão 2008 da Norma IEEE Std 829 (modelo de Domges e Pohl e Norma IEEE Std 829-2008);
- Modelar a semântica das diversas relações existentes entre os artefatos.

Apesar do foco no processo de teste, o desenvolvimento do modelo deixou claro seu potencial de expansão de forma a abranger outras fases do desenvolvimento de software (p.ex., Requisitos, Arquitetura, Codificação, Liberação e Manutenção). Assim, outra possibilidade de melhoria do modelo atual é criar um modelo de rastreabilidade para cada fase e, então, vincular o modelo específico de cada fase com o modelo do teste, por diversos pontos de conexão (elementos de informação em comum), conforme indicado nos documentos da Norma IEEE Std 829. Nesse sentido, os modelos de rastreabilidade apresentados neste trabalho representam um bom ponto de partida; como exemplo, pode-se utilizar o modelo de Wiegers e o modelo do RUP para criar um modelo de rastreabilidade para a fase de Requisitos e vincular esse modelo ao modelo de rastreabilidade do Teste.

Todas as sugestões de expansão e melhoria devem ser analisadas considerando os seguintes dados mencionados anteriormente: para a maioria dos projetos pode-se obter até 80% dos benefícios da rastreabilidade rastreando até 20% dos *links* possíveis (Kotonya e Sommerville, 1998; Wiegers, 2003). Considere-se também que geralmente os participantes de um projeto não dispõem de tempo para coletar dados que não tenham um objetivo claro de utilização.

5.2 Modelo de dados

A Figura 5.14 ilustra o modelo de dados de PROMETEU usando somente um subconjunto das tabelas e das relações para facilitar a visualização das entidades principais. Os tipos de documentos "Características", "Requisitos", "*Designs*", "Código-fonte", "Especificação de Caso de Teste", "Relatório de Incidente", "Diário de Teste" possuem tabelas específicas, dadas as peculiaridades de cada um; no caso dos tipos de documento Plano de Teste, Especificação de Projeto de Teste e Especificação de Procedimento de Teste, foi possível agrupá-los na tabela DOCUMENTOS. Em relação aos documentos preconizados pela Norma IEEE Std 829-1998, é possível gerar documentos personalizados usando os elementos de informação registrados no banco de dados para cada um deles.

A seguir são apresentadas as principais entidades do modelo de dados da PROMETEU e algumas de suas relações com outras entidades do modelo.

Pessoas: mantém dados sobre os atores (tanto técnicos como *stakeholders*) de um projeto. Esses dados são: nome, papel, email, localização (onde encontrar o ator), além de dados adicionais (em arquivos externos à ferramenta) que podem ser associados ao perfil de cada ator.

Documentos: armazena dados sobre os seguintes artefatos: Plano de Teste, Especificação de Projeto de Teste e Especificação de Procedimento de Teste. Exemplos de associações são: a) um documento Plano de Teste é detalhado por uma ou mais Especificações de Projeto de Teste, possui uma ou mais características associadas e pode ter um ou mais documentos extras associados; b) um documento Especificação de Projeto de Teste possui uma ou mais Especificações de Procedimento de Teste associadas e um ou mais casos de teste associados; c) um documento Especificação de Procedimento de Teste detalha um documento Especificação de Projeto de Teste e possui um ou mais casos de teste associados.

Casos_de_Teste: mantém dados relativos aos casos de teste de um projeto; exemplos de associações com outros artefatos são: a) está associado a pelo menos uma Especificação de Projeto de Teste e a pelo menos uma Especificação de Procedimento de Teste (em ambos os casos, pela tabela "Doc_Secao_Info", atributo "id_secao_infotable"); b) está associado a um ou mais requisitos, e (c) pode estar associado a um incidente de teste.

Diários_de_Teste: um diário de teste pode referenciar diretamente os seguintes artefatos: caso de teste, incidente de teste, plano de teste, projeto de teste e procedimento de teste; além disso, existem os campos descrição e observações para descrever as atividades realizadas.

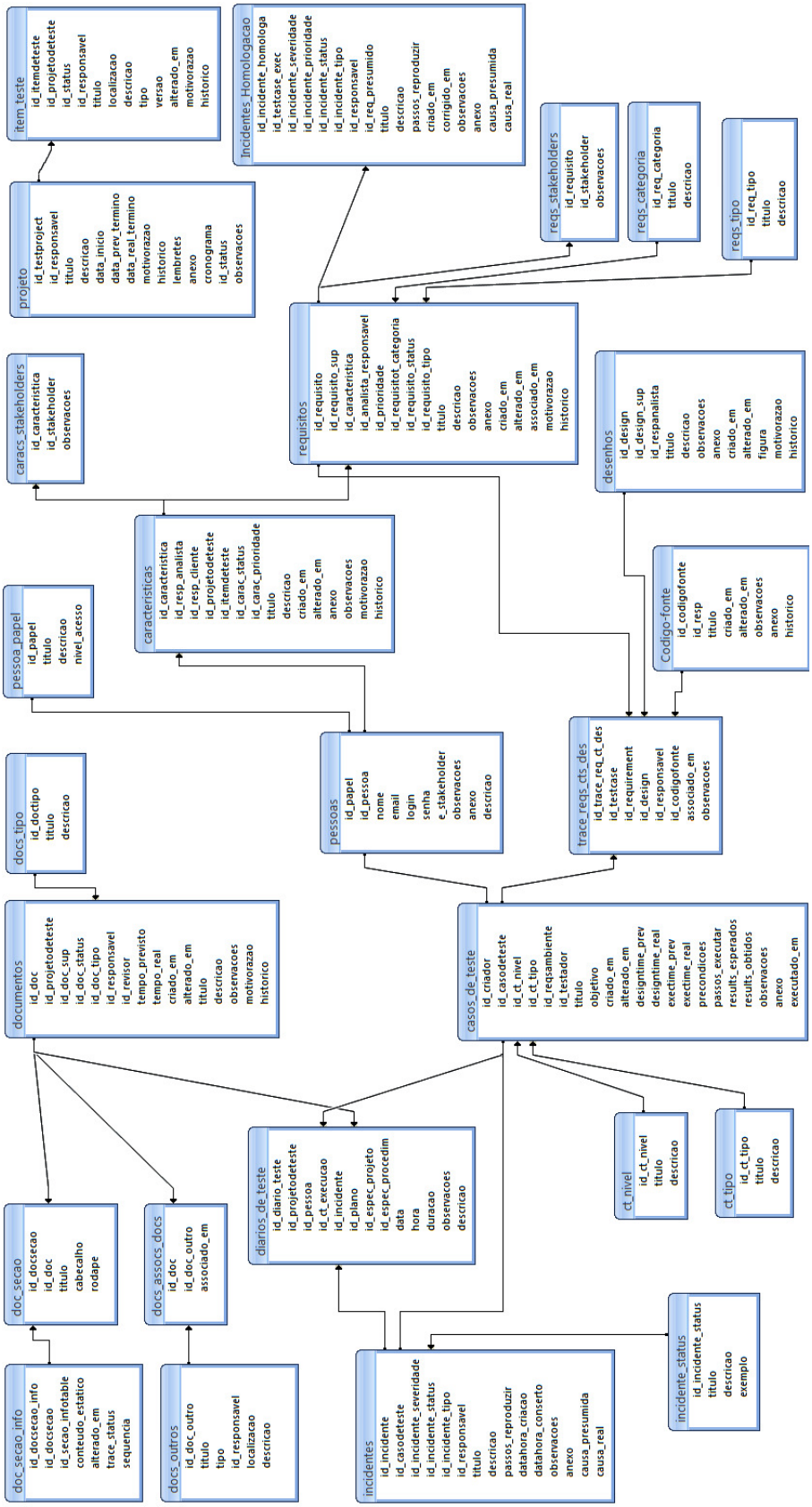

Figura 5.14: Modelo de dados (simplificado).

Incidentes: mantém dados relativos aos incidentes geralmente originados após a execução de um caso de teste que revelou a existência de um defeito no software sendo testado. Alguns desses dados são: título, descrição, tipo, *status* e passos para reprodução; um incidente está associado diretamente ao caso de teste que originou seu registro.

Código-fonte: armazena dados sobre os códigos-fonte do software sendo testado.

Desenhos: armazena dados sobre os artefatos da arquitetura do software; um desenho: (a) está associado a um ou mais requisitos, (b) pode ser validado por um ou mais casos de teste.

Características: esta entidade armazena dados relativos às características relacionadas a um projeto de teste; uma característica: (a) possui um ou mais requisitos subordinados, (b) pode estar associada a um ou mais documentos do tipo Plano de Teste (pela tabela "Doc_Secao_Info", atributo "id_secao_infotable").

Requisitos: armazena dados relativos aos requisitos associados a um projeto de teste; um requisito pode ser do tipo funcional ou não-funcional e: (a) está subordinado a uma característica, (b) pode estar subordinado a outro requisito, (c) pode ter um ou mais requisitos subordinados, (d) está associado a zero ou mais casos de teste, (e) possui associado a si zero ou mais *designs*, (f) está associado a um ou mais documentos do tipo Especificação de Projeto de Teste (pela tabela "Doc_Secao_Info", atributo "id_secao_infotable").

5.3.3.1 Análise do modelo de dados

Algumas melhorias que podem ser feitas no modelo são destacadas a seguir:

- Alterar o projeto para suportar o controle de versões para os artefatos da documentação. O controle de versões possibilita, por exemplo, acompanhar a evolução das modificações em cada artefato;
- Registrar automaticamente as ações dos usuários; para isso é necessário criar tabelas para esse registro, com benefícios que incluem: a) criar um *log* das ações de cada usuário, para acompanhamento do Gerente de Teste, b) oferecer ao usuário a opção de desfazer as modificações mais recentes nos documentos;
- Dadas as peculiaridades de cada tipo de documento e o conjunto específico de operações sobre os elementos de informação de cada um deles, para cada tipo de documento foi modelado um conjunto de elementos de informação; entretanto, considerando que a facilidade de adaptação ao contexto do usuário é uma

característica desejável, deve-se prever a possibilidade de criar campos personalizados para cada artefato. Dessa forma o usuário pode incluir campos não previstos originalmente, mas que são importantes para o seu contexto de trabalho;

- O modelo atual possibilita a geração/personalização de novos documentos a partir dos elementos de informação dos documentos previamente modelados. No entanto, as informações de personalização não são armazenadas no banco de dados, sendo perdidas após a geração dos documentos. Devem ser criadas tabelas para o registro das configurações dos documentos personalizados pelo usuário;
- Modelar artefatos de outras fases do processo de software, além dos artefatos atuais que foram modelados com foco no teste. Assim, será possível extrapolar o foco exclusivo no teste e expandir o uso da ferramenta também para outras fases do processo de software. Um exemplo são os artefatos dos requisitos, que podem sofrer melhorias como: a) modelagem de acordo com os modelos de Wiegers e do RUP, b) registro dos dados tanto como texto puramente seqüencial ou na forma de casos de uso (amplamente usado atualmente), c) modelagem do ciclo de vida de um requisito;
- Possibilitar o registro de dados semânticos associados aos *links* de rastreabilidade para complementar o significado de cada associação.

5.3 Síntese do Capítulo

Este capítulo apresenta o modelo de rastreabilidade e o modelo de dados usados pelo protótipo PROMETEU.

O modelo de rastreabilidade definido e adotado pelo protótipo está baseado na unificação dos conceitos dos modelos apresentados no Capítulo 4 juntamente com os artefatos e relacionamentos propostos na Norma IEEE Std 829-1998 e no Modelo V (Figura 2.2). O modelo de dados foi projetado para possibilitar o armazenamento e utilização das informações de rastreabilidade de acordo com o modelo de rastreabilidade definido.

Em relação aos modelos, é feita uma breve análise sobre cada um, destacando os aspectos atuais e possíveis pontos onde podem ser melhorados.

O próximo capítulo apresenta o protótipo PROMETEU.

Capítulo 6

Implementação da PROMETEU

Dada a complexidade de um processo de teste, sua documentação demanda registrar e manter uma grande quantidade de elementos de informação. Além disso, para auxiliar a manter a documentação atualizada e consistente, devem ser rastreadas as diversas associações entre os artefatos que a compõem; esse rastreamento também demanda o manuseio de uma grande quantidade de dados. Sendo assim, a existência de suporte automatizado adequado é fundamental.

PROMETEU é o protótipo de uma ferramenta projetada para dar suporte ao registro e manutenção dos documentos de um processo de teste de software; proporciona rastreabilidade entre os diversos artefatos que compõem essa documentação. O protótipo foi concebido e desenvolvido como parte de uma metodologia para a introdução ou melhoria do processo de teste em empresas desenvolvedoras de software; essa metodologia está baseada na Norma IEEE Std 829-1998 e foi criada pelo Grupo de Teste de Software do CTI (IEEE, 1998b; Crespo et al., 2002; Crespo et al., 2004). Um exemplo de utilização do protótipo é apresentado no Apêndice A.

O nome 'PROMETEU' significa PROjeto de MEstrado na área de TEste na Unicamp. Este nome foi inspirado em Prometeu, o titã da mitologia grega que trouxe o fogo e revelou o conhecimento para a humanidade; no contexto do software teste é o processo responsável por revelar a existência de defeitos, ou seja, melhorar o conhecimento sobre o software sob teste..

Este capítulo apresenta o protótipo desenvolvido e está dividido nas seguintes seções: a Seção 6.1 apresenta a arquitetura da ferramenta; a Seção 6.2 apresenta, em linhas gerais, as decisões adotadas para a implementação do protótipo; a Seção 6.3 apresenta os artefatos e exemplos da interface gráfica dos mesmos, além de exemplos de maneiras de visualizar informações de rastreabilidade no protótipo; a Seção 6.4 apresenta as características da ferramenta; a Seção 6.5 faz referência a um exemplo de aplicação do protótipo; a Seção 6.6

apresenta uma breve comparação entre a PROMETEU e outras ferramentas de apoio ao teste, com base nas características de cada uma; e a Seção 6.7 apresenta a síntese do capítulo.

6.1 Arquitetura

A Figura 6.1 ilustra a arquitetura da ferramenta. São apresentados os conjuntos de dados de entrada, os módulos da ferramenta e o conjunto de resultados que pode ser obtido.

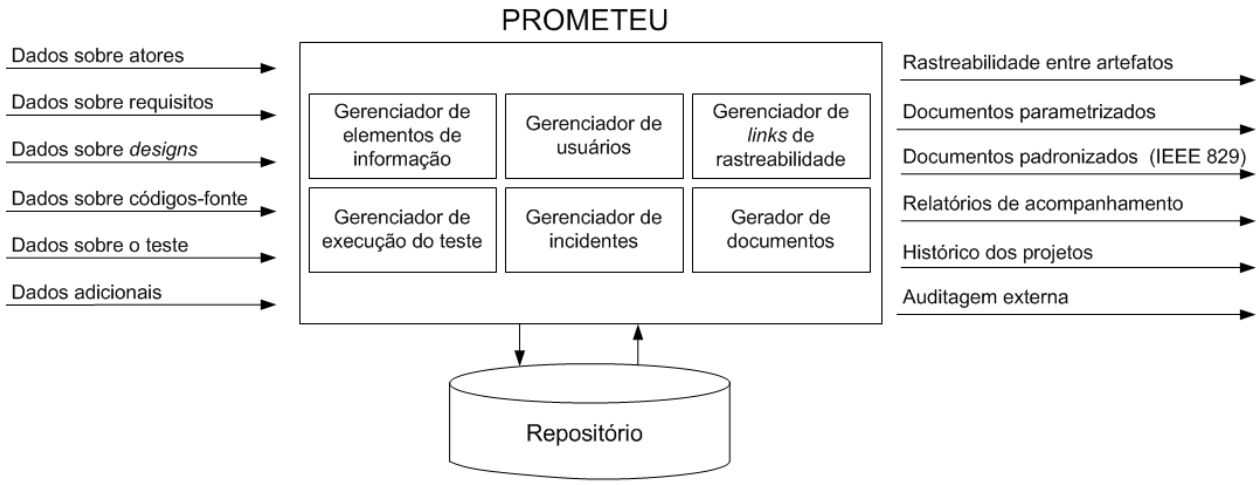

Figura 6.1: Arquitetura da ferramenta.

Usando o protótipo é possível registrar dados sobre: os atores do projeto (técnicos e *stakeholders*[10]); os requisitos do software que será testado; os *designs* da arquitetura; os documentos prescritos pela Norma IEEE Std 829-1998; e sobre documentos adicionais que porventura sejam referenciados na documentação do teste. A ferramenta está dividida em seis componentes principais:

- Gerenciador de elementos de informação: conjunto de funções para gravação e manutenção dos dados que compõem a documentação de um projeto de teste e, também, para controlar o acesso às funcionalidades da ferramenta de acordo com o papel de cada usuário;
- Gerenciador de usuários: conjunto de funções para cadastrar e gerenciar os usuários;
- Gerenciador de *links* de rastreabilidade: conjunto de funções para manutenção dos *links*, suporte à navegabilidade entre artefatos e relatório de *status* dos *links*;

[10] Ver a definição no Apêndice B – Glossário e Abreviaturas.

- Gerenciador da execução do teste: possibilita acompanhar o *status* da execução do teste por meio de um conjunto pré-definido de consultas sobre a base de dados;
- Gerenciador de incidentes: conjunto de funções que dão suporte ao registro, listagem e consulta de incidentes registrados;
- Gerador de documentos: funções que pesquisam a base de dados e geram documentação no formato padrão ou personalizado pelo usuário.

Uma vez que os artefatos que compõem a documentação de um projeto de teste estejam registrados no protótipo, por meio da ferramenta é possível manter a rastreabilidade entre eles, manter a atualidade e consistência dos dados, navegar facilmente entre os artefatos, além de gerar documentos no padrão da Norma IEEE Std 829-1998 ou de forma parametrizada (conforme a necessidade do usuário). Também é possível acompanhar a execução de um projeto por diversos relatórios, que podem ser usados tanto pela gerência do projeto como por auditores externos (para comprovar quais testes foram realizados e quais resultados foram obtidos); esses relatórios incluem os de acompanhamento da execução do teste e os de *status* das associações de rastreabilidade. Por fim, os dados armazenados ao longo da execução de vários projetos constituem uma base de conhecimento que pode ser de grande utilidade para a melhoria tanto do processo de teste como de novos projetos.

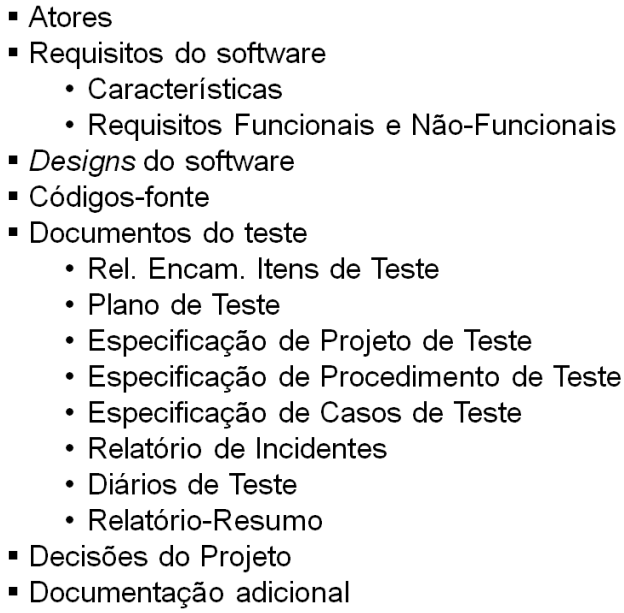

Figura 6.2: Artefatos controlados no protótipo.

A Figura 6.2 ilustra, na forma de árvore, os tipos de artefatos mantidos pelo protótipo para um projeto de teste. A figura mostra somente uma das maneiras que a ferramenta oferece para visualizar esses artefatos e suas relações.

6.2 Pressupostos adotados na implementação

A seguir são apresentados os recursos tecnológicos utilizados e os pressupostos adotados em relação à modelagem dos documentos e ao formato para entrada de dados.

6.2.1 Recursos tecnológicos

O protótipo foi desenvolvido utilizando a linguagem de programação Microsoft Visual C# Express (http://msdn.microsoft.com/vstudio/express/visualcsharp/) e funciona no sistema operacional Microsoft Windows XP. Para armazenar as informações é utilizado o gerenciador de banco de dados MySQL (http://www.mysql.org/).

6.2.2 Representação de documentos

O conteúdo de um documento é composto pelo conjunto de seus atributos que podem ser armazenados em uma ou mais entidades de um banco de dados (Hay, 1999). Exemplos de atributos ou elementos de informação de um documento são: título, autor, resumo, estado atual, observações, conteúdo de uma seção, etc. (Duarte e Grahl, 2000; Sommerville, 2003). A quantidade e o conjunto de atributos são estruturados de acordo com o contexto da aplicação.

Cada documento pode ser uma instância de um "tipo de documento", que representa uma determinada estrutura de atributos. Na PROMETEU, exemplos de tipos de documentos são: Plano de Teste, Especificação de Caso de Teste e Registro de Incidente.

Cada "tipo de documento" geralmente possui um modelo associado que, por sua vez, possui uma estrutura que representa a organização de um conjunto de dados a serem armazenados para registrar determinadas informações (Hay, 1999; Duarte e Grahl, 2000). A estrutura de um documento geralmente é composta por uma ou mais seções. A informação de cada seção é constituída geralmente de declarações textuais estáticas ou por referências ao conteúdo de outras seções de outros documentos ou por uma mistura de ambos (Hay, 1999). Na versão atual do protótipo a estrutura de um tipo de documento é fixa e organizada em seções seguindo o prescrito

na Norma IEEE Std 829-1998. A estrutura é fixa dadas as particularidades de cada tipo de documento e as necessidades específicas de manipulação de dados sobre cada um deles.

Também é desejável registrar as relações entre documentos e as pessoas que os manipulam (Hay, 1999). As pessoas podem assumir diversos papéis (criador, revisor, etc.). No protótipo é possível atribuir, conforme o documento, um responsável, um revisor e uma lista de atores envolvidos na concepção e/ou evolução do documento.

Quando necessário, deve ser previsto o registro das várias versões de um documento (Hay, 1999; Duarte e Grahl, 2000). Na versão atual da PROMETEU não foi implementado suporte ao registro de várias versões para um artefato.

Considerando o exposto nesta subseção, uma aplicação que necessite representar documentos, pode considerar a seguinte caracterização:

- Um ou mais atores, cada um com papel definido, manipula os documentos; cada ator pode ter uma relação específica com um documento como, por exemplo, responsável, revisor, fornecedor de informações;
- Cada documento pode ser de um tipo e, para cada tipo, pode existir um modelo específico;
- Um documento possui uma ou mais versões;
- Cada versão é composta por uma ou mais seções e cada seção pode ser composta por subseções.

Formato para entrada de dados

Ferramentas de suporte à documentação geralmente utilizam o formato de formulários para entrada de dados; a disposição e o desenho desses formulários variam de acordo com o tipo de documento e com os elementos de informação que serão registrados (Hay, 1999; DiMaggio, 2001).

A Figura 6.3 mostra um exemplo de formulário para entrada de dados relativos ao documento do tipo “Especificação de Projeto de Teste” na PROMETEU.

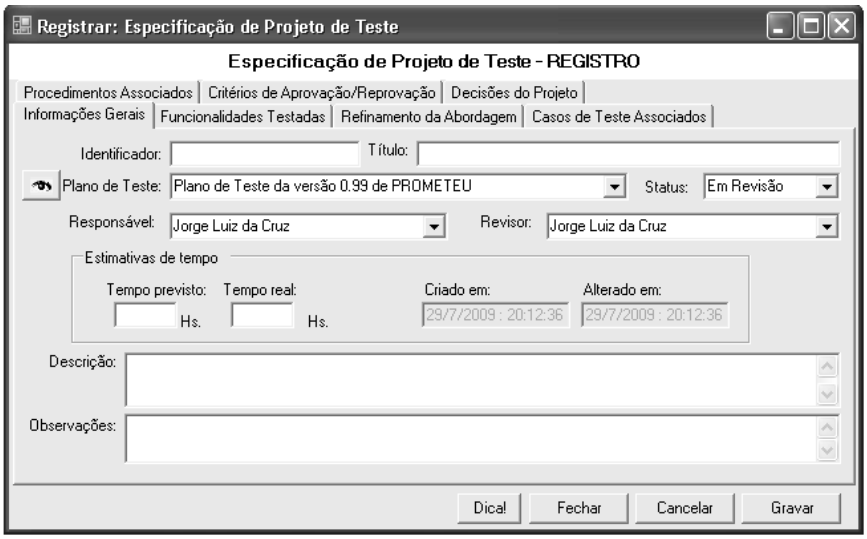

Figura 6.3: Exemplo de formulário para entrada de dados.

6.3 Artefatos e *links* de rastreabilidade do protótipo

PROMETEU possui foco em usuários *low-end* e disponibiliza vínculos pré-definidos entre os artefatos sob seu controle. Os artefatos e links de rastreabilidade do protótipo são apresentados a seguir. No Apêndice A é apresentado um exemplo de utilização da ferramenta.

6.3.1 Itens de rastreabilidade

Os principais itens de rastreabilidade em PROMETEU são:

- Atores do processo;
- Justificativas para decisões tomadas ao longo do processo;
- Requisitos do software;
- *Designs* (desenhos) do software;
- Códigos-fonte;
- Documentos do teste de acordo com a norma IEEE Std 829-1998;
- Documentação adicional específica do processo e não prevista na Norma IEEE Std 829-1998.

6.3.1.1 Atores

Os papéis dos atores são organizados em duas grandes categorias: *Stakeholder* e Técnico. *Stakeholders* usualmente representam clientes e seus representantes, enquanto os técnicos

representam a equipe técnica alocada ao projeto. A Figura 6.4 ilustra a interface para registro dos dados de um ator.

Figura 6.4: Registro de dados sobre um ator.

6.3.1.2 Razões para decisões do projeto

Os seguintes campos estão disponíveis na ferramenta para registrar os dados relativos às decisões do projeto: “observações”, “motivação”, “lembretes” e “histórico”. Esses campos podem ser acessados, quando estiverem disponíveis para um artefato, por meio da aba “Decisões do Projeto” e da aba “Informações Gerais”.

6.3.1.3 Requisitos do software

Na PROMETEU, os requisitos do software são representados, no nível mais alto, como características; cada característica é detalhada por um ou mais requisitos do tipo Funcional (RF) ou Não-Funcional (RNF).

A descrição de cada requisito deve possuir informações suficientes para possibilitar o *design* dos casos de teste para validá-lo (Pfleeger, 2004). Para propiciar seu rastreamento e associações com outros artefatos do teste, cada requisito deve possuir um identificador único.

Apesar de o protótipo desenvolvido não ser uma ferramenta especializada no gerenciamento de requisitos, ele oferece a possibilidade de o usuário registrar e controlar os requisitos do software por meio da sua interface. A Figura 6.5, a seguir, mostra a interface para registro dos elementos de informação relativos a um artefato “Característica”.

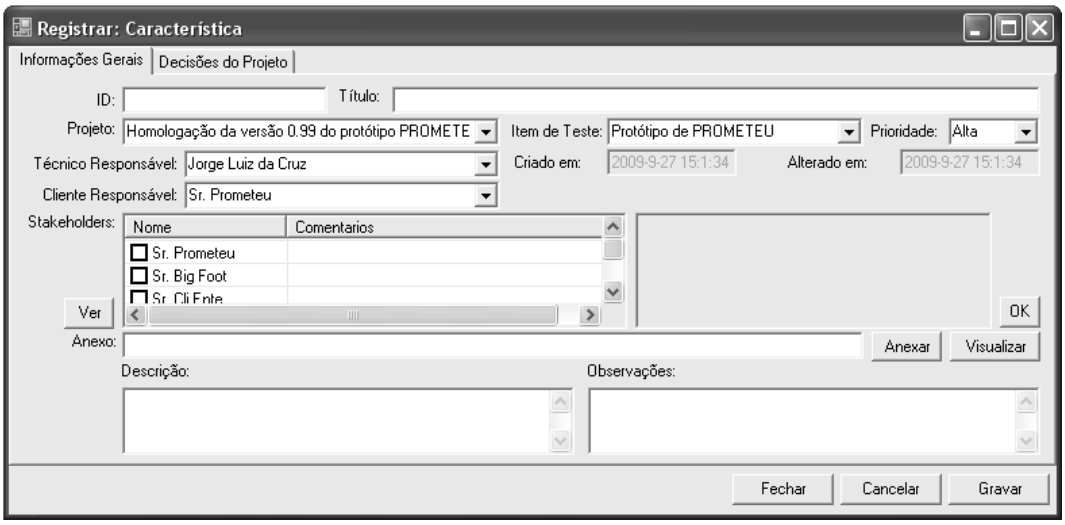

Figura 6.5: Registro de uma característica.

A Figura 6.6 ilustra a interface para registro dos elementos de informação relativos a um artefato “Requisito”. Na aba “Informações Gerais”, são registrados os seguintes elementos de informação para um requisito: identificador, título, prioridade do requisito para o teste, característica a qual o requisito será associado, “Req PAI” (caso o requisito seja subordinado a outro), categoria, tipo, *status*, responsável técnico, data do registro, data da última modificação, *stakeholders* associados (para cada *stakeholder* é possível inserir um comentário associado ao requisito), referência a um arquivo externo (para complementar a descrição do requisito), uma descrição e eventuais observações.

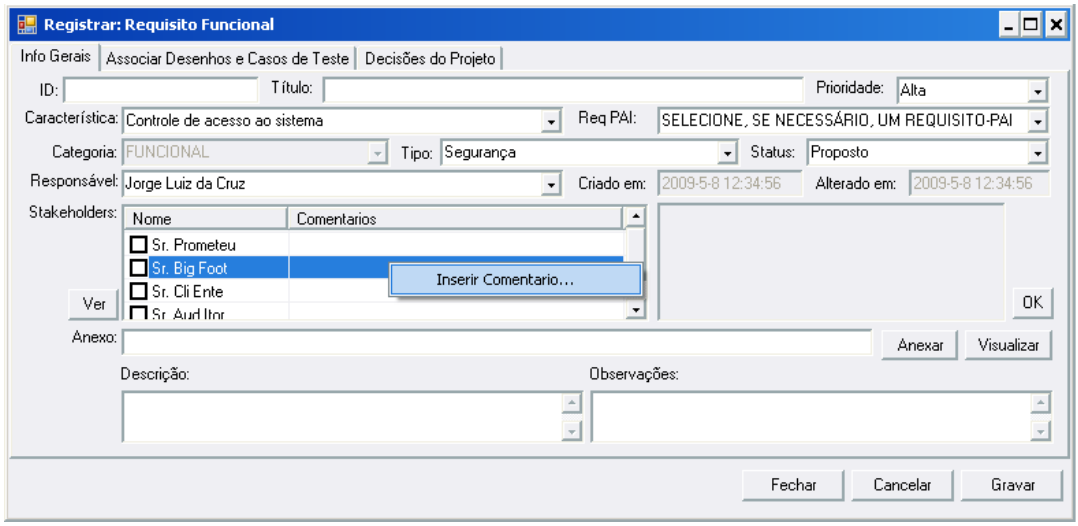

Figura 6.6: Registro de um requisito funcional.

A Figura 6.7 ilustra a seção “Associar Desenhos e Casos de Teste” de um requisito, onde é possível associar o requisito a *designs* e casos de teste registrados na ferramenta. Na aba “Decisões de Projeto” é possível registrar informações sobre a motivação para o registro do requisito, histórico de modificações e eventuais lembretes.

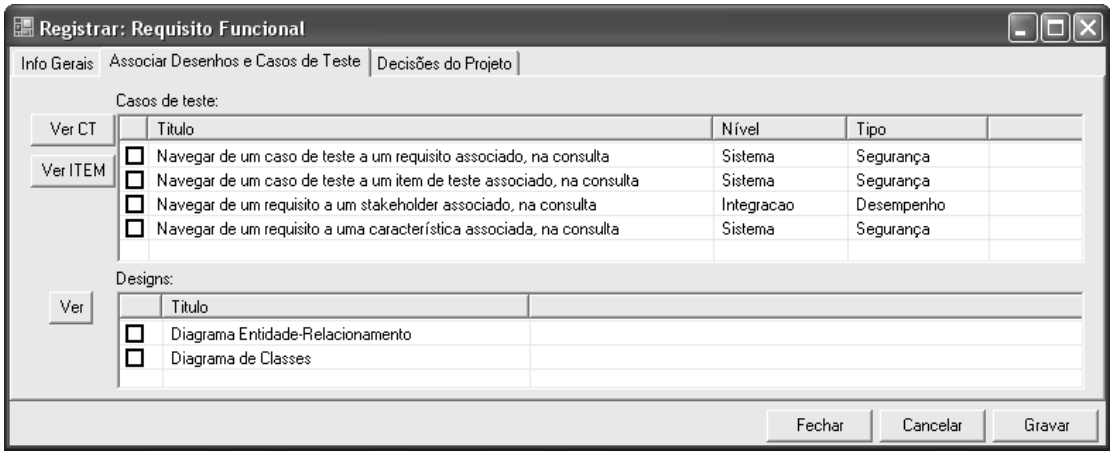

Figura 6.7: Associação entre requisitos, casos de teste e *designs*.

