ELISA CERRI E CERRI

UM MODELO DE RASTREABILIDADE ENTRE O DOCUMENTO DE ESPECIFICAÇÃO DE REQUISITOS E O MODELO DE CASOS DE USO DO SISTEMA

Dissertação apresentada como requisito parcial a obtenção do grau de Mestre em Ciência da Computação, pelo Programa de Pós-Graduação em Ciência da Computação da Faculdade de Informática, Pontifícia Universidade Católica do Rio Grande do Sul.

Orientador: PROF. DR. RICARDO MELO BASTOS

PORTO ALEGRE
MARÇO DE 2007

Pontifícia Universidade Católica do Rio Grande do Sul

Dados Internacionais de Catalogação na Publicação (CIP)

C417m Cerri, Elisa Cerri e
Um modelo de rastreabilidade entre o documento de especificação de requisitos e o modelo de casos de uso do sistema / Elisa Cerri e Cerri. – Porto Alegre, 2007.
190 f.

Diss. (Mestrado) – Fac. de Informática, PUCRS
Orientador: Prof. Dr. Ricardo Melo Bastos

1. Informática. 2. Engenharia de Requisitos. 3. Software. I. Título.

CDD 005.1

Ficha Catalográfica elaborada pelo
Setor de Tratamento da Informação da BC-PUCRS

PUCRS
Campus Central
Av. Ipiranga, 6681 – prédio 16 – CEP 90619-900
Porto Alegre – RS – Brasil
Fone: +55 (51) 3320-3544 – Fax: +55 (51) 3320-3548
Email: bceadm@pucrs.br
www.pucrs.br/biblioteca

Pontifícia Universidade Católica do Rio Grande do Sul
FACULDADE DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

TERMO DE APRESENTAÇÃO DE DISSERTAÇÃO DE MESTRADO

Dissertação intitulada ***"Um Modelo de Rastreabilidade Entre o Documento de Especificação de Requisitos e o Modelo de Casos de Uso do Sistema"***, apresentada por Elisa Cerri e Cerri, como parte dos requisitos para obtenção do grau de Mestre em Ciência da Computação, Sistemas de Informação, aprovada em 08/03/2007 pela Comissão Examinadora:

Prof. Dr. Ricardo Melo Bastos – Orientador — PPGCC/PUCRS

Prof. Dr. Toacy Cavalcante de Oliveira – — PPGCC/PUCRS

Prof. Dr. Sérgio Crespo Coelho da Silva Pinto– — UNISINOS

Homologada em..12../..12../..07...., conforme Ata No. ..025.... pela Comissão Coordenadora.

P/ Fernando Luís Dotti

Prof. Dr. Fernando Luís Dotti
Coordenador.

PUCRS
Campus Central
Av. Ipiranga, 6681 – P. 16 – sala 106 – CEP: 90619-900
Fone: (51) 3320-3611 – Fax (51) 3320-3621
E-mail: ppgcc@inf.pucrs.br
www.pucrs.br/facin/pos

Aos meus pais, aos meus irmãos e ao meu noivo,
pelo apoio e incentivo quando mais precisei!

AGRADECIMENTOS

Gostaria muito de agradecer, em primeiro lugar, a Deus e aos espíritos amigos que sempre estiveram comigo. É impossível deixar de fazer este agradecimento pois tenho certeza que foi Deles que recebi o maior incentivo, o maior apoio e toda a força que em alguns momentos nem eu mesma sei de onde tirei.

Aos meus pais, Geraldo e Carime, e aos meus irmãos, Livia e Arthur, por entenderem todas as vezes que minha ausência se fez necessária, por entenderem que eu precisava atingir este objetivo de vida e, acima de tudo, pelo incentivo em continuar, pelo amor dispensado e pelo carinho emocionante com o qual sempre me receberam nas vezes em que tive que me desvencilhar dos trabalhos a fim de matar uma saudade sufocante!

Ao meu noivo, Bruno, por ter estado, durante este último ano, o tempo todo do meu lado, me incentivando a continuar e me mostrando que valeria a pena! Obrigada, meu lindo, pelo teu amor, pelo teu carinho, pela tua amizade e por tudo que fazes sempre por mim! Só posso dizer... Te amo!

À minha grande amiga Fabiana Rocha, uma amizade de infância nascida durante o mestrado! Quantas risadas, quantas angústias, quantas festas, quanto desespero... mas valeu a pena, né amiga? J

Às amigas Eliana Pereira e Ana Paula Lemke pelas noites não dormidas, pelas festas e pela parceria, principalmente no primeiro ano. Vocês são exemplos de dedicação e de amizade!

Ao meu orientador Prof. Dr. Ricardo Melo Bastos, pela paciência e dedicação com que me orientou nestes dois anos.

Ao Prof. Dr. Toacy Cavalcante, pelo apoio dispensado durante o estágio docência e pela disposição em colaborar com a pesquisa!

Ao Prof. MsC. Rafael Prikladnicki, pela amizade e por sempre estar disponível quando precisei!

Ao Convênio Dell/PUCRS pelo apoio financeiro para realização deste trabalho.

E, por fim, agradeço a todos aqueles que me incentivaram a deixar a monotonia de lado e sair em busca de novos desafios! Obrigada a todos vocês!

“Muitas vezes, as pessoas tentam viver a vida às avessas: elas procuram ter mais coisas ou mais dinheiro, para poderem fazer o que querem, de modo que possam ser felizes. A coisa deve funcionar ao contrário: você primeiramente precisa ser quem você realmente é, para então fazer o que precisa ser feito, a fim de ter o que você deseja."

(Shakti Gawain)

RESUMO

Um produto final que represente realmente as necessidades dos usuários é, atualmente, uma busca constante das organizações de desenvolvimento de software. Um ponto crucial para o alcance deste objetivo é realizar um processo de requisitos que considere as necessidades organizacionais do início ao fim. O documento de especificação de requisitos (SRS) é utilizado para formalizar as necessidades dos clientes e o modelo de casos de uso (MCU) serve como base para o processo de desenvolvimento. Sendo assim, é importante que a SRS contenha todas as funcionalidades requisitadas pelos usuários e que o MCU represente-a fielmente, contendo todos os requisitos nela estabelecidos. Neste contexto, este trabalho apresenta um modelo de rastreabilidade entre o documento de especificação de requisitos e o modelo de casos de uso do sistema. Este modelo foi desenvolvido com base nos modelos conceituais destes artefatos e nas necessidades organizacionais, identificadas através de um estudo de caso realizado em uma organização de desenvolvimento de software. Sua principal contribuição é auxiliar a atividade de engenharia de requisitos, mantendo a integridade entre os artefatos gerados quando da mudança dos requisitos do sistema.

Palavras-chave: Engenharia de Requisitos, Gerência de Requisitos, Especificação de Requisitos, Modelo de Casos de Uso, SRS, Rastreabilidade.

ABSTRACT

To produce the right software is, nowadays, the main goal of the software development organizations. For that, the whole point is to have a requirements process that involves the organization's needs all the time. The software requirements specifications (SRS) is used to formalize the user's needs and the use case model (UCM) is used during the development process. So, the SRS should involve all the functionalities that reflect a user's true needs and the UCM should have all the requirements specified in that. In this context, this research presents a traceability model linking software requirements specification and system use case model. This model was developed starting from conceptual models of these artifacts and business needs, identified in a case study at a software development organization. The main contribution of this model is to help the requirements engineering activity, keeping the integrity between the generated artifacts when the system requirements are changed.

Keywords: Requirements Engineering, Requirements Management, Requirements Specification, Use Case Model, SRS, Traceability.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

AGRT	Actor Goal Reading Technique
BDL	Banco de Dados Lógico
CDPE	Centro de Desenvolvimento e Pesquisa Dell/PUCRS
DR	Documento de Requisitos
E/S	Entrada/Saída
ER	Engenharia de Requisitos
FAO	Formulário Ator x Objetivo
GUCCRA	Guidelines for Use Case Construction and Requirements document Analysis
IE	Interface de Entrada
IS	Interface de Saída
LN	Linguagem Natural
MCU	Modelo de Casos de Uso
MR	Modelo de Rastreabilidade
PPGCC	Programa de Pós Graduação em Ciência da Computação
RNE	Regra de Negócio Específica
RNFE	Requisito Não-Funcional Específico
RNFG	Requisito Não-Funcional Geral
RNG	Regra de Negócio Geral
SRS	Software Requirements Specification
SsD	Sistema Sob Discussão
TUCCA	Técnica de Leitura para apoiar a Construção de Modelos de Casos de Uso e Análise de Documentos de Requisitos
UCRT	Use Case Reading Technique
UML	Unified Modeling Language

SUMÁRIO

1 INTRODUÇÃO

Atualmente, as organizações de desenvolvimento de software estão cada vez mais interessadas em garantir que o software represente, realmente, uma solução para o sistema proposto. A maneira mais eficaz de alcançar este objetivo é a utilização de um processo de requisitos bem definido, visto que o motivo da maioria dos cancelamentos ou fracassos de projetos acontece por eles não atenderem completamente às necessidades dos clientes e excederem os prazos e os orçamentos estimados [STA95].

Embora não seja fácil, determinar requisitos é uma tarefa extremamente importante, pois eles formam a base para o planejamento, o acompanhamento do desenvolvimento e a aceitação dos resultados do projeto de software. Ter um processo bem definido significa ter maior possibilidade para se obter, entender e validar as necessidades e expectativas do cliente para, posteriormente, documentá-las [FIO98]. O principal objetivo deste processo é concluir, com êxito, um acordo entre quem solicita e quem desenvolve, estabelecendo clara e rigorosamente o que deverá ser produzido. Não utilizar requisitos é equivalente a pensar "não há tempo para descobrir o que se deve construir, devemos começar a construir agora" [HEU01].

O documento de especificação de requisitos é utilizado para formalizar as necessidades dos clientes e o modelo de casos de uso serve como base para o processo de desenvolvimento. Com isso, um ponto positivo para iniciar o processo de desenvolvimento é garantir que o documento de especificação de requisitos seja uma representação fiel daquilo que o usuário deseja e, com isso, desenvolver o modelo de casos de uso contendo todas as informações nele relacionadas.

Outro ponto importante é a rastreabilidade de requisitos, que vem sendo identificada na literatura como um fator de qualidade [POH96] e é vista como uma técnica fundamental no apoio às diversas atividades do projeto. Sua utilização facilita o controle de que sistemas e software estão em conformidade às mudanças dos requisitos [GOT97] e [RAM95].

Embora se saiba que o esforço necessário para sua aplicação pode ser elevado, como aumento de custo ou tempo, sem a utilização de mecanismos deste tipo podem surgir inconsistências durante o processo de adição, remoção ou modificação de requisitos. Ainda, de acordo com [KOT98], os requisitos não podem ser efetivamente gerenciados sem rastreabilidade.

Neste contexto, este trabalho apresenta um modelo de rastreabilidade entre o documento de requisitos, que se espera ser bem elicitado e conter todas as necessidades dos interessados, e o modelo de casos de uso, que deve conter todas as informações citadas neste documento. Seu

desenvolvimento se deu com base na literatura e no estudo de caso realizado.

Inicialmente o estudo da base teórica foi realizado e a partir dele foram encontradas lacunas na área e vários trabalhos relacionando problemas entre a má definição e especificação dos requisitos com os resultados não satisfatórios do produto final. A partir disto, e considerando o contexto de que o modelo de casos de uso fornece a base para o desenvolvimento de software, buscou-se identificar a relação entre os elementos dos dois documentos a fim de propor um modelo que permitisse que eles permanecessem consistentes durante todo o processo de desenvolvimento.

Neste sentido, foram desenvolvidos um modelo de rastreabilidade e um processo de apoio contendo relacionamentos, dependências e procedimentos para sua aplicação. Sua principal contribuição é permitir que os artefatos envolvidos desde o processo de levantamento de requisitos até o final do processo de desenvolvimento permaneçam consistentes mesmo quando alterações forem realizadas. A fim de suportar esses artefatos, um protótipo de uma ferramenta, também implementado no contexto desta pesquisa, é apresentado.

Com esta proposta, são oferecidos subsídios para manter os dois documentos consistentes entre si e, com isso, possibilitar que o resultado do processo seja satisfatório.

1.1 Questão de Pesquisa

A questão de pesquisa abordada no projeto aqui apresentado foi a seguinte:

Como garantir a integridade entre os artefatos do documento de especificação de requisitos e o modelo de casos de uso do sistema quando da alteração de algum de seus elementos?

1.2 Objetivos

Com base na contextualização apresentada na seção anterior, o objetivo geral desta pesquisa é propor um modelo de rastreabilidade entre o documento de especificação de requisitos e o modelo de casos de uso do sistema oferecendo mecanismos que mantenham a integridade entre os artefatos gerados quando da mudança nestes requisitos.

Para atender a este objetivo geral emergem os seguintes objetivos específicos:

- Estudo da base teórica, aprofundando conhecimentos sobre requisitos, processo de engenharia de requisitos e também em modelo de casos de uso, descrição de casos de uso, rastreabilidade e tópicos relacionados.

- Avaliar as propostas que relacionam o documento de requisitos com o modelo de casos de uso e, também, propostas que utilizem a rastreabilidade entre estes dois artefatos.
- Identificar os elementos do documento de especificação de requisitos a serem considerados para o desenvolvimento do modelo de caso de uso.
- Realização de um estudo de caso visando buscar elementos para o desenvolvimento do modelo de rastreabilidade, bem como identificar as semelhanças e diferenças entre as propostas encontradas na literatura e a prática nas empresas.
- Propor um modelo de rastreabilidade entre o documento de especificação de requisitos e o modelo de casos de uso.
- Propor um processo a ser utilizado junto com o modelo com a finalidade de contribuir para a realização da rastreabilidade.
- Verificar a aplicabilidade da proposta e os benefícios decorrentes de sua utilização.
- Desenvolver um protótipo para suportar o modelo e o processo propostos.

1.3 Organização do Volume

Este volume está organizado em oito capítulos. O capítulo 1 corresponde a esta introdução.

No capítulo 2 é apresentado o referencial teórico desta pesquisa, envolvendo os principais conceitos e áreas do estudo: requisitos, casos de uso e rastreabilidade.

O capítulo 3 apresenta os trabalhos relacionados. Foram selecionados artigos sobre estudos em áreas relacionadas ou complementares a pesquisa aqui apresentada. Durante este capítulo, são feitas considerações sobre estes trabalhos.

No capítulo 4 é detalhado o método de pesquisa utilizado. Nele são descritas todas as etapas do estudo e o desenho de pesquisa é apresentado.

O capítulo 5 descreve o estudo de caso realizado em uma empresa de desenvolvimento de software. Este estudo de caso teve como finalidade identificar os elementos presentes nos documentos de especificação de requisitos e a completude de suas informações, bem como, encontrar subsídios que indicassem se os casos de uso estavam corretamente relacionados aos requisitos presentes no documento.

No capítulo 6 é apresentado o modelo de rastreabilidade proposto. Neste capítulo são apresentados, também, os modelos conceituais do documento de especificação de requisitos e do

modelo de casos de uso, utilizados em conjunto com o modelo de rastreabilidade para a realização do processo. Ainda neste capítulo são apresentados todos os procedimentos que compõem o processo de rastreabilidade.

O capítulo 7 descreve a avaliação do modelo proposto através de sua aplicação em um documento de especificação de requisitos e em seu respectivo modelo de casos de uso, visando verificar a aplicabilidade da proposta e suas contribuições.

Por fim, no capítulo 8 são apresentadas as considerações finais deste volume. São descritas as contribuições deste estudo, bem como suas limitações e trabalhos futuros.

2 REFERENCIAL TEÓRICO

Esta seção descreve o referencial teórico deste trabalho apresentando os principais conceitos envolvidos com o tema de pesquisa. Serão apresentados, ainda, alguns trabalhos relacionados que serviram como referência para o desenvolvimento do processo.

2.1 Engenharia de Requisitos

Antes do desenvolvimento de qualquer sistema, é necessário entender qual o seu objetivo e como seu uso poderá ajudar nas metas dos clientes ou do negócio. Isto envolve compreender o domínio da aplicação (telecomunicações, jogos, varejo, e etc.), as restrições do sistema operacional, a especificação das funcionalidades requeridas pelos stakeholders[1] e as características essenciais do sistema como: performance, segurança, e dependência. De acordo com [KOT98] e [SOM97], a Engenharia de Requisitos (ER) é um termo que engloba todas as atividades envolvidas na descoberta, documentação e manutenção de um conjunto de requisitos para um sistema computacional.

O sucesso de um sistema de software é determinado pela maneira com que ele satisfaz as necessidades do cliente, onde esta satisfação é determinada pela forma com que a solução proposta está relacionada com as necessidades apontadas pelos stakeholders [NUS00]. O esforço de desenvolvimento é total ou parcialmente desperdiçado se o software construído não corresponde aos objetivos propostos. Além disso, se a base tecnológica (hardware, software e dispositivos) necessária ao software não for compatível com a base existente onde ele será utilizado, o esforço de desenvolvimento pode ser completamente perdido.

Para atingir o sucesso, é fundamental identificar e documentar as necessidades e objetivos do software. Esta tarefa exige o entendimento do ambiente onde o software será inserido, considerando as características de negócio, as possíveis mudanças e as necessidades reais envolvidas no processo. Os requisitos estão associados aos principais problemas do desenvolvimento de software, visto que requisitos mal elicitados e mal especificados podem acarretar em um resultado diferente do esperado pelo usuário.

Não existe maneira incontestável de assegurar que uma especificação de software satisfaz completamente as necessidades e características desejadas pelo cliente. Na tentativa de evitar este

[1] Um stakeholder é qualquer grupo ou indivíduo que pode afetar ou ser afetado pela obtenção dos objetivos da organização ([FRE84] apud [SHA99]).

tipo de problema é importante que as organizações disponham de um processo de requisitos bem definido, controlado, medido e aprimorado para guiar os desenvolvedores na árdua tarefa de definição de requisitos de um sistema [BLA05], já que a engenharia de requisitos vem sendo considerara, por vários autores, como a disciplina mais importante da engenharia de software [PRE01].

Nesta seção será apresentado o processo de engenharia de requisitos, alguns conceitos relacionados ao tema da pesquisa e a contextualização deste trabalho no âmbito da ER.

2.1.1 Processo de Engenharia de Requisitos

O processo fundamental de ER varia dependendo do tipo de aplicação que será desenvolvida, do tamanho e da cultura das companhias envolvidas e do processo de aquisição de software utilizado, porém, independente do tipo de processo realizado, algumas atividades são fundamentais: o reconhecimento do problema, a interpretação das necessidades que demandam o sistema, a modelagem, a especificação e a validação dos requisitos que atendem estas necessidades [PRE01].

Como em qualquer processo, o processo de ER pode apresentar várias dificuldades. A maior delas é a definição precisa sobre o que construir. Este é um problema antigo e que está presente até hoje no processo de software. Nenhuma outra parte é mais difícil de ser corrigida tardiamente [BER98].

De um modo geral, segundo Nuseibeh [NUS00], o processo de ER identifica os stakeholders, suas necessidades e as documenta de forma que possam ser analisadas e discutidas para a implementação.

De acordo com Pressman [PRE01], uma completa compreensão dos requisitos é fundamental para um desenvolvimento de um software bem sucedido. Não importa quão bem projetado ou quão bem codificado seja, um programa mal analisado e mal especificado desapontará o usuário e trará aborrecimentos para o desenvolvedor.

Independente do tipo de processo utilizado, algumas atividades são fundamentais [SOM05]:

- Elicitação: nesta fase são identificadas as informações sobre o sistema baseando-se nas necessidades e expectativas dos stakeholders.
- Análise: os requisitos iniciais, obtidos através da fase de elicitação, são utilizados como base para esta análise. Eles são distribuídos em categorias, as relações entre eles são exploradas e são classificados conforme sua importância, de acordo com as necessidades dos stakeholders.

- Validação: os requisitos são examinados quanto a sua pertinência, consistência e integralidade. Deve ser assegurado que todos os erros tenham sido detectados e corrigidos.
- Negociação: inevitavelmente as opiniões e exigências dos stakeholders poderão ser conflitantes. A negociação é realizada para decidir quais os requisitos devem ser aceitos, de forma a obter consenso para que seja gerado um conjunto consistente de requisitos.
- Documentação: os requisitos devem ser escritos de forma que os stakeholders e os desenvolvedores de software possam compreendê-los. Em geral é produzido um documento de especificação de requisitos.
- Gerenciamento: é responsável por controlar as modificações nos requisitos, que inevitavelmente acontecerão. Diversas pesquisas como as apresentadas em [MCC96] e [LEF00] apontam que os principais motivos que levam a atrasos na entrega, aumento nos custos, e funcionalidade deficiente de um software são todos relacionados com práticas de gerência de requisitos.

Algumas vezes estas atividades são apresentadas como se ocorressem em seqüência, onde a elicitação seria a primeira a ser realizada e a documentação, em conjunto com o gerenciamento de mudanças nos requisitos, seria a última. Entretanto a ER é sempre uma atividade cíclica [SOM05], conforme é apresentado na Figura 1.

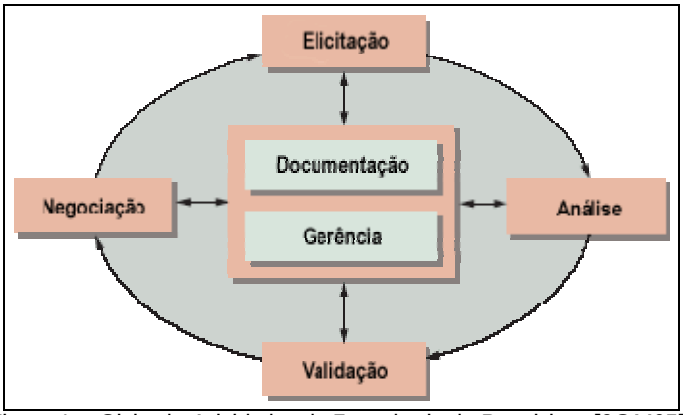


Figura 1 – Ciclo de Atividades da Engenharia de Requisitos [SOM05].

O foco deste trabalho está no resultado do processo de ER, o documento de requisitos, que define o que será implementado. Com isso, nota-se que se este documento não tiver bem especificado, a implementação pode não ser adequada.

2.1.2 Requisitos

Os requisitos são um componente fundamental no processo de software. Não utilizar requisitos é equivalente a pensar “não há tempo para descobrir o que se deve construir, devemos começar a construir agora” [HEU01].

Apesar de a literatura apresentar diversas definições para requisitos, neste trabalho optou-se por utilizar a definição de Goguen [GOG96] que diz que os requisitos são propriedades que um software deve possuir para funcionar com sucesso no ambiente para o qual foi proposto, considerando tanto o contexto social quanto o contexto técnico do ambiente, o que tem se mostrado importante na utilização de requisitos, pois muitas das informações utilizadas por eles estão ligadas ao ambiente social dos usuários e gerentes.

Alguns dos problemas que surgem durante o processo de engenharia de requisitos, resultam da falta de uma separação entre os níveis de descrição de requisitos. Os termos requisitos de usuário e requisitos de sistema são utilizados por [SOM03] para designar os requisitos abstratos de alto nível e para indicar a descrição detalhada do que o sistema deverá fazer, respectivamente. Este trabalho está focado, principalmente, na especificação dos requisitos do sistema.

Freqüentemente os requisitos de software são classificados em funcionais (declarações das funções que o sistema deve oferecer) e não-funcionais (restrições sobre os serviços ou funções oferecidas pelo sistema; como o software se comporta em relação a atributos observáveis como performance e confiabilidade, por exemplo.). Essa divisão é apresentada por diversos autores como [NUS00], [FRA98] e [KRU01].

Os requisitos funcionais representam a descrição das diversas funções que os clientes e os usuários querem ou precisam que o sistema ofereça. Dependem do tipo de software que está sendo desenvolvido. Em princípio devem ser escritos de forma completa e consistente, onde a completeza significa que todas as funções requeridas pelo usuário devem estar descritas e a consistência significa que os requisitos não devem ter definições contraditórias [SOM05].

Os requisitos não-funcionais são as propriedades de um software, como manutenibilidade, usabilidade, desempenho, custos, entre outros. Podem definir, também, restrições para o sistema, como capacidade de dispositivos E/S (entrada/saída) e as representações de dados utilizados nas interfaces do sistema.

Os requisitos não-funcionais surgem conforme a necessidade dos usuários, em razão de restrições de orçamento, de políticas organizacionais, pela necessidade de interoperabilidade com outros sistemas de software ou hardware, ou devido a fatores externos [SOM03].

A falha em se cumprir um requisito não-funcional pode ser muito pior do que a falha em se cumprir um requisito funcional. Muitos requisitos não-funcionais dizem respeito ao sistema como um

todo. Enquanto a falha em um requisito funcional pode degradar o sistema, a falha em um requisito não-funcional pode tornar todo o sistema inútil. Por exemplo, se um sistema de aviação não atender aos seus requisitos de confiabilidade, ele não será atestado como seguro [SOM03].

2.1.3 Contextualização

No contexto da engenharia de requisitos, este trabalho se enquadra na fase de especificação (documentação) e de gerenciamento de requisitos.

Desenvolver uma solução para um problema específico e não uma solução para um problema qualquer é uma dificuldade ainda constante nos processos de desenvolvimento de software. A fim de solucionar este problema, foi desenvolvido um modelo de rastreabilidade que busca fornecer meios para que todas as funcionalidades levantadas sejam realmente implementadas.

Assim, este trabalho possibilita que um processo consistente de levantamento de requisitos (elicitação) contribua para um processo completo de desenvolvimento de software. Tendo um documento de requisitos bem elicitado, que compreenda todas as funcionalidades necessárias para a satisfação do usuário, e fornecendo um processo para auxiliar que todas as funcionalidades presentes neste documento sejam completamente implementadas, as possibilidades de um resultado de sucesso crescem consideravelmente.

Este modelo permite averiguar se todas as funcionalidades descritas no documento de requisitos estão representadas no modelo de casos de uso e permite, também, verificar a consistência entre os elementos dos dois documentos durante o processo de mudança nos requisitos.

2.2 Especificação de Requisitos

Uma vez identificados e negociados, os requisitos devem ser documentados para que possam servir de base para o restante do processo de desenvolvimento [NAD02]. Um ponto importante, a ser sempre considerado, é que os requisitos devem ser escritos de forma que os stakeholders e os desenvolvedores de software possam compreendê-los da mesma maneira.

A documentação pode ser representada através das especificações de requisitos, que podem ser classificadas em: informais, semi-formais e formais. As informais são as representações em linguagem natural, as semi-formais podem ser representadas, por exemplo, por linguagens gráficas e as formais são representações matemáticas para o sistema [SOM03]. O estudo completo sobre os tipos de especificação de requisitos pode ser conferido em [CER05a].

Neste trabalho, serão utilizadas as representações informais, fazendo uso de linguagem

natural e as representações semi-formais, através dos modelos e descrições de casos de uso.

Os requisitos de sistema são geralmente documentados em um documento formal que é usado como meio de comunicação entre os stakeholders [KOT98]. Não há um padrão de nome para este documento, em organizações distintas ele pode ter nomes diferentes como documento de requisitos, especificação funcional, especificação de requisitos, etc. [KOT98]. Independente de nome, este documento deve trazer informações importantes como: os serviços e as funções que o sistema deve fornecer, as restrições sobre as quais ele deve operar, propriedades gerais do sistema, definições de outros sistemas que interagem com ele, etc.

A IEEE [IEE98] define o nome "Especificação de Requisitos de Software" (SRS – Software Requirements Specification), para este documento e este trabalho foi realizado com base neste padrão, que será apresentado a seguir.

2.3 SRS

A SRS é uma especificação para um produto de software específico, para um programa ou para um conjunto de programas que executam certas funções em um determinado ambiente. Deve ser escrito por um ou mais representantes do fornecedor, um ou mais representantes dos clientes ou por ambos [IEE98].

Uma razão para que estes documentos sejam frequentemente incompletos, é que o levantamento dos requisitos geralmente segue uma rota improdutiva. Por exemplo, alguns requisitos podem ser ignorados e, neste caso, o time de desenvolvimento desconsidera a solução, resultando em um design baseado em suposições de como o sistema deve funcionar [KUL03]. Documentos de requisitos mal escritos – muito longos ou voltados à visão de uma única pessoa, por exemplo – podem levar ao cancelamento de um projeto [IEE98].

Alguns pontos como documentar as necessidades dos usuários, evitar deduções prematuras de design, apresentar requisitos não conflitantes e não redundantes, devem sempre ser considerados para alcançar a qualidade de uma SRS. Uma maneira de garantir tais aspectos é permitir uma rastreabilidade entre os requisitos [KUL03]. Para isto, é fundamental que todos os passos do projeto sejam documentados relatando, por exemplo, a qual requisito uma classe específica se refere, qual caso de teste reflete um requisito específico e assim por diante. Tais cuidados podem garantir que uma mudança em determinado ponto não se torne uma catástrofe para o projeto final, já que se terá documentado quais artefatos serão atingidos com tal procedimento.

Existem diversas maneiras de escrever uma SRS, geralmente elas são adaptadas à realidade do projeto e/ou da empresa. Infelizmente, nem sempre seguem um padrão ideal e nem sempre

consideram todos os aspectos necessários para que o documento possa ser considerado completo. Com isso, o restante do processo fica debilitado sendo que se determinadas características foram esquecidas ou expressas de maneira inadequada, o resultado pode não ser o esperado.

Segundo a IEEE [IEE98], a SRS deve estar apta a responder algumas questões, tais como:

- O que o software pretende fazer?
- Como o software interage com as pessoas, com o hardware do sistema, outros hardwares e outros softwares?
- Qual a velocidade, disponibilidade, tempo de resposta, tempo de restauração, recuperação das várias funções do software?
- Qual a portabilidade, conectividade, manutenção, segurança?
- Há padrões efetivados, linguagem implementada, políticas de integridade para banco de dados, limites de recursos, ambientes operacionais?

Analisando as questões acima, pode-se perceber que elas estão relacionadas a alguns aspectos como funcionalidade, interfaces externas, performance, atributos e limites de implementação, respectivamente.

De acordo com a IEEE [IEE98] para que possa ser considerada boa, uma SRS deve ser: correta, não ambígua, completa, consistente, superior pela importância e/ou estabilidade, verificável, modificável e rastreável.

- Correta: uma SRS é correta se, e somente se, cada requisito expresso for encontrado também no software. Alternadamente o cliente ou usuário pode determinar se a SRS reflete corretamente as necessidades atuais.
- Não ambígua: uma SRS não é ambígua se, e somente se, cada requisito declarado seja suscetível a apenas uma interpretação. Isso requer, no mínimo, que cada característica do produto final utilize um único termo[2]. É fundamental que a SRS seja clara para aqueles que a criam e para aqueles que a usam.
- Completa: uma SRS é completa se, e somente se, inclui todos os requisitos significantes, inclui a definição das respostas do software a todos os tipos de dados de entrada em todas as situações possíveis, válidas ou inválidas, e se referencia todos os termos, figuras e/ou diagramas, além de unidades de medida;

[2] No caso onde o termo usado pode ter múltiplo significado, ele deve ser incluído em um glossário onde seu significado é obtido mais especificamente [IEE98].