6.3.1.4 *Design* (desenho) do software

Em relação aos *designs*, considera-se que são gerados por ferramentas especializadas, sendo armazenados em arquivos externos; no protótipo são registradas somente informações relevantes para as atividades do teste. Na Figura 6.8 é possível ver a aba para registrar dados gerais sobre um *design* do software. Na aba “Artefatos Associados” o *design* pode ser vinculado a requisitos, códigos-fonte e casos de teste. Na aba “Decisões de Projeto” podem ser registrados elementos de informação relativos às decisões tomadas no projeto em função deste *design*.

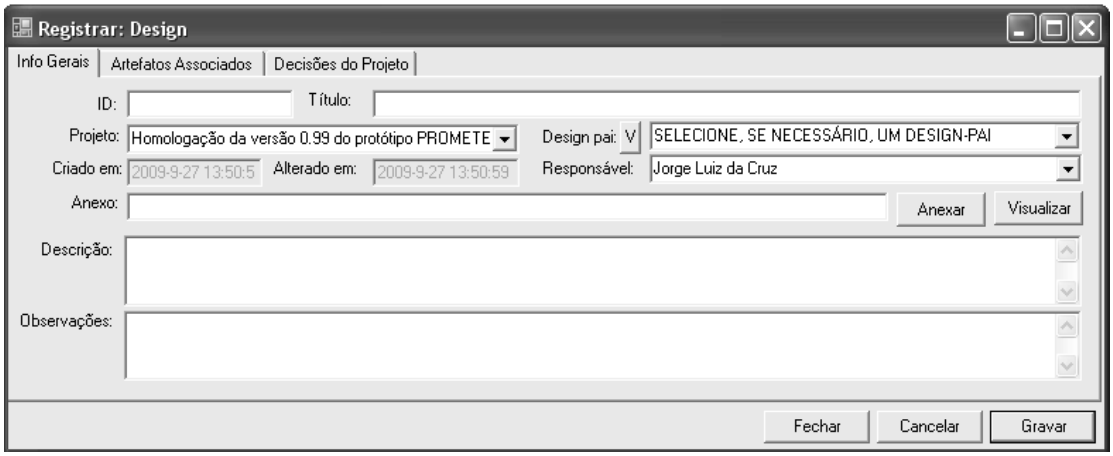

Figura 6.8: Registro de um *design*.

6.3.1.5 Código-fonte do software

No protótipo, os artefatos de código-fonte podem ser vinculados aos *designs* do software e também a um conjunto de casos de teste, possibilitando estabelecer um caminho que inicia nos atores que definiram os requisitos, passa pelos requisitos, *designs* relacionados aos requisitos, códigos-fonte associados aos *designs* e chega aos casos de teste projetados e executados para validá-los. Na Figura 6.9 é possível ver a aba para registrar dados gerais sobre um *código-fonte*. Na Figura 6.10 é possível ver a aba "Artefatos Associados", onde o código-fonte pode ser vinculado a *designs* e casos de teste.

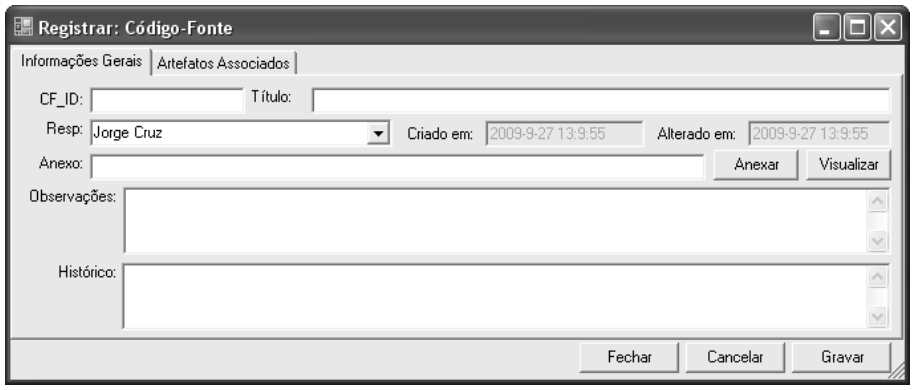

Figura 6.9: Registro de um código-fonte.

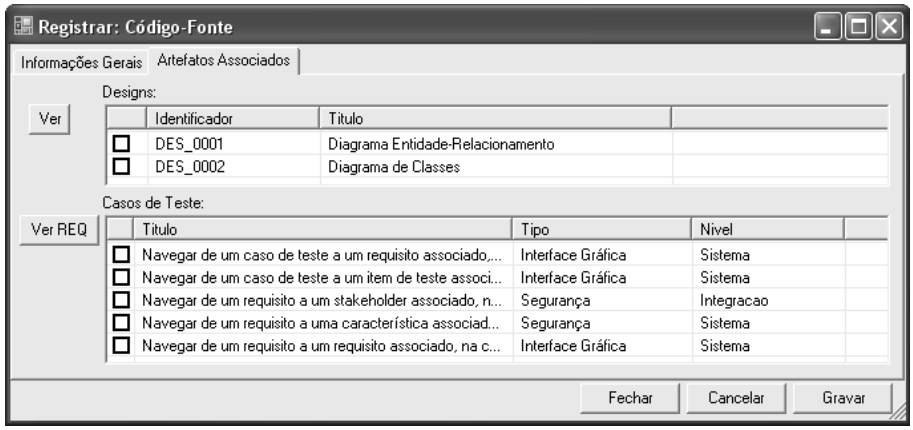

Figura 6.10: Associação de um código-fonte a *designs* e casos de teste.

6.3.1.6 Documentação adicional

Na ferramenta são registrados os seguintes elementos de informação: título, tipo, técnico responsável, localização no sistema de arquivos e uma descrição. Na versão atual do protótipo é

possível associar documentos adicionais a documentos do tipo “Plano de Teste”. Na Figura 6.11 é possível ver a aba para registrar dados gerais sobre um documento adicional.

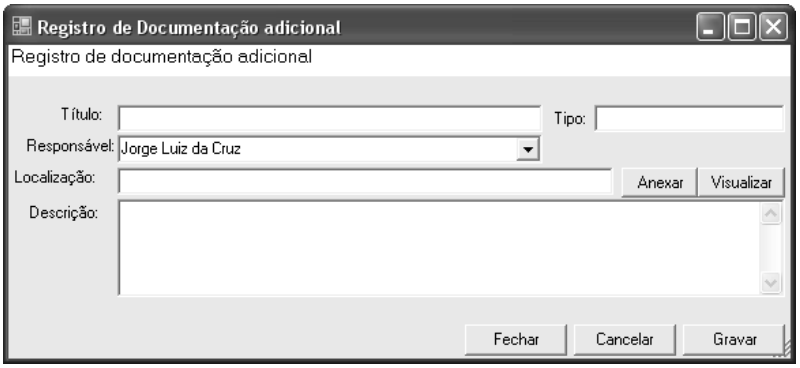

Figura 6.11: Registro de um “Documento Adicional”.

6.3.1.7 Documentos de teste

Conforme mencionado no Capítulo 5, os documentos de teste gerenciados por meio da PROMETEU são: Plano de Teste, Especificação de Projeto de Teste, Especificação de Procedimento de Teste, Especificação de Caso de Teste, Relatório de Incidente de Teste, Diário de Teste e Relatório de Encaminhamento de Item de Teste. Quanto ao Relatório-Resumo do Teste, o protótipo oferece diversas opções de relatório que cumprem o papel deste tipo de documento. Cada documento é composto por um conjunto de seções conforme descrito na Norma IEEE Std 829-1998. Cada seção pode ser preenchida com um ou mais elementos de informação, que podem ser estáticos (texto digitado pelo usuário, sem vínculo direto com outros artefatos e sem associações de rastreabilidade controladas pela ferramenta) ou dinâmicos (campos de um artefato vinculados a outros artefatos registrados e mantidos pela ferramenta).

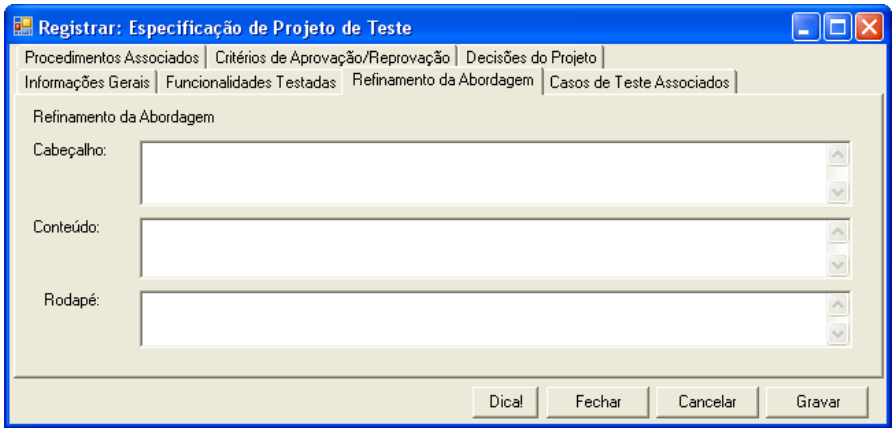

Figura 6.12: Especificação de Projeto de Teste – Seção “Refinamento da Abordagem”.

As Figuras 6.12 e 6.13 ilustram seções de um documento do tipo “Especificação de Projeto de Teste”. A Figura 6.12 mostra a seção “Refinamento da Abordagem”, com os campos de texto estático “Cabeçalho” (texto para o cabeçalho da seção), “Conteúdo” (campo para descrever o conteúdo da seção) e “Rodapé” (texto para o rodapé da seção). E a Figura 6.13 ilustra a seção “Funcionalidades Testadas”, com os campos de texto estático “Cabeçalho” e “Rodapé” e com o campo “Conteúdo” com elementos de informação dinâmicos (neste caso são listados todos os requisitos associados às características que fazem parte do Plano de Teste ao qual esta especificação de projeto está vinculada).

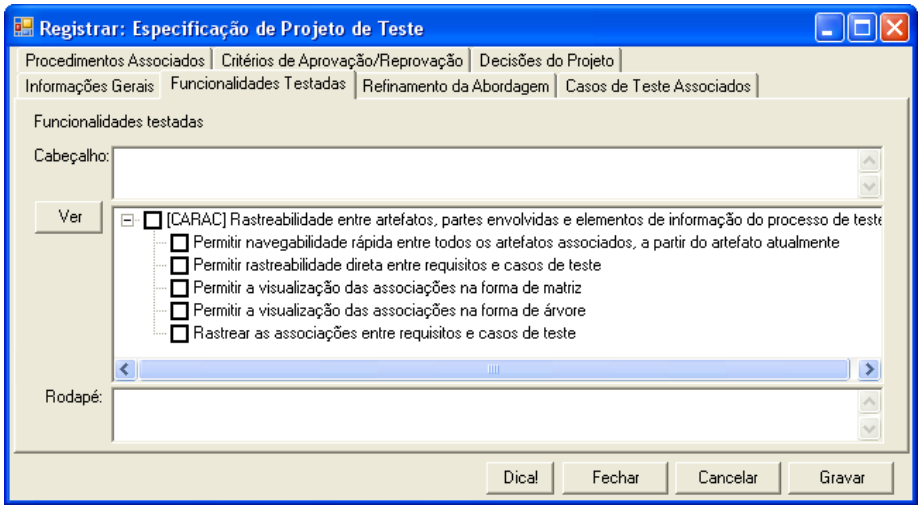

Figura 6.13: Especificação de Projeto de Teste – Seção “Funcionalidades Testadas”.

A seguir são apresentados os oito documentos de teste e alguns exemplos de telas utilizadas para registrar os elementos de informação para cada um deles.

Plano de Teste

Um documento “Plano de Teste” possui três seções com itens dinâmicos: “Funcionalidades que Serão Testadas”, “Funcionalidades que Não Serão Testadas” e “Documentos Adicionais”. A Figura 6.14 mostra a Seção “Informações Gerais” para registro de elementos de informação em um artefato “Plano de Teste”. A Figura 6.15 mostra a Seção “Características Testadas”, na qual são selecionadas todas as características que serão testadas – a seleção é feita a partir de todas as características previamente registradas na ferramenta; a partir desta seção é possível visualizar o conteúdo de cada característica, bem como dos requisitos associados a elas.

Figura 6.14: Plano de Teste – Seção “Informações Gerais”.

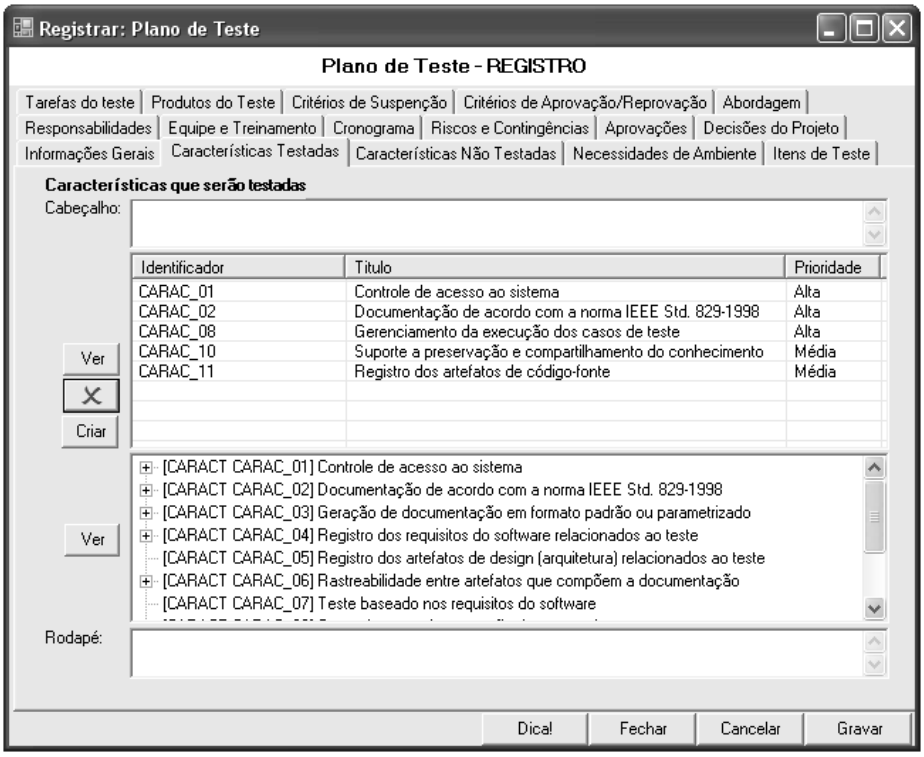

Figura 6.15: Plano de Teste – Seção “Características Testadas”.

Especificação de Projeto de Teste

A Figura 6.16 ilustra a seção “Informações Gerais” e a Figura 6.17 ilustra a seção “Casos de Teste Associados”, onde são selecionados os requisitos que farão parte da especificação. A Figura 6.18 ilustra a seção “Funcionalidades Testadas”, onde é possível vincular os casos de teste que farão parte da especificação.

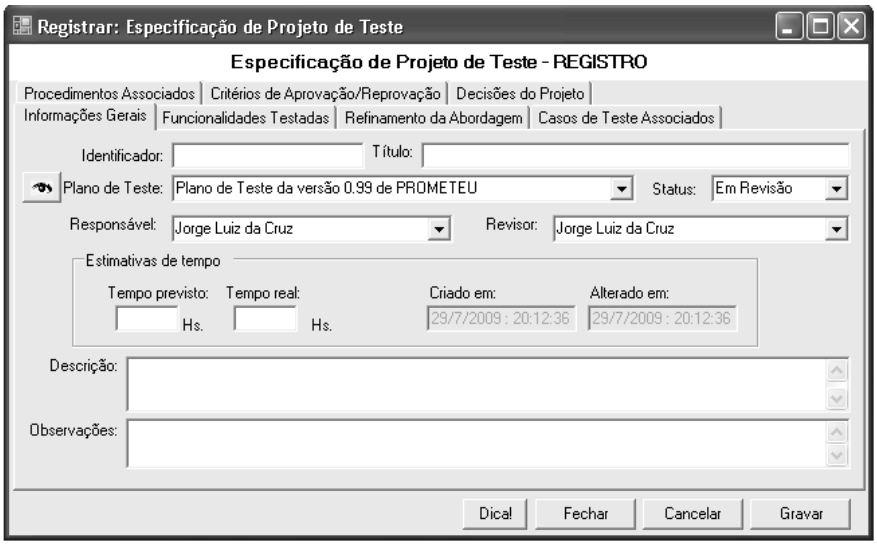

Figura 6.16: Especificação de Projeto de Teste – Seção “Informações Gerais”.

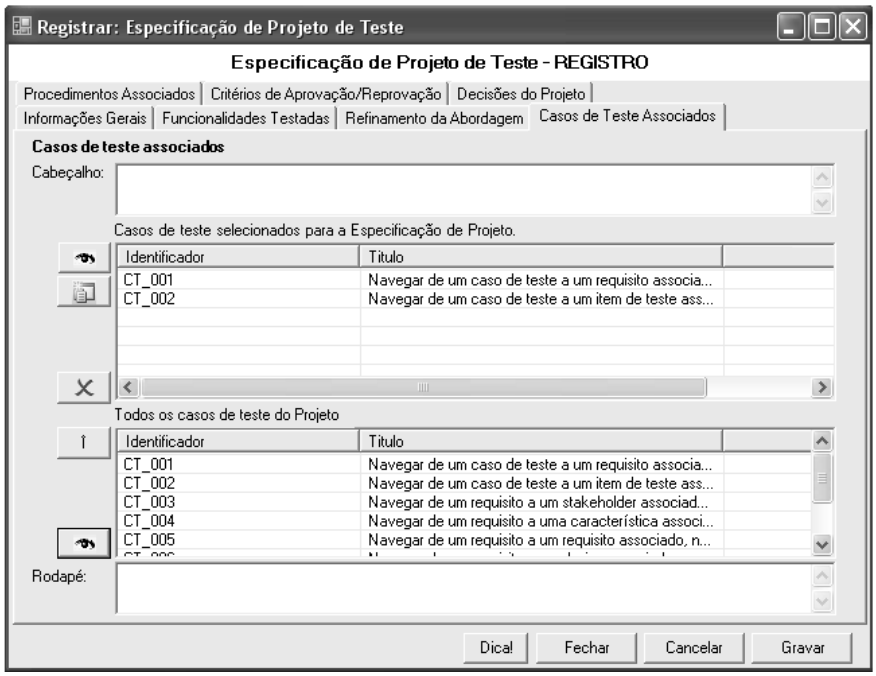

Figura 6.17: Especificação de Projeto de Teste – Seção “Casos de Teste Associados”.

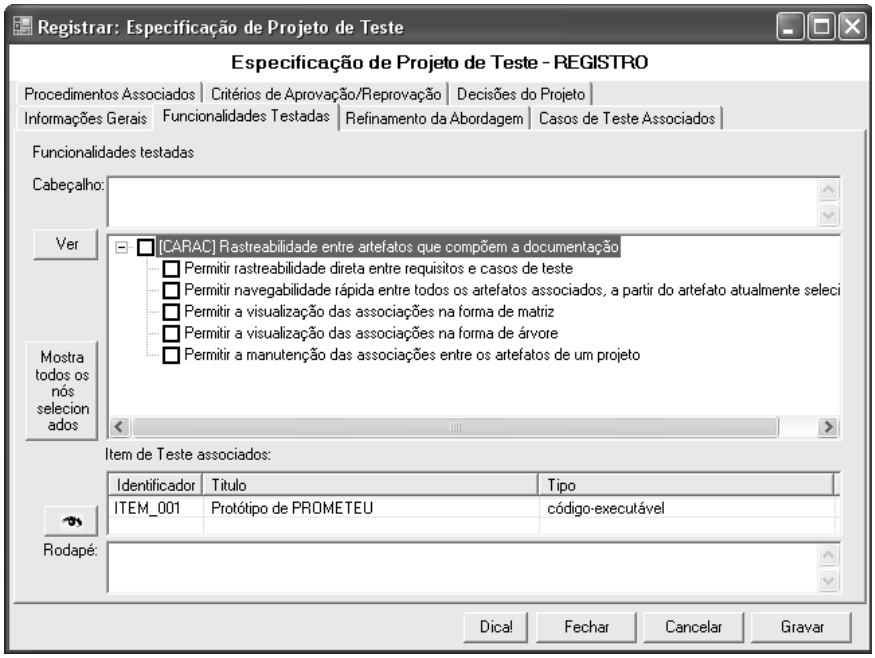

Figura 6.18: Especificação de Projeto de Teste – Seção “Funcionalidades Testadas”.

Especificação de Procedimento de Teste

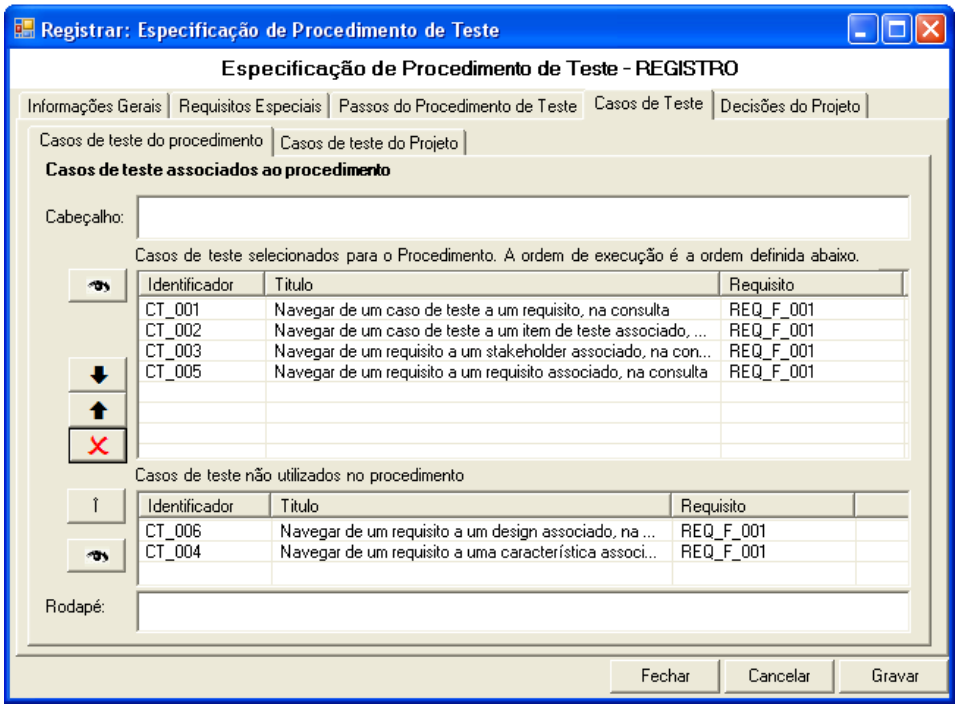

Figura 6.19: Especificação de Procedim. de Teste – Seção “Casos de Teste do Procedimento”.

A Figura 6.19 ilustra a Seção “Casos de Teste do Procedimento”, onde os casos de teste são vinculados ao procedimento de teste. Nesta seção, os casos de teste são vinculados na ordem na qual serão executados.

Especificação de Caso de Teste

Para cada caso de teste são registrados dados de entrada, resultados esperados, ações, condições gerais para sua execução, associações com requisitos, *designs* e códigos-fonte, além do nível de teste para o qual o caso de teste foi projetado e o tipo de teste que ele representa.

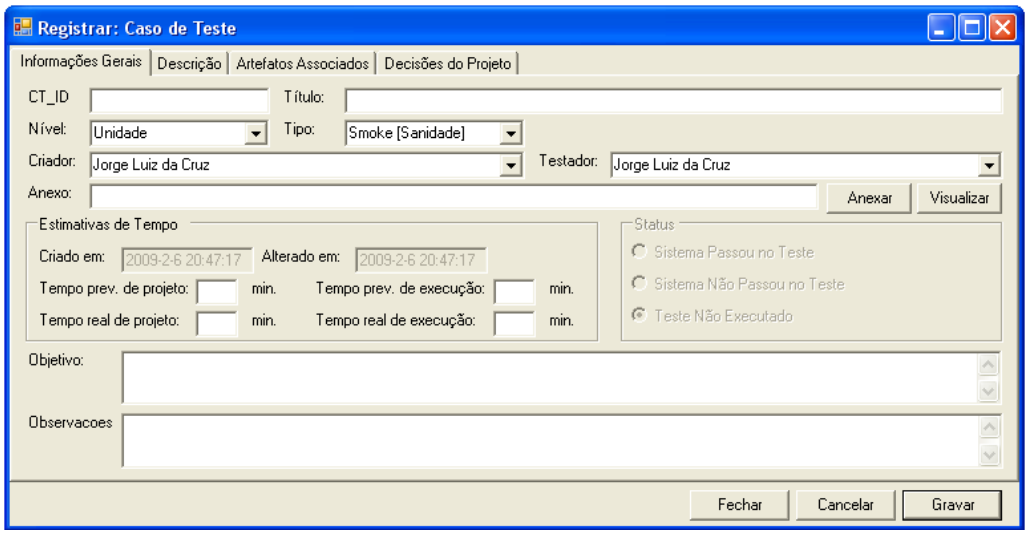

Figura 6.20: Registro de um Caso de Teste – Seção “Informações Gerais”.

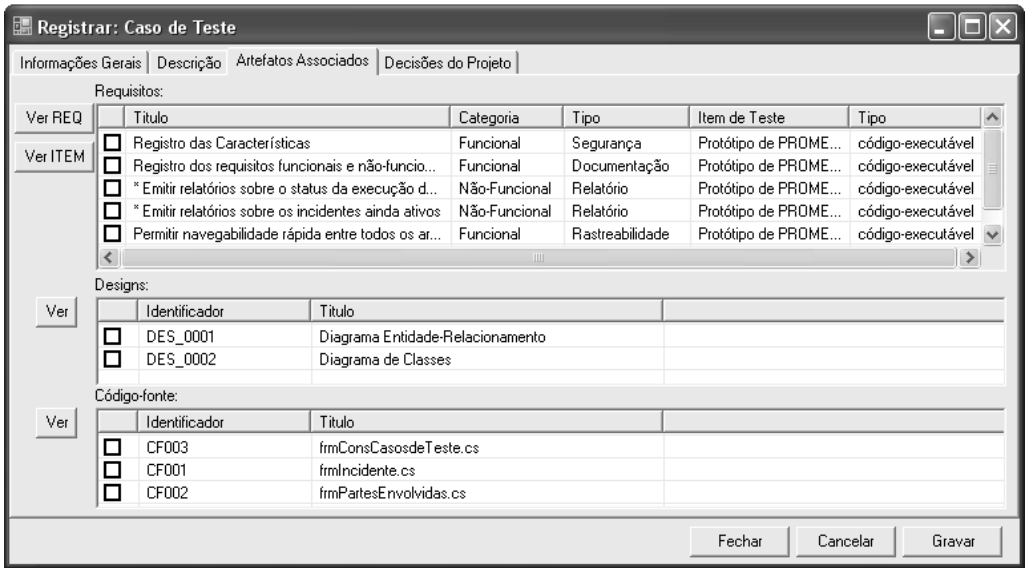

Figura 6.21: Caso de teste: associação com requisitos, *designs* e códigos-fonte.

A Figura 6.20 ilustra uma das telas para registro de informações relativas a um caso de teste e a Figura 6.21 mostra a tela “Artefatos Associados”, onde é possível associar um caso de teste a requisitos, *designs* e códigos-fonte.

Diário de Teste

Neste tipo de documento devem ser registradas tarefas relevantes relacionadas com a execução dos testes. A Figura 6.22 ilustra a interface gráfica de um diário de teste.

Figura 6.22: Diário de Teste.

Relatório de Incidente de Teste

A Figura 6.23 ilustra a interface gráfica para registro dos dados de um incidente de teste.

Figura 6.23: Relatório de Incidente de Teste.

6.3.2 Associações (links) de rastreabilidade na ferramenta

Como o foco do protótipo é em usuários do tipo *low-end*, os *links* e as operações de consulta sobre eles são pré-determinados.

Os *links* são implementados como relações entre os atributos das tabelas em um sistema gerenciador de banco de dados relacional.

PROMETEU possui *links* bidirecionais entre os diversos artefatos rastreados. Na versão atual da ferramenta não é possível caracterizar semanticamente um *link* de rastreabilidade. O protótipo oferece suporte a rastreabilidade vertical e rastreabilidade horizontal entre artefatos. A Figura 5.13 (Capítulo 5) ilustrou um exmplo do que pode ser feito por meio da PROMETEU. O Apêndice A mostra exemplos de utilização dos *links* de rastreabilidade no protótipo.

Utilizar atributos em *links* de rastreabilidade é uma decisão de projeto e depende dos recursos oferecidos pela ferramenta utilizada; na versão atual da PROMETEU não é possível caracterizar semanticamente um *link* de rastreabilidade.

6.3.3 Representação das associações

Na PROMETEU os *links* de rastreabilidade podem ser visualizados como matrizes e como árvores de rastreabilidade. A Figura 6.24 mostra um exemplo de matriz de rastreabilidade, a matriz "Casos de Teste x Requisitos x *Designs* x Incidentes", onde é possível visualizar, por exemplo, que a execução do caso de teste CT_001 implicou a descoberta e registro do incidente INCIDENTE_001.

Figura 6.24: Matriz de rastreabilidade "Casos de Teste x Requisitos x *Designs* x Incidentes".

Na ferramenta também implementa as matrizes “Requisitos x *Designs* x Casos de Teste”, “*Designs* x Requisitos x Casos de Teste”, “Casos de Teste x Requisitos x *Designs* x Códigos-fonte”, “Códigos-fonte x *Designs* x Casos de Teste x Incidentes”.

A Figura 6.25 mostra um exemplo de árvore de rastreabilidade de um projeto de teste na PROMETEU, na qual é possível visualizar os artefatos associados ao plano de teste “Plano-Mestre de Teste 3”.

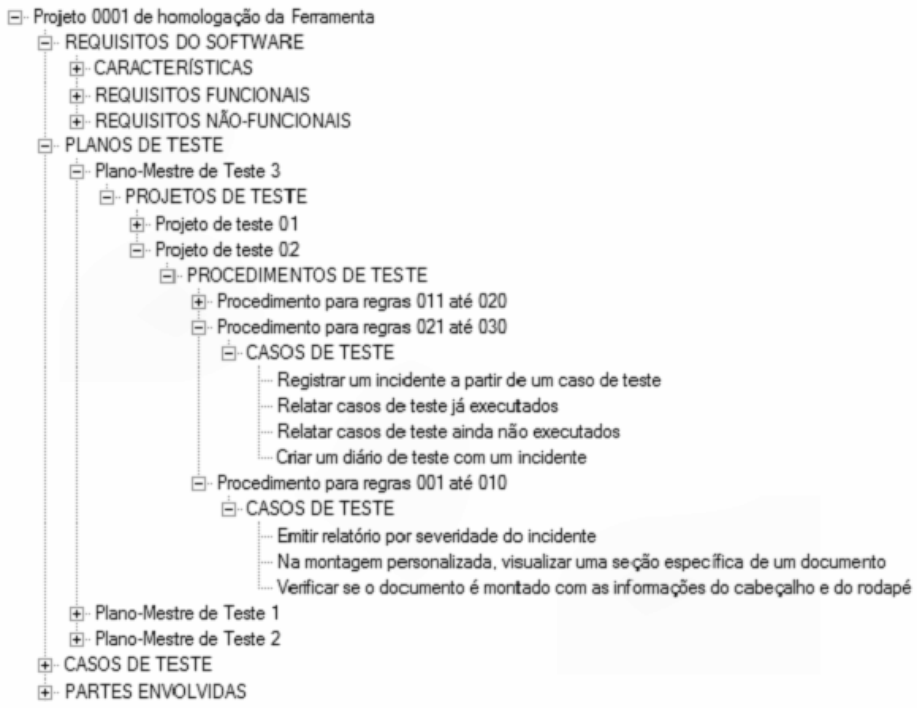

Figura 6.25: Árvore de rastreabilidade de um projeto na PROMETEU.

Para os itens de rastreabilidade do tipo Característica, Requisito, *Design*, Código-fonte, Especificação de Caso de Teste, Registro de Incidente e Ator é possível, na tela de consulta, visualizar uma árvore de rastreabilidade específica do artefato sendo consultado.

6.4 Características do protótipo

As principais características do protótipo são:

- Controle de acesso ao sistema: o acesso às funcionalidades do protótipo é liberado de acordo com as permissões relativas ao papel do usuário. Os papéis possuem duas categorias genéricas: *Stakeholder* e Técnico. A categoria “Stakeholder” não possui papéis pré-definidos, já que um *stakeholder* pode assumir diversos papéis: “Gerente

de Marketing", "Representante do Cliente", entre outros. Sendo assim, os papéis são cadastrados de acordo com o contexto do processo para, então, ser vinculados aos atores que pertencem à categoria *Stakeholder*. Já a categoria "Técnico" possui os seguintes papéis pré-definidos: a) Gerente de Teste: possui acesso total aos recursos do sistema e é o responsável direto pelo cadastro dos requisitos a testar e dos atores em um projeto; b) Engenheiro de Teste: os mesmos direitos do Gerente de Teste, menos a possibilidade de criar processos e registrar usuários; c) Testador: possui acesso aos casos de teste a ele atribuídos, ao registro de incidentes e a relatórios associados; d) Engenheiro da Qualidade: acesso de consulta para revisão de casos de teste e incidentes; e) Desenvolvedor: possui acesso somente aos incidentes a ele atribuídos para correção.