- Consistente: uma SRS é consistente se, e somente se, nenhum dos requisitos do documento, tomado individualmente, está em conflito com qualquer outro requisito do mesmo documento. Alguns conflitos comuns podem ser os seguintes:
    1. Formatos de relatórios de saída com padrões diferentes;
    2. Requisitos especificando adição de determinadas entradas enquanto outros especificam que será realizada a multiplicação delas;
    3. Requisitos descrevendo o mesmo objeto, porém usando termos diferentes para ele.
- Classificação: uma SRS ocupa um lugar de destaque pela importância e/ou estabilidade se existem indicações no documento quanto à importância ou estabilidade do requisito;
- Verificável: uma SRS é verificável se, e somente se, para cada um dos requisitos contidos no documento, existe um processo finito e economicamente viável através do qual uma pessoa ou máquina possa assegurar que o produto de software atende ao requisito;
- Modificável: uma SRS é modificável se, e somente se, modificações possam ser agregadas ao documento de forma fácil, completa e consistente, com relação a sua estrutura e estilo;
- Rastreável: uma SRS é rastreável se, e somente se, a origem de cada um de seus requisitos é clara e a referência a cada um deles é facilitada nos documentos subseqüentes do processo ou em uma melhoria da documentação do sistema (através da numeração dos requisitos, por exemplo).

Além disso, a IEEE [IEE98] sugere várias seções a serem utilizadas neste documento. Estas seções foram interpretadas e a partir delas foi desenvolvido um modelo conceitual para a SRS, detalhado na seção 4.1.1, a fim de facilitar a identificação de seus elementos no momento da realização da rastreabilidade.

Kotonya & Sommerville [KOT98] afirmam que a linguagem natural é a única notação existente passível de interpretação por todos os potencias leitores do documento de requisitos, porém, reconhecem que podem ser ambíguas e mal interpretadas. Recomenda-se, então, que sejam usadas técnicas, no mínimo, estruturadas sempre que não se possa correr o risco de ter o requisito mal interpretado.

Assim, para este trabalho optou-se pela utilização de uma estrutura para especificação de requisitos em linguagem natural. Seu desenvolvimento se deu em um projeto de pesquisa realizado

no CDPe (Centro de Desenvolvimento e Pesquisa Dell/PUCRS) e, atualmente, é utilizada em vários projetos desta empresa. Esta estrutura é apresentada em detalhes na próxima seção.

2.4 Estrutura para especificação de requisitos em linguagem natural

A estrutura para especificação de requisitos em linguagem natural foi inicialmente estruturada por meio de uma pesquisa desenvolvida no Centro de Desenvolvimento e Pesquisa DELL/PUCRS (CDPe). A idéia de seu desenvolvimento surgiu devido a diversos desafios identificados na engenharia de requisitos em ambientes distribuídos de software, que podem ser conferidos em [LOP04]. Esta estrutura é atualmente utilizada em projetos da empresa participante do estudo de caso, e por este motivo pode ser adicionada ao contexto deste trabalho. A pesquisa tratou, especificamente, os seguintes pontos:

- Ambigüidade e falta de clareza;
- Múltiplas visões de requisitos;
- Múltiplos níveis de detalhe na especificação;
- Estimativas e cálculos de métricas;

O modelo inclui um metamodelo de requisitos e uma estrutura textual. O metamodelo de requisitos apresenta um conjunto de definições utilizadas para classificar as informações obtidas durante a especificação, bem como seus relacionamentos. A estrutura textual define estruturas de frase, para cada tipo de informação, com o objetivo de simplificar sua compreensão.

O metamodelo, apresentado na Figura 2, visa tratar problemas causados pelas múltiplas visões de requisitos – fornecendo um conjunto de definições a ser utilizado junto com a estrutura de texto – bem como estruturar o relacionamento entre as informações obtida.

Os autores concluíram, através de um estudo de caso, que este modelo é importante para reduzir as diferenças no nível de detalhe das especificações entre as equipes e por isso ele foi implantado e é, atualmente, utilizado na maioria dos projetos da empresa.

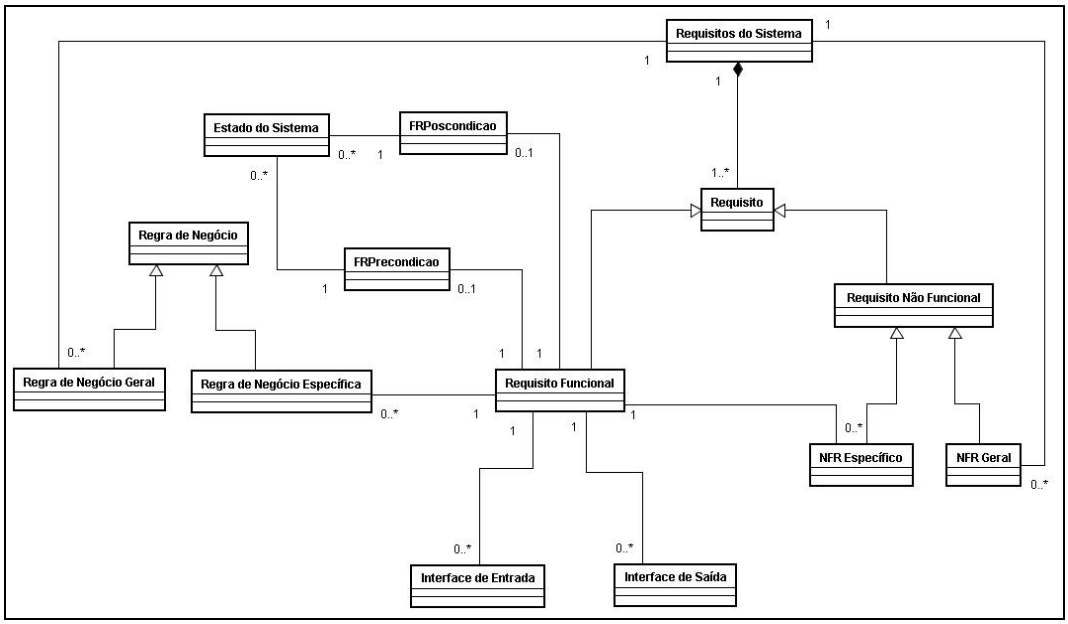


Figura 2 – Meta-modelo de requisitos.

A estrutura de frase é composta de campos padrão que devem ser preenchidos de acordo com regras para cada categoria de informação. Quando a estrutura de frase é preenchida, a frase final é obtida concatenando-se os campos em uma ordem pré-definida. O resultado final representa a informação desejada. Esta estrutura tem como objetivo, neste trabalho, oferecer mais possibilidades para que a técnica de rastreabilidade entre os requisitos e os casos de uso possa ser aplicada efetivamente.

A estrutura de frase principal é composta pelos elementos mostrados na Figura 3.

ESPECIFICAÇÃO DE REQUISITOS	
Identificação da Informação (obrigatório):	
Categoria da informação (obrigatório):	
Origem:	
1. Ator/iniciador da ação (obrigatório):	
2. Condição:	
3. Ação (obrigatório):	
4. Objeto da ação (obrigatório):	
5. Refinamento (ou destino) do objeto:	
6. Refinamento (ou destino) da ação:	
7. Pós-condições:	
Frase final:	

Figura 3 – Estrutura de frases em linguagem natural.

- Identificação da informação (obrigatório): utilizado para identificação da informação. Será representado pelo número correspondente à informação a ser registrada na SRS;
- Categoria da informação (obrigatório): categoria de acordo com o metamodelo. Categorias válidas são: RF (requisito funcional), RNFG (requisito não-funcional geral),

RNFE (requisito não-funcional específico), RNG (regra de negócio geral), RNE (regra de negócio específica), IE (interface de entrada) e IS (interface de saída).

- Origem: Origem da informação. Pode ser uma pessoa, grupo, documentos ou uma reunião documentada;
- Ator/Iniciador da ação (obrigatório): define o sujeito da sentença;
- Condição: define as condições necessárias para que se execute a ação. Utilizado para as pré-condições em requisitos funcionais;
- Ação (obrigatório): verbo que define a ação envolvida;
- Objeto da ação (obrigatório): objeto sobre o qual ocorre a ação;
- Refinamento (ou destino) do objeto: qualifica em detalhe o objeto;
- Refinamento (ou destino) da ação: qualifica em detalhe a ação;
- Pós-condições: define o estado do sistema ao concluir a ação;

De acordo com a categoria de informação, algumas regras são definidas para guiar o preenchimento da estrutura. As regras são as seguintes:

2.4.1 Requisitos Funcionais

Quando preenchendo a estrutura de frases com requisitos funcionais, o campo "2 - Condição" deve conter as pré-condições; o campo "7 – Pós-condições" deve ser preenchido com as pós-condições, conforme apresentado no metamodelo. A frase final é obtida pela concatenação dos campos na seguinte seqüência: 1 + 3 + 4 + 5 + 6 + "." + 2 + "." + 7 + ".". O preenchimento é exemplificado na Figura 4.

ESPECIFICAÇÃO DE REQUISITOS	
Identificação da Informação (obrigatório):	1
Categoria da informação (obrigatório):	RF
Origem:	
1. Ator/iniciador da ação (obrigatório):	Usuário
2. Condição:	
3. Ação (obrigatório):	deve ser capaz de adicionar
4. Objeto da ação (obrigatório):	novas ordens de serviço
5. Refinamento (ou destino) do objeto:	
6. Refinamento (ou destino) da ação:	
7. Pós-condições:	Após a execução, o estoque deve ser atualizado.
Frase final:	Usuário deve ser capaz de adicionar novas ordens de serviço. Após a execução, o estoque deve ser atualizado.

Figura 4 – Requisito Funcional.

2.4.2 Requisitos Não-funcionais

Para escrita de requisitos não-funcionais na estrutura de frase principal, as seguintes regras devem ser observadas: o campo “1 - Ator” deve ser preenchido com “Sistema”; as condições que se aplicam ao requisito não-funcional, se houverem, devem ser preenchidas no campo “2 - Condição”; o campo “7 - Pós-condição” deve ser deixado em branco.

A frase final é obtida pela concatenação dos campos 2 + “,” + 1 + 3 + 4 + 5 + 6 + “.”. O preenchimento é exemplificado na Figura 5.

ESPECIFICAÇÃO DE REQUISITOS	
Identificação da Informação (obrigatório):	1
Categoria da informação (obrigatório):	RNFG
Origem:	
1. Ator/iniciador da ação (obrigatório):	Sistema
2. Condição:	
3. Ação (obrigatório):	deve ter
4. Objeto da ação (obrigatório):	uma interface amigável.
5. Refinamento (ou destino) do objeto:	
6. Refinamento (ou destino) da ação:	
7. Pós-condições:	
Frase final:	O sistema deve ter uma interface amigável.

Figura 5 – Requisito Não-Funcional.

2.4.3 Regras de Negócio

Quando preenchendo a estrutura de frases para representação de regras de negócio, as seguintes regras se aplicam: o campo “1 - Ator” deve ser preenchido com “Sistema”; as condições que se aplicam ao requisito não-funcional, se houverem, devem ser preenchidas no campo “2 - Condição”; o campo “7 - Pós-condição” deve ser deixado em branco.

A frase final é obtida pela concatenação dos campos 2 + “,” + 1 + 3 + 4 + 5 + 6 + “.”. A Figura 6 exemplifica o preenchimento de regras de negócio.

ESPECIFICAÇÃO DE REQUISITOS	
Identificação da Informação (obrigatório):	1
Categoria da informação (obrigatório):	RNG
Origem:	
1. Ator/iniciador da ação (obrigatório):	Sistema
2. Condição:	Se o pedido é para o governo
3. Ação (obrigatório):	deve calcular
4. Objeto da ação (obrigatório):	o valor total do pedido
5. Refinamento (ou destino) do objeto:	
6. Refinamento (ou destino) da ação:	somando os valores de cada item do pedido.
7. Pós-condições:	
Frase final:	Se o pedido é para o governo, o sistema deve calcular o valor total do pedido, somando os valores de cada item do pedido.

Figura 6 – Regra de Negócio.

2.4.4 Interfaces

Interfaces devem ser representadas usando as seguintes regras: o campo “1 - Ator” deve ser preenchido com “Sistema”; o campo “3 - Ação” deve ser preenchido com “deve obter” em interfaces de entrada ou com “deve fornecer” em interfaces de saída; o campo “4 - Objeto da ação” deve ser preenchido com “a seguinte informação como entrada para {<ação>, objeto} ({obrigatória, opcional}):” para interfaces de entrada, ou “a seguinte informação como saída para {<ação>, objeto} ({obrigatório, opcional})” para interfaces de saída. O campo “5 - Refinamento do Objeto” deve ser preenchido com as informações requeridas pela interface. O campo “7 - Pós-condição” deve ser deixado em branco. A Figura 7 apresenta uma interface de entrada.

ESPECIFICAÇÃO DE REQUISITOS	
Identificação da Informação (obrigatório):	1
Categoria da informação (obrigatório):	IE
Origem:	
1. Ator/iniciador da ação (obrigatório):	Sistema
2. Condição:	
3. Ação (obrigatório):	deve obter
4. Objeto da ação (obrigatório):	a seguinte informação como entrada para um pedido (obrigatório):
5. Refinamento (ou destino) do objeto:	nome do comprador, endereço do comprador.
6. Refinamento (ou destino) da ação:	
7. Pós-condições:	
Frase final:	O sistema deve obter a seguinte informação como entrada para um pedido (obrigatório): nome do comprador, endereço do comprador.

Figura 7 – Interface de Entrada.

Neste trabalho, a estrutura de frases em linguagem natural será utilizada para padronizar a descrição dos requisitos e de outras informações, apresentadas acima, a fim de facilitar a compreensão dos requisitos da mesma maneira por pessoas diferentes e, também, a fim de facilitar o

mapeamento entre os elementos da SRS e do modelo de casos de uso.

A seguir, os modelos de caso de uso serão explanados. Serão apresentadas suas características, seus elementos e informações necessárias para dar continuidade à compreensão do modelo proposto neste trabalho.

2.5 Modelo de Casos de Uso

Um modelo de casos de uso (MCU) é um modelo das funções a serem realizadas pelo sistema e seus limites. Funcionam como um contrato entre os clientes e os desenvolvedores [RUP06]. Os casos de uso servem como uma linha base a ser seguida durante todo o desenvolvimento do sistema e são um resultado obtido através da análise da especificação dos requisitos.

Existem muitas maneiras de modelar um sistema, cada qual servindo para uma finalidade diferente. Entretanto, o propósito mais importante de um modelo de casos de uso é comunicar o comportamento do sistema ao cliente ou ao usuário final [RUP06]. Consequentemente, o modelo deve ser de fácil compreensão.

Um caso de uso individual descreve como um ator interage com o sistema para alcançar um resultado. Um conjunto de casos de uso, por sua vez, descreve o comportamento completo do sistema. Entretanto, descrever cada caso de uso antes de determinar o modelo completo não é uma boa prática. Primeiramente o modelo geral do sistema deve ser construído para, então, sucessivamente refiná-lo até que os comportamentos detalhados possam ser compreendidos. Porém, cada caso de uso é refinado somente dentro do contexto geral do sistema. O conjunto completo de casos de uso, atores e suas interações constituem o modelo de casos de uso do sistema [LEF00].

A seguir serão explicados, detalhadamente, os elementos que compõem um modelo de casos de uso.

2.5.1 Diagrama de casos de uso

O diagrama de casos de uso, Figura 8, é uma representação pictórica do sistema sendo construído [JAC92] e são compostos de atores, de casos de uso e de relacionamentos [RUP06].

Os diagramas de casos de uso são muito úteis para a visualização do contexto geral do sistema.

A notação utilizada neste trabalho é a proposta pela UML (Unified Modeling Language) [UML05] e consiste de elipses, setas e bonecos de palito. As elipses e setas mostram o empacotamento e a decomposição dos casos de uso, não seus conteúdos.

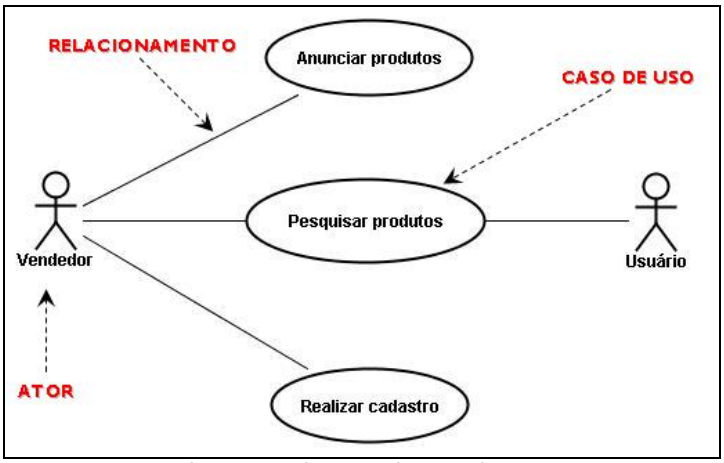


Figura 8 – Diagrama de caso de uso.

2.5.2 Ator

Um ator pode ser uma pessoa, uma companhia ou organização, um programa de computador ou um sistema computacional (hardware, software ou ambos). O importante é saber que um ator é qualquer coisa que tem um comportamento. Ele representa o usuário do sistema.

Um ator primário é aquele que utiliza o sistema diretamente. Este tipo de ator realiza uma ou mais das principais tarefas do sistema e é responsável pelas informações inseridas e recuperadas do sistema em modelagem [JAC92].

Um ator secundário é aquele que supervisiona e mantém o sistema. Desta forma, só existem atores secundários caso exista ao menos um ator primário [JAC92]. É um ator externo que provê um serviço ao SsD[3] [COC01].

É importante considerar que um ator pode ser primário em um caso de uso e secundário em outro [COC01]. Através da generalização de atores, podem ser definidos papéis para os usuários do sistema, como mostrado na Figura 9 [JAC92].

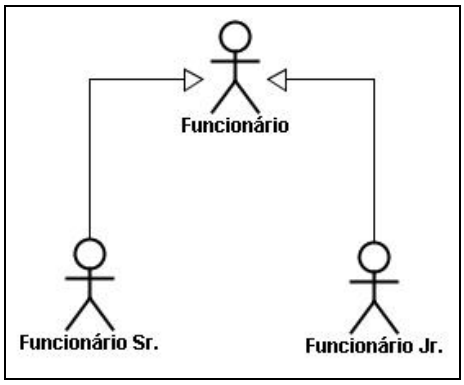


Figura 9 – Generalização de atores.

[3] SsD: Sistema sob discussão.

Vários autores propõem diversas questões para a identificação de atores. A Tabela 1mostra estas questões:

Tabela 1 – Questões básicas para identificação de atores.

QUESTÃO	REFERÊNCIA
Quem usa o sistema?	[LEF00], [SCH01], [RUP06]
Quem recebe informações do sistema?	[LEF00], [SCH01] [RUP06]
Quem fornece informações para o sistema?	[LEF00], [SCH01], [RUP06]
Onde o sistema é usado na companhia?	[LEF00], [RUP06]
Quem dá suporte e mantém o sistema?	[LEF00], [SCH01], [RUP06]
Quais outros sistemas usam este sistema?	[LEF00], [SCH01], [RUP06]
Quem instala o sistema?	[SCH01]
Quem inicia o sistema?	[SCH01]
Quem finaliza o sistema?	[SCH01]
Alguma coisa acontece automaticamente em um determinado tempo?	[SCH01]
Quem usará a funcionalidade?	[RUP06]
Quem está interessado no requisito?	[RUP06]
Quais são os recursos externos?	[RUP06]

2.5.3 Caso de uso

Os casos de uso servem como uma linha base a ser seguida durante todo o desenvolvimento do sistema e são um resultado obtido através da análise da especificação dos requisitos. Segundo [KRU00] um caso de uso é uma seqüência de ações que um sistema realiza para gerar um resultado observável de valor para um ator em particular, sabendo-se que um ator é alguém ou alguma coisa de fora do sistema que interage com ele.

Um caso de uso deve ser bastante detalhado, já que seu detalhamento será usado por todas as outras partes do desenvolvimento [MED04]. Porém este detalhamento deve ser caixa-preta, ou seja, não deve conter informações internas do sistema.

Após os atores terem sido identificados, pode-se passar para a identificação dos casos de uso. Da mesma maneira que para os atores, são propostas diversas questões para facilitar esta identificação, mostradas na Tabela 2.

Tabela 2 – Questões básicas para identificação de atores.

QUESTÃO	REFERÊNCIA
Para que o ator utilizará o sistema?	[LEF00]
Quais atores irão criar, ler, atualizar ou excluir informações no sistema?	[LEF00], [SCH01]
O ator precisará informar ao sistema sobre eventos externos ou mudanças?	[LEF00], [SCH01], [RUP06]
O ator precisa ser informado sobre certas ocorrências no sistema?	[LEF00],[SCH01], [RUP06]
Para cada ator identificado, quais as tarefas em que o sistema estará envolvido?	[RUP06]
O sistema apresenta o comportamento correto com relação ao negócio?	[RUP06]
Todas as características podem ser executadas pelos casos de uso identificados?	[RUP06]
Quais casos de uso darão suporte e manutenção ao sistema?	[RUP06]
Quais informações devem ser modificadas ou criadas no sistema?	[RUP06]

2.5.4 Relacionamentos

Os casos de uso se relacionam uns com os outros através dos relacionamentos de extensão (extend), de inclusão (include) e de generalização [RUP06] [UML05].

Um caso de uso base inclui outro caso de uso se um passo de ação no caso de uso base chama o nome do caso de uso incluído. As associações de <include> são usadas para evitar a duplicação de passos através de múltiplos casos de uso e, também, como uma estratégia de reuso para texto de casos de uso [KUL03].

Um caso de uso de extensão estende um caso de uso base indicando qual seu caso de uso base e definindo as circunstâncias sob as quais ele o interrompe. É importante considerar que o caso de uso base não menciona o de extensão [COC01]. Desta forma, adiciona-se comportamento ao caso de uso sem mudar o base [SCH01].

A associação de generalização, por sua vez, é usada quando um caso de uso especializa um outro caso de uso mais geral, ou seja, diz-se que o caso de uso é “um tipo de” outro caso de uso [COC01].

Segundo a UML [UML05] um relacionamento de generalização entre casos de uso implica que o caso de uso filho contém todos os atributos, seqüências de comportamento, e pontos de extensão

definidos no caso de uso pai, e participa de todos os seus relacionamentos.

2.5.5 Especificação de casos de uso

A especificação de casos de uso tem como objetivo descrever o que eles fazem. Apesar de normalmente esta especificação ser feita em linguagem natural, as linguagens formais também podem ser utilizadas.

Fala-se de caso de uso caixa branca quando um caso de uso é utilizado para documentar a cooperação entre as partes do sistema, quando os componentes são nomeados ou quando se consegue perceber o comportamento interno do sistema. Entretanto, este trabalho está focado nos casos de uso caixa-preta, aqueles dentro dos quais não se pode ver. Este tipo de caso de uso não menciona qualquer componente dentro do SsD [COC01], ou seja, não especifica detalhes de interface ou detalhes de baixo nível na execução do sistema.

Este trabalho faz uso da linguagem natural não só para a especificação dos requisitos na SRS mas também para a especificação dos casos de uso, visto que facilita a compreensão por todos os stakeholders. Diversos formulários ([KUL03], [RUP06], [COC01], [SCH01]) são propostos na literatura para este fim, com o objetivo de estabelecer algum tipo de estruturação. Todos eles utilizam características comuns, por vezes representadas com nomes diferentes, que são necessárias para o entendimento dos casos de uso. O padrão proposto por Cockburn [COC01] (Figura 10) será utilizado como base para a especificação dos casos de uso desta proposta. Seus campos são descritos a seguir.

Caso de uso:
 # <nome>
Contexto de Uso:
Escopo:
Nível:
Ator:
***Stakeholders*:**
Pré-Condição:
Garantias Mínimas:
Garantias de Sucesso:
Acionador:
Cenário de Sucesso Principal:
<# passo> <descrição da ação>
Fluxo Alternativo:
<# passo> <condição> : <ação ou sub caso de uso>
Variações Tecnológicas e de Dados:
Informações Relacionadas:

Figura 10 – Padrão de especificação de casos de uso proposto por Cockburn [COC01].

- Caso de uso: todo caso de uso deve possuir um nome único, que o identifique. Cockburn [COC01] propõe que este nome seja escrito com uma pequena frase de verbo ativo.
- Contexto de uso: representa uma sentença maior do objetivo e, se necessárias, suas condições de ocorrência normal.
- Escopo: define qual a funcionalidade está sendo considerada neste caso de uso.
- Nível: um dentre estes – resumo, objetivo de usuário, subfunção. Neste trabalho serão descritos apenas casos de uso a nível de objetivo de usuário.
- Ator: define o nome do ator primário.
- Stekeholders: relaciona todos os interessados no caso de uso, apresentando uma lista de stakeholders e atores secundários participantes.
- Pré-condição: representa o que já deve ser verdadeiro para a execução do caso de uso.
- Garantias mínimas: define como os interesses são protegidos sob todas as saídas, ou seja, se o caso de uso falhar quais são as garantias dadas aos interessados.
- Garantias de sucesso: define quais os interesses dos stakeholders são satisfeitos depois de uma conclusão bem-sucedida do caso de uso. Também chamados de pós-condições.
- Acionador: mantém a informação referente ao que dá inicio ao caso de uso.
- Cenário de sucesso principal: contém todos os passos do cenário do acionamento a entrega do objetivo e qualquer limpeza posterior. Responsável por manter uma execução de sucesso para o caso de uso.
- Fluxo alternativo: define os fluxos alternativos, ou seja, condições de extensão para o caso de uso.
- Variações tecnológicas e de dados: lista das variações que podem causar eventuais bifurcações no cenário.
- Informações Relacionadas: tudo que o projeto precisa de informação adicional. Deve conter todos os requisitos não-funcionais e/ou regras de negócio que influenciem a descrição do objetivo, também chamada de requisitos especiais [RUP06].

Observa-se que a especificação dos casos de uso é de extrema importância para as demais fases do processo de desenvolvimento, já que é utilizada durante a elaboração de outros diagramas e, também, para a elaboração dos casos de teste.

Apesar de serem encontrados, na literatura, diversos trabalhos que fornecem “melhores maneiras” para o desenvolvimento dos casos de uso [AND05] [COC97] [COC01] [COX01] [GOT06], nenhum apresenta uma relação direta entre o documento de especificação de requisitos e o modelo de casos de uso. Apenas indicam qual seria a melhor forma de realizar determinada atividade, sem indicar de onde as informações podem ser adquiridas.

2.6 Rastreabilidade

De acordo com Ramesh [RAM01], rastreabilidade de requisitos é “uma característica de sistemas nos quais requisitos são claramente ligados às suas fontes e aos artefatos criados durante o ciclo de vida de desenvolvimento do sistema baseado nesses requisitos. Rastreabilidade de requisitos estabelece um elo entre mudanças das necessidades dos usuários e evolução dos sistemas de computação, sendo uma base para o gerenciamento do conhecimento organizacional”.

O rastreamento dos requisitos, para [GOT97] e [RAM95], refere-se à habilidade para descrever e seguir a vida de um requisito em ambas as direções, para frente e para trás, conforme Figura 11. A pré-rastreabilidade documenta o contexto a partir do qual emergem os requisitos e a pós-rastreabilidade vincula os requisitos ao desenho do sistema e sua implementação [DAV93].

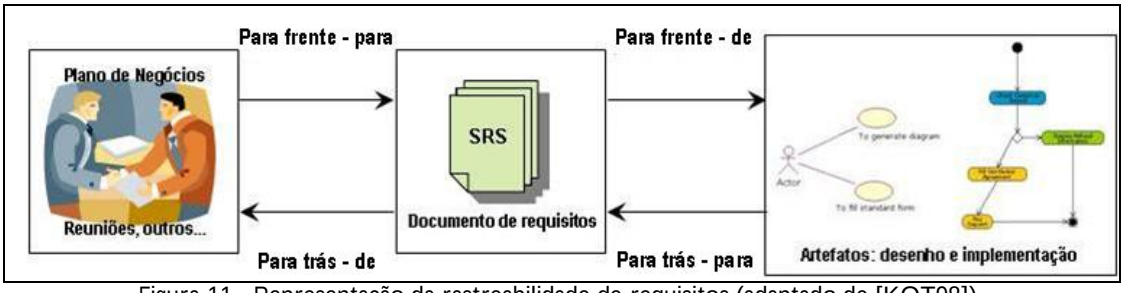


Figura 11 - Representação da rastreabilidade de requisitos (adaptado de [KOT98]).

Na pré-rastreabilidade tem-se a rastreabilidade forward-to (para frente – para), que liga documentos obtidos no processo de elicitação a requisitos relevantes, e a rastreabilidade backward-from (para trás – de), que liga os requisitos às suas fontes. Na pós-rastreabilidade tem-se a rastreabilidade forward-from (para frente – de), que liga os requisitos a artefatos de desenho e implementação e a rastreabilidade backward-to (para trás – para), que liga os artefatos de desenho e implementação de volta aos requisitos.

A rastreabilidade é vista como uma técnica fundamental no apoio às diversas atividades do projeto, assegurando que sistemas e software estão em conformidade às mudanças dos requisitos, entretanto é citada como uma área problema pelos desenvolvedores [GOT97] e [RAM95].

Apesar de se saber que o esforço necessário para a aplicação da rastreabilidade pode ser elevado, como aumento de custo ou tempo, sem a utilização de mecanismos deste tipo podem surgir

inconsistências durante o processo de adição, remoção ou modificação de requisitos. De acordo com [KOT98], os requisitos não podem ser efetivamente gerenciados sem rastreabilidade.

Além disso, o rastreamento de requisitos é reconhecido como um importante pré-requisito para o desenvolvimento de sistemas com alta qualidade, pois ajuda a manter consistentes um conjunto de informações relacionadas do processo de desenvolvimento [POH96]. Diversos trabalhos relacionam a rastreabilidade como fator de qualidade para o desenvolvimento de software [SPA02], [RAM01], [TOR99], [JAR98], [KOT98] e [PAL97]. A fim de manter um sistema de qualidade é necessário aplicar rastreamento de requisitos no processo de desenvolvimento, já que os requisitos originam a produção de outros artefatos intermediários que, ao final do desenvolvimento, deverão estar de acordo com as funcionalidades oferecidas no produto final. Desta forma, é importante que os relacionamentos entre os requisitos e os artefatos sejam gerenciados, a fim de facilitar a manutenção e verificação do sistema [TOR99].

Em [SAY05] são apresentadas várias situações em que a rastreabilidade pode auxiliar gerentes e desenvolvedores. As situações mais importantes, no contexto deste trabalho, são as seguintes: verificação da alocação de requisitos a componentes do software, correção de defeitos, validação e análise de impacto na evolução do sistema.

- Verificação da alocação de requisitos a componentes do software: é possível identificar requisitos ainda não alocados ou implementados através da avaliação dos elos de rastreabilidade de requisitos e artefatos de desenho e implementação. Neste trabalho, este tipo de situação é verificada com relação aos requisitos e aos casos de uso, podendo-se averiguar se existe algum requisito ainda não compreendido no modelo de casos de uso.
- Correção de defeitos: é possível que, após a identificação do componente que causou o erro, identifique-se que a origem do defeito não está no código propriamente dito. Os elos de rastreabilidade indicarão quais artefatos, além do código, deverão ser revistos para identificação do erro. Neste trabalho, esta situação é considerada quando for verificado, por exemplo, que um erro parte de determinado caso de uso. Com isto, seu requisito de origem poderá ser identificado.
- Validação: nesta situação a rastreabilidade permite mostrar se a implementação atende ao conjunto de requisitos acordados entre clientes e desenvolvedores. Neste trabalho, esta situação é considerada para verificar, por exemplo, se todos os requisitos estão compreendidos no modelo de casos de uso.
- Análise de impacto na evolução do sistema: a identificação dos componentes afetados por mudanças em um requisito, ou mesmo por inclusão de novos requisitos, fica facilitada pela existência de elos de rastreabilidade entre requisitos e componentes,

sem que haja necessidade de consultas a diferentes artefatos. Caso o requisito alterado esteja ligado a outros requisitos (dependência entre requisitos), estes também deverão ser avaliados.