- Documentação de acordo com a norma IEEE Std 829-1998: é possível registrar os elementos de informação relativos aos oito tipos de documentos propostos pela norma. A interface gráfica usa como referência o padrão do sistema operacional Microsoft Windows XP, sendo do tipo "formulário" com diversas "*tab pages*" que procuram refletir a estrutura dos oito tipos de documentos propostos pela norma;
- Geração de documentação em formato padrão ou parametrizado: os documentos do teste poderão ser visualizados no formato padrão da Norma IEEE Std 829-1998 ou pela combinação de seções de documentos diferentes de acordo com a necessidade em um determinado momento. Em relação aos requisitos e *designs* do software relacionados ao teste é possível gerar os documentos no formato padrão oferecido pela ferramenta; a visualização pode ser no formato HTML (*HyperText Markup Language*) ou via interface gráfica da ferramenta;
- Registro dos requisitos do software relacionados ao teste: os requisitos são organizados como "Características" no seu nível mais alto. Cada característica pode ser detalhada por um ou mais requisitos do tipo funcional ou não-funcional. Sobre os requisitos são registrados diversos atributos relacionados às atividades de teste;
- Registro dos artefatos de *design* (arquitetura) relacionados ao teste: os *designs* são artefatos criados posteriormente aos requisitos, para detalhar sua implementação; no protótipo são registrados atributos relacionados às atividades de teste;

- Registro dos artefatos de código-fonte, sendo possível associá-los a *designs* e a casos de teste;
- Rastreabilidade entre artefatos que compõem a documentação: existem diversos vínculos de rastreabilidade pré-definidos entre os artefatos. Esses vínculos permitem localizar rapidamente um determinado elemento de informação, além de contribuir para manter a documentação consistente e atualizada (a ferramenta monitora modificações nos elementos de informação e emite alertas sobre os artefatos potencialmente afetados por uma modificação e sobre eventuais associações suspeitas ou desatualizadas). Os *links* podem ser visualizados na forma de matriz e de árvore, sendo também possível acompanhar o estado das associações entre os artefatos através de um relatório geral de pendências de rastreabilidade;
- Teste baseado nos requisitos do software: a rastreabilidade direta entre requisitos e casos de teste contribui para comprovar que todos os requisitos foram validados bem como facilita visualizar qual a extensão do teste (quantidade de casos de teste) projetado e executado para cada um;
- Gerenciamento da execução dos casos de teste: é possível acompanhar, a qualquer momento, os testes em andamento, quais os testadores envolvidos e quais os resultados obtidos. A cada caso de teste está associado um nível e um tipo de teste;
- Registro e controle dos incidentes de teste: é possível acompanhar todos os incidentes registrados e o estado atual das correções a qualquer momento do projeto;
- Suporte à preservação e compartilhamento do conhecimento: os elementos de informação registrados ao longo de diversos projetos possibilitam, por exemplo: a) reutilizar casos de teste e outros documentos, reduzindo o retrabalho; b) descobrir incidentes e ocorrências típicas que prejudicaram os projetos de forma a definir ações para evitá-los e, conseqüentemente, melhorar o processo de teste.

6.5 Exemplo de utilização

Um exemplo de utilização da PROMETEU é apresentado no Apêndice A, ilustrando as seguintes ações: a) acessar o sistema, b) registrar um projeto, c) registrar as justificativas para as decisões do projeto, d) cadastrar os atores, e) registrar os requisitos do software, f) registrar a arquitetura do software, g) registrar os códigos-fonte, h) registrar elementos de informação da

documentação do teste, i) executar os testes e registrar incidentes, j) gerenciar os incidentes, l) gerenciar as associações de rastreabilidade, m) analisar o impacto de uma modificação, n) analisar as justificativas para decisões do projeto, o) consultar os elementos de informação da documentação, p) manter os elementos de informação da documentação, q) gerar documentação padronizada e personalizada, r) visualizar e imprimir relatórios, e s) analisar incidentes que ocorrem após a entrega do software.

6.6 PROMETEU e outras ferramentas

Esta seção apresenta algumas ferramentas comerciais (com a necessidade de pagamento de licenças de uso) e *freeware* (sem a necessidade de pagamento de licenças de uso) de apoio ao processo de teste; a apresentação é resumida e feita com foco nas características de suporte à documentação e à rastreabilidade entre os artefatos que compõem a documentação do teste. Finalmente, é feita uma breve comparação entre as funcionalidades da PROMETEU e as das ferramentas descritas. A comparação é feita sem a intenção de desqualificar ou recomendar uma ferramenta em detrimento de outras.

Bugzilla (http://www.bugzilla.org/download/)

Ferramenta *freeware* usada para registrar defeitos e incidentes de teste observados ao longo do processo e para gerenciar as atividades para correção dos mesmos. Esta ferramenta pode ser configurada para funcionar em conjunto com a Testlink.

Mantis (http://www.mantisbt.org/download.php)

Ferramenta *freeware* usada para registrar defeitos e incidentes de teste observados ao longo de um projeto e para gerenciar o processo de correção dos mesmos. Com esta ferramenta é possível gerar o documento Relatório de Incidentes de Teste. Esta ferramenta pode ser configurada para funcionar em conjunto com a Testlink.

Testlink 1.7.4 (http://www.teamst.org/)

Ferramenta *freeware* na qual é possível registrar os requisitos do software e os casos de teste, podendo-se estabelecer *links* de rastreabilidade entre esses dois tipos de artefatos. É possível gerar os seguintes tipos de documentos: requisitos e casos de teste. Não existe o recurso de registro e controle de defeitos nesta ferramenta; entretanto, é possível configurar uma integração rudimentar com ferramentas de gerenciamento de defeitos, como Mantis e Bugzilla.

IBM Rational TestManager 7.0

Ferramenta comercial da IBM na qual os tipos de artefatos armazenados são casos de teste. Nesta ferramenta um plano de teste representa somente a organização de um conjunto de casos de teste, diferente do conceito de "Plano de Teste" da Norma IEEE Std 829-1998 (o plano é um documento para planejar e gerenciar um projeto de teste). Nesta ferramenta é possível gerenciar os casos de teste ao longo de diversos *builds* (compilação para código executável de uma versão específica de um software).

IBM Rational RequisitePRO 7.0

Ferramenta comercial da empresa IBM (http://www.ibm.com), na qual os tipos de artefatos gerenciados são os requisitos do software. A rastreabilidade existe somente entre os requisitos, sendo possível estabelecer associações hierárquicas ente eles. Todas as modificações nos requisitos são registradas, gerando um histórico de modificações.

IBM Rational ClearQuest 7.0

Ferramenta comercial da IBM na qual os tipos de artefatos armazenados são os defeitos revelados e solicitações de modificações em artefatos do software. Quando a ferramenta TestManager está instalada no mesmo ambiente de operação da ClearQuest é possível configurar a integração entre ambas, o que possibilita associar casos de teste e defeitos revelados.

6.6.1 Comparações das funcionalidades

Na PROMETEU são registradas informações sobre requisitos e *designs* do software que sejam pertinentes ao planejamento, projeto e execução dos testes; ou seja, em princípio o protótipo não contempla ser o substituto de uma ferramenta para gerenciamento de requisitos e nem de uma ferramenta para gerenciamento de artefatos da arquitetura. Já a documentação específica do teste é feita de acordo com a Norma IEEE Std 829-1998. Uma vez que os elementos de informação dos artefatos tenham sido registrados, existe rastreabilidade entre atores, requisitos, elementos da arquitetura, e todos os artefatos prescritos pela norma (entre eles, casos de teste e defeitos). Os documentos podem ser gerados de acordo com o padrão da norma ou personalizados a partir da seleção de elementos de informação previamente registrados em documentos de tipos diferentes.

Nas ferramentas da IBM Rational, RequisitePro é usada para gerenciar os requisitos e permitir rastreabilidade entre eles, TestManager é usada para gerenciar os casos de teste e ClearQuest é usada para gerenciar os defeitos descobertos. A rastreabilidade entre requisitos e casos de teste e entre defeitos e casos de teste é obtida somente após cada ferramenta ser instalada e, além disso, ser configurada a integração entre elas; isto dificulta a rastreabilidade sob o ponto de vista do teste e implica um investimento muito alto em termos de licenças de software e despesas com treinamento. Além disso, a TestManager não oferece suporte ao registro e manutenção de todos os elementos de informação prescritos na Norma IEEE Std 829-1998.

Quando a ferramenta RequisitePRO está instalada no mesmo ambiente de operação da TestManager, é possível configurar a integração entre ambas, o que possibilita associar casos de teste a requisitos. No entanto, mesmo com a integração configurada, no momento da modificação de um artefato (requisito ou do caso de teste) não existe um alerta automático indicando quais artefatos associados podem ficar obsoletos como conseqüência dessa alteração de conteúdo. Na PROMETEU isso é possível: toda vez que um artefato vai ser modificado, a ferramenta emite um alerta indicando todos os artefatos associados. Assim, se necessário, é possível revisar todos os artefatos logo após a modificação ser feita. Tanto a RequisitePRO como a PROMETEU produzem um relatório de *status* indicando as associações entre artefatos que estão desatualizadas.

PROMETEU foi projetada para ser uma ferramenta de apoio à documentação de um processo de teste e não para gerenciamento de requisitos. No entanto, dados sobre os requisitos da aplicação que é testada são registrados no protótipo uma vez que os requisitos são fundamentais para o planejamento, projeto, execução e acompanhamento do teste. A ênfase é em relação ao registro de elementos de informação e geração de documentos de acordo com a Norma IEEE Std 829-1998; o protótipo também possibilita a geração parametrizada de documentos (podem-se combinar seções de dois ou mais documentos em um único documento). A rastreabilidade não é somente vertical entre os mesmos tipos de artefatos; ela é horizontal também, existindo associações entre artefatos de fases distintas do processo de desenvolvimento (exemplo: requisitos, *designs* e casos de teste). No caso da PROMETEU, as associações entre requisitos, desenhos e artefatos do teste, incluindo os defeitos revelados, são controladas em uma mesma ferramenta, o que simplifica o gerenciamento das atividades do teste, além de reduzir custos com treinamentos e aquisição licenças de utilização para várias ferramentas.

A Tabela 6.1 mostra as ferramentas analisadas e os artefatos por elas gerenciados. No caso dos artefatos "*Design* (Desenho)" e "Código-fonte", PROMETEU armazena atributos destes artefatos que sejam necessários ao teste, mas pressupõe-se que esses artefatos são gerados e mantidos por meio de ferramentas especializadas.

Tabela 6.1: Ferramentas e artefatos por elas gerenciados.

Ferramenta	**Características de documentação – artefatos gerenciados através ferramentas**										
	Requisitos	Designs (*)	Código-fonte (*)	Defeitos	Plano de Teste	Espec. Projeto de Teste	Espec. Procedim. de Teste	Espec. de Caso de Teste	Diário de Teste	Rel. de Encam. de Item de Teste	Rel. Resumo de Teste
IBM Rational TestManager 7.0							X	X			
IBM Rational RequisitePRO 7.0	X										
IBM Rational ClearQuest 7.0				X							
Testlink 1.7.4	X							X			X
Bugzilla				X							
Mantis				X							
PROMETEU	X	X	X	X	X	X	X	X	X	X	X

Tabela 6.2: Rastreabilidade entre artefatos controlados pelas ferramentas.

Ferramenta	**Rastreabilidade entre artefatos controlados por uma ferramenta ou através da integração entre ferramentas**										
	Requisitos	Designs	Código-fonte	Defeitos	Plano de Teste	Espec. Projeto de Teste	Espec. Procedim. de Teste	Espec. de Caso de Teste	Diário de Teste	Rel. de Encam. de Item de Teste	Rel. Resumo de Teste
IBM Rational TestManager 7.0							X	X			X
IBM Rational RequisitePRO 7.0	X										
IBM Rational ClearQuest 7.0				X							
IBM Rational TestManager 7.0 + IBM Rational RequisitePRO 7.0 + IBM Rational ClearQuest 7.0	X			X			X	X			X
Testlink 1.7.4 + Bugzilla	X			X				X			X
Testlink 1.7.4 + Mantis	X			X				X			X
PROMETEU	X	X	X	X	X	X	X	X	X	X	X

A Tabela 6.2 apresenta a rastreabilidade que pode ser estabelecida usando cada uma das ferramentas. Existem casos nos quais a rastreabilidade entre os artefatos pode ser expandida pela integração entre duas ou mais ferramentas como a integração entre as ferramentas Testlink e Mantis.

6.7 Síntese do Capítulo

Este capítulo apresentou o protótipo PROMETEU, sua origem, os pressupostos de implementação, suas funcionalidades; foi feita uma breve comparação com outras ferramentas, com ênfase nas características de suporte à documentação e à rastreabilidade em um processo de teste.

A implementação de um protótipo, com foco em usuários *low-end* foi bastante importante para validar a aplicação da rastreabilidade entre os elementos de informação que compõem a documentação de um processo de teste: informações atualizadas e consistentes contribuem para a tomada de decisões de boa qualidade.

Apesar de ainda não existirem resultados da aplicação do protótipo em empresas desenvolvedoras de software, a vivência profissional na área de teste e os comunicados existentes nos diversos fóruns de discussão sobre teste de software destacam a necessidade de uma ferramenta com suporte à rastreabilidade, simples de usar (foco em usuários *low-end*) e que possibilite documentar e gerar evidências atualizadas e consistentes do teste do software; preferencialmente esta ferramenta deve ter sua arquitetura e filosofia baseadas em uma norma internacional, que represente a unificação do conhecimento de diversos especialistas da área. PROMETEU representa um passo importante neste sentido.

O capítulo a seguir apresenta as conclusões deste trabalho, destacando as suas contribuições e futuras melhorias a serem realizadas no protótipo e nos modelos de rastreabilidade e de dados.

Capítulo 7

Conclusões

7.1 Considerações gerais

As aplicações de software aumentam progressivamente sua influência na execução das atividades humanas. Melhorar e assegurar a qualidade dos produtos de software tornou-se uma exigência constante por parte dos envolvidos na sua produção, utilização e comercialização. Neste contexto, o teste é reconhecido como um processo fundamental para obtenção de um produto de software de boa qualidade.

Testar software é um processo complexo e dispendioso, tanto que a devida atenção com as questões gerenciais representa, na maioria das vezes, a chave para atingir os objetivos de um projeto de teste. Um aspecto fundamental para o planejamento e gerenciamento adequado do teste é uma documentação de boa qualidade, composta por elementos de informação atualizados, precisos e consistentes entre si. Documentação de boa qualidade possibilita, por exemplo: a) capacidade de relatar rapidamente o *status* do projeto (p.ex. informar como e quais testes já foram realizados, quais foram os resultados obtidos até o momento); b) capacidade de repetir testes já executados; c) capacidade de identificar todos os requisitos que foram testados, quais casos de teste foram executados para cada requisito, quais resultados foram obtidos e quais requisitos não foram testados; d) capacidade para prever a abrangência dos impactos das mudanças nos artefatos e revisar todos os artefatos afetados.

Apesar dessa importância, ainda persiste uma cultura negativa que procura justificar a ausência de documentação com argumentos como: “não se sabe o que deve ser documentado”, ou “os documentos ficam rapidamente desatualizados e inconsistentes, pois é extremamente difícil atualizá-los ao longo do processo, dados todos os vínculos existentes entre os elementos de informação”. Uma possível solução para este problema é a utilização de padrões de

documentação e a implantação de mecanismos que contribuam fortemente para manter os documentos atualizados e consistentes.

Neste trabalho: a) utilizou-se a norma IEEE Std 829-1998 como padrão para documentação do teste; b) propôs-se um modelo de rastreabilidade entre os elementos de informação que compõem a documentação como mecanismo para auxiliar a manter a documentação atualizada e consistente; c) propôs-se um modelo de dados que fornece suporte tanto à documentação segundo a norma como à rastreabilidade; d) implementou-se um protótipo de ferramenta, a PROMETEU, que viabiliza a utilização prática destes modelos: as funcionalidades do protótipo possibilitam manter atualizados os elementos de informação da documentação e os vínculos entre eles.

Documentação de boa qualidade é um aspecto crítico para o sucesso de um projeto de teste. A Norma IEEE Std 829-1998 é um padrão de documentação que pode ser usado por qualquer empresa que queira documentar seu processo de teste. A norma é composta por um conjunto de oito documentos que podem ser combinados em um número menor de documentos de forma a tornar a documentação menos burocrática. Ressalte-se que a norma não deve ser utilizada sem ter sido feito um estudo cuidadoso do seu conteúdo para, então, adaptá-la ao contexto específico do seu usuário. Devem ser documentados somente os elementos de informação necessários de acordo com o contexto da empresa, sem levar a situações como documentar em excesso ou não documentar o suficiente. Essa norma é utilizada com sucesso desde o ano 2000 pelo Grupo de Teste do CTI, contexto no qual se originou este trabalho. Uma das funcionalidades do protótipo é, justamente, possibilitar a geração parametrizada de documentos, com um número menor de elementos de informação do que aquele sugerido pela norma.

Em 2008 foi lançada a nova versão da norma, denominada “IEEE Std 829-2008 Standard for Software and System Test Documentation”. A mudança da versão de 2008 em relação à de 1998 é, fundamentalmente: na versão de 1998 existe somente a dimensão dos documentos do teste e, na versão de 2008, existe adicionalmente a dimensão que trata das atividades de um processo de teste. Como este trabalho está fundamentado na versão de 1998, foram feitas somente observações relativas à organização dos documentos da versão 2008 e sua equivalência com os documentos da versão de 1998. A versão atual do protótipo está preparada para gerar documentos de acordo com a versão de 1998; as modificações necessárias para atender à versão de 2008 fazem parte dos trabalhos futuros decorrentes desta dissertação.

"Talvez a pior documentação seja aquela que é bem estruturada e formatada, mas incorreta ou desatualizada, levando seu usuário a tomar decisões baseadas em suposições incorretas e/ou incompletas" (Lewis, 2000).

Neste trabalho, rastreabilidade foi o mecanismo adotado para auxiliar a manter os documentos atualizados e consistentes. Rastreabilidade pode ser entendida como o grau de relacionamento que pode ser estabelecido entre dois ou mais artefatos do processo de desenvolvimento, especialmente artefatos que, entre si, possuam relacionamento do tipo predecessor-sucessor ou mestre-subordinado (como o relacionamento existente entre um requisito e um ou mais casos de teste que o validam).

Foram apresentados nove modelos de rastreabilidade, sete deles com foco no processo de requisitos e dois com foco no processo de teste. Os modelos estudados indicam os tipos de artefatos e os relacionamentos entre eles; os modelos foram usados como base para a definição de um modelo de rastreabilidade para a documentação de um projeto de teste. Na definição deste modelo também foram considerados a Norma IEEE Std 829-1998 e o modelo V; a norma indica os elementos de informação a serem documentados em um projeto de teste além de servir para corroborar as relações entre tipos de artefatos de diferentes fases do processo de software; o modelo V, por sua vez, também corrobora as relações entre tipos de artefatos de diferentes fases do processo de software. Em relação aos elementos de informação utilizados para os requisitos, *designs* e códigos-fonte, eles foram baseados na literatura estudada.

Convém ressaltar que apesar de ser possível estabelecer *links* de rastreabilidade entre todos os artefatos de um processo, deve-se planejar a estratégia de rastreabilidade com cuidado, pois quanto maior o número de artefatos a rastrear e quanto maior o número de *links* a definir e manter maiores são a complexidade e custo para a implementação e gerenciamento da rastreabilidade. Além disso, para a maioria dos projetos pode-se obter até 80% dos benefícios da rastreabilidade rastreando até 20% dos *links* possíveis (Kotonya e Sommerville, 1998; Wiegers, 2003).

Durante a execução do teste a documentação sofrerá diversas modificações. Para manter os documentos atualizados e consistentes devem-se rastrear as diversas associações entre os vários artefatos que compõem a documentação; isto demanda controlar uma grande quantidade de dados. Assim, a existência de suporte automatizado adequado é fundamental. Este trabalho apresentou o protótipo de ferramenta PROMETEU, suas características, as decisões de implementação, sua arquitetura, o modelo de rastreabilidade utilizado, o modelo de dados

implementado para suportar a rastreabilidade, além de uma breve comparação entre suas funcionalidades e as de outras ferramentas de apoio ao teste quanto ao suporte à rastreabilidade e à documentação. O protótipo foi concebido e desenvolvido como parte de uma metodologia para a introdução ou melhoria do processo de teste em empresas de software; esta metodologia é fruto do trabalho do Grupo de Teste de Software do CTI e a utilização da Norma IEEE Std 829-1998 é uma das suas características.

Como o foco da PROMETEU é em usuários "*low-end*" (tipo de usuário de informações de rastreabilidade), ela foi projetada para tornar mais fácil seu aprendizado e a obtenção dos benefícios da rastreabilidade: os *links* de rastreabilidade, possibilitam ao usuário concentrar-se apenas no seu trabalho ao invés de se preocupar em definir e manter inúmeros *links* entre o grande número de artefatos de um projeto de teste.

Para tornar mais claros as funcionalidades da ferramenta e os potenciais benefícios advindos da sua utilização, no Apêndice A foi apresentado um exemplo de sua utilização.

Tendo em vista o contexto da sua origem, PROMETEU foi pensada e projetada para ser uma ferramenta de suporte à geração e manutenção da documentação somente de um processo de teste; entretanto, considerando as características da ferramenta, é possível perceber seu potencial para servir de suporte à documentação e rastreabilidade ao longo de todo o processo de software. Alguns tópicos na Seção "Trabalhos Futuros" tratam deste assunto.

A vivência como profissional na área de Qualidade de Software, aliada à experiência do Grupo de Teste do CTI ao longo dos serviços prestados e cursos ministrados bem como os depoimentos constantes em fóruns de discussão relacionados ao Teste de Software fornecem subsídios que corroboram a utilidade de uma ferramenta como a PROMETEU.

7.2 Contribuições

Algumas contribuições deste trabalho são:

- Proposição de um modelo de rastreabilidade para documentação de um processo de teste de software;
- Proposição de um modelo de dados para dar suporte à documentação e rastreabilidade em um processo de teste;
- Implementação de um protótipo de ferramenta para suporte à documentação do processo de teste e à rastreabilidade entre os artefatos que compõem os documentos,

que serve de base para estudos posteriores sobre o tema; esse protótipo também pode ser usado em sala de aula para auxiliar no ensino da área de Teste de Software.

7.3 Trabalhos Futuros

Com relação à continuidade deste trabalho, algumas ações possíveis são:

- Utilização em ambiente real de produção de software para coletar dados que fundamentem melhorias da usabilidade, das características e dos modelos da ferramenta;
- Aperfeiçoar a interface gráfica da ferramenta para, por exemplo, (a) facilitar a visualização dos vínculos de rastreabilidade, (b) facilitar a navegação entre os elementos de informação registrados;
- Implementar suporte a importação e exportação de dados usando os formatos de arquivo XML e CSV, para facilitar a troca de dados com outras ferramentas;
- Aumentar o nível de segurança relativo à confidencialidade dos dados, aperfeiçoando os mecanismos de segurança que controlam o acesso às funcionalidades da ferramenta e aos elementos de informação sob seu controle.

Em relação ao modelo de dados, algumas ações possíveis são:

- Implementar suporte a execução dos testes ao longo de vários ciclos;
- O modelo atual possibilita a geração parametrizada de novos documentos a partir dos elementos de informação dos documentos previamente modelados. No entanto, as informações de personalização não são armazenadas no banco de dados, sendo perdidas após a geração dos documentos. Devem ser criadas tabelas para o registro das escolhas dos usuários relativas às configurações dos documentos personalizados, possibilitando o reuso das estruturas de documentos montadas pelos usuários no momento da geração de documentos;
- Alterar o projeto para suportar o controle de versões para os artefatos da documentação. O controle de versões possibilitará, por exemplo, acompanhar a evolução das modificações em cada artefato;
- Registrar automaticamente as ações dos usuários; para isso é necessário criar tabelas para possibilitar esse registro, com benefícios que incluem: a) criar um *log* das ações

de cada usuário, para acompanhamento do Gerente de Teste; e b) oferecer ao usuário a opção de desfazer as modificações mais recentes nos documentos;

- Dadas as peculiaridades de cada tipo de documento, e o conjunto específico de operações sobre os elementos de informação de cada um deles, para cada tipo de documento foi modelado um conjunto de elementos de informação; entretanto, considerando que a facilidade de adaptação ao contexto do usuário é uma característica desejável, deve-se prever a possibilidade de criar campos personalizados para cada tipo de artefato modelado. Dessa forma o usuário pode incluir campos não previstos originalmente no projeto da ferramenta, mas que são importantes para o seu contexto de trabalho;
- Possibilitar o registro de dados semânticos associados aos *links* de rastreabilidade;
- Implementar um mecanismo de *keywords*: palavras-chave anexadas aos artefatos e que podem ser usadas para complementar sua descrição além de criar uma outra dimensão de rastreabilidade além daquelas pré-definidas na ferramenta;
- Modelar mais artefatos do processo de software, com as características exclusivas de cada fase, além dos artefatos atuais que foram modelados com foco no processo de teste. Com isso, será possível extrapolar o foco exclusivo no teste (a proposta inicial) e expandir o uso da ferramenta também para outras fases do desenvolvimento do software. Um exemplo claro são os artefatos dos requisitos, que podem sofrer melhorias como: a) modelagem de acordo com os modelos de Wiegers e do RUP, b) possibilitar o registro no formato puramente textual ou na forma de casos de uso (amplamente usado atualmente), c) modelagem do ciclo de vida de um requisito (recursos para gerência das modificações nos requisitos).

Em relação ao modelo de rastreabilidade, com foco exclusivo no processo de teste, as ações possíveis são:

- Modelar as estruturas de contribuição relacionadas a um processo de teste;
- Modelar a dimensão do processo, possibilitando rastrear atividades e tarefas e seus vínculos com a documentação;
- Modelar a semântica das diversas relações existentes entre os artefatos do teste.

Em relação ao modelo de rastreabilidade proposto e as outras fases do desenvolvimento de software, as ações possíveis são: considerar a criação de um modelo de rastreabilidade para cada

fase e, então, vincular esses modelos com o modelo do teste por diversos pontos de conexão (que podem ser os elementos de informação em comum entre as fases). Como sugestão, o primeiro modelo de rastreabilidade que pode ser desenvolvido é para a fase de Requisitos, até mesmo usando os modelos de rastreabilidade apresentados neste trabalho como ponto de partida. A opção pelo Modelo de Requisitos se baseia na intenção de acrescentar à ferramenta a capacidade de gerência de requisitos.

Apêndice A

Exemplo de uso da PROMETEU

As seções deste apêndice estão organizadas de forma a ilustrar um cenário de utilização do protótipo, de acordo com a versão de 1998 da norma IEEE 829. Será demonstrado como: a) acessar o sistema, b) registrar um projeto, c) registrar as justificativas para as decisões do projeto, d) cadastrar os atores, e) registrar os requisitos do software, f) registrar a arquitetura do software, g) registrar os códigos-fonte, h) registrar elementos de informação da documentação do teste, i) executar os testes e registrar incidentes, j) gerenciar os incidentes, l) gerenciar as associações de rastreabilidade, m) analisar o impacto de uma modificação, n) analisar as justificativas para decisões do projeto, o) consultar os elementos de informação da documentação, p) manter os elementos de informação da documentação, q) gerar documentação padronizada e personalizada, r) visualizar e imprimir relatórios, e s) analisar incidentes que ocorrem após a entrega.

A aplicação para a qual será criado um projeto de teste será o próprio protótipo.

A.1 Acessando o sistema

As funcionalidades do protótipo são acessadas através da tela “Login”, onde se digita um nome de usuário e senha válidos, seleciona-se um projeto e, por fim, seleciona-se o botão “Entrar”. Através da tela “Login” também é possível criar um novo projeto.

Para a primeira utilização da ferramenta, logo após sua instalação, existe um projeto previamente cadastrado chamado “Prometeu”, com usuário “prometeu”, senha “prometeu” e papel “Gerente de Teste”.

Na versão atual do protótipo é utilizado um controle rudimentar de acesso sobre a utilização de determinadas funções de acordo com o papel do usuário. Cada papel faz parte de uma

categoria, que pode ser Stakeholder ou Técnico. Os papéis pré-definidos para a categoria "Técnico" são:

- Gerente de Teste: possui acesso total aos recursos do sistema e é o responsável direto pelo cadastro dos requisitos a testar e dos atores em um projeto;
- Engenheiro de Teste: executa todas as funções de um Gerente de Teste, menos criação de projeto e criação de usuários;
- Testador: possui acesso aos casos de teste a ele atribuídos, ao registro de incidentes e a relatórios associados;
- Engenheiro da Qualidade: revisa casos de teste e incidentes a ele atribuídos para correção;
- Desenvolvedor: possui acesso somente aos incidentes que ele deve corrigir.

A demonstração descrita neste apêndice será baseada em um projeto chamado "Homologação da versão 0.99 do protótipo PROMETEU". Para esta demonstração assume-se a existência do usuário "jorge", cujo papel é "Gerente de Teste" (único papel que permite registrar um projeto); esse usuário criará o projeto e registrará os atores.

A.2 Registrando um projeto

Um novo projeto pode ser registrado a partir do menu Registro, opção "Projeto de Teste" ou através da janela "Login", botão "Criar Projeto". Também pode ser criado selecionando-se o botão "Criar Projeto" na tela "Login".

Na Seção "Informações Gerais" da janela "Projeto de Teste" (Figura A.1), registram-se os seguintes elementos de informação pertinentes ao projeto: um identificador (regra de formação é definida pela política da empresa), título, *status* (Ativo, Encerrado ou Cancelado), responsável, tempo previsto em horas, data de criação e da última atualização e uma breve descrição do projeto. Também é possível referenciar um arquivo externo (todo arquivo armazenado fora do banco de dados controlado pelo protótipo) para complementar as informações registradas.

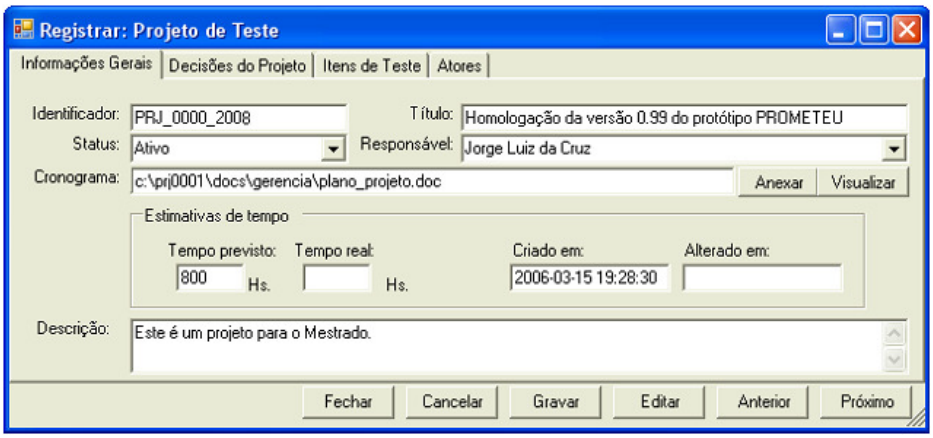

Figura A.1: Cadastrando um projeto de teste – Informações Gerais.

Todo projeto de teste possui um ou mais itens de teste associados. Eles podem ser registrados na janela "Projeto de Teste" a partir da seção "Itens de Teste", botão  . O botão  permite acessar o item de teste selecionado. O botão  permite associar ao projeto um ou mais itens de teste presentes na lista "Todos os itens de teste registrados". A Figura A.2 ilustra a seção "Itens de Teste" de um projeto.

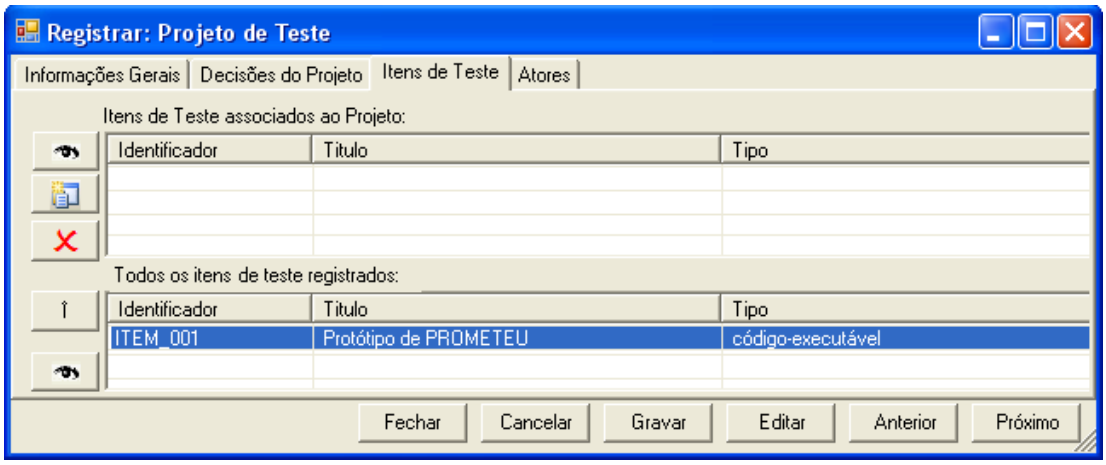

Figura A.2: Cadastrando um projeto de teste – Itens de Teste.

Justificativas para decisões significativas tomadas ao longo do projeto podem ser registradas na seção "Decisões do Projeto".

Os atores participantes de um projeto podem ser registrados a partir da seção "Atores", selecionando-se o botão  . O botão  permite acessar elementos de informação relativos ao ator selecionado.