A existência de elos de rastreabilidade possibilita identificar as origens de cada funcionalidade presente no sistema.

A rastreabilidade, neste trabalho, será usada para garantir que todos os requisitos expressos na SRS estejam englobados no MCU. O inverso também é verdadeiro, já que se, por ventura, forem encontradas dúvidas na especificação de determinado caso de uso estas podem ser sanadas na descrição de seu requisito de origem.

Outra aplicação importante deste conceito se dará no momento em que forem realizadas alterações no documento de especificação de requisitos. Por intermédio da rastreabilidade, apresentada no modelo proposto, será possível alcançar todos os elementos (representados no modelo através das classes) impactados quando desta alteração. Neste trabalho, são consideradas as alterações correspondentes à inclusão, exclusão e substituição de requisitos. É importante considerar que, como em [HAZ03], a alteração de um requisito se caracteriza quando existe mudança em sua funcionalidade.

2.7 Considerações do capítulo

Este capítulo apresentou conceitos referentes à engenharia de requisitos e à rastreabilidade objetivando demonstrar a importância de seu uso para as organizações de desenvolvimento de software. Através deste estudo, percebeu-se que manter o documento de especificação de requisitos e o modelo de casos de uso consistentes entre si durante todo o processo de desenvolvimento é uma forma de permitir que o resultado represente uma solução para o sistema proposto.

Neste contexto, o próximo capítulo apresenta os trabalhos relacionados ao tema de pesquisa.

3 Estudos Relacionados

Após a pesquisa bibliográfica realizada, foram encontrados alguns trabalhos relacionados ao tema de pesquisa apresentado nesta dissertação. Este capítulo apresenta-os, juntamente com suas principais características.

3.1 BELGAMO e FABBRI (2005)

O trabalho apresentado por Belgamo e Fabbri [BEL05] propõe uma técnica para construção do modelo de casos de uso e para análise do documento de requisitos que surgiu com a idéia de reduzir o grau de subjetividade e a necessidade de experiência do projetista. Seu objetivo principal é auxiliar na construção de modelos de casos de uso, fornecendo procedimentos sistemáticos que direcionem a modelagem e permitam que o modelo construído não apresente tanta variação se desenvolvido por diferentes pessoas. À medida que o modelo é construído, essa técnica também propicia condições de inspecionar o documento de requisitos (DR).

TUCCA (Technique for Use Case model construction and Construction-based requirements document Analysis) é composta por duas técnicas de leitura: AGRT (Actor Goal Reading Technique) e UCRT (Use Case Reading Technique) cujos propósitos são, respectivamente, determinar os atores do sistema e seus objetivos e determinar o modelo de casos de uso. Ambas as técnicas são utilizadas por meio de passos que dão suporte à construção do MCU e incorporam, também, uma revisão do documento de requisitos.

TUCCA utiliza por padrão a SRS da IEEE [IEE98] como artefato básico para ambas as técnicas. Porém, nem todas as seções da SRS são necessárias, a premissa é de que se a SRS incluir as seções Definições, Funções do Produto, Características do Usuário, Requisitos Funcionais e Requisitos Não-Funcionais já é possível a aplicação da técnica. Contudo o campo “Requisitos Não-Funcionais” não faz parte do modelo sugerido em [IEE98]. Aparentemente este campo foi definido para facilitar a aplicação da técnica, entretanto, não se pode dizer que ele é proposto na srs da IEEE.

Inicialmente a técnica AGRT recebe como entrada o DR e através de algumas diretrizes tem como saída um formulário (FAO) com uma lista de atores e seus objetivos. Nesta lista são feitas algumas manutenções como: inclusão de objetivos e atores esquecidos e exclusão de redundâncias. A técnica UCRT, por sua vez, tem como entrada o DR e o FAO gerado e, através da aplicação de diretrizes, tem como saída o diagrama de casos de uso, as especificações dos casos de uso e um

relatório de discrepâncias que conterá inconsistências encontradas durante a aplicação das técnicas.

O trabalho apresenta considerações importantes para identificação de atores e objetivos em busca de encontrar os casos de uso do sistema. Porém apenas cita que existem sub-passos a serem aplicados para a especificação detalhada destes casos de uso. Outro aspecto identificado nesta pesquisa foi a maneira como o requisito está descrito. O trabalho não se preocupa em definir um padrão de escrita para os requisitos, o que pode fazer com que a técnica tenha um desempenho diferente se os requisitos não forem escritos de maneira semelhante aos do estudo de caso apresentado em [BEL05].

Embora tais limitações tenham sido identificadas, este trabalho contribuiu para esta pesquisa no sentido de definir alguns relacionamentos e também na definição de procedimentos para utilização do processo proposto.

3.2 SOMÉ (2005)

O estudo de Somé [SOM06] apresenta uma abordagem para dar suporte aos casos de uso, baseando-se na elicitação, clarificação, composição e simulação dos requisitos. Oferece uma forma restrita de linguagem natural para casos de uso, dependente de contexto, e uma derivação automatizada da especificação. O modelo de domínio é usado para capturar os conceitos do domínio relevantes ao contexto, bem como as definições das operações de pré e pós-condições em paralelo com os casos de uso.

Este trabalho considera como base o template proposto por Cockburn [COC01], onde os casos de uso são descritos usando um texto estruturado. Não faz referências ao uso de documentos de requisitos, porém apresenta interessantes maneiras de representar condições e operações fazendo uso de uma estrutura de linguagem natural buscando, com isso, evitar alguns problemas referentes desta área. Como o objetivo principal da pesquisa de Somé [SOM06] não é simplesmente a descrição dos casos de uso, alguns detalhes importantes como a representação dos requisitos não-funcionais são deixados de lado.

Sabendo-se que os casos de uso possuem como componentes básicos condições e operações, é proposta uma forma de linguagem natural para descrição concreta destes elementos. Para as condições é proposta uma gramática contendo explicações sobre o que são condições, o que elas descrevem e outras definições. Para as operações de casos de uso é proposta, também, uma gramática que se refere a algumas definições feitas anteriormente na gramática para condições.

Além disso, o trabalho apresenta uma sintaxe abstrata para os casos de uso, fazendo uso da UML [UML05], onde são distinguidos dois tipos de descrição que correspondem aos dois tipos de

casos de uso: casos de uso normais e casos de uso de extensão. Entretanto, como seu objetivo principal é a derivação da máquina de estados, mais detalhes sobre as especificações em si não são considerados.

O trabalho de Somé [SOM06] contribuiu para a opção feita em utilizar uma estrutura de especificação de requisitos em linguagem natural. Isto facilita a compreensão por todos os interessados e ao mesmo tempo oferece certa formalidade, já que a representação é estruturada. Além disso, a sintaxe abstrata para os casos de uso, contribuiu para a definição do modelo conceitual do MCU, desenvolvido no decorrer desta pesquisa.

3.3 SMIALEK (2005)

O trabalho apresentado por •mia•ek [SMI05] propõe uma notação para cenários de casos de uso e relata sobre a necessidade de diferentes papéis nos projetos de desenvolvimento de software. De acordo com o autor, alguns papéis requerem sentenças simples e informais com referências ao vocabulário de domínio. Outros papéis necessitam de uma relação e de um mapeamento com elementos de interface de usuário ou mensagens circulando dentro do sistema de desenvolvimento. Com isso, o autor conclui que estes requisitos contraditórios sugerem que a notação dos casos de uso deve ser uma composição de varias notações com regras precisamente definidas para sua transformação.

De acordo com o descrito acima, o trabalho apresenta quatro notações baseadas em estrutura de texto, diagramas de interação e diagramas de atividades. Além disso, é apresentado um mapeamento entre elementos específicos destas notações e um mapeamento com os elementos dos modelos de design.

O trabalho faz uso de elementos previamente definidos na UML [UML03] como a notação de Caso de Uso, Diagrama de Atividades, Casos de Teste e Gráficos de Seqüência. Ainda, apresenta um metamodelo para a representação de texto estruturado, Figura 12, que foi utilizado na composição do modelo conceitual do MCU.

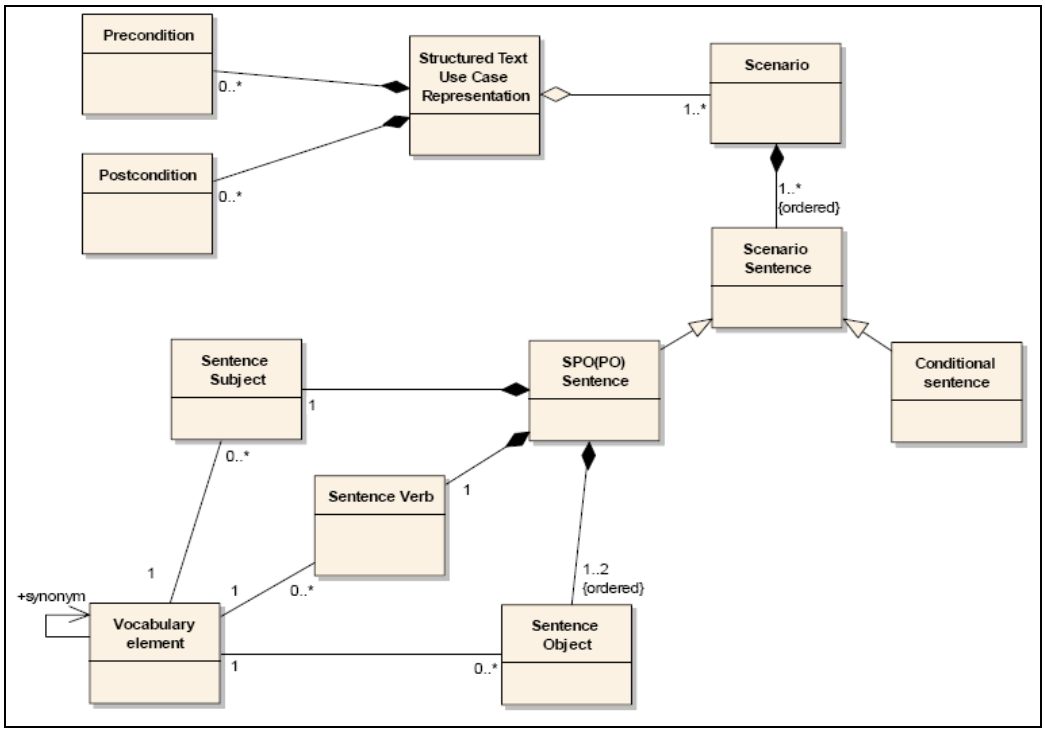


Figura 12 – Metamodelo para representação textual estruturada [SMI05].

Além disso, o autor propõe um mapeamento entre alguns elementos deste metamodelo e os diagramas de atividade e os nodos de decisão. De acordo com ele, as instâncias da classe SPO(PO)Sentence serão mapeadas em uma ação e cada instância da classe Conditional sentence será mapeada em um nodo de decisão do diagrama de atividades. Afirma, ainda, que todos os cenários representados textualmente serão mapeados em uma ou mais atividades que conterão ações e nodos de decisão.

A pesquisa apresentada em [SMI05] mostra uma forma interessante de estruturação para os cenários dos casos de uso e contribuiu para o desenvolvimento do modelo conceitual do MCU, como é apresentado na seção 6.2.

3.4 TORANZO (2002)

Toranzo [TOR02] [CES02] apresenta uma proposta que visa melhorar o rastreamento de requisitos. As estratégias apresentadas são: fornecer uma classificação das informações a serem rastreadas; um metamodelo; um modelo intermediário de requisitos que possui muitos dos artefatos que geralmente são encontrados nos modelos de rastreamento; e um processo que reúne e aplica as três estratégias anteriores.

As informações a serem rastreadas são classificadas em quatro níveis distintos, conforme Figura 13.

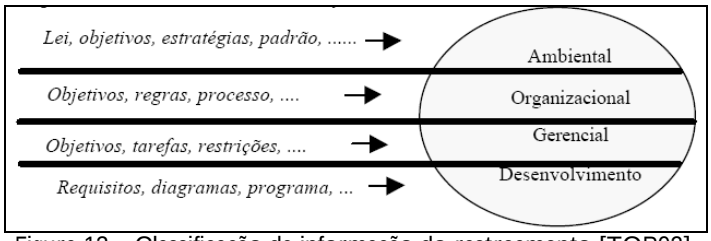


Figura 13 – Classificação da informação do rastreamento [TOR02].

Os dois últimos níveis são os que mais se enquadram no contexto deste trabalho, pois tratam de ligar tarefas a requisitos e apresentam informações relacionadas aos diversos artefatos gerados no processo de desenvolvimento.

O metamodelo proposto apresenta classes que classificam, por exemplo, os tipos de elementos, relacionamentos e associações a serem utilizadas. São definidos seis tipos de elos que dão suporte a rastreabilidade: satisfação, recurso, responsabilidade, representação, alocado e agregação.

Além disso, a proposta compreende um modelo intermediário de rastreamento, Figura 14, que consolida observações da prática, de estudos de caso e da aplicação das estratégias de classificação das informações e o metamodelo de rastreabilidade. A idéia é que este modelo possa ser encarado como base para discussão e definição de um modelo de rastreabilidade focado num projeto em particular.

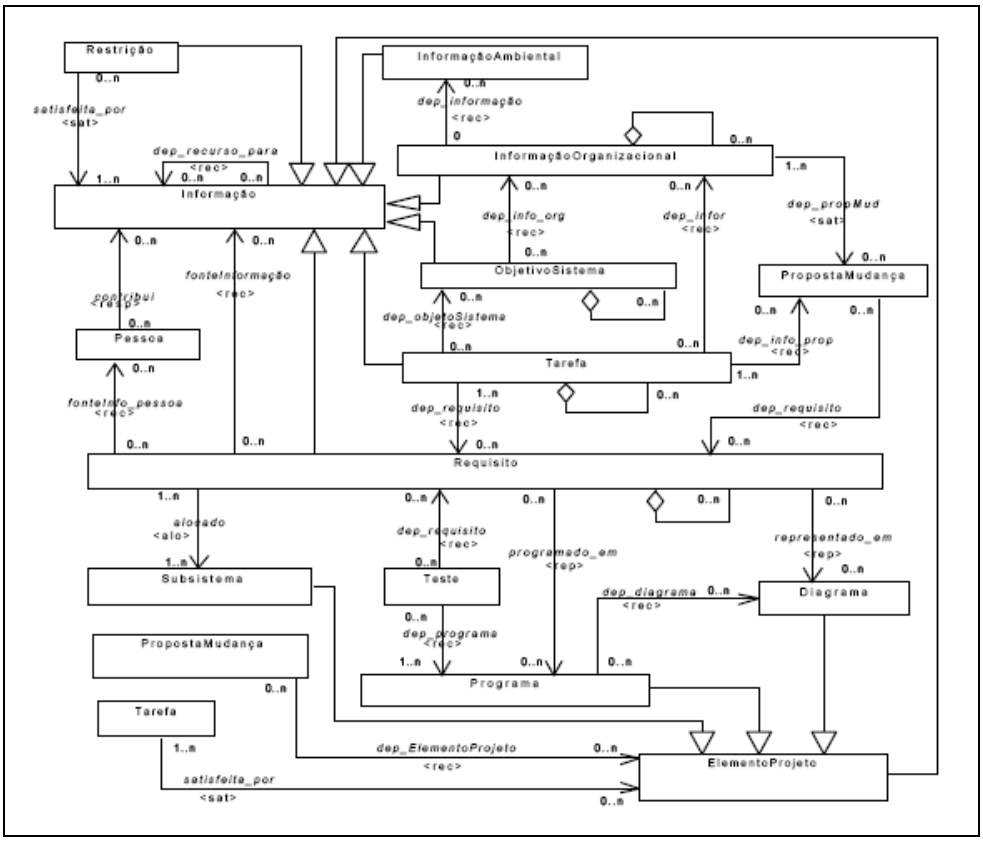


Figura 14 – Modelo intermediário para o rastreamento de requisitos [TOR02].

Alguns relacionamentos e dependências, apresentados no modelo de rastreabilidade proposto nesta dissertação, foram identificados com o auxílio dos relacionamentos apresentados no modelo intermediário de Toranzo [TOR02].

3.5 RAMESH E JARKE (2001)

O trabalho apresentado em [RAM01] propõe um modelo para rastreabilidade que possibilita a captura de informações relacionadas a stakeholders (interessados), fontes (documentos que remetem a origem dos requisitos) e objetos (objetos do produto ou do processo).

Os autores agrupam os elos de rastreabilidade em duas categorias: relacionados ao produto e relacionados ao processo. A primeira representa os elos que descrevem propriedades e relacionamentos dos objetos, onde estes elos são subdivididos em elos de satisfação e de dependência. A segunda representa os elos relacionados ao histórico de ações executadas no próprio processo e estes são subdivididos em elos de evolução e de rationale.

A Figura 15 apresenta o metamodelo de rastreabilidade de requisitos proposto por [RAM01] e apresenta a distinção dos três níveis da rastreabilidade.

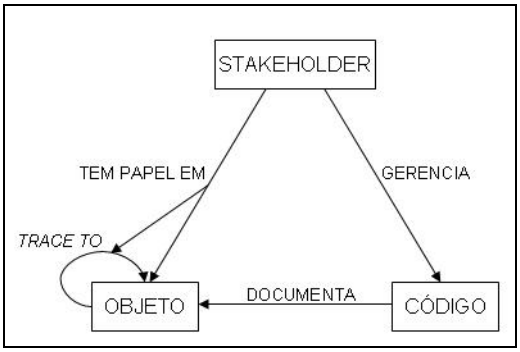


Figura 15 – Metamodelo de rastreabilidade (adaptado de [RAM01]).

Os elos de satisfação buscam assegurar que os requisitos sejam atendidos pelo sistema, ou seja, a cada requisito foi associado um componente que deverá atendê-lo.

Os elos de evolução registram relacionamentos que levam de objetos existentes para objetos novos ou modificados.

O propósito dos elos de rationale é representar as motivações subjacentes aos objetos existentes ou documentar as razões para a evolução.

E, finalmente, os elos de dependência têm por propósito apoiar o gerenciamento de dependências entre objetos, frequentemente impostas por restrições de recurso, de competências ou de compatibilidade, sendo úteis para registrar a composição e hierarquia dos objetos e apoiar o

gerenciamento do impacto das alterações num objeto sobre os objetos que dele dependem.

Os elos de dependência são utilizados no modelo de rastreabilidade, proposto nesta dissertação, através das relações de dependência da UML [UML05] e apresentam, também, as características de impacto propostas em [RAM01].

3.6 Considerações do capítulo

Como visto, existem diversos trabalhos envolvidos com o tema de pesquisa, porém não diretamente ligados a ele. Foram encontrados poucos trabalhos relacionando o documento de especificação de requisitos aos modelos de casos de uso. Neste contexto foi apresentado o trabalho proposto por Belgamo [BEL05]. Os trabalhos de Somé [SOM06] e •mia•ek [SMI05] apresentam, também, informações relevantes sobre o MCU.

Apesar de serem encontrados na literatura vários trabalhos relacionados à rastreabilidade de requisitos, nenhum deles integra os elementos da SRS com os elementos do MCU. Em sua maioria, os trabalhos apresentam modelos, metamodelos e relacionamentos referentes aos artefatos de todo o processo de desenvolvimento de software. Neste sentido, a única relação indicada é a de requisitos com seus casos de uso. [TOR02] e [RAM01], por exemplo, são alguns dos trabalhos estudados neste contexto e, apesar de não estarem diretamente focados no assunto desta proposta, os metamodelos e relacionamentos apresentados por eles foram úteis na definição dos elos de rastreabilidade do modelo.

O próximo capítulo apresenta o método de pesquisa utilizado neste trabalho bem como considerações importantes sobre seu desenvolvimento.

4 MÉTODO DE PESQUISA

Após ampla revisão teórica, percebeu-se que o problema apresentado ainda não foi abordado sob a mesma perspectiva. Assim, esta pesquisa se caracteriza como um estudo predominantemente exploratório.

Foram utilizadas duas metodologias de apoio: revisão bibliográfica e o estudo de caso. Além disso, foi realizada a aplicação do modelo proposto. A pesquisa bibliográfica é um método que possibilita um levantamento dos trabalhos realizados anteriormente sobre o mesmo tema estudado no momento, além de fornecer subsídios para a revisão da literatura do projeto ou trabalho [CRU03]. O estudo de caso é um método que pode ter por finalidade "entender como e por que funcionam as coisas" [YIN01]. Por fim, a aplicação tem o objetivo de validar a proposta através de um exemplo prático.

A primeira etapa desta pesquisa foi a pesquisa bibliográfica, que teve como objetivo adquirir conhecimento e base teórica sobre os assuntos diretamente abordados neste trabalho. Assim, foram realizados estudos detalhados sobre requisitos, abordando os documentos de especificação de requisitos, os modelos de casos de uso, a rastreabilidade e trabalhos relacionados ao contexto desta pesquisa.

A partir da base teórica, partiu-se para a segunda etapa, onde foram definidos os modelos conceituais da SRS e do MCU. A partir deles, foi desenvolvido um modelo de integração identificando as relações entre os elementos dos dois primeiros modelos desenvolvidos.

Tendo os resultados da segunda etapa, pode-se partir para a terceira etapa. Nesta, foi realizado um estudo de caso em uma organização de desenvolvimento de software, onde se buscou avaliar a aplicabilidade da proposta, considerando que a prática realizada na empresa pudesse contribuir com a teoria estudada e com o modelo proposto, o que realmente aconteceu. Com o estudo de caso concluído, foi possível aprimorar o modelo de integração, chegando a construção de um modelo de rastreabilidade e de um processo de apoio para sua realização.

Por fim, a quarta etapa consistiu na consolidação da proposta, que se deu na aplicação do processo de rastreabilidade, fazendo uso do modelo, em uma SRS e em seus respectivos casos de uso. Este estudo contribuiu para a avaliação da proposta e após ter sido realizado, partiu-se para o protótipo da ferramenta, que atualmente está em fase de desenvolvimento.

Todas as etapas descritas acima podem ser visualizadas no desenho de pesquisa, apresentado na Figura 16.

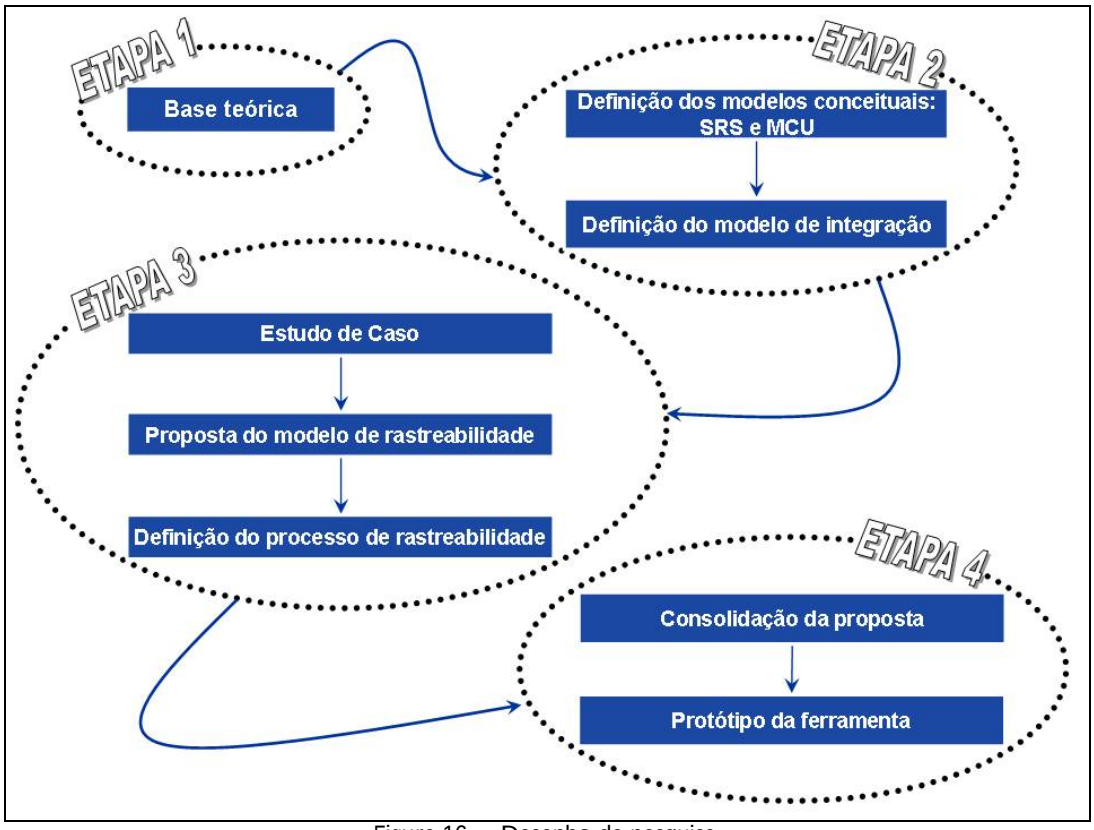


Figura 16 – Desenho de pesquisa.

4.1 Considerações sobre a evolução da pesquisa

A proposta aqui apresentada considera que o documento de requisitos, artefato inicial deste trabalho, está completo, ou seja, abrange todas as funcionalidades desejadas pelos clientes e por todos os stakeholders envolvidos no projeto.

Com base na literatura e no estudo de caso, foi proposta a adaptação de um template para o documento de especificação de requisitos (SRS) que deve ser utilizado contendo as informações sugeridas neste trabalho. Com base neste template, deu-se continuidade à pesquisa, onde foi proposto um modelo de rastreabilidade e um processo para sua aplicação. É importante considerar que o modelo de rastreabilidade trará mais benefícios quando os documentos da SRS e do MCU estiverem de acordo com os padrões estabelecidos neste trabalho. Entretanto, com um mínimo de organização nestes documentos já é possível sua aplicação.

Outro ponto a ser considerado é que na segunda etapa desta pesquisa, foram desenvolvidos, em conjunto com o grupo de pesquisas em engenharia de requisitos do CDPe (Centro de Desenvolvimento Dell/PUCRS), os modelos conceituais da SRS, do MCU e um modelo de integração. Estes modelos serviram como ponto inicial para o restante do desenvolvimento deste trabalho e à medida que a pesquisa foi evoluindo, eles foram sendo adaptados conforme as

necessidades identificadas na literatura e no estudo de caso.

Estes ajustes foram discutidos e avaliados com pesquisadores e colegas do PPGCC durante o seminário de andamento e durante todo o ciclo de pesquisas envolvendo esta pesquisa de mestrado.

A seguir serão apresentadas algumas considerações sobre os modelos inicialmente desenvolvidos neste trabalho, apresentados em [CER06a], [ROC06a] e no Apêndice I. Estes modelos foram desenvolvidos como ponto de partida para alcançar as versões finais, que serão apresentados na seção 6. A partir destes modelos foi desenvolvido, também, um modelo para a avaliação da qualidade da tradução dos requisitos para os casos de uso, que pode ser conferido no artigo apresentado em [ROC06b].

4.1.1 Modelo conceitual inicial da SRS

A partir dos campos e características apresentadas na documentação da SRS da IEEE [IEE84] [IEE98] foi construído seu modelo conceitual. A decisão da utilização deste modelo de especificação de requisitos se deu devido a sua ampla utilização no mercado e, também, devido a sua grande aceitação no meio acadêmico.

Os elementos e seus relacionamentos, indicados nas práticas recomendadas para especificação de requisitos definidas pela IEEE [IEE84] [IEE98], foram representados em um diagrama de classes [UML05].

Schimidt [SCH00] coloca que [IEE98] fornece diretrizes detalhadas das informações que devem estar contidas em cada uma das seções da SRS, porém afirma que, ainda assim, existem dúvidas sobre quais informações detalhar em determinadas seções.

Um ponto comum de confusão, de acordo com Schimidt [SCH00], é a diferença entre a seção "Propósito" e a seção "Escopo", subseções da "Introdução". Segundo ele, a primeira deve delinear a finalidade da SRS e identificar sua audiência enquanto a segunda deve identificar o software em questão, explicando o que ele fará, descrevendo sua aplicação, seus benefícios e objetivos.

Outra possível confusão geralmente ocorre em relação às informações presentes nas seções 2 e 3, já que ambas devem incluir informações sobre as interfaces, funcionalidades e restrições do software. Entretanto a seção 2 (Descrição Geral) não especifica os requisitos, apenas dá uma base destes que serão detalhados na seção 3 da SRS (Requisitos Específicos).

Assim, o modelo conceitual da SRS foi inicialmente desenvolvido contendo todos os elementos indicados pela documentação da IEEE [IEE84] [IEE98]. Os elementos que fizeram parte do modelo inicial podem ser vistos na estrutura apresentada na Figura 17.

Após ter sido realizado o estudo de caso, este modelo foi adaptado conforme as necessidades

identificadas. O modelo final pode ser conferido na seção 6.1.

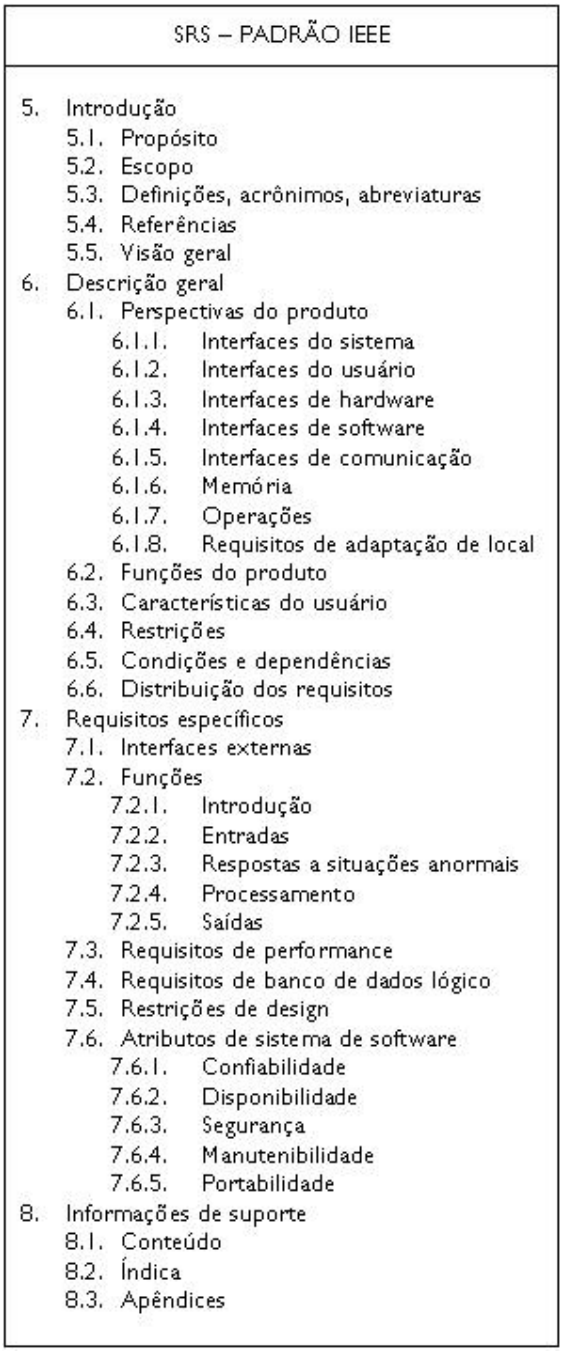


Figura 17 – Estrutura da SRS da IEEE [IEE98].