Visão parcial da árvore de rastreabilidade do projeto

A Figura A.3 apresenta a estrutura inicial do projeto "Homologação da versão 0.99 do protótipo Prometeu", somente com a indicação dos tipos de artefatos, mas ainda sem nenhum artefato registrado. Essa estrutura padrão pode ser acessada através do menu Rastreabilidade, opção "Árvore".

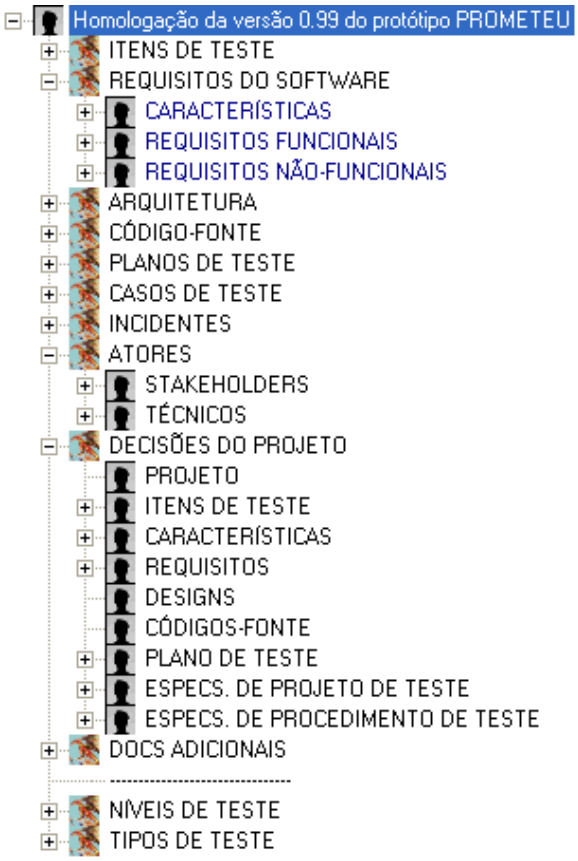

Figura A.3: Estrutura inicial dos artefatos de um projeto.

A.3 Registrando justificativas para as decisões do projeto

Justificativas para decisões significativas tomadas ao longo de um projeto podem ser registradas e atualizadas na seção "Decisões do Projeto", nos campos Motivação, Histórico e Lembretes. A Figura A.4 ilustra a interface para registro desses elementos de informação.

Os seguintes tipos de artefato possuem a seção "Decisões do Projeto": Projeto de Teste, Item de Teste, Característica, Requisito, *Design*, Código-fonte, Caso de Teste, Plano de Teste, Especificação de Projeto de Teste e Especificação de Procedimento de Teste.

Figura A.4: Cadastrando um projeto de teste – Decisões do Projeto.

As justificativas para as decisões do projeto podem ser rastreadas a partir do menu "Rastreabilidade", opção "Árvore", item "Decisões do Projeto".

A.4 Cadastrando os atores

Um projeto de teste possui diversos atores. Cada um deles pode criar, evoluir, corrigir ou consultar um artefato. Além disso, cada ator fornece informações importantes que devem ser rastreadas, como observações e justificativas que fundamentam a existência e evolução de um artefato.

Elementos de informação sobre os atores são registrados através do menu Registro, item "Informações Auxiliares", subitem "Atores", ou através da seção "Atores" de um projeto (janela "Projeto de Teste"). Os elementos de informação sobre cada ator são: nome, email, nome de usuário, senha de acesso, papel a desempenhar, descrição resumida, eventuais observações adicionais e, se necessário, é possível associar um arquivo externo com outros dados que sejam considerados importantes.

Para simplificar esta demonstração, não será ilustrado o registro dos atores; assume-se a existência dos seguintes usuários: elisa (testador), jorge (gerente de teste), miguel (gerente de teste), jino (gerente de teste), adalberto (testador), Sr. Prometeu (stakeholder), Sr. Big Foot (stakeholder) e Sr. Cli Ente (stakeholder). A Figura A.5 mostra uma visão parcial da árvore de rastreabilidade do projeto, com destaque para os atores registrados para esta demonstração e os papéis atribuídos a cada um deles.

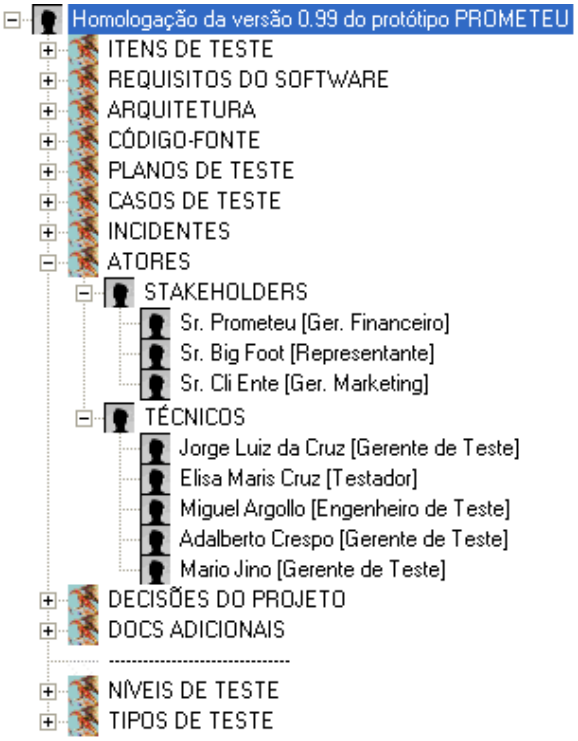

Figura A.5: Atores em um projeto de teste.

A.5 Registrando os requisitos do software

Após acessar o sistema, registrar um novo projeto e os atores, o próximo passo é registrar os requisitos que serão testados. Os requisitos do software são representados na ferramenta, no nível mais alto, como características; cada característica é detalhada por um ou mais requisitos do tipo Funcional ou Não-Funcional. Cada requisito pode ser detalhado por um ou mais requisitos, em um número máximo de três níveis.

Considera-se que os requisitos podem ser armazenados exclusivamente na ferramenta (suporte básico para registro e controle dos requisitos) ou em arquivos externos separados. Quando os requisitos forem registrados fora da ferramenta (p.ex.: em um arquivo para cada requisito ou em um único arquivo com todos os requisitos – isto depende dos procedimentos de cada empresa), no protótipo devem ser registrados, para cada requisito que será testado, um identificador único e elementos de informação suficientes para propiciar o planejamento do teste e o rastreamento entre os requisitos e outros artefatos do teste. Neste ponto, existe a necessidade de grande interação entre o responsável pelos requisitos (informará os identificadores dos

requisitos) e o responsável pelos testes (registrará elementos de informação sobre os requisitos na PROMETEU).

Características

A seguir são apresentadas as características que serão referenciadas na demonstração (e seus identificadores), além de um número reduzido de requisitos para algumas delas (o número é reduzido para simplificar a demonstração):

- Controle de acesso ao sistema (CARAC_01);
- Documentação de acordo com a norma IEEE Std 829-1998 (CARAC_02)
    - Permitir o registro dos elementos de informação relativos aos 8 tipos de documentos da norma IEEE Std 829-1998;
    - As seções dos documentos devem ser representadas na forma de "*tab pages*", padrão Microsoft Windows.
- Geração de documentação em formato padrão ou parametrizado (CARAC_03);
- Registro dos requisitos do software relacionados ao teste (CARAC_04)
    - Registro das características;
    - Registro dos requisitos funcionais e não-funcionais;
- Registro dos artefatos de *design* (desenho) relacionados ao teste (CARAC_05);
- Rastreabilidade entre artefatos que compõem a documentação (CARAC_06)
    - Permitir a manutenção das associações entre os artefatos de um projeto;
    - Permitir a visualização das associações na forma de matriz;
    - Permitir a visualização das associações na forma de árvore;
    - Permitir rastreabilidade direta entre requisitos e casos de teste;
    - Permitir navegabilidade rápida entre todos os artefatos associados, a partir do artefato atualmente selecionado.
- Teste baseado nos requisitos do software (CARAC_07);
- Gerenciamento da execução dos casos de teste (CARAC_08);
- Registro e controle dos incidentes de teste (CARAC_09);

- Suporte a preservação e compartilhamento do conhecimento (CARAC_10);
- Registro dos artefatos de código-fonte (CARAC_11).

Considerando os objetivos deste exemplo de utilização, será ilustrado o registro de somente uma das características da PROMETEU.

Para registrar uma característica, seleciona-se o menu Registro, item “Requisitos do Software”, subitem “Características”.

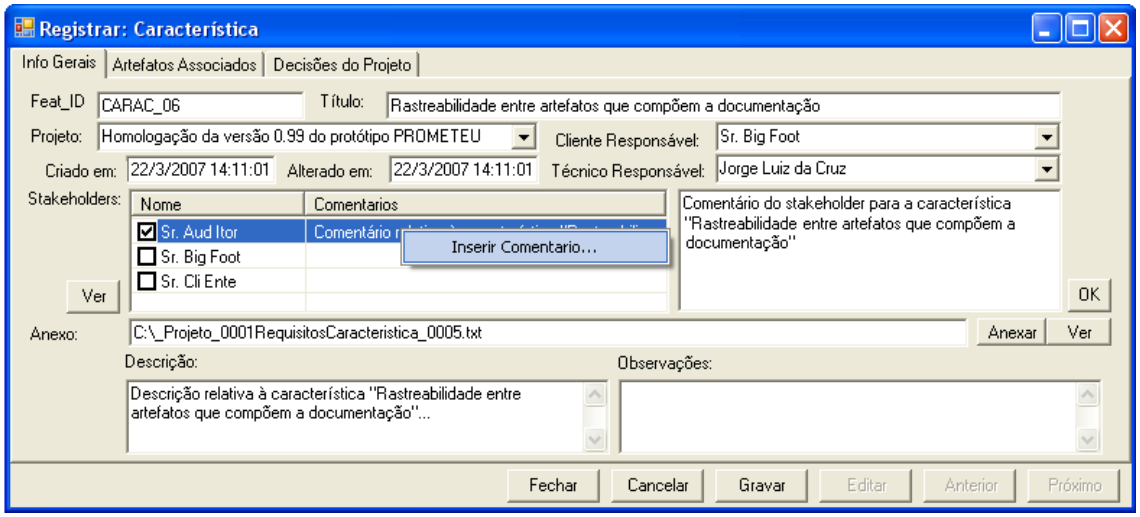

Figura A.6: Registro de uma Característica.

A Característica a ser registrada possui o título “Rastreabilidade entre artefatos que compõem a documentação”. Seu identificador será “CARAC_06” e ela será testada no projeto “Homologação da versão 0.99 do protótipo PROMETEU” (Figura A.6). O item de teste associado é “Protótipo de PROMETEU” (o registro de itens de testes será abordado na seção 3.5.2). O técnico responsável será ”Jorge Luiz da Cruz” e o responsável por parte do cliente será “Sr. Big Foot”. O Sr. “Audi Tor” será outro *stakeholder* associado à característica. Um clique com o botão direito do mouse sobre um ator presente na lista “Stakeholders” ativa o menu de contexto “Inserir Comentário...” que permite associar comentários de um *stakeholder* à característica sendo registrada. O botão “Ver”, à esquerda da lista de *stakeholders*, permite consultar informações sobre aquele que estiver selecionado.

Dados relativos a decisões significativas sobre o registro e evolução de uma característica podem ser registrados na seção “Decisões do Projeto”.

Para registrar as demais características citadas previamente na introdução deste apêndice, basta seguir os mesmos passos apresentados acima e registrar os elementos de informação

específicos de cada uma. A Figura A.7 mostra uma parte da janela “Árvore de Rastreabilidade”, onde são visualizadas as onze características registradas para o projeto.

Figura A.7: Características visualizadas na árvore de rastreabilidade.

Após o registro das características, os próximos artefatos a serem registrados são os requisitos funcionais e não-funcionais associados a elas.

Requisitos

Para registrar um requisito, seleciona-se o menu Registro, item “Requisitos do Software”, subitem “Requisito Funcional” ou “Requisito Não-Funcional”. Um requisito pode ser da categoria funcional ou não-funcional. As duas categorias possuem o mesmo conjunto de elementos de informação, mudando somente o valor do campo “Categoria”, que recebe o texto “Funcional” ou “Não-Funcional”, de acordo com o contexto.

Para esta demonstração serão registrados dois requisitos funcionais para a característica CARAC_06: “Permitir rastreabilidade direta entre requisitos e casos de teste” - REQ_F_001 e “Permitir navegabilidade rápida entre todos os artefatos associados, a partir do artefato atualmente” - REQ_F_002. A Figura A.8 mostra alguns dos elementos de informação registrados para o requisito REQ_F_001.

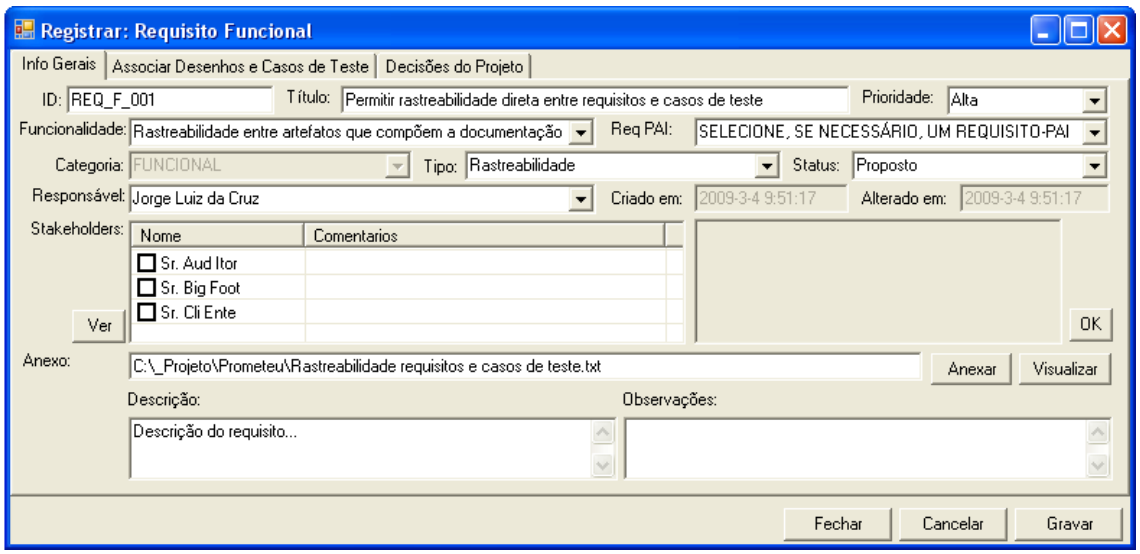

Figura A.8: Registro de um Requisito Funcional.

São registrados os seguintes elementos de informação sobre os requisitos: identificador, título, prioridade para o teste, característica e requisito aos quais está associado, categoria, tipo, *status*, responsável, datas de criação e alteração, *stakeholders* associados e suas observações, vínculo com um arquivo externo (para detalhamento de informações, se necessário), uma descrição e eventuais observações.

Na aba "Associar desenhos e casos de teste" é possível definir vínculos com estes tipos de artefatos. E na aba "Decisões do Projeto" é possível registrar dados relativos a justificativas para decisões tomadas ao longo do projeto que envolvam o requisito em questão

A.6 Registrando a arquitetura do software

Os artefatos da arquitetura do software podem ser registrados antes ou após o registro dos requisitos, uma vez que a associação entre estes tipos de artefatos pode ser estabelecida a qualquer momento no projeto. No exemplo de utilização descrito neste apêndice não será demonstrado o registro dos desenhos; assume-se que o artefato "Diagrama Entidade-Relacionamento" - DES_001, já foi registrado e associado aos requisitos REQ_F_001 e REQ_F_002. Além de DES_001, também o artefato "Diagrama de Classes" - DES_002 está cadastrado.

Considera-se que os artefatos da arquitetura são gerados e armazenados em arquivos externos; na ferramenta são registrados somente os elementos de informação que possam auxiliar as atividades de planejamento e desenho dos artefatos do teste.

A.7 Registrando os códigos-fonte

Os códigos-fonte do software podem ser registrados antes ou após o registro dos *designs*, uma vez que a associação entre estes tipos de artefatos pode ser estabelecida a qualquer momento no projeto. No exemplo de utilização descrito neste apêndice não será demonstrado o registro dos códigos-fonte; assume-se que os artefatos "frmIncidente.cs", "frmPartesEnvolvidas.cs" e "frmConsCasosdeTeste.cs" já foram registrados e associados ao *design* DES_002.

Os códigos-fonte são gerados externamente à ferramenta; no protótipo são registrados somente os elementos de informação que possam auxiliar as atividades de planejamento e projeto dos artefatos do teste.

Visão parcial da árvore de rastreabilidade do projeto

A Figura A.9 mostra uma visão parcial da árvore de rastreabilidade do projeto, considerando todos os artefatos registrados até o momento na demonstração. À esquerda percebe-se que o requisito funcional "Permitir rastreabilidade direta entre requisitos e casos de teste" está selecionado. O requisito pode ser visualizado como filho da característica CARAC_06 e, também, no nó "Requisitos Funcionais". Após o requisito ser selecionado pode-se ver à direita, na parte superior, informações resumidas a seu respeito e, na parte inferior, uma árvore que mostra as associações atuais entre o requisito selecionado e os demais artefatos do projeto: neste momento ele está associado à característica CARAC_06, ao desenho "Diagrama Entidade-Relacionamento", não é referenciado por nenhuma documentação do teste, e é possível ver que o *stakeholder* (Sr. Prometeu) e o técnico (Jorge Luiz da Cruz) estão associados a ele; na árvore de rastreabilidade também é mostrado o papel específico de cada participante do projeto: Sr. Prometeu (Gerente Financeiro do cliente) e Jorge (Gerente de Teste).

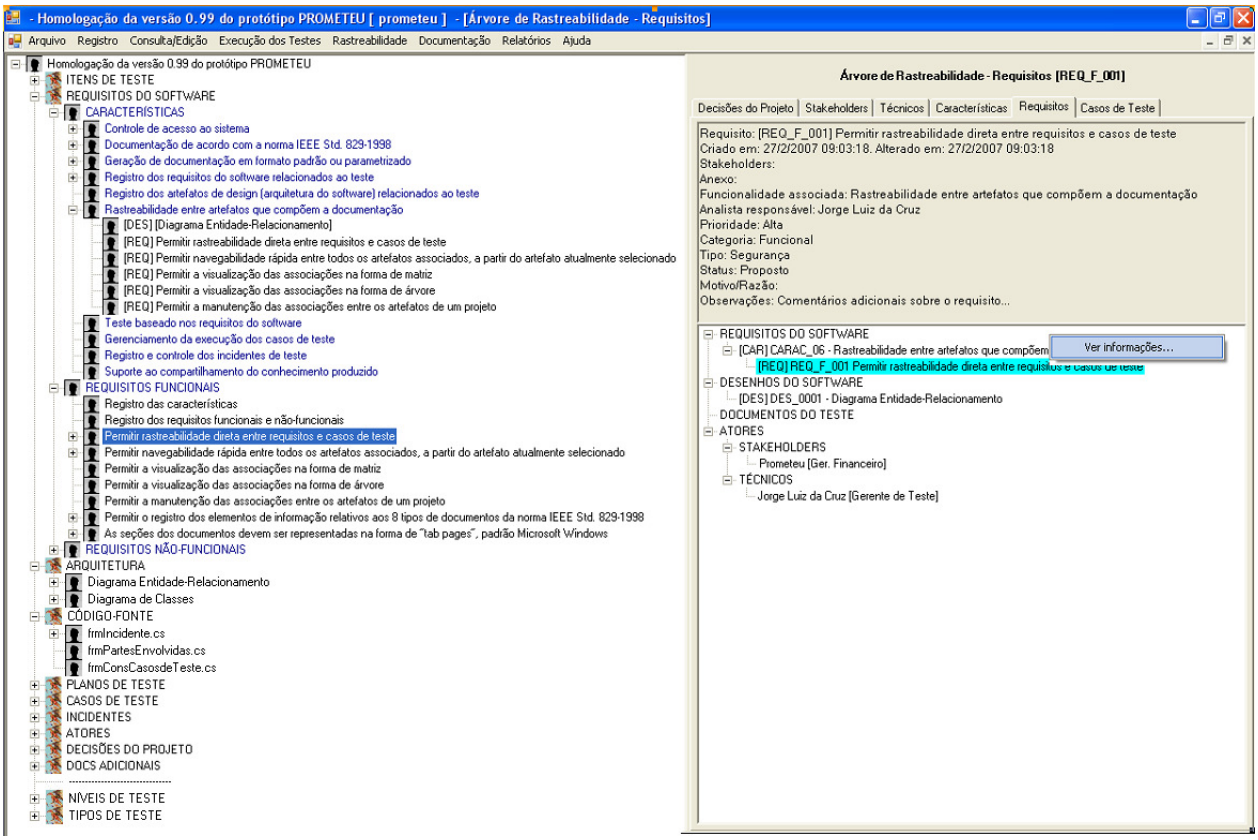

Figura A.9: Árvore de rastreabilidade dos requisitos do software.

Para cada artefato nas árvores de rastreabilidade existe o menu de contexto “Ver informações...”, através do qual é possível acessar os elementos de informação do artefato.

A.8 Registrando elementos de informação da documentação do Teste

Os elementos de informação da documentação do teste são baseados na norma IEEE Std 829-1998 e são registrados a partir do menu “Registro”, nas opções: “Item de Teste”, “Plano de Teste”, “Especificação de Projeto de Teste”, “Especificação de Procedimento de Teste”, “Especificação de Caso de Teste”, “Incidente de Teste” e “Diário de Teste”.

Cada documento é composto por um conjunto de seções, tendo por base as seções dos documentos prescritos pela norma. A interface gráfica para registro dos elementos de informação de cada documento é composta de formulários com diversas folhas (*tab pages*), onde cada uma representa uma seção do documento. Cada seção pode ser preenchida com um ou mais elementos de informação, que podem ser estáticos (texto digitado pelo usuário, sem vinculação direta com

outros tipos de documentos) ou dinâmicos (elementos de informação vinculados a seções de outros artefatos controlados através do protótipo).

Além da documentação proposta pela norma IEEE Std 829-1998, existem diversos tipos de outros documentos que são criados em um projeto e que podem ser referenciados nos documentos do teste; a própria norma faz referência a estes documentos, que neste trabalho serão tratados como “documentação adicional”.

Documentação adicional

Uma vez que não é possível definir com antecedência a estrutura dos documentos que compõem a documentação adicional, PROMETEU possibilita registrar informações genéricas sobre os mesmos, de maneira que seja possível referenciá-los nos documentos gerenciados pelo protótipo.

Para registrar elementos de informação sobre um documento adicional, seleciona-se o menu Registro, item “Informações Auxiliares”, subitem “Documentação Adicional”. Os campos existentes são: título, tipo, responsável, localização e descrição.

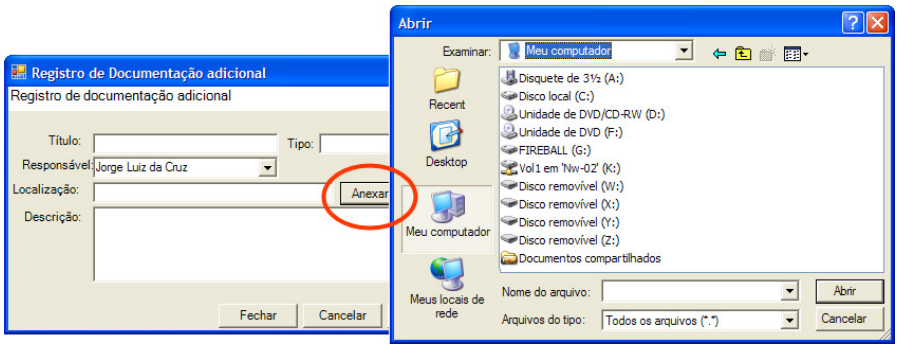

Figura A.10: Registro de documentação adicional.

A Figura A.10 ilustra a interface para registro de elementos de informação relativos a um documento adicional. Para esta demonstração será considerado que os seguintes documentos adicionais foram registrados: “Manual de Usuário do Protótipo PROMETEU”, “Manual de Instalação do Protótipo PROMETEU” e “Manual de Operação do Protótipo PROMETEU”. Esses documentos são listados na seção “Informações Gerais” de um Plano de Teste.

Itens de teste

Prosseguindo com o exemplo de utilização da ferramenta, devem ser registrados os itens de teste. Pelo menos um item de teste deve ser registrado e associado a um projeto, uma vez que toda característica está associada a um item de teste. Para o projeto "Homologação da versão 0.99 do protótipo PROMETEU", um item de teste será registrado: o código executável do protótipo.

Na Figura A.11 vê-se a interface para o preenchimento dos dados que caracterizam o Item de Teste. O campo "Localização" pode ser o caminho de um arquivo em disco, na rede ou em um repositório de arquivos.

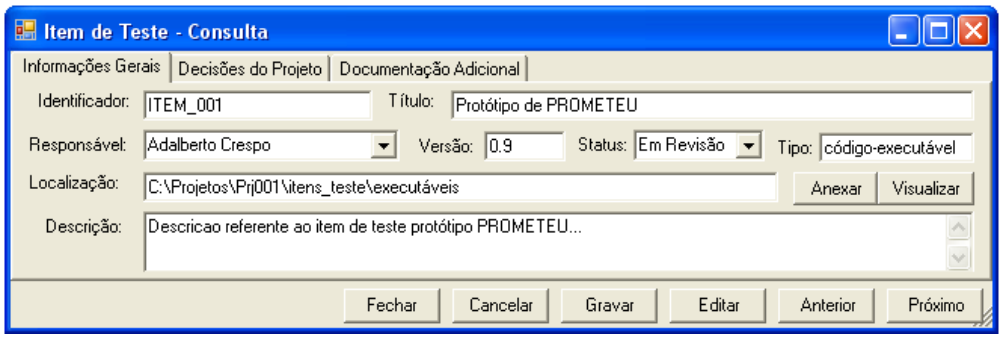

Figura A.11: Item de Teste – Informações Gerais.

Itens de teste também possuem uma seção "Decisões do Projeto", para possibilitar o registro de informações sobre sua utilização ao longo do projeto. Além disso, é possível associar uma documentação adicional a um item de teste, através da janela de registro de um item de teste, como pode ser visto na Figura A.12.

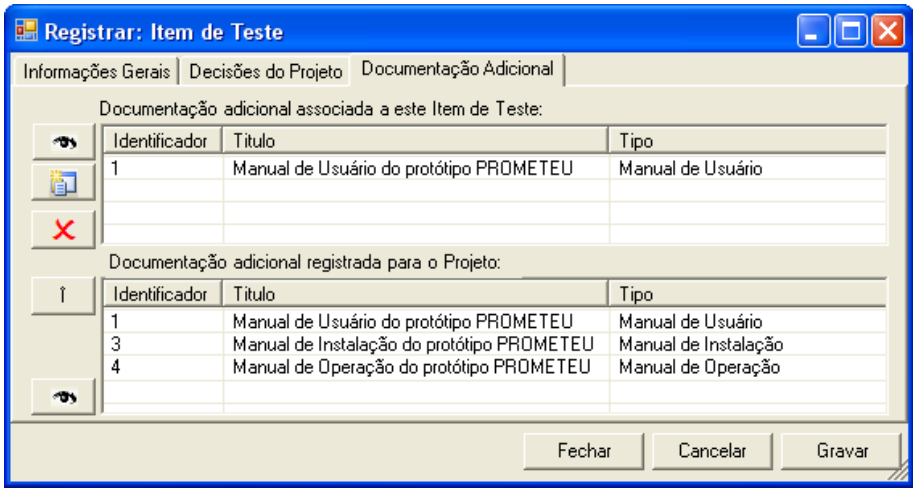

Figura A.12: Item de Teste – Documentação Adicional.

O botão  permite visualizar um documento selecionado. O botão  permite registrar informações sobre um novo documento, o botão  permite adicionar um documento à lista de documentos associados ao item de teste. O botão  possibilita remover um documento da lista de documentos associados.

Para esta demonstração assume-se que já tenham sido registradas informações relativas aos três documentos que compõem a documentação adicional do projeto: “Manual de Usuário de PROMETEU”, “Manual de Instalação de PROMETEU” e “Manual de Operação de PROMETEU”.

Especificação de Caso de Teste

Após o registro dos requisitos e desenhos, registram-se os casos de teste e suas associações com os requisitos. No momento do registro das especificações de projeto e de procedimento de teste, a ferramenta utiliza os vínculos já existentes entre requisitos e casos de teste; assim, se sugere definir essas associações antes de registrar as especificações de projeto e de procedimento de teste.

Para esta demonstração serão registrados seis casos de teste para o requisito REQ_F_002 (“Permitir navegabilidade rápida entre todos os artefatos associados, a partir do artefato atualmente selecionado”): “Navegar de um caso de teste a um requisito associado, na consulta” – CT_001, “Navegar de um caso de teste a um item de teste associado, na consulta” – CT_002, “Navegar de um requisito a um *stakeholder* associado, na consulta” – CT_003, “Navegar de um requisito a uma característica associada, na consulta” – CT_004, “Navegar de um requisito a um requisito associado, na consulta” – CT_005 e “Navegar de um requisito a um *design* associado, na consulta” – CT_006. Além desses seis casos de teste, um sétimo será registrado, mas sem ser associado a algum requisito: “Navegar de um incidente a um artefato associado” – CT_007.

As Figuras A.13, A.14 e A.15 ilustram detalhes dos dados registrados para o caso de teste “Navegar de um caso de teste a um requisito associado, na consulta”.

Na seção “Informações Gerais” (Figura A.13) são registrados os seguintes elementos de informação: identificador único, título, nível, tipo, criador e testador alocado para executar o caso de teste, um arquivo anexo (referência a um arquivo externo para complementar as informações relativas ao caso de teste), objetivo do caso de teste, eventuais observações, além de dados sobre estimativas de tempo. O item “Status” é atualizado em função da execução do caso de teste (ver

Seção 6.8 – Executando os testes e registrando incidentes). Não são indicados nem a técnica e nem o critério de teste utilizados no *design* de cada caso de teste.

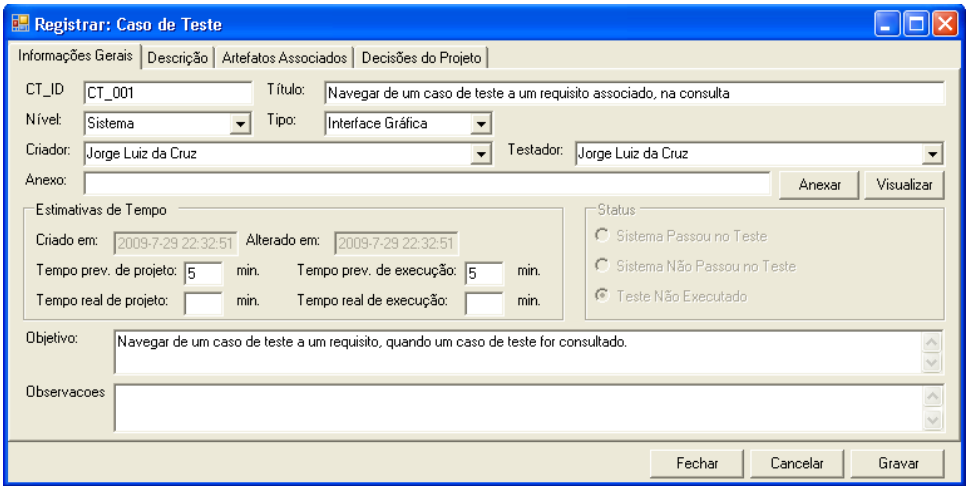

Figura A.13: Especificação de Caso de Teste – Informações Gerais.

Na seção "Descrição" (Figura A.14) são registradas as condições gerais para execução do caso de teste: os dados de entrada, passos a executar, resultados esperados e resultados obtidos na execução.

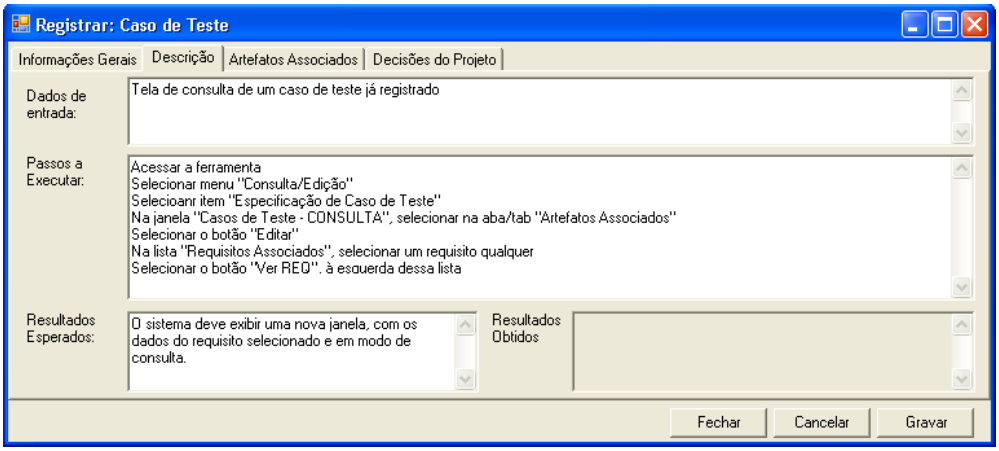

Figura A.14: Descrição dos passos para execução do caso de teste.