4.1.2 Modelo conceitual inicial do modelo de casos de uso

Da mesma forma que no modelo anterior, este foi desenvolvido em conjunto pelo grupo de Engenharia de Requisitos do Centro de Desenvolvimento e Pesquisas DELL/PUCRS (CDPe). Este modelo teve como base o template de casos de uso de Cockburn [COC01], onde para cada campo

foi definida uma classe, responsável por armazenar as devidas informações dos casos de uso. As pesquisas realizadas em [RUP06], [SOM06] e [COC01], contribuíram para a definição da utilização deste template e, também, para a identificação das informações que deveriam estar presentes em cada classe do modelo.

Um ponto importante é que a seção Cenário de sucesso principal, é onde será contada a história sobre o que o sistema entrega como resultado. Antes de tentar identificar todas as exceções ou condições de falha, deve ser mostrado como o sistema funciona em caso de sucesso. Este tipo de cenário pode gerar exceções, o que ocasiona a criação de um Fluxo alternativo para seu tratamento. Tanto o Cenário de sucesso principal quanto o Fluxo alternativo são compostos de passos que explicam sua execução.

O nível indicado na especificação é considerado como sendo sempre "Objetivo de Usuário", conforme explicado anteriormente na seção 2.5.

O template utilizado para compor o modelo é apresentado na Figura 18.

TEMPLATE PARA ESPECIFICAÇÃO DE CASOS DE USO	
NOME	**IDUC**
CONTEXTO	
ESCOPO	
NÍVEL	
Objetivo de Usuário	
ATOR	
STAKEHOLDERS	
PRÉ-CONDIÇÕES	
GARANTIAS MÍNIMAS	
GARANTIAS DE SUCESSO	
ACIONADOR	
CENÁRIO DE SUCESSO PRINCIPAL	
PASSOS	
FLUXO ALTERNATIVO	
PASSOS	
LISTA DE VARIAÇÕES TECNOLÓGICAS E DE DADOS	
INFORMAÇÕES RELACIONADAS	

Figura 18 – Template para especificação dos casos de uso (adaptado de [COC01]).

Outros detalhes sobre a adaptação do modelo, bem como a apresentação do modelo final, serão vistos na seção 6.2.

4.1.3 Modelo conceitual de integração

O modelo de integração é resultado de uma análise realizada sob os modelos conceituais da SRS e do MCU. Através de estudos e pesquisas realizadas em artigos, relatórios e documentos [RUP06] [IEE84] [IEE98] [SCH00], foram identificadas as relações entre os elementos de ambos os modelos para que fosse possível definir de onde poderiam ser retiradas, na SRS, as informações para as descrições dos casos de uso.

Cada elemento foi analisado individualmente e suas características foram sendo comparadas uma a uma para que fosse possível identificar as relações de umas com as outras. Após amplas pesquisas e estudos sob exemplos e materiais teóricos encontrados na literatura, foram definidas as relações mostradas na Figura 19.

CLASSES CASOS DE USO	CLASSES SRS
Nome	IntroducaoRF
Escopo	Escopo
Contexto	FuncoesProduto
Ator	IntroducaoRF
Stakeholders	FuncoesProduto
PreCondicoes	Processamento
GarantiasMinimas	SuposicoesDependencias
GarantiasMinimas	RespostasSituacoesAnormais
GarantiasSucesso	Saida
CenarioSucessoPrincipal	FuncoesProduto
FluxoAlternativo	RespostasSituacoesAnormais
Variacoes	Processamento
Variacoes	FuncoesProduto
InformacoesRelacionadas	Atributo
InformacoesRelacionadas	Performance

Figura 19 – Integração entre as classes dos modelos.

O desenvolvimento do modelo de integração se deu para que ficasse facilitada a construção do modelo de rastreabilidade. A partir dos relacionamentos identificados neste modelo, foi possível verificar sua viabilidade de aplicação devido ao estudo de caso realizado e devido a sua aplicação em exemplos reais.

A explicação da origem e dos relacionamentos identificados para este modelo, podem ser vistos em [CER06a], encontrado também no Apêndice 1.

4.2 Considerações do capítulo

Este capítulo apresentou o método de pesquisa utilizado bem como as etapas e considerações sobre sua evolução. Neste sentido foram detalhadas as atividades iniciais realizadas para o desenvolvimento do modelo e do processo propostos.

No próximo capítulo é apresentado o estudo de caso realizado.

5 ESTUDO DE CASO

O estudo de caso teve como objetivo identificar características específicas relacionadas à documentação dos requisitos e os principais problemas enfrentados neste contexto, a fim de comparar e propor um modelo que englobasse tanto as características estudadas na literatura como a realidade encontrada nas empresas. Com este propósito foram realizadas análises, através de pesquisa documental, em sete documentos de especificação de requisitos (SRS). A seção 5.1 apresenta a descrição do estudo realizado. A seção 5.2 apresenta o roteiro das atividades. A seção 5.3 apresenta as considerações finais deste estudo de caso.

5.1 Descrição

Quanto ao processo de pesquisa, o estudo de caso foi realizado em três etapas. A primeira etapa foi a de planejamento, onde foi desenvolvido o roteiro para realização do estudo de caso. Para isto foram realizadas três atividades. Primeiro foi realizado um levantamento dos objetivos do estudo e, com isso, foram relacionados tópicos a serem analisados. Em seguida foi realizada uma reunião entre a pesquisadora e o professor orientador para fins de avaliação dos tópicos. E, por fim, foi definido o roteiro para o estudo de caso.

A segunda etapa do estudo de caso consistiu na realização da análise da documentação. Esta etapa, bem como todas as outras, foi realizada seguindo o roteiro estipulado. Foram analisados sete SRS com o objetivo de identificar os elementos presentes nestes documentos e a completude de suas informações, buscando conclusões sobre as possibilidades de geração de uma especificação de casos de uso de qualidade a partir deles. Foi realizada uma análise em cada seção da SRS a fim de identificar possíveis falhas, falta de informações e/ou características especificadas em locais não adequados do documento. Ainda nesta etapa realizou-se um estudo considerando as SRS e seus respectivos casos de uso. Este estudo foi realizado para identificar a existência ou não de informações incompletas e/ou apenas presentes em um dos documentos, visando encontrar subsídios que indicassem se o caso de uso estava realmente de acordo com a SRS e se todos os requisitos apresentados na SRS estavam também compreendidos no modelo de casos de uso.

A última etapa do estudo de caso consistiu na análise dos resultados adquiridos através da realização da etapa dois deste estudo. Nesta etapa todos os resultados das análises feitas em cada SRS foram analisados pela pesquisadora e um relatório preliminar foi apresentado ao orientador. Ao todo foram realizadas três reuniões entre o orientador e a pesquisadora para discussão dos

resultados a fim de identificar as adaptações que seriam realizadas no modelo, inicialmente baseado na literatura. Como resultado foi redigida esta seção que apresenta os resultados obtidos.

5.1.1 Caracterização da Organização

O estudo de caso foi desenvolvido em uma unidade de desenvolvimento de software de uma organização de grande porte. A organização possui escritórios em mais de 34 países em todo o mundo, incluindo o Brasil. Segundo informação fornecida pela própria organização, possui em torno de 50.000 colaboradores em todo o mundo.

A unidade onde o estudo foi aplicado está localizada na cidade de Porto Alegre, estado do Rio Grande do Sul, Brasil. Ela possui mais de 400 colaboradores trabalhando em projetos que atendem as necessidades da área de TI da empresa.

5.1.2 Caracterização das SRS analisados

Nesta pesquisa foram analisadas 7 (sete) SRS, chamadas doravante de SRS1, SRS2, SRS3, SRS4, SRS5, SRS6 e SRS7. As SRS tratam de aspectos necessários para o desenvolvimento dos projetos e especificam as funcionalidades necessárias para o desenvolvimento do software.

As informações contidas nas SRS são oriundas de documentos, sem nenhuma espécie de padronização, escritos por analistas de outra unidade da empresa. De acordo com os analistas da unidade participante do estudo de caso, as informações presentes nestes documentos quase nunca contemplam todas as informações necessárias para o desenvolvimento das SRS. Normalmente as dúvidas são esclarecidas por telefone ou diretamente com os clientes.

5.2 Roteiro das atividades

A primeira etapa do estudo de caso teve como objetivo levantar informações referentes as SRS estudados. Sendo assim, foram definidas algumas questões a serem respondidas durante o andamento das pesquisas, representando o roteiro das atividades. Para este roteiro foram definidas três fases: análise individual da SRS, análise geral das SRS e análise dos resultados.

As fases de análise individual e geral das SRS foram utilizadas para a realização da etapa 2 do estudo de caso que corresponde à análise da documentação. A fase de análise dos resultados corresponde à etapa 3 deste estudo e foi realizada após todas as etapas anteriores do roteiro terem sido finalizadas. Os resultados obtidos após a análise são apresentados na seção 5.3.

5.2.1 Fase 1 – Análise individual da srs

Para a análise individual da SRS, buscou-se sanar alguns tópicos levantados durante as pesquisas realizadas. Os tópicos são apresentados em forma de questões, apresentadas no Apêndice II.

5.2.2 Fase II – Análise geral das SRS

Após ter sito realizado o estudo em todas as SRS, foram realizadas outras questões, também apresentadas no Apêndice II.

5.2.3 Fase III – Análise dos resultados

Foram analisadas 7 (sete) SRS contendo, em média, 10 requisitos funcionais e 17 requisitos não-funcionais, considerando os requisitos não-funcionais específicos. Com relação aos casos de uso, observou-se que todos os requisitos funcionais em todas as SRS estavam relacionados a 1 (um) caso de uso, ou seja, relação de 1 pra 1 entre requisitos funcionais e casos de uso.

Nem todas as SRS apresentaram os requisitos em níveis diferenciados, o que dificultou, algumas vezes, a identificação do tipo de requisito especificado.

Após terem sido finalizadas todas as atividades, presentes no roteiro deste estudo de caso, todos os resultados foram analisados e discutidos e, a partir disto, tiraram-se algumas conclusões, apresentadas na seção 5.3.

5.3 Considerações finais e apresentação dos resultados

Em um primeiro momento, buscou-se, através da revisão bibliográfica, a determinação de melhores práticas para escrita de documentos de requisitos e de casos de uso. Foram analisadas quais as informações importantes, de acordo com a literatura, deveriam estar presentes nos documentos de especificação de requisitos. Além disso, foram analisadas, também, as informações indispensáveis para a composição de um modelo de casos de uso e para sua descrição.

Através deste estudo, buscou-se levantar a completude das informações analisadas nos documentos oferecidos pela empresa participante do estudo de caso e, também, identificar as diferenças entre eles, tendo em vista comparar as informações levantadas na pesquisa bibliográfica com as informações adquiridas através da realização do estudo de caso.

Sabendo-se que a prática nem sempre corresponde à teoria ideal, o estudo de caso buscou identificar se as práticas realizadas pela empresa correspondiam ao levantado na literatura. Sendo assim, os próximos tópicos apresentam as conclusões alcançadas com o estudo de caso.

5.3.1 Padrão da SRS

Conforme já citado no decorrer deste texto, existem vários padrões de escrita para SRS e estes podem e devem ser adaptados conforme as necessidades das empresas. Porém, nem sempre as organizações optam por um padrão ideal ou que, pelo menos, compreendam as informações mínimas para um resultado de sucesso.

A empresa analisada neste estudo de caso, utiliza como referência o documento de especificação de requisitos da IEEE, apresentado em [IEE98]. Porém, não são consideradas todas as seções deste documento e algumas, importantes, são deixadas de lado.

A Figura 20 mostra o padrão de SRS proposto pela IEEE [IEE84] e [IEE98] e o padrão de SRS utilizado pela empresa.

SRS – PADRÃO IEEE	SRS – PADRÃO DA EMPRESA
1. Introdução 1.1. Propósito 1.2. Escopo 1.3. Definições, acrônimos, abreviaturas 1.4. Referências 1.5. Visão geral 2. Descrição geral 2.1. Perspectivas do produto 2.1.1. Interfaces do sistema 2.1.2. Interfaces do usuário 2.1.3. Interfaces de hardware 2.1.4. Interfaces de software 2.1.5. Interfaces de comunicação 2.1.6. Memória 2.1.7. Operações 2.1.8. Requisitos de adaptação de local 2.2. Funções do produto 2.3. Características do usuário 2.4. Restrições 2.5. Condições e dependências 2.6. Distribuição dos requisitos 3. Requisitos específicos 3.1. Interfaces externas 3.2. Funções 3.2.1. Introdução 3.2.2. Entradas 3.2.3. Respostas a situações anormais 3.2.4. Processamento 3.2.5. Saídas 3.3. Requisitos de performance 3.4. Requisitos de banco de dados lógico 3.5. Restrições de design 3.6. Atributos de sistema de software 3.6.1. Confiabilidade 3.6.2. Disponibilidade 3.6.3. Segurança 3.6.4. Manutenibilidade 3.6.5. Portabilidade 4. Informações de suporte 4.1. Conteúdo 4.2. Índica 4.3. Apêndices	Histórico de aprovação do documento Conteúdo Histórico de revisão do documento 1. Introdução 1.1. Propósito do documento 1.2. Escopo do documento 1.3. Acrônimos e definições 1.4. Referências 1.5. Visão geral 2. Requisitos específicos 2.1. Interfaces 2.1.1. Interfaces do usuário 2.1.2. Interfaces de hardware 2.1.3. Interfaces de software 2.1.4. Interfaces de comunicação 2.2. Requisitos funcionais 2.2.1. *Feature* 2.2.1.1. Introdução/Propósito da *feature* 2.2.1.2. Seqüência de ação/reação 2.2.1.3. Requisitos associados 2.3. Requisitos de performance 2.4. Requisitos de banco de dados lógico 2.5. Restrições de design e implementação 2.6. Restrições de memória 2.7. Operações 2.8. Requisitos de adaptação de local 2.9. Requisitos de documentação 2.9.1. Manual de usuário 2.9.2. Guia de instalação, configuração, leia-me 2.9.3. Documentação On-line 2.9.4. Etiquetagem e Empacotamento 2.10. Atributos de sistema de software 2.10.1. Segurança 2.10.2. Manutenibilidade 2.10.3. Confiabilidade 2.10.4. Portabilidade 2.10.5. Reusabilidade 2.10.6. Testabilidade 2.10.7. Disponibilidade 2.10.8. Eficiência 2.10.9. Flexibilidade 2.10.10. Interoperabilidade 2.10.11. Robustez 2.10.12. Usabilidade 2.10.13. Outros requisitos 3. Histórico de revisão de *template* 4. Apêndices

Figura 20 – Padrões de SRS.

A Tabela 3 apresenta as relações entre as seções correspondentes nos dois documentos. Estas relações foram identificadas com base na bibliografia estudada e nas informações levantadas no estudo de caso.

Tabela 3 – Relações entre as seções das SRS.