Na seção "Artefatos Associados" o caso de teste pode ser vinculado a um ou mais requisitos, *designs* e códigos-fonte. A Figura A.15 mostra que o caso de teste CT_001 será associado ao requisito REQ_F_002. Qualquer requisito da lista pode ser acessado a partir do botão "Ver REQ" e qualquer item de teste pode ser acessado a partir do botão "Ver ITEM". Para registrar os demais casos de teste, CT_002, CT_003, CT_004, CT_005, CT_006 e CT_007, basta

seguir os mesmos passos apresentados anteriormente e registrar os elementos de informação específicos de cada um.

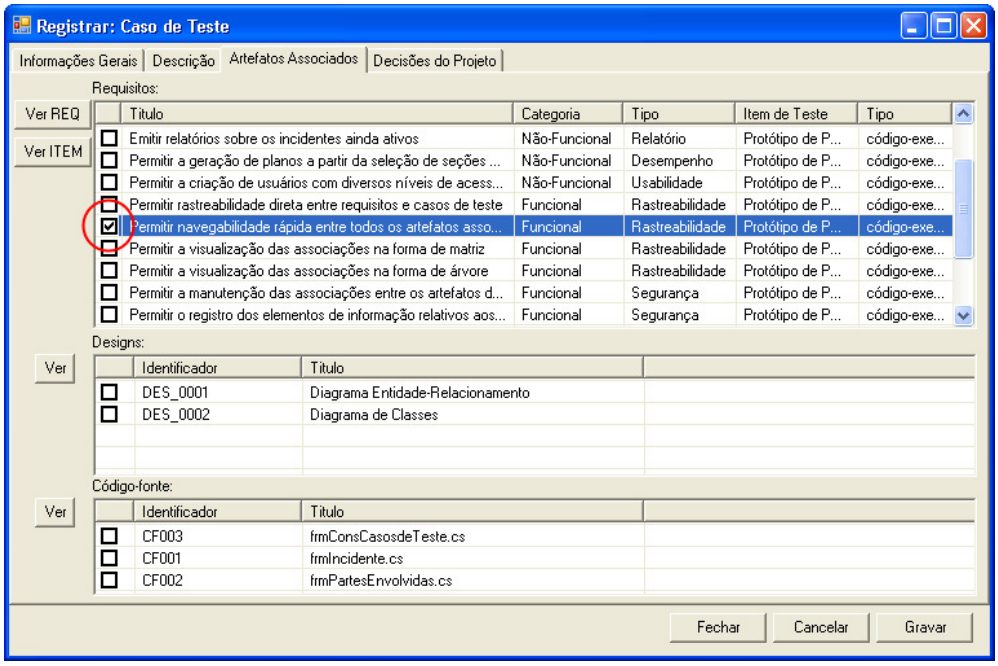

Figura A.15: Associação com requisitos e desenhos.

Figura A.16: Árvore de rastreabilidade do projeto.

A Figura A.16 mostra uma parte da árvore de rastreabilidade do projeto, parcialmente expandida, após o registro dos sete casos de teste supracitados.

Plano de Teste

Uma vez registrados os elementos de informação relativos ao projeto, atores, itens de teste, requisitos, arquitetura, casos de teste e documentação adicional, pode-se registrar o Plano de Teste.

Para cria um plano de teste, seleciona-se o menu “Registro”, item “Plano de Teste”. No exemplo em curso, o Plano de Teste será criado para testar a característica CARAC_06.

A Figura A.17 ilustra a seção “Informações Gerais”: é possível visualizar identificador do plano (PLANO_M01), o projeto ao qual está associado, seu *status* (concluído; em revisão; em criação), o responsável pelo documento e o revisor do documento. É possível registrar uma descrição e algumas observações, se necessário. Os campos “Tempo previsto” e “Tempo real” permitem rastrear desvios na estimativa de tempo para criação do documento. E a lista “Documentação adicional associada” permite associar fontes adicionais de informação ao documento.

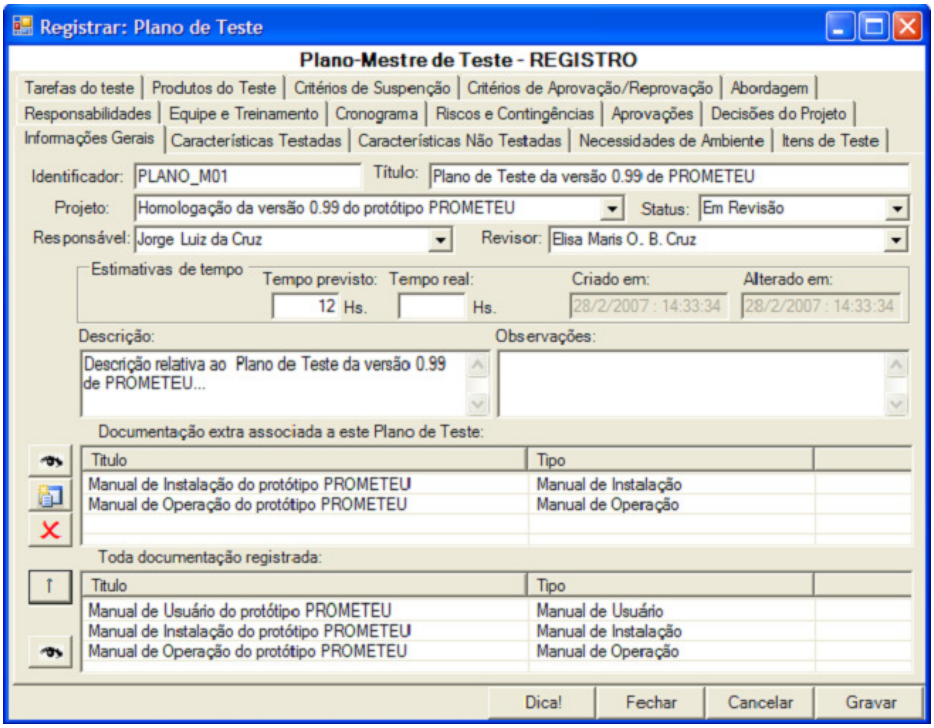

Figura A.17: Plano de Teste – Informações Gerais.

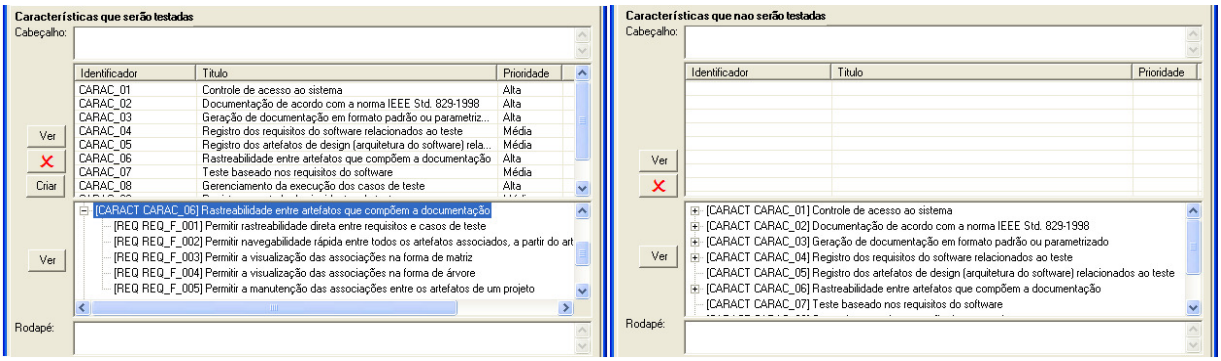

Figura A.18: Plano de Teste – Características Testadas e Não Testadas.

A Figura A.18 mostra as seções “Características que Serão Testadas” e “Características que Não Serão Testadas”. Na árvore abaixo da lista de características que serão testadas, e da lista de características que não serão testadas, existe uma árvore com todas as características registradas no projeto e que podem ser associadas ao documento. Também é possível visualizar todos os requisitos associados às características. O botão “Ver” permite navegar até o artefato selecionado na lista ou na árvore.

No momento da criação de um Plano de Teste, todas as Características registradas na ferramenta são listadas na seção “Características que serão testadas”. Para definir quais características serão testadas de acordo com o plano em criação, basta selecionar as características que não serão testadas e clicar no botão  para removê-la da lista de características a testar. Quando uma característica é removida da lista de características a serem testadas, automaticamente ela será listada na seção “Características que não serão testadas”, e vice-versa, como pode ser visto na Figura A.19.

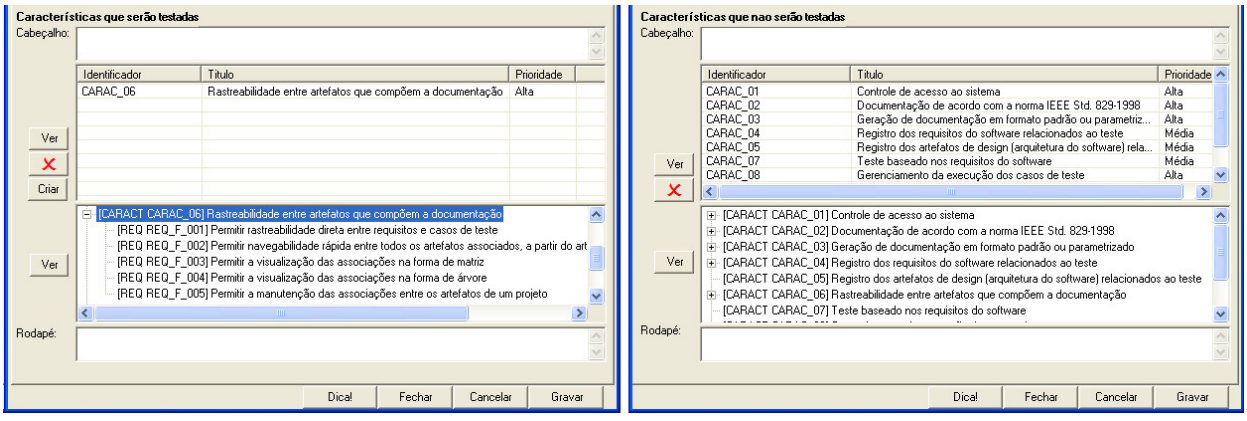

Figura A.19: Plano de Teste – Características Testadas e Não Testadas.

O botão “Criar” possibilita incluir uma nova característica ao projeto de teste. O botão “Ver” serve para visualizar uma característica e o botão  permite remover uma característica da lista. Os campos “Cabeçalho” e “Rodapé” são campos para texto de conteúdo estático (não possuem relação com outros documentos), que podem ser usados para preencher um texto introdutório e um texto de conclusão para uma seção.

A Figura A.20 mostra parte da árvore de rastreabilidade do projeto, após a criação do Plano de Teste. Pode-se perceber que ainda não existe nenhuma Especificação de Projeto de Teste associada ao Plano de Teste e, conseqüentemente, nenhuma Especificação de Procedimento de Teste registrada.

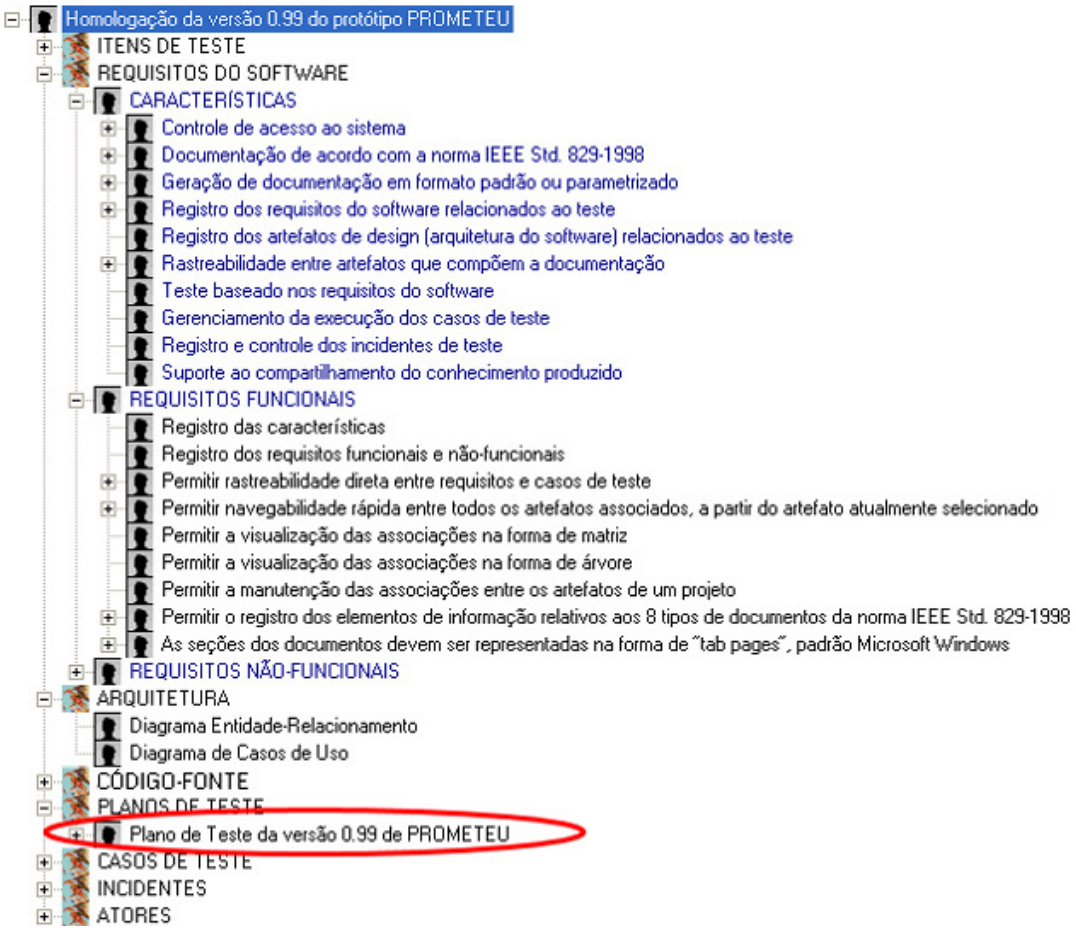

Figura A.20: Árvore de Rastreabilidade - Plano de Teste.

Especificação de Projeto de Teste

Após o registro de um Plano de Teste, são criadas as especificações de projeto de teste e as especificações de procedimento de teste. No Plano de Teste PLANO_M01 a característica a ser

testada é “Rastreabilidade entre artefatos que compõem a documentação” - CARAC_06. Ela possui cinco requisitos associados: REQ_F_001, REQ_F_002, REQ_F_003, REQ_F_004 e REQ_F_005.

Para refinar o planejamento do teste de CARAC_06, definida em PLANO_M01, serão criados dois documentos do tipo “Especificação de Projeto de Teste”: EPROJ_001 para o requisito REQ_F_002 e EPROJ_002 para o requisito REQ_F_003.

A Figura A.21 mostra a Seção “Informações Gerais’ de EPROJ_001: percebe-se que o documento está associado ao plano de teste PLANO_M01, que pode ser acessado clicando-se no botão . Além do identificador e título da especificação de projeto, são definidos o *status* do documento, o responsável e o revisor. Pode ser registrada uma estimativa de tempo para criação do documento, uma descrição e eventuais observações. Todo documento “Especificação de Projeto de Teste” também possui uma seção “Decisões do Projeto”.

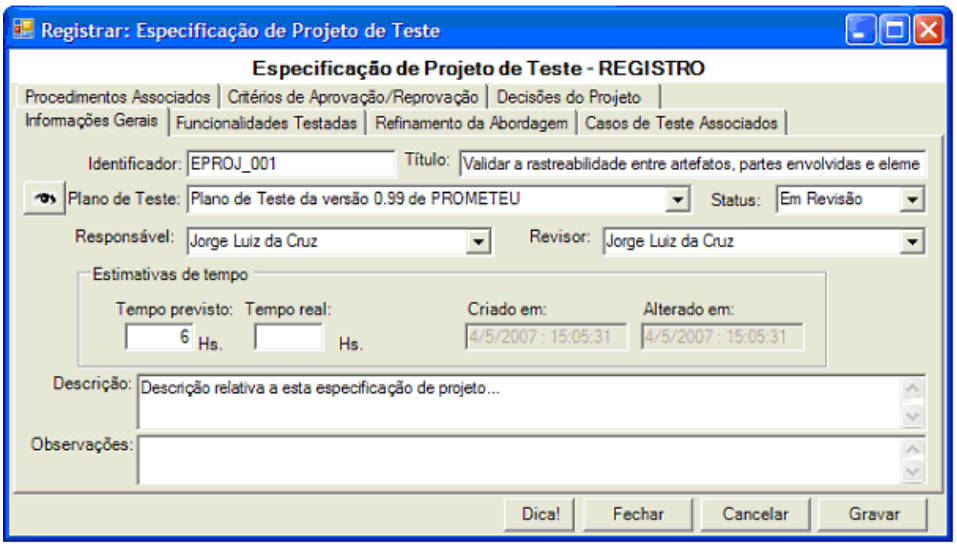

Figura A.21: Especificação de Projeto de Teste - Informações Gerais.

O requisito “Permitir navegabilidade rápida entre todos os artefatos associados, a partir do artefato atualmente”, REQ_F_002, está selecionado e definido como requisito a ser testado (Figura A.22). O botão “Ver” possibilita visualizar a característica ou o requisito selecionado. E os itens de teste associados à característica e aos requisitos, listados em “Item de Teste associados”, podem ser visualizados clicando-se no botão .

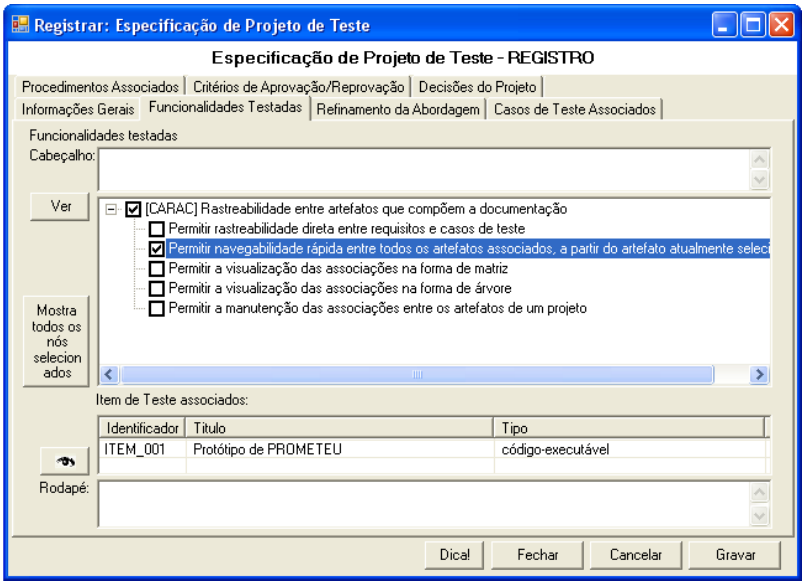

Figura A.22: Especificação de Projeto de Teste – Funcionalidades testadas.

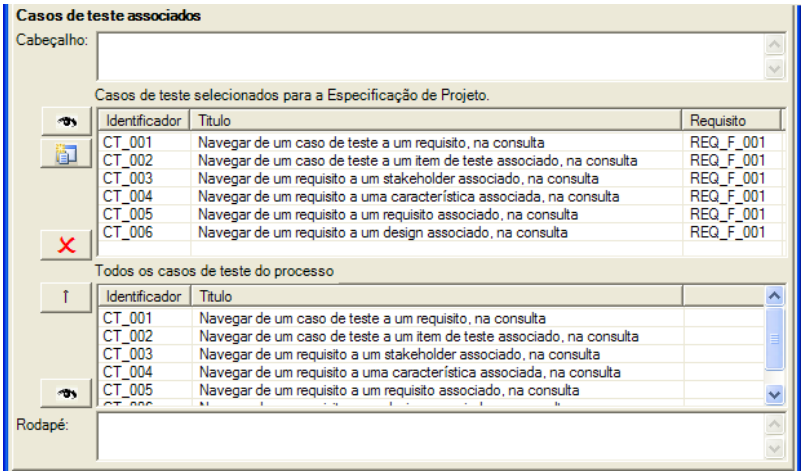

Figura A.23: Especificação de Projeto de Teste – Casos de Teste Associados.

Quando um requisito é selecionado, automaticamente os casos de teste associados ao requisito (neste caso, REQ_F_002) são inseridos na seção “Casos de Teste Associados” da especificação de projeto de teste em criação (Figura A.23); para o documento EPROJ_001 serão utilizados os casos de teste CT_001, CT_002, CT_003 e CT_004, CT_005 e CT_006.

Para definir quais casos de teste serão associados à EPROJ_001, basta selecionar os casos de teste que não farão parte da especificação e clicar no botão  para removê-los da lista de casos de teste associados à especificação de projeto. O botão  possibilita registrar um novo caso de teste. O botão  serve para visualizar um caso de teste que esteja selecionado na lista.

Para registrar a segunda especificação de projeto de teste, cujo identificador será EPROJ_002, basta seguir os passos descritos anteriormente nesta seção; assume-se que nesta demonstração não serão criados casos de teste para EPROJ_002.

Especificação de Procedimento de Teste

Especificações de Procedimento de Teste definem os passos para a execução de um conjunto de casos de teste relacionados a uma Especificação de Projeto de Teste. Considerando a demonstração em curso, serão criadas duas especificações de procedimento de teste que serão associadas à especificação de projeto EPROJ_001; o foco serão os testes relativos ao requisito REQ_F_002 ("Permitir navegabilidade rápida entre todos os artefatos associados, a partir do artefato atualmente").

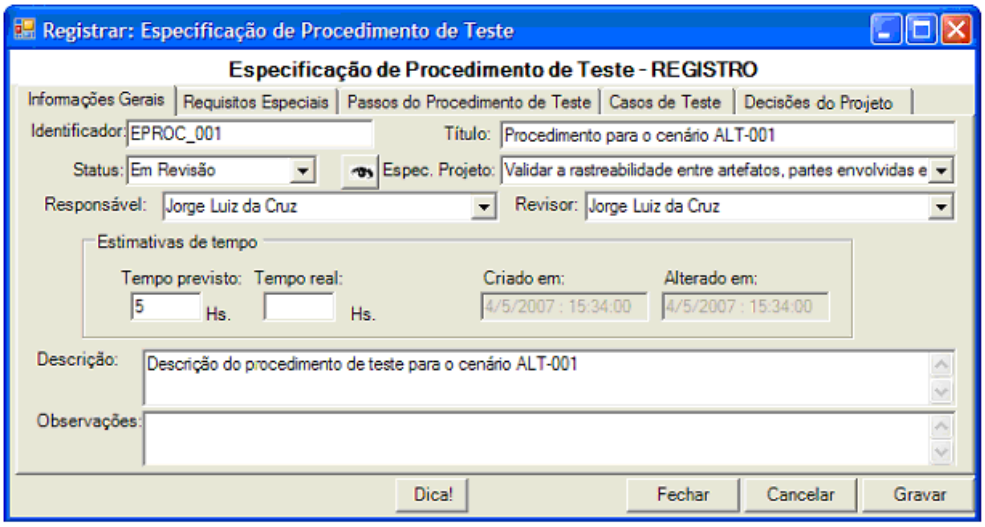

Figura A.24: Especificação de Procedimento de Teste – Informações Gerais.

A Figura A.24 mostra a seção "Informações Gerais" do procedimento em criação, onde são definidos seu identificador (neste caso "EPROC_001"), um título, a especificação de projeto (que pode ser visualizada clicando-se no botão) a qual está associado (EPROJ_001), seu *status*, os atores, estimativas de tempo, uma breve descrição e, se necessário, algumas observações.

Na Figura A.25, a lista "Casos de teste selecionados para o procedimento" mostra os casos de teste associados ao procedimento em criação, "EPROC_001": CT_005, CT_006, CT_004, CT_003. Os casos de teste devem ser executados na ordem em que aparecem na lista.

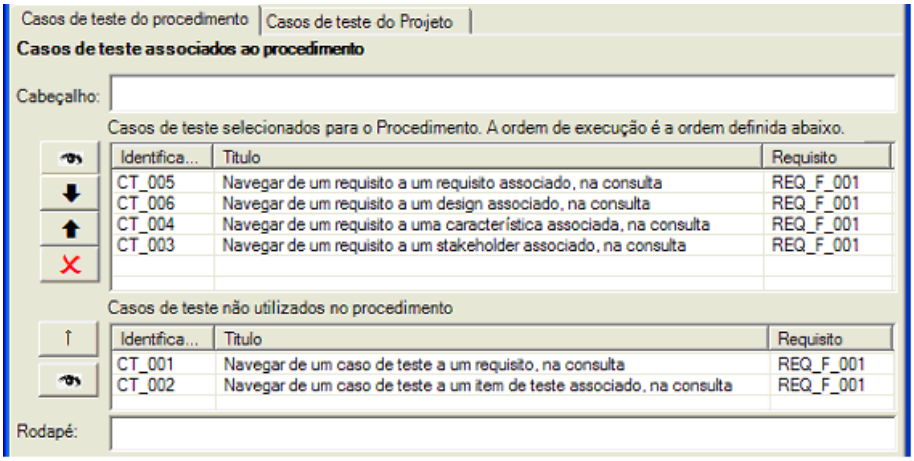

Figura A.25: Casos de teste associados a um procedimento.

Os botões  e  possibilitam ordenar os casos de teste. O botão  exclui um caso de teste da lista de casos de teste do procedimento e o botão  inclui um caso de teste. A lista "Casos de teste não utilizados no procedimento" mostra o restante dos casos de teste associados aos requisitos referenciados na especificação de projeto à qual o procedimento de teste está vinculado. A aba "Casos de teste do projeto" lista todos os casos de teste registrados no projeto. Os casos de teste CT_001 e CT_002, não utilizados em "EPROC_001", serão utilizados na Especificação de Procedimento de Teste "EPROC_002", que não será detalhada nesta demonstração.

Diário de Teste

Nesta demonstração não será ilustrada a utilização de documentos do tipo "Diário de Teste".

Visão parcial da árvore de rastreabilidade do projeto

A Figura A.26 mostra grande parte dos artefatos citados até aqui (o conteúdo foi resumido para que coubesse em uma página).

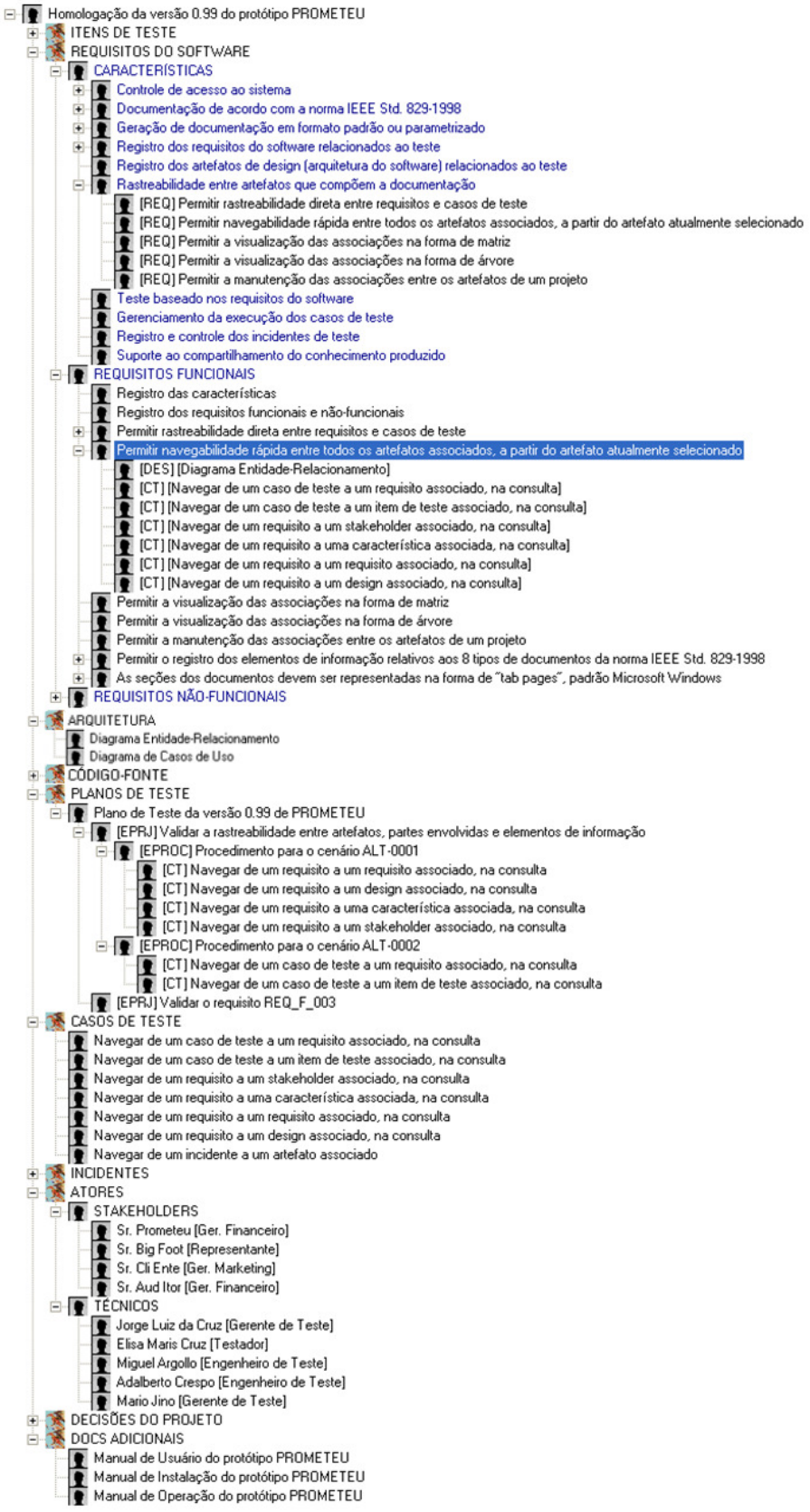

Figura A.26: Árvore de rastreabilidade do projeto.

Os nós de nível mais alto da árvore apresentam os principais tipos de artefatos controlados pelo protótipo; abaixo de cada um dos nós da árvore é possível visualizar todos os artefatos registrados para o projeto atual.

A.9 Executando os testes e registrando incidentes

Para registrar a execução dos casos de testes, seleciona-se o menu “Execução dos Testes”, item “Procedimentos de Teste”. O sistema exibe todos os procedimentos de teste atribuídos ao usuário que estiver utilizando a ferramenta. A Figura A.27 mostra os casos de teste dos procedimentos “EPROC_001” e “EPROC_002”, atribuídos ao usuário “jorge”. Como pode ser visto na coluna “status”, nenhum dos casos de teste foi executado até o momento.

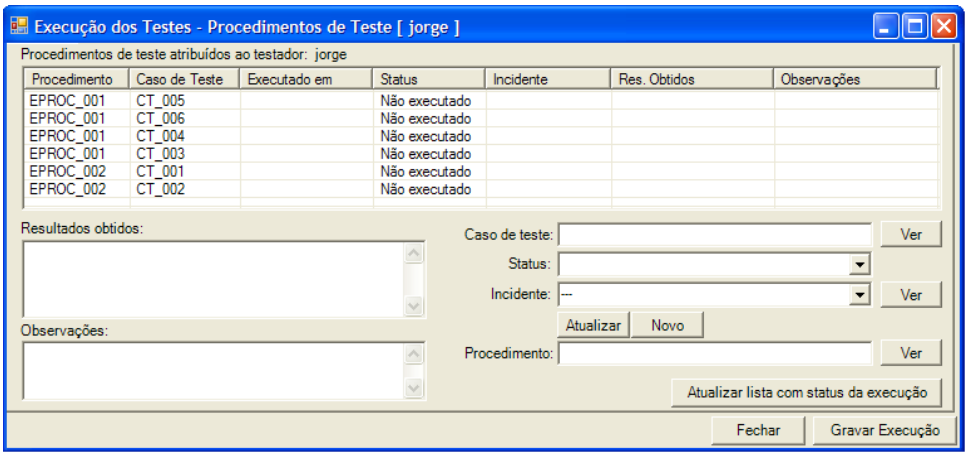

Figura A.27: Execução dos testes.

O próximo passo é executar os casos de teste, definir o *status* da execução e registrar os dados necessários em decorrência da execução do caso de teste. Para executar um caso de teste o testador pode consultar seus dados (informações do caso de teste e do procedimento) diretamente na ferramenta ou em um relatório impresso.

Após a execução de um caso de teste deve-se registrar o resultado obtido (se o sistema passou ou falhou no teste) e eventuais observações decorrentes da execução dos testes. Para esta demonstração, considera-se que os casos de teste CT_005, CT_006, CT_004 e CT_003, do procedimento “EPROC_001”, foram executados e o sistema passou em todos. Já na execução do caso de teste CT_001, do procedimento “EPROC_002”, assume-se que foi revelado um defeito.

Quando a execução de um caso de teste revela um defeito, faz-se um registro em um documento do tipo “Registro de Incidente de Teste”.

Para registrar a falha do sistema na execução de CT_001, o caso de teste deve ser selecionado e, em seguida, o campo “Status” deve ser alterado para “Sistema Falhou”. O resultado obtido é registrado, assim como eventuais observações, caso necessário. Após isso, clica-se no botão “Novo”, abaixo do campo “Incidente”, para registrar um novo incidente de teste (Figura A.28).

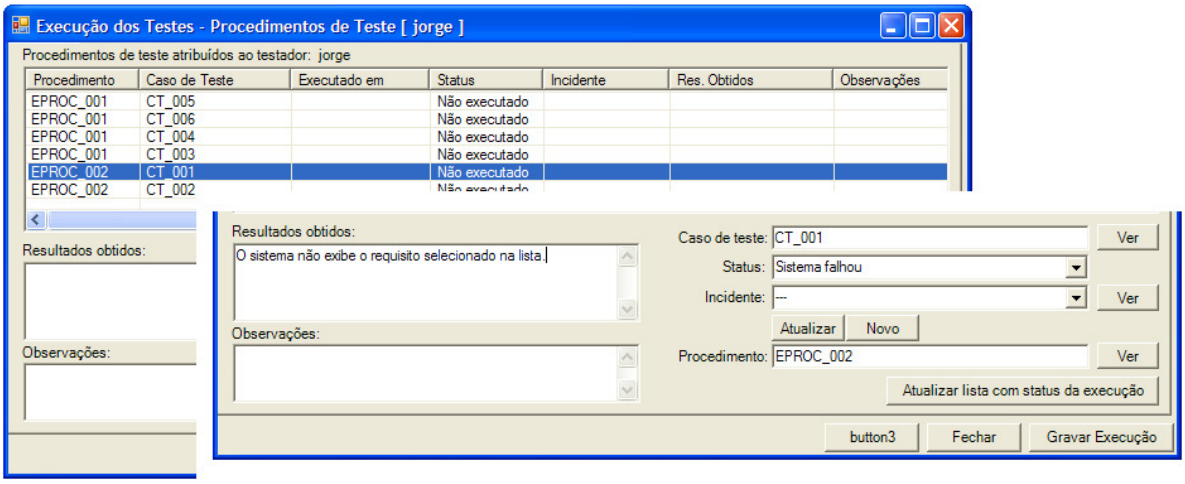

Figura A.28: Registro de um Incidente a partir da execução de um Caso de Teste.

Registrar um incidente de teste

Cada incidente possui um identificador, um título, o identificador do caso de teste que o revelou, um tipo (defeito; incidente; melhoria), um grau de severidade (Nível 1, Nível 2, Nível 3, Nível 4 e Nível 5 – o significado desses níveis é definido pela política de teste da empresa), o *status* atual (Em aprovação, Em conserto ou Corrigido), uma descrição, passos a serem executados para reproduzir o incidente, eventuais observações e a possibilidade de referenciar um arquivo externo à ferramenta.

Na Figura A.29, janela “Registrar Incidente de Teste”, é possível visualizar os elementos de informação que foram registrados relativos ao incidente originado a partir da execução de CT_001.

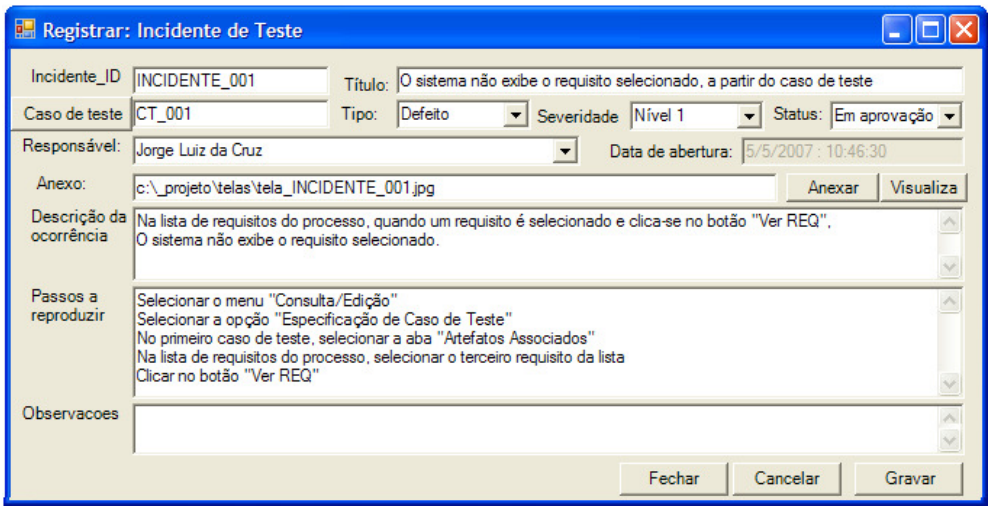

Figura A.29: Registro de um Incidente de Teste.

A.10 Gerenciando os incidentes de teste

A ferramenta possibilita listar todos os incidentes e *status* associado, além de possibilitar o rastreio dos artefatos associados ao incidente, o que facilita a correção dos motivos que ocasionaram os incidentes.

Listar os incidentes do projeto

Todos os incidentes do projeto podem ser listados a partir do menu “Rastreabilidade”, item “Incidentes”. A partir da janela “Lista de Incidentes” é possível descobrir e acessar todos os artefatos relacionados a um incidente.

Identificação dos artefatos associados a um defeito

Artefatos associados a um incidente são fontes de informações que podem contribuir para descobrir sua causa e facilitar sua correção. Além disso, todos esses artefatos são associados a atores do projeto, que também são fontes de informações. A partir de cada incidente, pode-se navegar rapidamente até os artefatos associados e consultá-los e, a partir de cada artefato, é possível consultar dados sobre os atores relacionados à criação e manutenção de cada artefato associado ao incidente.

É possível descobrir os artefatos associados a um incidente na operação de consulta aos dados do incidente ou a partir do menu “Rastreabilidade”, item “Incidentes”. O incidente “INCIDENTE_001” será utilizado como exemplo.

Para visualizar os artefatos associados ao incidente, seleciona-se o menu “Consulta/Edição”, item “Incidentes”. Na janela “Incidentes – Consulta”, clica-se no botão “Artefatos associados...”, que ativa a janela “Incidente – Artefatos afetados” (Figura A.30), na qual os artefatos associados são mostrados na forma de matriz. Uma vez descobertos os artefatos associados a um incidente, para descobrir os atores relacionados basta navegar até cada um dos artefatos e acessar a lista de atores vinculados ao artefato.

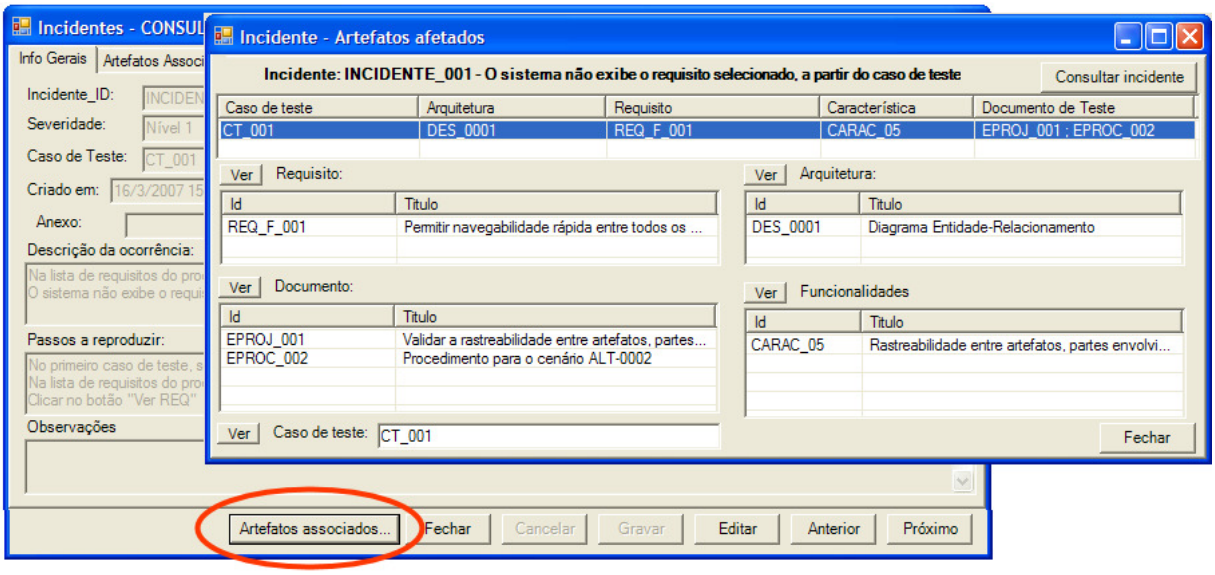

Figura A.30: Artefatos associados a um Incidente de Teste.

Para visualizar na forma de árvore os vínculos entre artefatos e incidentes, é necessário ativar a janela “Incidentes - Consulta”, através do menu “Consulta/Edição”, selecionar o incidente e, depois, selecionar a aba “Artefatos Associados” (Figura A.31).

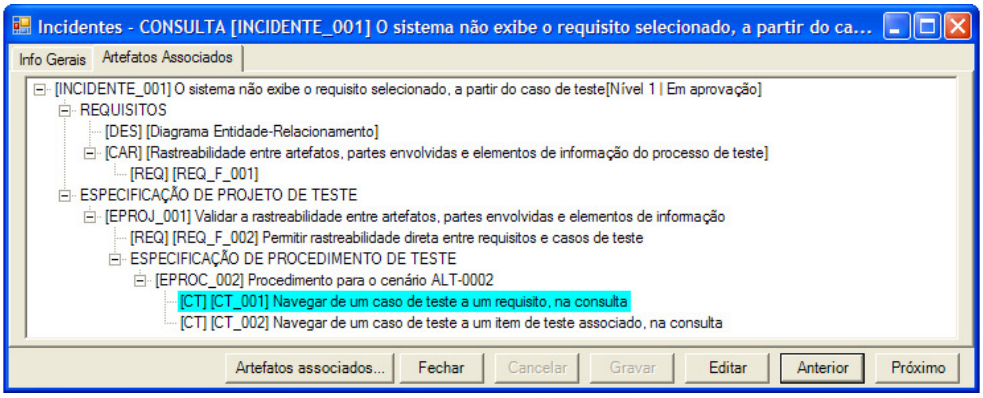

Figura A.31: Registro de um Incidente.

Considerando o exemplo deste apêndice, as formas de visualização ilustradas nas Figuras A.30 e A.31 auxiliam a descobrir os artefatos associados ao incidente INCIDENTE_001: ele foi gerado através da execução do caso de teste CT_001, que está associado ao requisito REQ_F_002 e ao procedimento EPROC_002. REQ_F_002 detalha a característica CARAC_04, é implementado pelo desenho DES_001 e está associado à especificação de projeto EPROJ_001. Todos os artefatos vinculados ao incidente podem ser acessados diretamente, independentemente da forma de visualização: matriz ou árvore. E, conforme mencionado anteriormente, os atores relacionados a um incidente podem ser descobertas através da consulta a cada artefato associado ao incidente.

A.11 Gerenciando as associações de rastreabilidade

Existem diversos vínculos predefinidos entre os tipos de artefatos gerenciados no protótipo, sendo possível visualizar as associações entre os artefatos, controlar o *status* dessas associações e, se necessário, atualizar o estado dessas associações.

A.11.1 Visualizar as associações na forma de árvore ou de matriz

Os formatos para visualizar as associações de rastreabilidade no protótipo são: matriz e árvore.

Matriz de rastreabilidade

Uma matriz de rastreabilidade permite visualizar as associações na forma de linhas e colunas. No protótipo existem cinco tipos pré-definidos: “Requisitos x *Designs* x Casos de Teste”, “Casos de Teste x Requisitos x *Designs* x Incidentes”, “Designs x Requisitos x Casos de

Teste", "Casos de Teste x Requisitos x *Designs* x Códigos-fonte" e "Códigos-fonte x *Designs* x Casos de Teste x Incidentes". Todos os artefatos listados podem ser acessados a partir das matrizes.

A Figura A.32 mostra a matriz de rastreabilidade "Casos de Teste x Requisitos x *Designs* x Incidentes", gerada com base nos artefatos e associações definidas até o presente nesta demonstração. Percebe-se que os casos de teste CT_001, CT_002, CT_003, CT_004, CT_005 e CT_006 estão vinculados a REQ_F_002 e que REQ_F_002 está associado ao desenho DES_001. Nota-se, também, que a execução de dois casos de teste resultou no registro de dois incidentes. Colunas vazias podem indicar possíveis problemas, como no caso de CT_007, que foi registrado na ferramenta, mas não foi associado a nenhum requisito e a nenhum *design*.

Figura A.32: Matriz Casos de teste x Requisitos x *Designs* x Incidentes.

A Figura A.33 mostra a matriz "Requisitos x *Designs* x Casos de Teste".

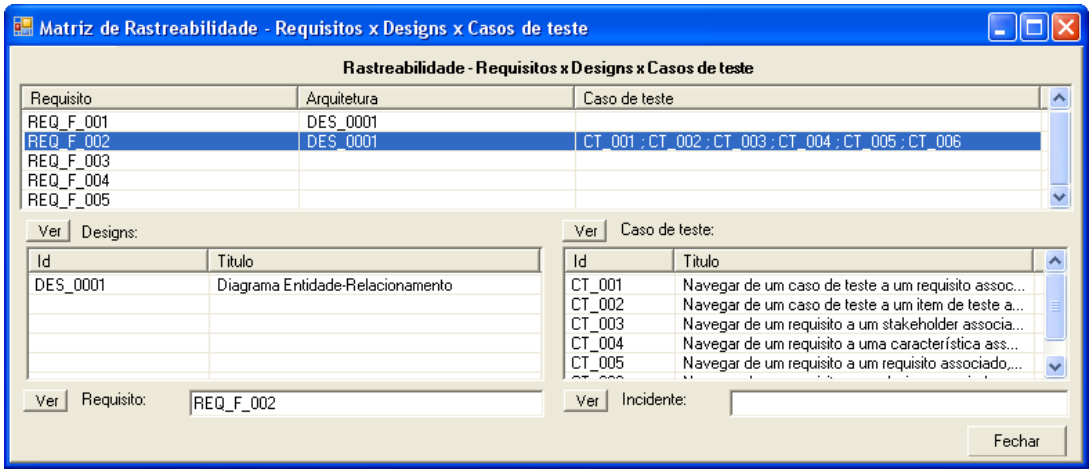

Figura A.33: Matriz Requisitos x *Designs* x Casos de Teste.

Árvore de rastreabilidade

Neste tipo de visualização, as associações de rastreabilidade são mostradas de maneira hierárquica. Os itens de rastreabilidade que podem ser vistos na árvore de rastreabilidade de um projeto são: itens de teste, requisitos do software (característica, requisito funcional e requisito não-funcional), elementos da arquitetura (referenciados ora como desenhos ora como *designs*), códigos-fonte, planos de teste, especificações de projeto de teste, especificações de procedimento de teste, especificações de casos de teste, registros de incidentes, atores, decisões do projeto, documentos adicionais, além da organização dos casos de teste por níveis e tipos.

Árvores de rastreabilidade também podem ser visualizadas na consulta aos seguintes itens de rastreabilidade: Característica, Requisito, *Designs*, Códigos-fonte, Especificação de Caso de Teste, Registro de Incidente e Ator.

A Figura A.34 mostra a árvore de rastreabilidade de um projeto de teste, logo após ser registrado na ferramenta.

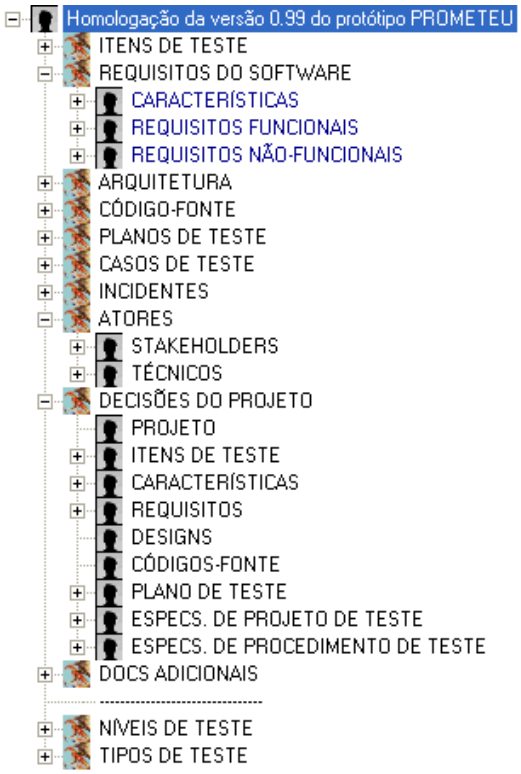

Figura A.34: Tipos de artefatos na árvore de rastreabilidade de um projeto.

A Figura A.35 mostra um exemplo na consulta ao caso de teste CT_001.

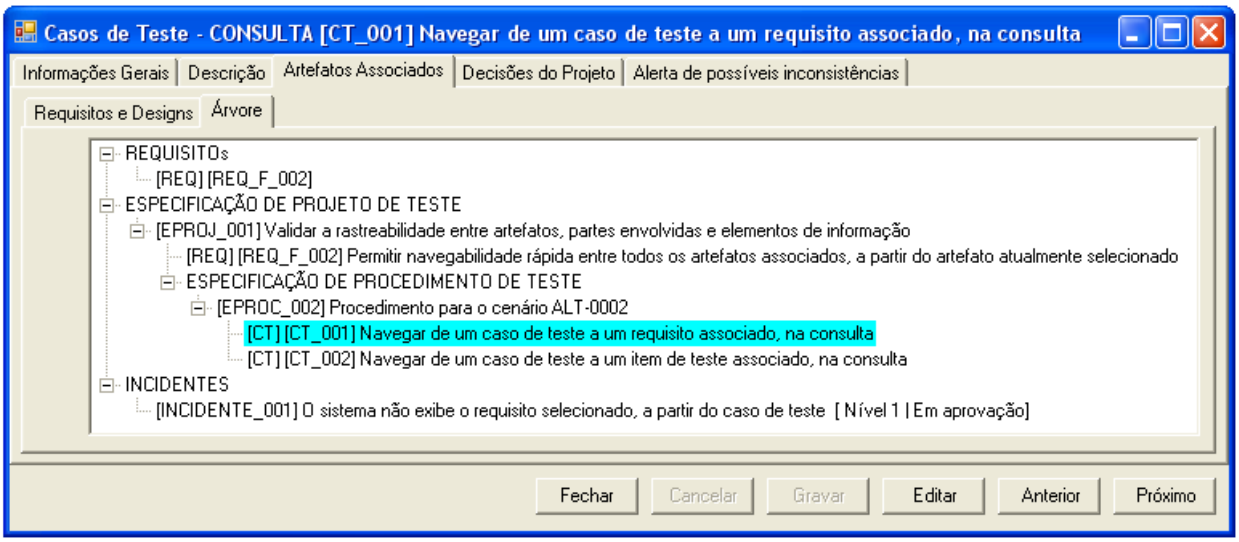

Figura A.35: Caso de teste – árvore de rastreabilidade.

A Figura A.36 mostra a tela que contém a árvore de rastreabilidade do projeto, gerada com base nos artefatos registrados até este ponto da demonstração. À esquerda são mostrados os artefatos registrados na ferramenta para o projeto de teste desta demonstração. À direita, após selecionar um artefato na árvore, é possível visualizar informações resumidas e a árvore de rastreabilidade deste artefato. Na figura, o artefato selecionado é a característica CARAC_06: à direita visualizam-se informações resumidas sobre ela, além da sua árvore de rastreabilidade. Qualquer um dos artefatos listados nas duas árvores pode ser consultado a partir do menu de contexto “Ver Informações...”, ativado ao clicar-se com o botão direito do mouse sobre o nome do artefato.

Para cada artefato selecionado na árvore à esquerda, à direita da tela serão mostrados os elementos de informação associados ao artefato.

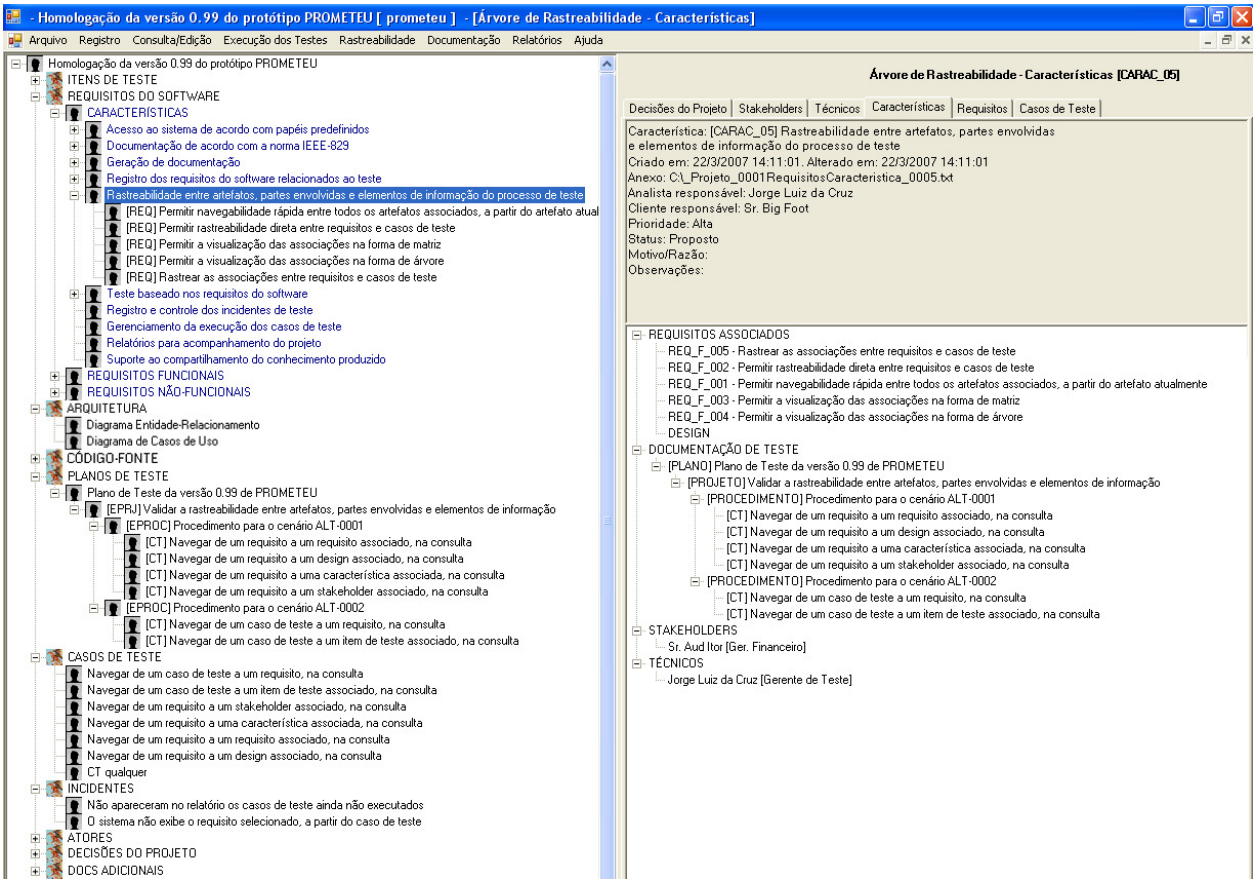

Figura A.36: Árvore de rastreabilidade do projeto da demonstração.

A.11.2 Analisar o status das associações entre artefatos

Associações de rastreabilidade obsoletas (quando um dos artefatos vinculados é atualizado após a definição da associação é não é feita uma revisão nos outros artefatos e nos *links* entre eles) podem indicar situações de informações desatualizadas e inconsistentes na documentação do projeto. A ausência de vínculos entre artefatos, em alguns casos, pode significar problemas de rastreabilidade.

PROMETEU possibilita identificar associações desatualizadas através de matrizes de rastreabilidade, através do relatório geral de pendências de rastreabilidade, na consulta a um artefato ou no momento da atualização de dados de um artefato.

Através de matrizes de rastreabilidade

Colunas vazias em uma matriz indicam ausência de associações, o que pode significar eventuais problemas, como por exemplo, um requisito que foi registrado, mas não possui nenhum

caso de teste associado. A Figura A.37 ilustra a rastreabilidade entre casos de teste, requisitos, desenhos e incidentes. Analisando a figura, percebe-se que CT_007 não está associado a nenhum requisito, uma situação que deve ser corrigida.

Figura A.37: Matriz Casos de Teste x Requisitos x *Designs* x Incidentes.

Através do relatório geral de pendências

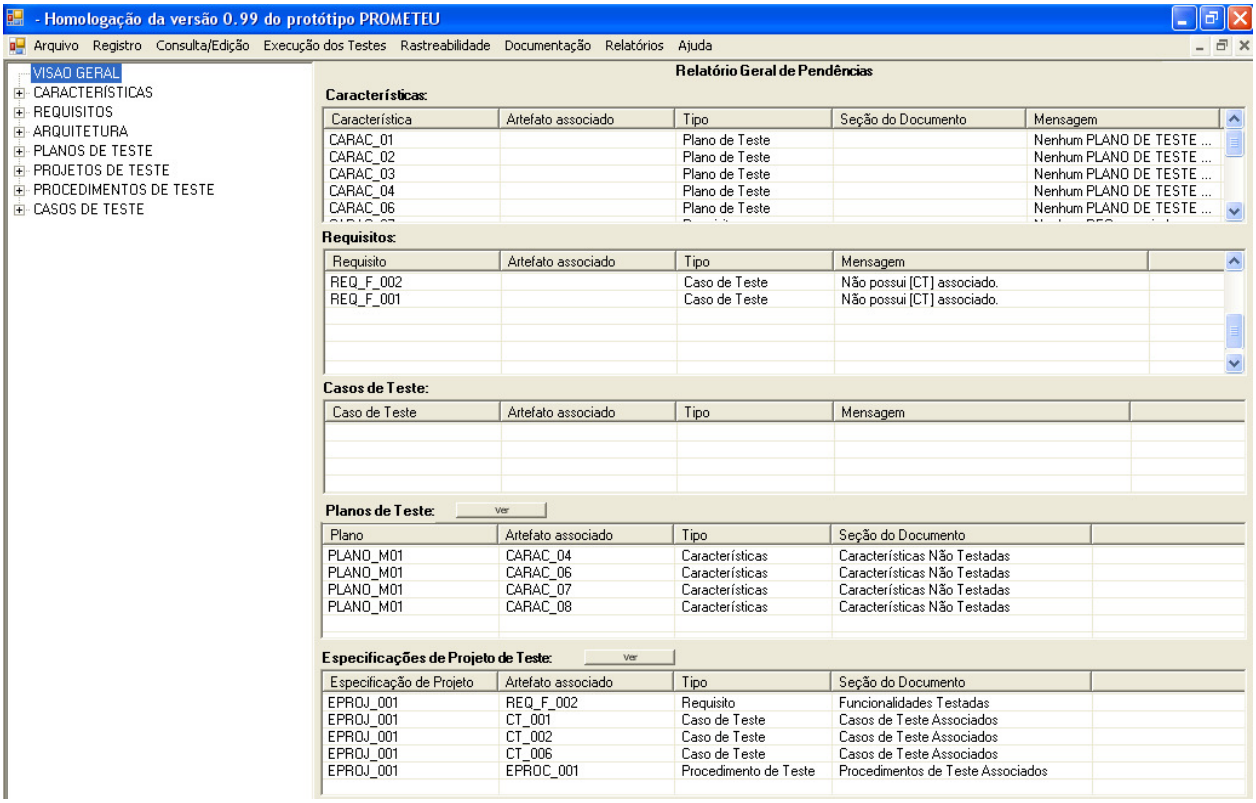

Figura A.38: Relatório geral de pendências de rastreabilidade.

Através do menu “Rastreabilidade”, item “Pendências”, é possível consultar um relatório geral de pendências que mostra, na forma de matrizes, as associações de rastreabilidade do projeto que estejam, como mostra a Figura A.38. Um menu de contexto permite acessar cada artefato para que o estado das associações seja atualizado.

Na consulta a um artefato

Quando um artefato é consultado, caso existam associações desatualizadas, PROMETEU exibe uma mensagem de alerta e disponibiliza uma lista indicando os artefatos associados cujos vínculos estiverem desatualizados. Através dessa lista é possível acessar os artefatos e revisar e atualizar as associações.

A Figura A.39 mostra um exemplo desta situação para o documento EPROJ_001.

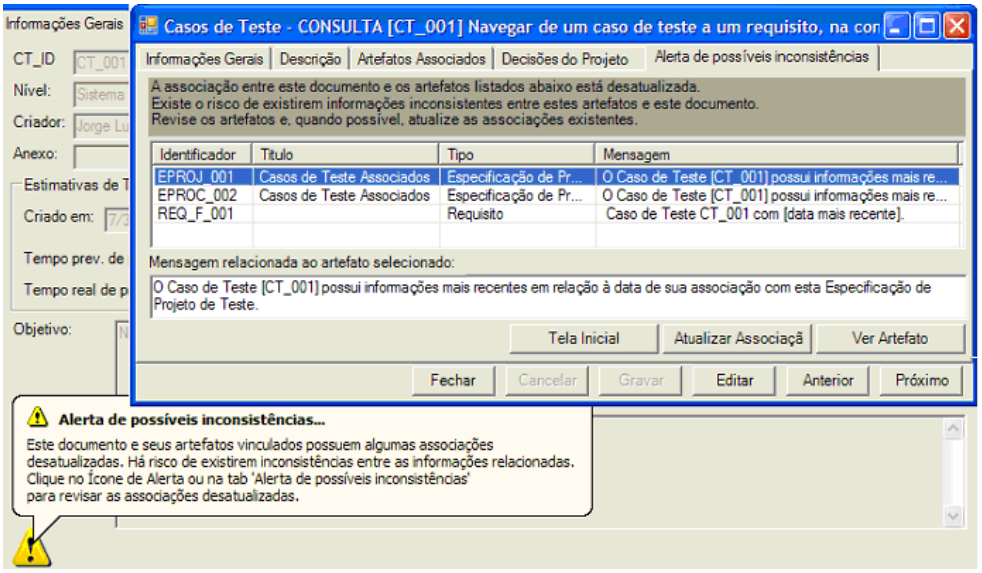

Figura A.39: Identificando problemas de rastreabilidade entre artefatos.

Na atualização de um artefato

Antes de gravar uma alteração nos elementos de informação de um artefato, PROMETEU verifica se existem associações com outros artefatos. Caso existam, é emitido um alerta indicando que possíveis inconsistências poderão ocorrer caso os artefatos vinculados não sejam revisados. A Figura A.40 ilustra esta situação durante a alteração de CT_001.

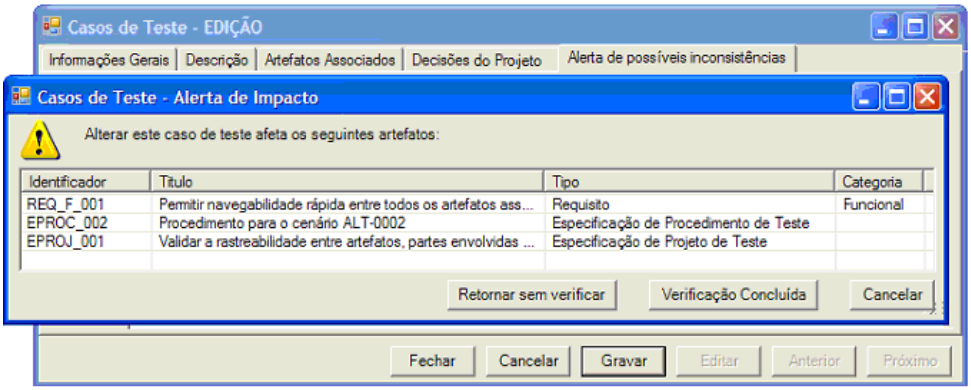

Figura A.40: Alerta de artefatos afetados no momento de uma alteração.

É possível revisar um a um todos os artefatos e posteriormente atualizar suas associações. Mas também é possível selecionar a opção "Verificação Concluída", que atualiza todas as associações de uma vez só, sem ser necessário percorrer os artefatos uma a um.

Entre os atores e os artefatos a eles associados.

É possível descobrir os atores associados a um artefato de duas maneiras: a) através do menu "Rastreabilidade", item "Árvore", nó "Atores", b) através do menu "Consulta/Edição", item "Atores".

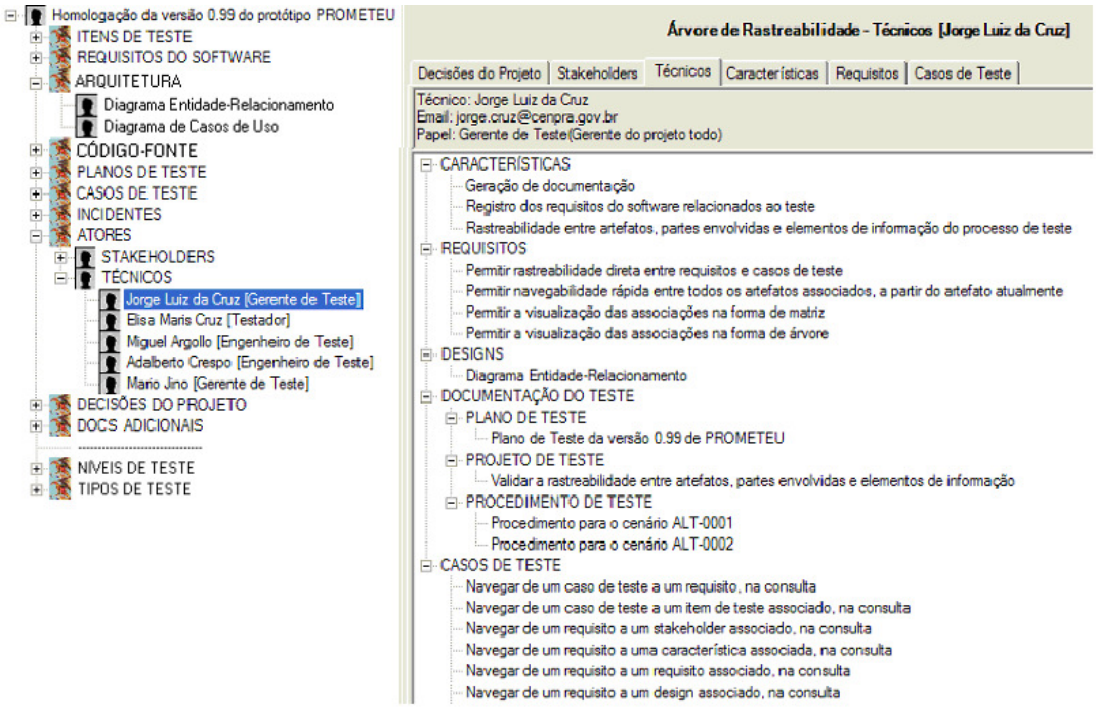

Figura A.41: Visualização dos técnicos e dos artefatos a eles associados.

A Figura A.41 demonstra a primeira opção: para cada ator, tanto da categoria “Stakeholder” como da categoria “Técnico”, é possível visualizar os artefatos associados e as contribuições do ator para cada artefato. Basta selecionar ator na árvore à esquerda para visualizar informações resumidas a seu respeito à direita, além da árvore de artefatos associados ao ator. É possível acessar qualquer artefato da árvore através de um menu de opções sensível ao contexto.

A Figura A.42 representa a segunda opção: quando se consulta os dados relativos a um ator (neste caso, “elisa”), é possível visualizar em forma de árvore todos os artefatos relacionados. Neste caso, uma característica, dois requisitos e nenhum outro artefato.

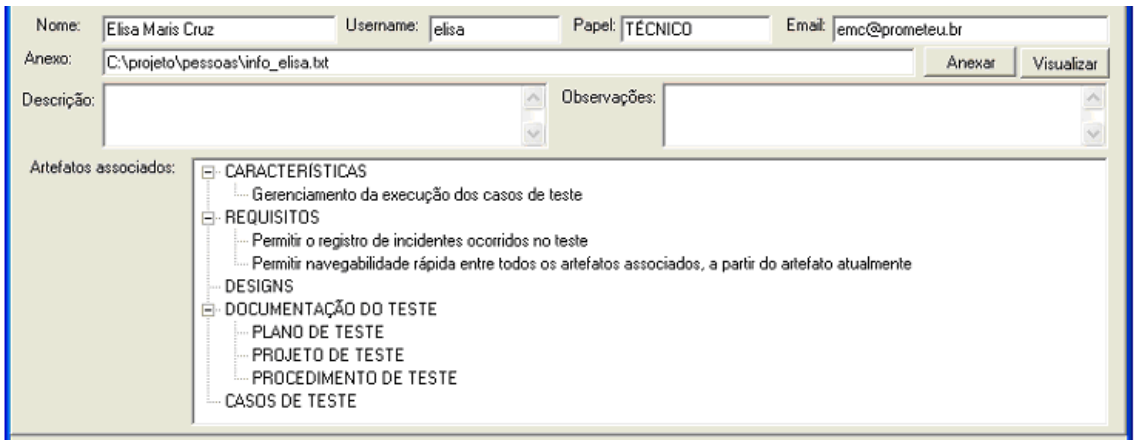

Figura A.42: Ator e artefatos associados.

A.12 Analisando o impacto de uma modificação

O impacto de uma modificação pode ser estimado através do menu “Rastreabilidade”, item “Árvore”. Seleciona-se o artefato desejado para visualizar todos os artefatos associados a ele; os dados de cada artefato podem ser consultados a partir da árvore de rastreabilidade do artefato selecionado (a árvore que aparece à direita).

Como exemplo, caso tenha sido descoberto que houve um engano na especificação do requisito REQ_F_002 e, em função disso, ela deva ser alterada, torna-se necessário estimar o impacto dessa modificação. O primeiro passo é selecionar o requisito na árvore de rastreabilidade à esquerda, sob o nó “REQUISITOS DO SOFTWARE”. Após o requisito ser selecionado, à direita é montada sua árvore de rastreabilidade, mostrando suas associações com outros artefatos (Figura A.43). Através dessa árvore é possível descobrir rapidamente todos os artefatos associados e, através de cada artefato, descobrir todos os atores relacionados à modificação.

Dessa forma, é possível estimar com maior exatidão os impactos que uma modificação neste requisito causará no projeto.

Além da análise através da árvore de rastreabilidade, toda a vez que um artefato estiver prestes a ter seu conteúdo alterado, o sistema emite um alerta mostrando todos os artefatos que serão afetados pela modificação. Dessa maneira também é possível perceber o impacto dessa modificação para o projeto.

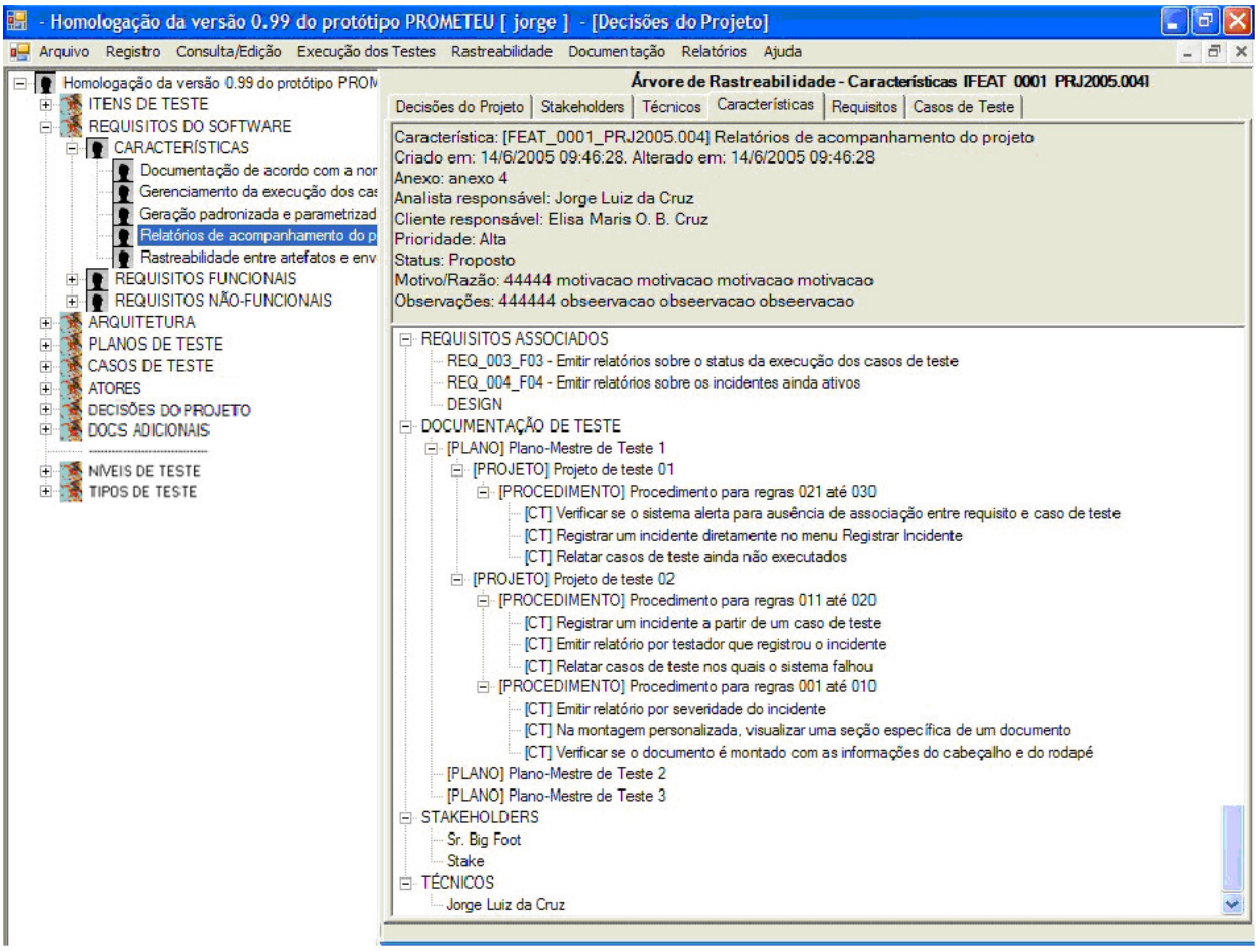

Figura A.43: Análise do impacto de uma modificação em um artefato.

A.13 Analisando as justificativas para decisões do projeto

Ao longo da execução do projeto pode ser necessário rastrear observações, comentários e justificativas que fundamentam determinadas ações e decisões. Os artefatos registrados em PROMETEU que possuem uma seção chamada “Decisões do Projeto” são: Projeto de Teste, Item de Teste, Característica, Requisito, *Design*, Código-fonte, Plano de Teste, Especificação de

Projeto de Teste e Especificação de Procedimento de Teste e Especificação de Caso de Teste. Essas decisões podem ser rastreadas de duas maneiras:

- Na consulta a um artefato: seleciona-se o menu “Consulta/Edição”, logo após o artefato desejado e, em seguida, a aba “Decisões do Projeto”;
- Na árvore de rastreabilidade do projeto: seleciona-se o menu “Árvore”, item “Rastreabilidade”. Na árvore de rastreabilidade, seleciona-se o nó “Decisões do Projeto” e, em seguida, seleciona-se o artefato desejado. A Figura A.44 mostra a sub-árvore “Decisões do Projeto”, parcialmente expandida.

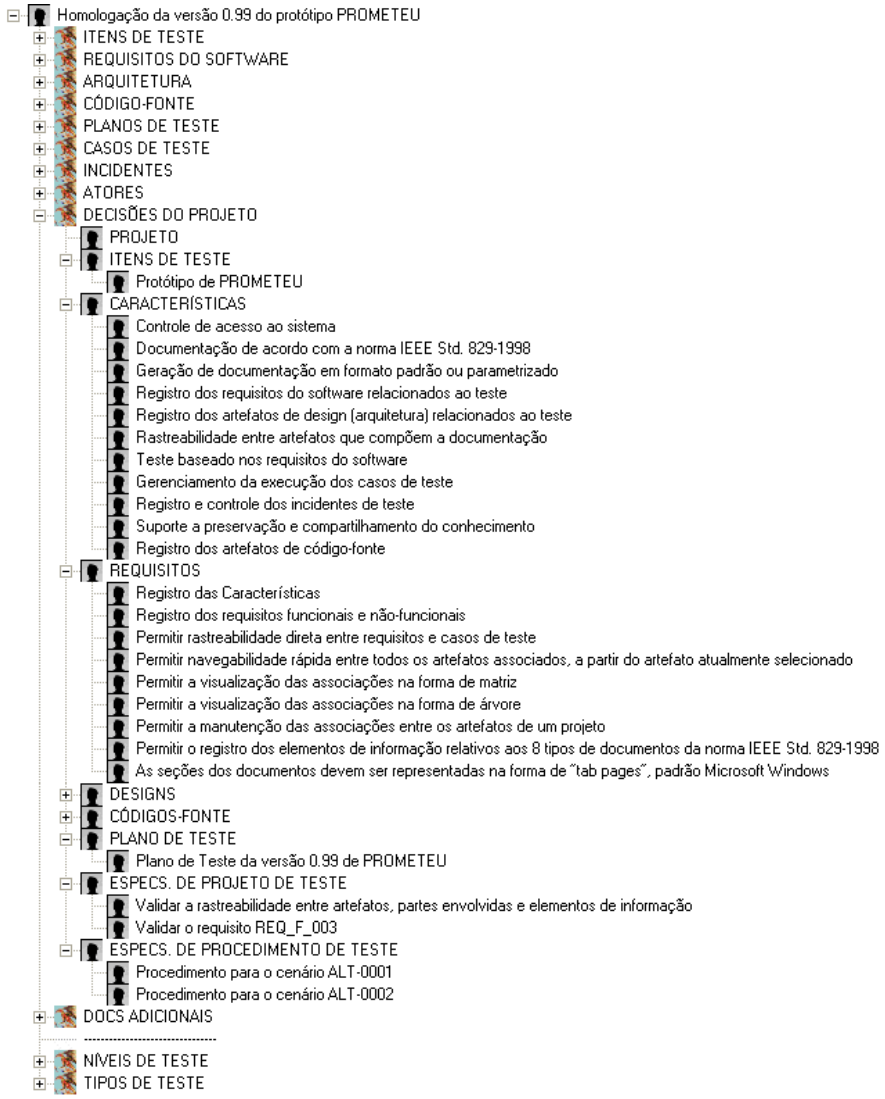

Figura A.44: Sub-árvore “Decisões do Projeto”.

A.14 Consultando os elementos de informação da documentação

A consulta aos elementos de informação de um projeto de teste é feita através do menu "Consulta/Edição". A interface gráfica é padrão do sistema operacional Microsoft Windows, organizada como formulários, nos quais as seções de documentos sugeridos pela norma IEEE Std 829-1998 são representadas como abas nos formulários (*tab pages*).

A Figura A.45 ilustra a tela de consulta ao Plano de Teste PLANO_001, seção "Características que Serão Testadas". Pode-se perceber que as seções sugeridas para este tipo de documento estão presentes e são organizadas em abas (*tab pages*).

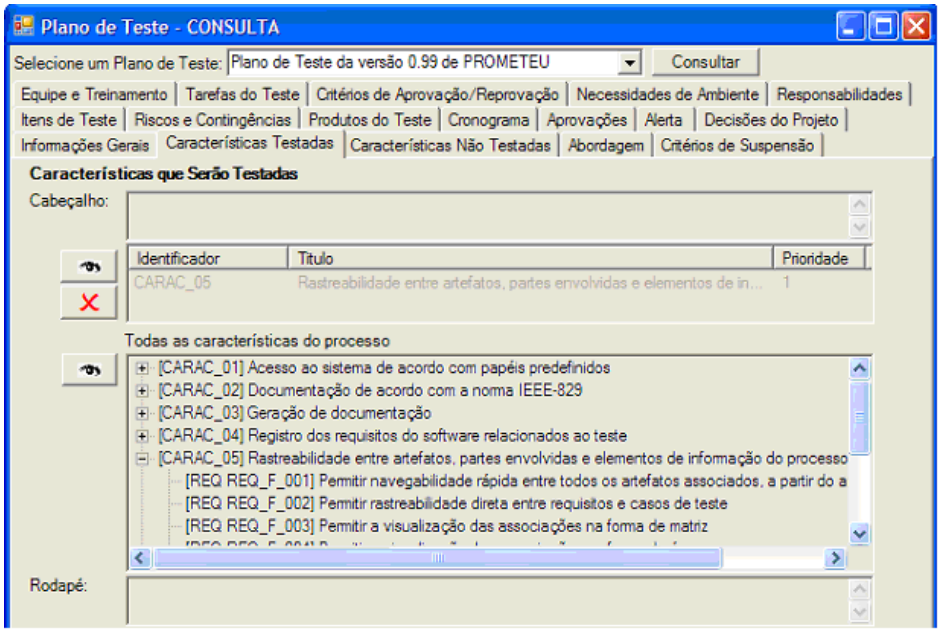

Figura A.45: Artefato PLANO_M01, seção "Características que serão testadas".

Quando um artefato é consultado, caso existam associações desatualizadas com os artefatos associados, o sistema exibe um alerta.

A Figura A.46 mostra um alerta para o documento EPROJ_001; ao selecionar o símbolo de alerta, é exibida uma lista (Figura A.47) dos artefatos vinculados cujas associações estão desatualizadas.

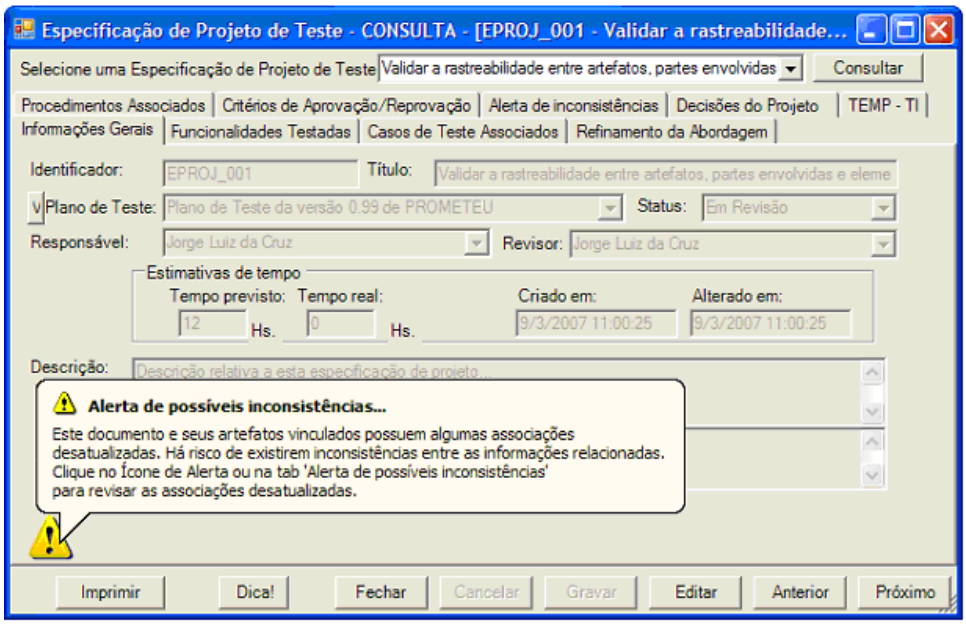

Figura A.46: Alerta de possíveis inconsistências entre artefatos associados.

Através desta tela (Figura A.47) é possível navegar até cada artefato listado, revisar seu conteúdo e realizar, se necessário, as devidas modificações. Também é possível atualizar todas as associações de uma vez através do botão “Atualizar Associações”. Neste caso, a Figura A.47 mostra que durante a consulta ao documento EPROJ_001, o sistema alertou para a existência de possíveis inconsistências relacionadas aos artefatos REQ_F_002, CT_001, CT_002, CT_006 e EPROC_001.

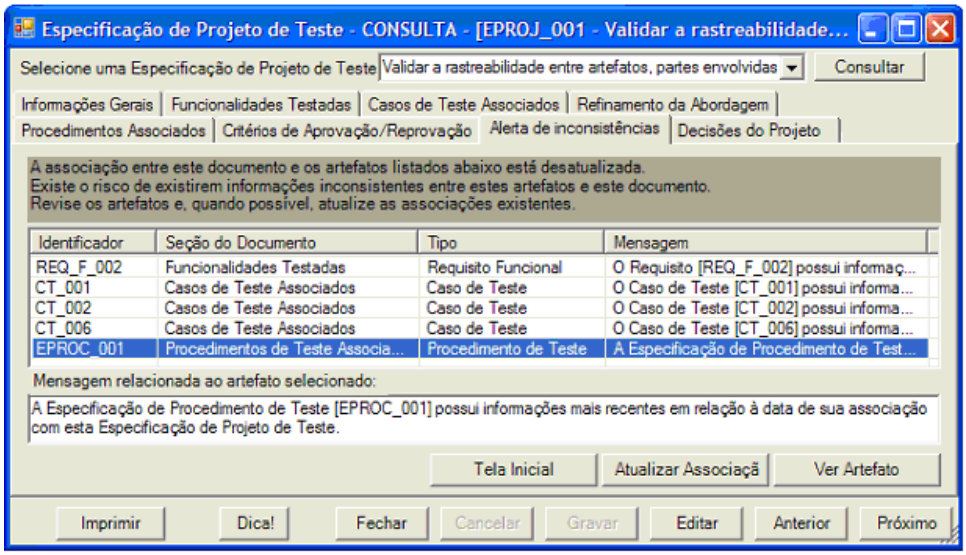

Figura A.47: Lista de possíveis associações inconsistentes.

A.15 Mantendo os elementos de informação da documentação

A manutenção do conteúdo dos artefatos é feita através do menu “Consulta/Edição”. Os elementos de informação de todos os tipos de artefato podem ser editados selecionando-se o botão “Editar”.

Modificar o conteúdo de artefatos ao longo do projeto pode gerar inconsistências entre diversos elementos de informação dos diferentes artefatos associados àquele que está sendo modificado. Para reduzir esse risco, PROMETEU emite um alerta antes de gravar uma modificação, indicando todos os artefatos vinculados cujo conteúdo pode ser afetado pela operação em curso. A Figura A.48 ilustra um exemplo deste tipo de situação para o documento CT_001: a partir da lista de alerta de impacto, todos os artefatos podem ser revisados individualmente e todas as associações podem ser atualizadas. Também é possível atualizar todas as associações de uma vez através do botão “Verificação Concluída”.

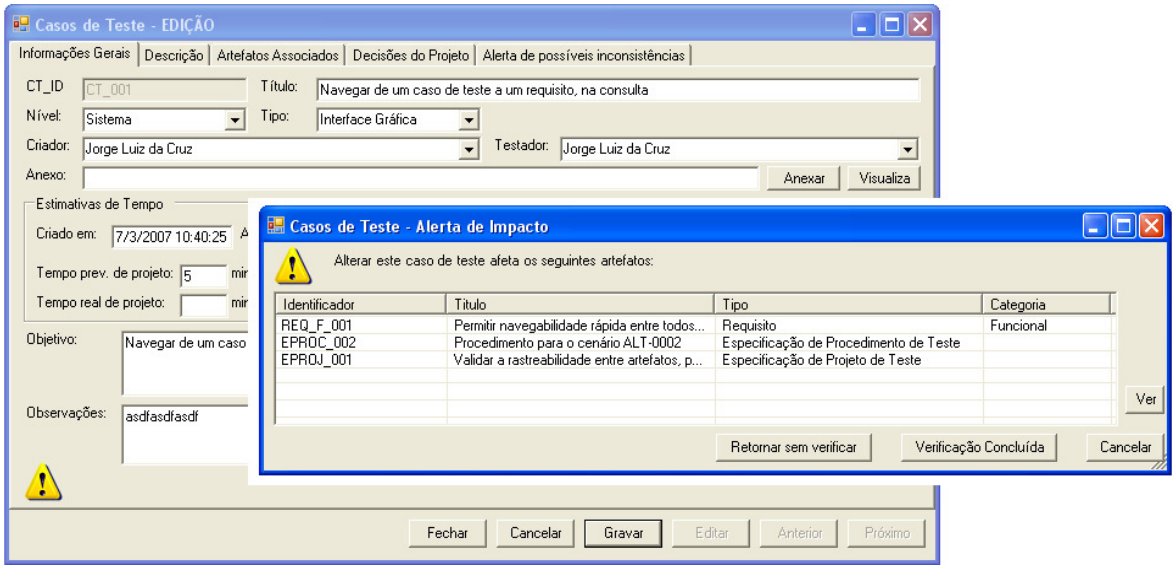

Figura A.48: Modificação em um Caso de Teste.

A.16 Gerando documentação

Ao longo do projeto será necessário gerar documentos tanto no formato padrão pré-definidos na ferramenta como com conteúdo parametrizado (composição de um documento a partir de partes, ou da totalidade, do conteúdo de outros documentos).

Geração padronizada

Um documento é gerado através do menu “Documentação”, item “Documentos do Teste”. Na janela “Geração de Documentação”, escolhe-se o tipo de documento, depois o documento a ser gerado e, caso se aplique, opções extras para detalhar a geração do documento. Pode ser gerada documentação para os requisitos do software e para os documentos de teste preconizados pela norma IEEE Std 829-1998. Em ambos os casos a documentação é gerada no formato HTML.

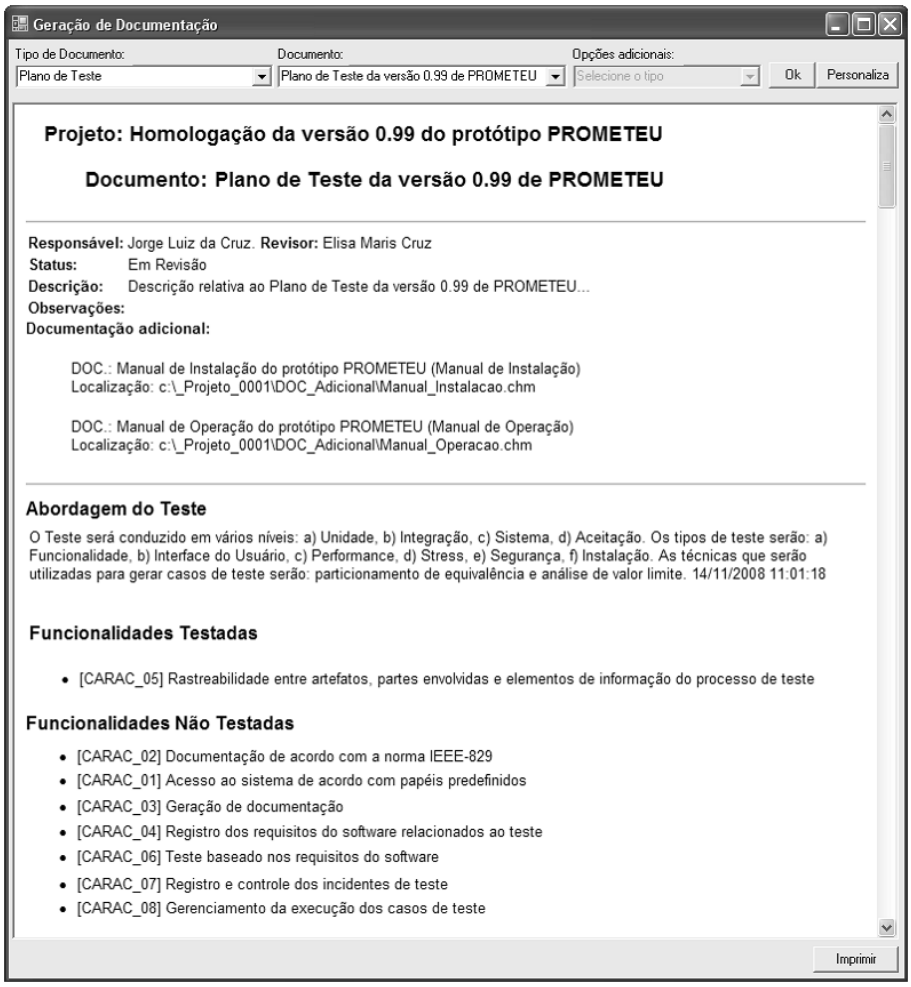

Figura A.49: Visão parcial do documento Plano de Teste.

A Figura A.49 é uma visão parcial do documento “Plano de Teste da versão 0.99 de PROMETEU”. As opções para geração padronizada de documentos do teste são:

- Para os planos de teste: “Por plano”;
- Para as especificações de projeto de teste: “Por especificação”;

- Para as especificações de procedimento de teste: “Por especificação”;
- Para os casos de teste: “Todos os casos de teste”, “Todos os casos de teste e Requisitos associados”, “Um caso de teste”, “Casos de teste por testador”, “Casos de teste por procedimento”;
- Para os incidentes de teste: “Todos os incidentes”, “Todos os incidentes, com artefatos associados”;
- Para os diários de teste: “Todos os diários”, “Diários por participante no projeto”. Todos os diários são impressos baseados em um intervalo entre duas datas;
- Para os itens de teste: “Todos os itens de teste”.

A Figura A.50 mostra uma das opções para geração de documentação dos casos de teste, que é “Lista de casos de teste por testador”.

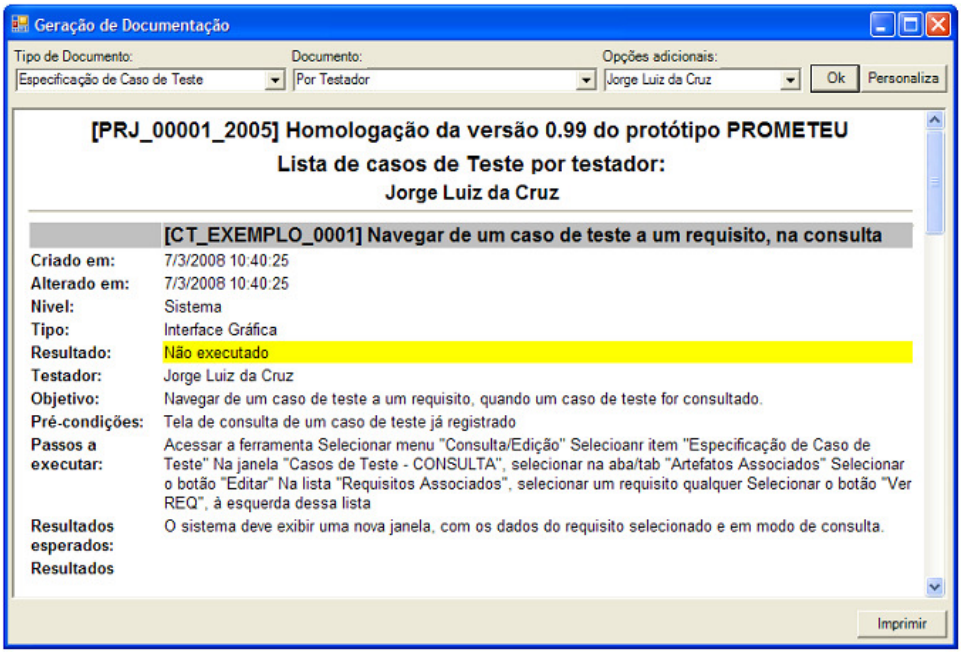

Figura A.50: Lista de casos de teste por testador.

Documentação dos requisitos do software

Além da opção de gerar documentação específica do teste, também é possível gerar documentos dos requisitos do software. A documentação dos requisitos do software (características, requisitos funcionais e não-funcionais) é gerada a partir do menu “Documentação”, item “Requisitos do Software”.

A Figura A.51 mostra parte de um documento composto por todos os requisitos do software sendo testado; a seta destaca a parte do documento relativa ao requisito funcional REQ_F_002.

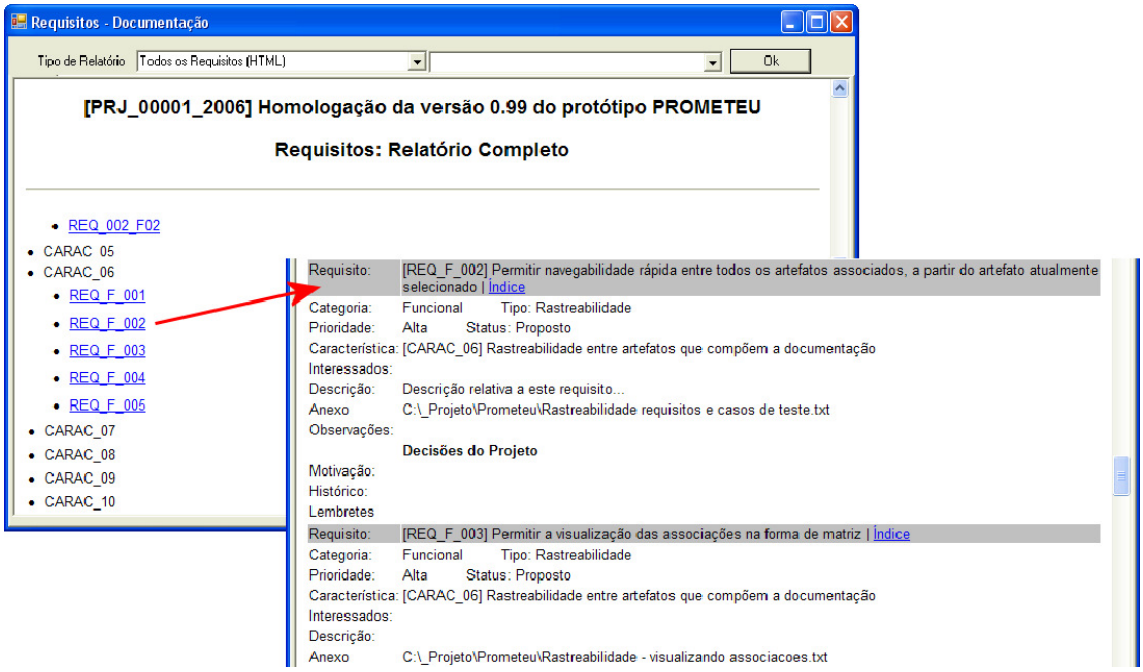

Figura A.51: Documentação dos requisitos.

Podem ser impressas todas as características, ou somente uma. E podem ser impressos todos os requisitos, ou somente um.

Geração parametrizada

Além da geração padronizada, é possível parametrizar a geração dos documentos do teste. Documentos personalizados são gerados através da combinação de seções específicas de um ou mais documentos previamente registrados. Também podem ser um conjunto resumido das seções de um único documento. Os documentos são gerados através do menu “Documentação”, item “Montagem Parametrizada”.

Na janela “Geração Parametrizada de Documentos – Passo 1” (Figura A.52), seleciona-se os documentos que fornecerão os elementos de informação. É possível visualizar um documento selecionado ao clicar-se no botão “Ver”.

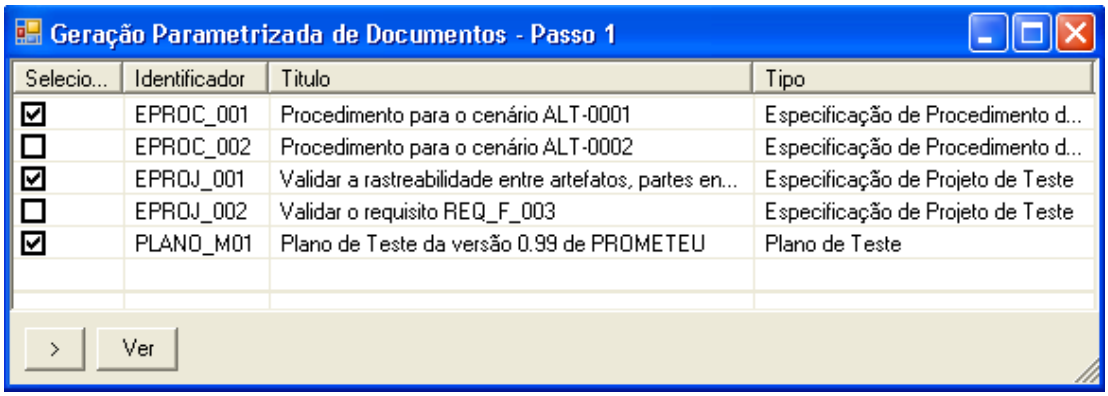

Figura A.52: Seleção de documentos para personalização.

A Figura A.53 ilustra a tela para seleção das seções que comporão o documento. Qualquer seção apresentada na lista pode ser consultada através do botão "Ver". O botão V copia a seção selecionada para a lista de seções que fazem parte do documento sendo gerado. Os botões e ordenam as seções de acordo com a necessidade. O botão exclui uma seção da lista. E o botão gera o documento.

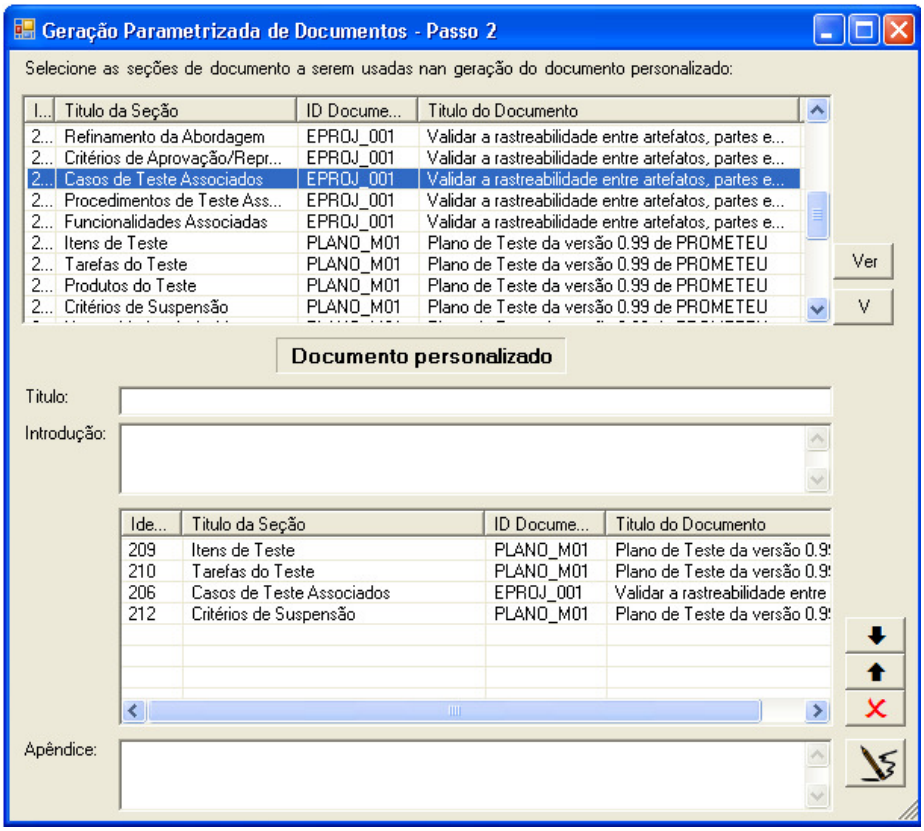

Figura A.53: Consulta a um elemento de informação específico.

Na versão atual da ferramenta, o documento resultante será gerado e exibido no formato HTML, mas não será armazenado no banco de dados.

A.17 Relatórios

Os relatórios disponíveis podem ser visualizados na interface da ferramenta ou impressos no formato HTML. As opções existentes são: decisões do projeto, requisitos, casos de teste, incidentes e relatório-resumo.

Decisões do projeto

Esses relatórios são gerados através do menu "Relatórios", item "Decisões do Projeto". Podem ser consultadas decisões relativas aos seguintes tipos de artefatos: o próprio projeto, os itens de teste, as características, os requisitos, os planos de teste, as especificações de projeto de teste e as especificações de procedimento de teste.

Requisitos

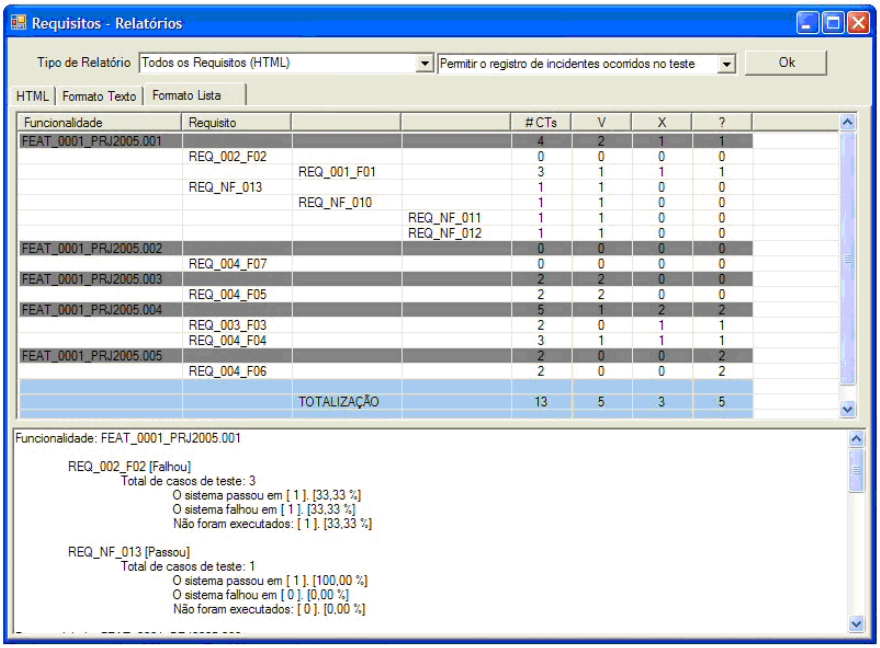

Figura A.54: Relatório por Característica.

Esses relatórios são gerados através do menu Relatórios, item "Requisitos", subitens "Características" ou "Requisitos". Pode-se consultar o *status* dos testes relativos a todos os

requisitos, organizados tanto por Característica como por prioridade do requisito. São mostrados todos os casos de teste associados e é feita uma totalização indicando em quantos casos de teste o sistema passou, em quantos o sistema falhou e quantos casos de teste ainda não foram executados. A Figura A.54 mostra um exemplo de relatório ordenado por característica.

Casos de Teste

Esses relatórios são consultados através do menu Relatórios, item "Casos de Teste". É exibida uma lista com todos os casos de teste e o *status* da sua execução, quem executou o teste, quando, e qual o tipo e o nível do teste. Também é exibida uma totalização resumida da execução dos casos de teste, indicando o número total de casos de teste, quantos foram executados e em quantos o sistema falhou ou passou.

A Figura A.55 mostra um exemplo deste tipo de relatório.

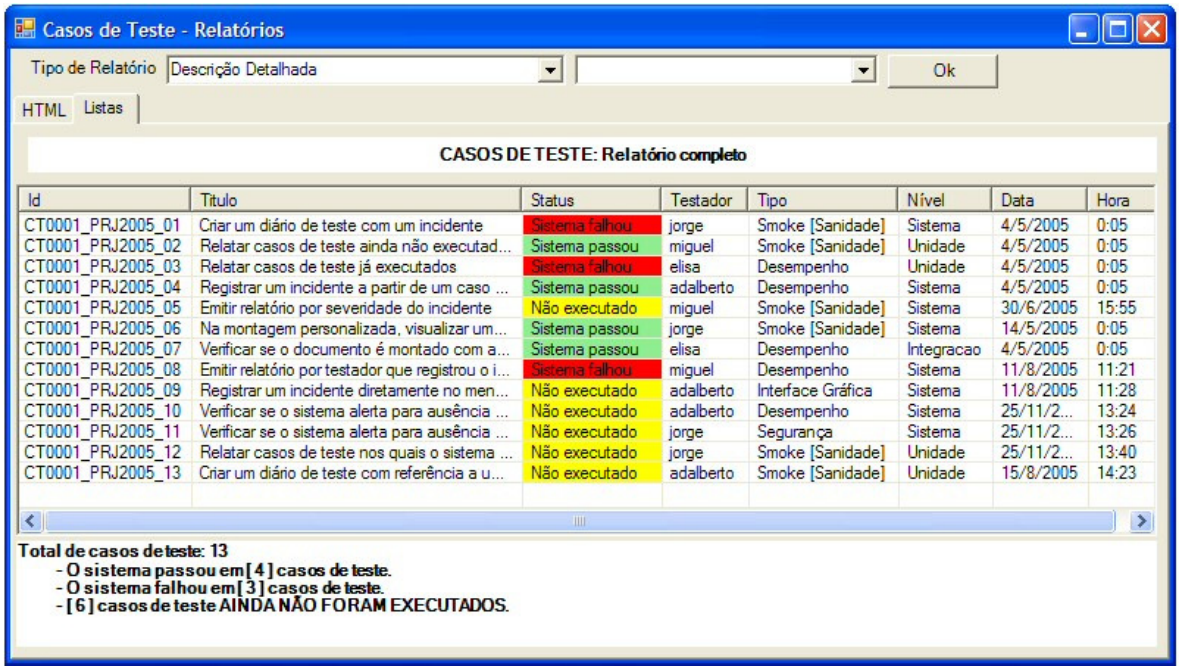

Figura A.55: Resumo da execução dos casos de teste.

Incidentes

Esse relatório fornece uma totalização resumida de todos os incidentes ocorridos no projeto e permite consultar detalhes sobre um determinado incidente e sobre todos os artefatos associados.

Relatório-resumo

Este relatório é gerado através do menu Relatórios, item “Relatório-Resumo”. Ele apresenta um resumo do estado atual da execução do teste com base nas recomendações existentes no tipo de documento “Relatório-Resumo” da norma IEEE Std 829-1998.

A.18 Analisar incidentes que ocorrem após a entrega do software

Mesmo após todo o esforço dos atores envolvidos como teste, o produto de software pode ser liberado com defeitos não revelados. Caso o cliente descubra algum defeito do produto durante sua operação, é possível registrar esse incidente na ferramenta, através do menu “Registro”, item “Incidentes após a entrega”; a interface para esse registro é apresentada na Figura A.56.

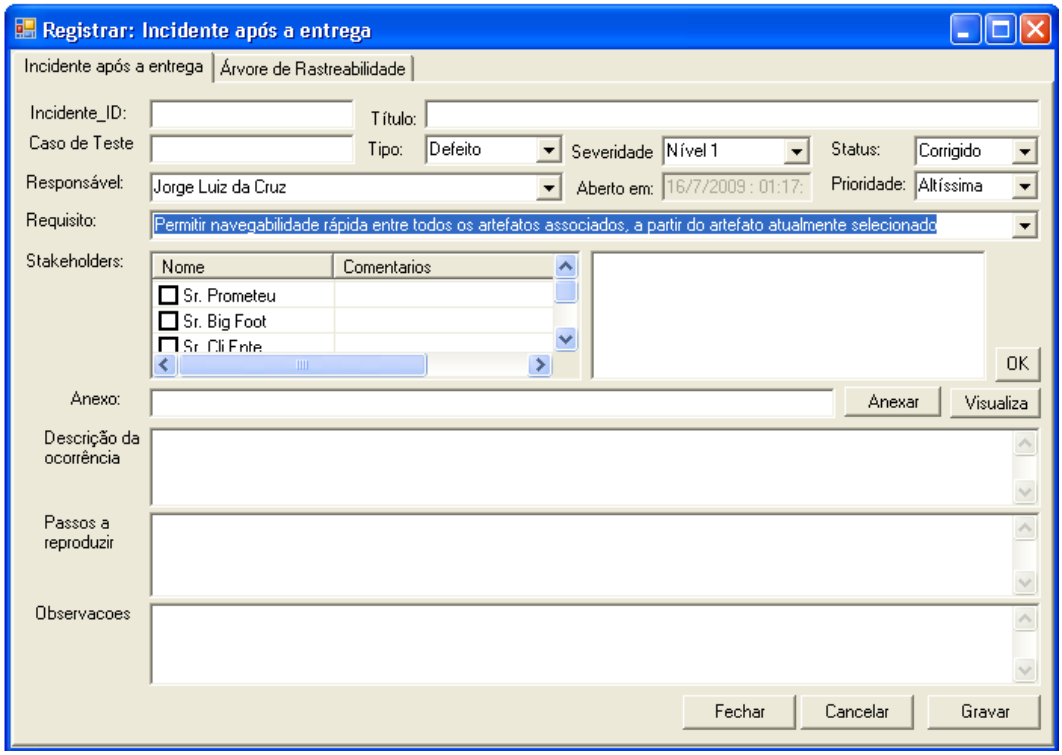

Figura A.56: Registrar um incidente que ocorre após a entrega do software.

Além da necessidade de corrigir o defeito, existe outra questão importante nesta situação: descobrir o motivo deste defeito não ter sido revelado durante a fase de testes, antes da sua

liberação para o cliente. A abordagem adotada no protótipo pressupõe a seguinte seqüência de passos: a) o cliente relata a falha ocorrida, b) um responsável técnico registra a falha e realiza uma análise prévia associando a falha à um possível requisito de origem, c) uma vez que um requisito é associado à falha relatada, a ferramenta constrói a árvore de rastreabilidade com todos os artefatos associados.

A Figura A.57 representa a árvore de rastreabilidade montada a partir da identificação do requisito relacionado à falha relatada pelo cliente; neste exemplo, o requisito REQ_F_002. Na arvore é possível visualizar os artefatos associados ao requisito.

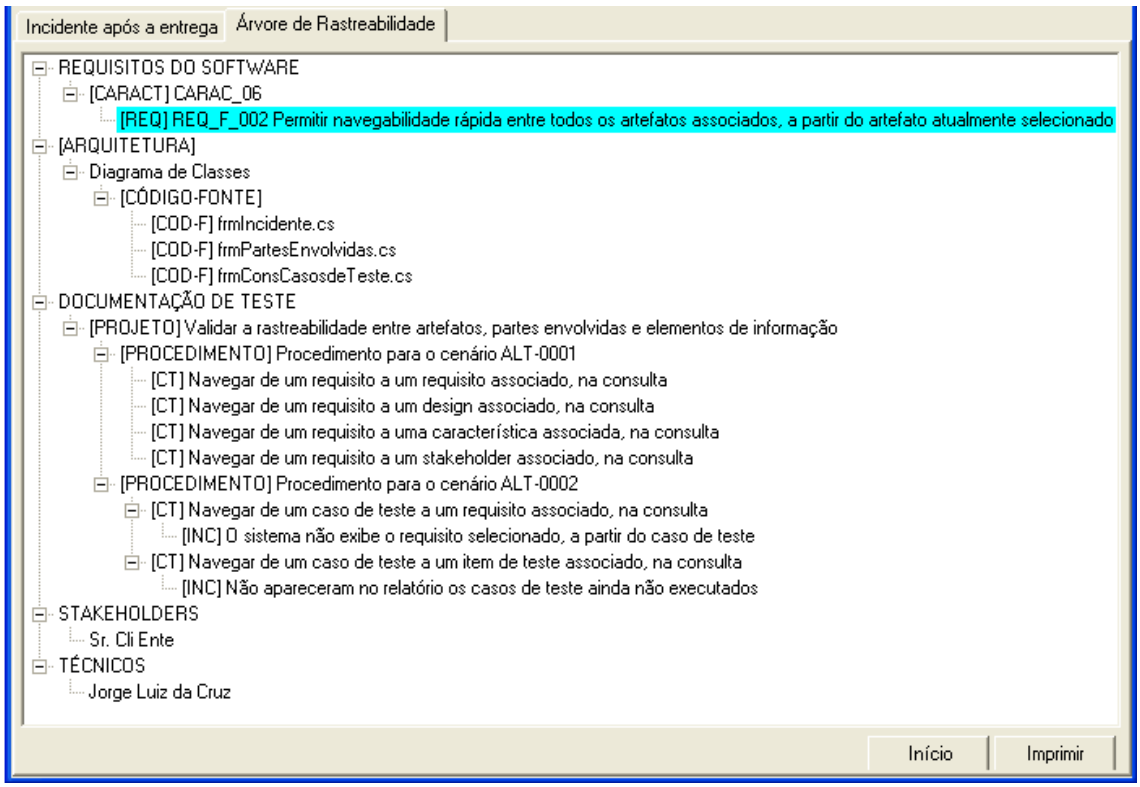

Figura A.57: Pesquisar as causas para a ocorrência da falha.

Através da árvore de rastreabilidade é possível analisar os artefatos relacionados e descobrir, por exemplo, por que o defeito não foi revelado antes da liberação do produto; exemplos de motivos são: a) requisito mal especificado, b) número insuficiente de casos de teste, c) projeto inadequado dos casos de teste, que desconsiderou determinadas particularidades do requisito.

Apêndice B

Glossário e Abreviaturas

Artefato – a) é uma parte relevante de informação produzida, registrada, utilizada e mantida em um processo de engenharia de software; são produtos tangíveis que um projeto produz ou utiliza até a entrega do resultado final (Ramesh e Jarke, 2001; Pfleeger, 2004); b) é uma informação que é produzida, modificada ou usada por um processo de desenvolvimento de software; pode ser um modelo, um elemento de modelo, um documento ou até mesmo o próprio software (Kruchten, 2003);

Alguns exemplos de artefatos são: a) Modelo de Casos de Uso, b) Modelo de Entidade-Relacionamento, c) um elemento de um modelo, como somente um caso de uso ou somente uma classe, d) Especificação de Requisitos do Software, e) Plano de Teste, f) Especificação de Caso de Teste, g) arquivos de código-fonte. Artefatos podem ser tanto documentos de papel como arquivos de computador gerenciados por ferramentas específicas usadas para criá-los e mantê-los (quando necessário, pode-se gerar cópias estruturadas e impressas das informações armazenadas em repositórios gerenciados por essas ferramentas) (DiMAggio, 2001; Kruchten, 2003).

Ator – pessoa envolvida de alguma forma na execução de um projeto de software (Gotel e Finkelstein, 1995; Ramesh e Jarke, 2001; Wiegers, 2003). Essa pessoa pode assumir um ou mais papéis ao longo de um projeto, como: a) representante do cliente, b) usuário do sistema, c) analista do sistema, d) gerente do projeto, e) desenvolvedor, f) gerente de teste, g) testador.

Caso de teste – artefato que especifica um conjunto específico de dados de entrada para executar um programa, os passos necessários para essa execução e o resultado esperado dessa execução (Pfleeger, 2004).

CenPRA – Centro de Pesquisas Renato Archer.

CTI – Centro de Tecnologia da Informação Renato Archer (antigo CenPRA). O CTI, outrora CenPRA, é uma instituição do Ministério da Ciência e Tecnologia e possui 25 anos de atividades tecnológicas relacionadas ao desenvolvimento de tecnologias de software, componentes eletrônicos, protótipos e produtos na área de Tecnologia da Informação, além de ferramentas e aplicações para a internet. O endereço na Internet é http://www.cti.gov.br.

Defeito – conjunto de instruções (que pode ser ou imperfeito ou incompleto ou ausente ou extra) no software que, ao ser executado, pode causar uma falha.

Design – palavra originária do idioma inglês e geralmente traduzida como "desenho" ou "projeto"; o termo refere-se geralmente a artefatos da fase da arquitetura do software ou a nomes de documentos que possuem o esboço ou detalhamento de algo (p.ex.: o documento "*Test Design Specification*", que é traduzido como "Especificação de Projeto de Teste"). Como as palavras *project* e *design* são traduzidas como "projeto", a diferenciação entre cada uma se dá pelo contexto onde está sendo empregada no texto.

Documentação – a) conjunto de documentos destinado a esclarecer ou provar determinado assunto ou fato (Ferreira, 1999); b) qualquer informação na forma de texto ou de figura, descrevendo, definindo, especificando, relatando ou certificando atividades, requisitos, procedimentos ou resultados (IEEE, 1990).

Documentar – registrar, em documentos, as informações que explicitem as decisões tomadas no desenvolvimento de um software, visando maior organização e referência para posteriores alterações (Ferreira, 1999).

Documento – a) qualquer base de conhecimento, fixada materialmente e disposta de maneira que se possa utilizar para consulta, estudo ou prova de algo (Ferreira, 1999); b) o meio, e a informação nele registrada, geralmente de armazenamento permanente, que podem ser lidos por uma pessoa ou máquina (planos de projeto, especificações de casos de teste, planos de teste, manuais de usuário) (IEEE, 1990); c) uma coleção de informações que são representadas em papel ou em uma mídia que representa um papel. Exemplos: documentos de processadores de texto, planilhas, agendas, gráficos de Gantt, páginas da Web e apresentações de *slide* (Kruchten, 2003).

Elemento de informação – corresponde a um dado ou parte de informação que será rastreado ao longo de um projeto. Pode ser um ator ou um dos seus atributos, um artefato ou um dos seus atributos (p.ex. uma seção de um documento).

Erro – estado incorreto, intermediário ou final, de execução do software que pode produzir um resultado incorreto, ou seja, pode levar a uma falha no software.

Falha – manifestação de um defeito no software; é a ocorrência de uma discrepância entre o resultado observado do software e o resultado prescrito pelos requisitos.

HTML – sigla para "*HyperText Markup Language*", ou "Linguagem de Marcação de Hipertexto".

Processo – conjunto de atividades inter-relacionadas que transforma entradas em saídas (IEEE, 2008).

Processo de software – conjunto de processos organizados de tal modo que, juntos, produzam um código testado. Exemplo de conjunto de processos: Requisitos, Desenho (*Design*), Codificação, Teste, Liberação, Manutenção (Pfleeger, 2004).

Project – palavra originária do idioma inglês e geralmente traduzida como "projeto"; o termo refere-se a um projeto de software, um empreendimento com objetivos e prazos bem definidos. Como as palavras *project* e *design* são traduzidas como "projeto", a diferenciação entre cada uma se dá pelo contexto onde está sendo empregada.

Rastreabilidade – grau de relacionamento que pode ser estabelecido entre dois ou mais artefatos do processo de desenvolvimento, especialmente artefatos que, entre si, possuam relacionamento do tipo predecessor-sucessor ou mestre-subordinado, como o relacionamento existente entre um requisito e um ou mais artefatos do design que o implementam (IEEE, 1990).

Software – programas de computador, procedimentos, documentação e dados relativos à operação de um sistema baseado em computador (Pressman, 2002).

Stakeholder – a) pessoas de dentro ou de fora das organizações que são, serão ou poderão ser afetadas pelo projeto, pelo sistema ou pela qualidade do sistema (Black, 2003); b) qualquer pessoa que pode ser materialmente afetada pela implementação de um novo sistema ou aplicação (Leffingwell e Widrig, 2000).

Referências Bibliográficas

Ahern, D.M., Clouse, A., Turner, R. (2003), CMMI Distilled: A Practical Introduction to Integrated Process Improvement, 2nd Ed., Addison Wesley.

Antoniol, G., Canfora, G., Casazza, G., De Lucia, A., Merlo, E. (2002), "Recovering Traceability Links Between Code and Documentation", IEEE Transactions on Software Engineering, pp. 970-983, vol. 28, no 10, October.

Beizer, B. (1990), Software Testing Techniques 2nd Edition, International Thomson Computer Press.

Black, R. (2003) Critical Testing Processes: Plan, Prepare, Perform, Perfect. Addison-Wesley – Pearson Education, Inc., Boston, United States of America, 571p.

Chrissis, M.B., Konrad, M., Shrum, S. (2003), CMMI – Guidelines for Process Integration and Product Improvement, Addison-Wesley.

Cleland-Huang, J., Chang, C.K., Christensen, M. (2003) "Event-based traceability for managing evolutionary change", IEEE Transactions on Software Engineering, Vol. 29, No 9, September, p. 796 – 810

Cockburn, A. (2005), Escrevendo Casos de Uso Eficazes – Um Guia Para Desenvolvedores de Software, Bookman Editora.

Copeland, L. (2004), A Practitioner's Guide to Software Test Design, Artech House Publishers.

Craig, R.D. e Jaskiel, S.P. (2002), Systematic Software Testing, Artech House Publishers.

Crespo, A.N., Martinez, M.R., Jino, M., Argolo, M.T. (2002) "Application of the IEEE 829 Standard as a Basis for Structuring the Testing Process", The Journal of Software Testing Professionals, Vol. 3, No. 3, December.

Crespo, A.N., Silva, O.J., Borges, C.A., Salviano, C.F., Argolo, M.T., Jino, M. (2004) "Uma Metodologia para Teste de Software no Contexto da Melhoria de Processo", Simpósio Brasileiro de Qualidade de Software, Maio, p. 271-285.

Crespo, A.N. e Jino, M. (2005), "Processo de Teste de Software", Relatório Técnico, CTI.

Davis, A.M. (1990) “The analysis and specification of systems and software requirements”. In Systems and Software Requirements Engineering. IEEE Computer Society Press, 1990, 119–144.

DiMaggio, L. (2001), Bringing Your Test Data to Life. STQE – The Software Testing & Quality Engineering Magazine, vol. 3, issue 2. March/April.

Domges, R. e Pohl, K. (1998) “Adapting traceability environments to project-specific needs”, Communications of the ACM, Vol. 41, No. 12, December, p. 54-62.

Duarte, A.S. e Grahl, E.A. (2000) Software de Apoio ao Processo de Documentação baseado em Normas de Qualidade - FURB, Simpros 2000

Dustin, E. (2003), Effective Software Testing: 50 Specific Ways to Improve Your Testing, Addison-Wesley – Pearson Education, Boston, USA, 271p.

Ferreira, A. B. H. (1999), Novo Aurélio Século XXI: o dicionário da língua portuguesa, Editora Nova Fronteira, 1999, 2128p.

Fewster, M. e Graham, D. (2000), Software Test Automation, Addison Wesley.

Garfinkel, S. (2005), History's Worst Software Bugs. Disponível em: http://www.wired.com/software/coolapps/news/2005/11/69355?currentPage=1. Acesso em: 01/10/2008.

Gelperin, D. (1982) “A Software Test Documentation Standard”. ACM Proceedings of the 1st annual International Conference on Systems Documentation, p. 61-63.

Gills, M. (2005) “Software Testing and Traceability”. Doctoral Thesis. University of Latvia, Latvia (Letônia).

Gotel, O.C.Z. e Finkelstein, C.W. (1994) “An analysis of the requirements traceability problem. Requirements Engineering”, Proceedings of the First International Conference on Requirements Engineering, pp. 94 – 101, April.

Gotel, O. e Finkelstein, A. (1995) “Contribution Structures”, Proceedings of 2nd International Symposium on Requirements Engineering. IEEE Computer Society Press, pp. 100-107, March.

Hay, D.C. (1999), Princípios de Modelagem de Dados. Makron Books.

Herbert, J.S. (2000) Teste Cooperativo de Software. Porto Alegre, RS: PGCC da UFRGS. Tese de Doutorado.

Huckle, T. (2008), Collection of Software Bugs. Institut für Informatik. Disponível em: http://www5.in.tum.de/~huckle/bugse.html. Acesso em: 01/11/2008.

IBM (2001), "Use Case Management with Rational Rose and Rational RequisitePro". Disponível em: http://www.ibm.com/developerworks/rational/library/835.html. Acesso em: 05/01/2009.

IEEE (1990), The Institute of Electrical and Electronics Engineers, "IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries", IEEE Computer Society.

IEEE (1991), The Institute of Electrical and Electronics Engineers, "IEEE Std 610: Standard Computer Dictionary", IEEE Computer Society.

IEEE (1998a), The Institute of Electrical and Electronics Engineers, "IEEE Std 830-1998: IEEE Recommended Practice for Software Requirements Specifications", IEEE Computer Society, Junho.

IEEE (1998b), The Institute of Electrical and Electronics Engineers, "IEEE Std 829-1998: Standard for Software Test Documentation", IEEE Computer Society, Setembro.

IEEE (2008), The Institute of Electrical and Electronics Engineers, "IEEE Std 829-2008: Standard for Software and System Test Documentation", IEEE Computer Society, Julho.

ISO (1995), ISO/IEC 12207 – Information Technology – software life cycle processes. International Standard Organization.

Jarke, M. (1998) "Requirements tracing", Communications of the ACM, Vol 41, Issue 12, December 1998, 32-36.

Kaner, C., Bach, J., Pettichord, B. (2001), Lessons Learned in Software Testing, Wiley.

Kit, E. (1995), Software Testing in the Real World, Addison-Wesley.

Kotonya, G. e Sommerville, I. (1998), Requirements Engineering – Processes and Techniques, John Wiley & Sons.

Kruchten, P. (2003), The Rational Unified Process: an Introduction - Third Edition, Addison-Wesley.

Leffingwell, D. e Widrig, D. (2000), Managing Software Requirements – A Unified Approach. Addison-Wesley.

Lewis, W. E. (2000), Software Testing and Continuous Quality Improvement. Auerbach, 620p.

Maletic, J.I., Munson, E.V., Marcus, A., Nguyen, T.N. (2003), "Using a Hypertext Model for Traceability Link Conformance Analysis" in Proceedings of the 2nd International Workshop on Traceability in Emerging Forms of Software Engineering (TEFSE '03) (Montreal, Canada, October 7th), pp. 47-54.

Marciniak, J. J. (1994), Encyclopedia of Software Engineering v. II. John Wiley & Sons.

Mosley, D.J. (1993), The Handbook of MIS Application Software testing, Prentice-Hall.

Myers, G. J. (1979), The Art of Software Testing, Wiley.

Neto, A. C. D. e Travassos, G. H. (2006) "Maraká: Infra-estrutura Computacional para Apoiar o Planejamento e Controle dos Testes de Software", V Simpósio Brasileiro de Qualidade de Software – SBQS´2006.

Paulk, M.C., Curtis, B., Chrissis, M.B., Weber, C.V. (1993), Capability Maturity Model, Version 1.1, CMU/SEI-93-TR-24.

Perry, W. E. (2006), Effective Methods for Software Testing, 3rd Ed. John Wiley & Sons, 973p.

Pfleeger, S.L. e Bohner, S. A. (1990) "A framework for software maintenance metrics", IEEE Conference on Software Maintenance, Novembro.

Pfleeger, S.L. (2004), Engenharia de Software – Teoria e Prática. Prentice-Hall, 576p.

Phillips, D. (1998), The Software Project Manager's Handbook. IEEE Computer Society.

Pinheiro, F.A.C. e Goguen, J.A. (1996) "An Object-Oriented Tool for tracing Requirements", IEEE Software, March, 1996.

Pohl, K. (1996) "PRO-ART: Enabling Requirements Pre-Traceability", Proceedings of the IEEE Intl. Conference on Requirements Engineering (ICRE'96), April 15–28th, 1996.

Pressman, R.S. (2002), Engenharia de Software, McGraw-Hill.

Ramesh, B. e Jarke, M. (2001) "Toward reference models for requirements traceability", IEEE Transactions on Software Engineering, Vol. 27, Issue 1, January, pp. 58 – 93

Reed, K. (2000) *Software Engineering - a new millenium?* IEEE Software, jul.- ago.

Robertson, S. e Robertson, J. (1999), Mastering the Requirements Process, Addison-Wesley.

Rocha, A.R.C.; Maldonado, J.C.; Weber, K.C. (2001), Qualidade de Software – Teoria e Prática. Prentice-Hall, São Paulo, Brasil.

Sametinger, J. e Riebisch, M. (2002), "Evolution support by homogeneously documenting patterns, aspects and traces". In: Proceedings of the 6th European Conference on Software Maintenance and Reengineering.

Sommerville, I. (2003), Engenharia de Software, 6a Edição, Pearson Education do Brasil.

Spence, I. e Probasco, L. (1998), "Traceability Strategies for Managing Requirements with Use Cases", Rational Software White Paper.

Tassey, G. (2002), The Economic Impacts of Inadequate Infrastructure for Software Testing - Final Report, NIST - National Institute of Standards and Technology.

Telelogic (2007), Telelogic DOORS. Disponível em: http://www.telelogic.com/products/doors/index.cfm. Acesso em: 07/01/2008.

Visconti, M. e Cook C. R. (2002), "An Overview of Industrial Software Documentation Practice", In: Proceedings of the XXII International of the Chilean Computer Science Society (SCCC'02).

Watkins, R. e Neal, M. (1994) "Why and how of requirements tracing", IEEE Software, Vol. 11, Issue 4, July, pp. 104 – 106.

Westfall, L. (2001) How to Create Useful Software Process Documentation, International Conference on Software Quality (11th ICSQ), Pittsburgh, PA, Vol. 11, pp. 1-13

Wiegers, K. E. (2003), Software Requirements – Second Edition, Microsoft Press.