SRS – PADRÃO IEEE	SRS – PADRÃO EMPRESA
1. Introdução	1. Introdução
1.1 Propósito	1.1 Propósito do documento
1.2 Escopo	1.2 Escopo do documento
1.3 Definições, acrônimos e abreviaturas	1.3 Acrônimos e definições
1.4 Referências	1.4 Referências
1.5 Visão geral	1.5 Visão geral
2.1.2 Interfaces de usuário	2.1.1 Interfaces de usuário
2.1.3 Interfaces de hardware	2.1.2 Interfaces de hardware
2.1.4 Interfaces de software	2.1.3 Interfaces de software
2.1.5 Interfaces de comunicação	2.1.4 Interfaces de comunicação
2.1.6 Restrições de memória	2.6 Restrições de memória
2.1.7 Operações	2.7 Operações
2.1.8 Requisitos de adaptação de local	2.8 Requisitos de adaptação de local
3.2 Funções	2.2 Requisitos funcionais
3.2.1 Introdução	2.2.1.1 Introdução/Propósito da feature
3.2.2 Entradas	2.2.1.3 Requisitos funcionais associados
3.2.4 Saídas	2.2.1.3 Requisitos funcionais associados
3.3 Requisitos de performance	2.3 Requisitos de performance
3.4 Requisitos de banco de dados lógico	2.4 Requisitos de banco de dados lógico
3.5 Restrições de design	2.5 Restrições de design e implementação
3.6 Atributos de sistema de software	2.10 Atributos de sistema de software
3.6.1 Confiabilidade	2.10.3 Confiabilidade
3.6.2 Disponibilidade	2.10.7 Disponibilidade
3.6.3 Segurança	2.10.1 Segurança
3.6.4 Manutenibilidade	2.10.2 Manutenibilidade
3.6.5 Portabilidade	2.10.4 Portabilidade
4.3 Apêndices	4. Apêndices

Realizando uma análise detalhada na tabela de relações apresentada acima, pode-se perceber que a maioria das seções sugeridas pela IEEE [IEE84] e [IEE98] são utilizadas pela empresa. Porém,

algumas são desconsideradas. As seções, propostas pela IEEE e não utilizadas pela empresa, bem como as seções adicionadas pela empresa, em relação ao padrão IEEE são apresentadas, respectivamente, na Tabela 4 e na Tabela 5.

Tabela 4 – Seções, do padrão IEEE, não utilizadas pela empresa.

SRS – PADRÃO IEEE
2.1.1 Interfaces do sistema
2.2 Funções do produto
2.3 Características do usuário
2.4 Restrições
2.5 Suposições e dependências
2.6 Distribuição dos requisitos
3.1 Interfaces externas
3.2.3 Respostas às situações anormais
3.2.4 Processamento
4.1 Tabela de conteúdo
4.2 Índice

Tabela 5 – Seções adicionadas, pela empresa, na SRS.

SRS – PADRÃO EMPRESA
2.2.1.2 Seqüência de ação/reação
2.9 Documentação dos requisitos
2.9.1 Manual de usuário
2.9.2 Guias de instalação, configuração, leia-me
2.9.3 Documentação on-line
2.9.4 Etiquetagem e empacotamento
2.10.5 Reusabilidade
2.10.6 Testabilidade
2.10.8 Eficiência
2.10.9 Flexibilidade
2.10.10 Interoperabilidade
2.10.11 Robustez
2.10.12 Usabilidade

Com base nas tabelas acima, percebe-se que algumas seções importantes, considerando o estudo realizado e o modelo de integração desenvolvido anteriormente, não estão inseridas no documento. As principais seções desconsideradas no documento da empresa são: Funções do produto,

Respostas às situações anormais e Processamento.

A seção "Funções do produto" deve conter uma relação detalhada de todas as funções do sistema, contendo todos os seus objetivos e todas as informações que se fizerem necessárias para um bom entendimento sobre o produto. A seção "Respostas às situações anormais" define todas as possibilidades de exceção na execução de um determinado requisito. A seção "Processamento" descreve todas as funções realizadas pelo sistema em resposta a uma entrada ou em suporte a uma saída. Nesta seção devem estar escritas todas as informações relacionadas às entradas.

No estudo de caso, pôde-se perceber que as funcionalidades do produto são apresentadas, apenas, na descrição do requisito funcional, apresentada na seção "Introdução/Propósito da feature" do padrão da empresa, não existindo outras seções que contextualizem sobre o objetivo do produto. Como o requisito é escrito de uma forma padronizada, nem sempre engloba todas as informações necessárias para o entendimento da funcionalidade e se englobar pode significar que ele esteja fora do padrão estipulado pela empresa, sendo que o padrão busca ser objetivo e claro.

Sendo assim, considera-se, com base na literatura e nos padrões estudados, que as seções acima descritas são fundamentais para os documentos de especificação de requisitos e devem estar presentes a fim de proporcionar que as informações ali contidas sejam mais abrangentes e mais compreensíveis no contexto do projeto como um todo.

Na seção "Funções do produto", por exemplo, baseando-se em Cockburn [COC01], sugere-se que seja criada uma lista "ator x objetivo" que relacione todos os atores envolvidos, juntamente com seus objetivos, que devem ter sido detalhados anteriormente. Esta abordagem também é adotada no trabalho apresentado por Belgamo [BEL05].

As seções de "Respostas às situações anormais" e "Processamento" permitirão que as informações sejam melhor organizadas e separadas dentro do documento. Essas informações até podem ser encontradas nas SRS da empresa, contudo não são identificadas como tal e muitas vezes se encontram em locais inadequados e diferentes, dependendo do documento.

A seção "Seqüência de ação/reação", encontrada no padrão da empresa, geralmente apresenta a descrição de um caso de uso, correspondente a um requisito funcional. Contudo, em alguns documentos podem ser encontrados, por exemplo, workflows ou textos descritivos explicando o funcionamento do produto. Sugere-se que todas as informações referentes aos requisitos funcionais sejam escritas nas seções de Funções do produto, Requisitos funcionais, Entradas, Respostas às situações anormais, Processamento e Saídas.

A principio, seria interessante que existisse um outro documento contendo o modelo de casos de uso do sistema contemplando todos os casos de uso, as relações entre eles e suas descrições [RUP06].

5.3.2 Casos de Uso

Da mesma forma que para os documentos de especificação de requisitos, existem diversas propostas para o modelo de casos de uso e suas descrições. Com base nas informações apresentadas por [COC01], [RUP06], [UML05] e outros autores alcançou-se um vasto conhecimento teórico e, a partir deste conhecimento, buscou-se identificar as semelhanças e diferenças com a prática, através do estudo de caso.

Com base nos estudos realizados, percebeu-se que a empresa não adota um padrão de especificação de casos de uso e nem sempre utiliza estes documentos. Considerando as SRS estudadas, a empresa faz uso da descrição dos casos de uso no mesmo documento de especificação de requisitos, na seção "Seqüência de ação/reação". Em nenhum documento foi encontrado um modelo de casos de uso; em quatro documentos foram encontrados diagramas de casos de uso.

Apesar de não adotar um padrão para a descrição dos casos de uso, alguns campos são comumente utilizados nos documentos. Considerando todas as SRS analisadas, os campos utilizados são: Nome do caso de uso, Objetivo, Atores, Pré-condições, Pós-condições, Curso básico, Curso alternativo e Exceção. A Figura 21 mostra o padrão para especificação de requisitos proposto por Cockburn [COC01] e os campos, para descrição dos casos de uso, utilizados nos documentos da empresa.

PADRÃO COCKBURN	CAMPOS NA EMPRESA
Caso de uso: # *<nome>* **Contexto de Uso:** **Escopo:** **Nível:** Objetivo de Usuário **Ator:** ***Stakeholders*:** **Pré-Condição:** **Garantias Mínimas:** **Garantias de Sucesso:** **Acionador:** **Cenário de Sucesso Principal:** *<# passo> <descrição da ação>* **Fluxo Alternativo:** *<# passo> <condição> : <ação ou sub caso de uso>* **Variações Tecnológicas e de Dados:** **Informações Relacionadas:**	**Nome do caso de uso:** **Objetivo:** **Atores:** **Pré-condições:** **Pós-condições:** **Curso básico:** **Curso alternativo:** **Exceção:**

Figura 21 - Relação entre o padrão de Cockburn [COC01] e os campos utilizados pela empresa.

Diferente do padrão de Cockburn [COC01], as descrições dos casos de uso apresentadas nos documentos analisados não apresentam uma padronização. Cada campo é preenchido da maneira que o responsável considerar mais adequada.

A Tabela 6 apresenta as relações entre as seções correspondentes nas duas especificações. Estas relações foram identificadas com base na bibliografia estudada e nas informações levantadas no estudo de caso.

Tabela 6 – Relações entre as especificações de casos de uso.

PADRÃO DE COCKBURN	CAMPOS UTILIZADOS NA EMPRESA
Caso de uso	Nome do caso de uso
Contexto do uso	Objetivo
Ator	Atores
Pré-condições	Pré-condições
Garantias de sucesso	Pós-condições
Cenário de sucesso principal	Curso básico
Fluxo alternativo	Curso alternativo
Fluxo alternativo	Exceção

Nota-se, pela tabela acima, uma relação dupla do campo "Fluxo Alternativo" com os campos "Curso alternativo" e "Exceção". Isto se explica pelo fato de que nas especificações de casos de uso, apresentadas nos documentos analisados, a empresa, aparentemente, utiliza estes campos para objetivos diferentes. O primeiro "Curso alternativo" seria utilizado para apresentar as diversas maneiras de executar o caso de uso com sucesso, ou seja, outras opções de seguir o caminho de sucesso sem que seja a possibilidade apresentada no "Curso Básico". Já o segundo "Exceção" seria utilizado para apresentar as condições em que o caso de uso poderia falhar. Contudo, nestes documentos foram encontradas informações incorretas ou trocadas nestes dois campos mostrando, mais uma vez, a falta de um padrão.

Outro ponto analisado é a inexistência de uma relação com os passos apresentados no curso básico. Tanto em "Curso alternativo" como em "Exceção" não são indicados os passos que o geraram. Neste trabalho, optou-se, com base em Cockburn [COC01] que não haja uma separação do fluxo alternativo em condições de sucesso ou falha, já que ele apresenta os pontos de extensão para o cenário de sucesso principal. Apenas indicar o passo ao qual determinado fluxo se refere e especificá-lo corretamente já permite que as informações sejam compreendidas de maneira adequada.

As "Pré-condições" são frases escritas, na maioria das vezes, no padrão da estrutura utilizada pela empresa. Não existe uma relação com outros casos de uso. Por exemplo, mesmo que uma pré-condição esteja diretamente ligada a outro caso de uso, esta relação não é apresentada. As "Pós-condições" na maioria das vezes não são apresentadas.

Analisando a Figura 21 percebe-se que alguns campos não foram utilizados nos documentos analisados. Estes campos podem ser vistos na Tabela 7.

Tabela 7 – Campos não utilizados pela empresa.

PADRÃO DE COCKBURN
Escopo
Nível
Stakeholders
Garantias Mínimas
Acionador
Variações Tecnológicas e de Dados
Informações Relacionadas

O escopo do caso de uso, geralmente é o escopo do produto em desenvolvimento. Sendo assim, o escopo estará implícito já que o documento dos casos de uso estará relacionado ao documento de especificação de requisitos.

A não ser que o nível das especificações de casos de uso fique claramente especificado em um documento geral, ou em algum outro local de fácil acesso a todos os interessados, é importante que este esteja especificado no template. A princípio o processo que está sendo desenvolvido propõe as especificações sempre no nível de usuários, conforme explicado anteriormente.

É importante que os interessados na execução do caso de uso estejam relacionados, para que se tenha uma noção sobre o contexto geral deste caso de uso e sobre quais os atingidos com sua execução.

As garantias mínimas são de ampla importância para o desenvolvimento de um caso de uso, já que nela estarão relacionadas todas as garantias que os interessados terão se, por ventura, o caso de uso terminar sem que a transação seja totalmente concluída.

Outro campo importante é o acionador do caso de uso, que indica como este caso de uso é iniciado.

As variações tecnológicas e de dados expressam que há diversas maneiras de o sistema fazer alguma coisa. É diferente do fluxo alternativo, pois apenas são indicadas as possibilidades de execução de uma determinada ação [COC01]. Essas possibilidades podem estar relacionadas a outros casos de uso, caso exista esta necessidade.

Nas informações relacionadas devem estar descritas todas as informações adicionais que se fizerem necessárias, indicando requisitos não-funcionais e/ou regras de negócio, por exemplo.

Sendo assim, seria interessante que a especificação de casos de uso incluísse os campos relacionados a fim de facilitar seu entendimento e de evitar a falta de informação.

5.4 Considerações finais

Este capítulo apresentou o estudo de caso realizado em uma organização de desenvolvimento de software. Teve como objetivo analisar documentos de especificação de requisitos e casos de uso a fim de identificar semelhanças e diferenças entre as propostas da literatura e as aplicações práticas. Este estudo contribuiu para o desenvolvimento do modelo e do processo propostos nesta dissertação.

O próximo capítulo apresenta o modelo proposto, juntamente com o processo e outros elementos importantes neste contexto.

6 MODELO PROPOSTO

Independentemente da natureza do projeto de software, o surgimento de novos requisitos e as mudanças dos requisitos existentes do sistema são inevitáveis [KOT98]. Acrescentando-se a complexidade, o tamanho do sistema e a enorme quantidade de requisitos a gerenciar, fica claro que várias dificuldades irão surgir, tais como: manter consistentes todos os artefatos relacionados a um requisito quando de sua alteração, gerenciar quais os artefatos devem estar relacionados e completos quando da inclusão de novos requisitos, garantir que o sistema implementado compreenda todos os requisitos e todas as informações previamente levantadas e documentadas. Para evitar esses problemas existe a necessidade de gerenciar requisitos. Para isto, o modelo proposto neste trabalho visa realizar a rastreabilidade entre o documento de especificação de requisitos e o modelo de casos de uso do sistema a fim de permitir a consistência entre seus elementos quando da alteração de artefatos do documento de especificação de requisitos.

Este modelo busca oferecer possibilidades para que o sistema a ser desenvolvido compreenda todas as funcionalidades descritas no documento de especificação de requisitos e não mais do que o especificado. Com isso, será possível permitir que o sistema resultante seja completamente aproveitado, reunindo todas as características necessárias para sua efetividade e satisfazendo todas as expectativas dos usuários. Com sua utilização, é possível indicar se alguma funcionalidade, levantada no documento de requisitos, foi desconsiderada podendo gerar conflitos no resultado.

Inicialmente foram desenvolvidos os modelos conceituais da SRS, do modelo de casos de uso e de integração, apresentados no Apêndice I. Conforme explicado anteriormente, o objetivo do desenvolvimento destes modelos era ter um ponto de partida para iniciar o processo de rastreabilidade entre os documentos da SRS e do MCU. Entretanto, estes modelos foram desenvolvidos de forma abrangente e totalmente baseados nos documentos propostos por [IEE84], [IEE98] e [COC01]. Através das outras pesquisas e, também, do estudo de caso realizado encontrou-se a necessidade de efetuar algumas adaptações a fim de facilitar o processo de rastreabilidade.

Esta seção apresenta as adaptações realizadas no modelo conceitual da SRS e no modelo conceitual do MCU, a fim de compor um modelo de rastreabilidade específico para o processo proposto. Também são apresentadas algumas considerações sobre a utilização do modelo de integração no auxílio à composição deste modelo.

6.1 Modelo conceitual da SRS

A adaptação do modelo conceitual da SRS foi realizada com base no estudo de caso realizado, sempre tendo por base o apoio da literatura. Após terem sido realizadas análises minuciosas em sete documentos de requisitos pôde-se concluir que algumas seções propostas pela IEEE [IEE84] e [IEE98] foram desconsideradas pela empresa que, por sua vez, adicionou outras seções a seu documento.

A fim de facilitar a aplicabilidade e a visualização do modelo de rastreabilidade, foco principal neste trabalho, optou-se pela adaptação do modelo conceitual da SRS com base no estudo de caso desenvolvido.

Assim sendo, foram mantidas as seções “Funções do produto”, “Respostas às situações anormais” e “Processamento” que são relacionadas nos documentos da IEEE [IEE84] e [IEE98] e não consideradas pela empresa. Por outro lado, passaram a ser considerados outros atributos de software adicionados pela empresa ao padrão do documento de requisitos que são “Reusabilidade”, “Testabilidade”, “Eficiência”, “Flexibilidade”, “Interoperabilidade”, “Robustez” e “Usabilidade”.

Além disso, outras adaptações foram sendo consideradas à medida que iam sendo realizadas aplicações com os modelos e, com isso, chegou-se ao modelo conceitual da SRS apresentado na Figura 22, cujas classes serão explicadas a seguir. Os relacionamentos utilizados no processo de rastreabilidade serão denominados posteriormente, no momento da explicação deste processo.

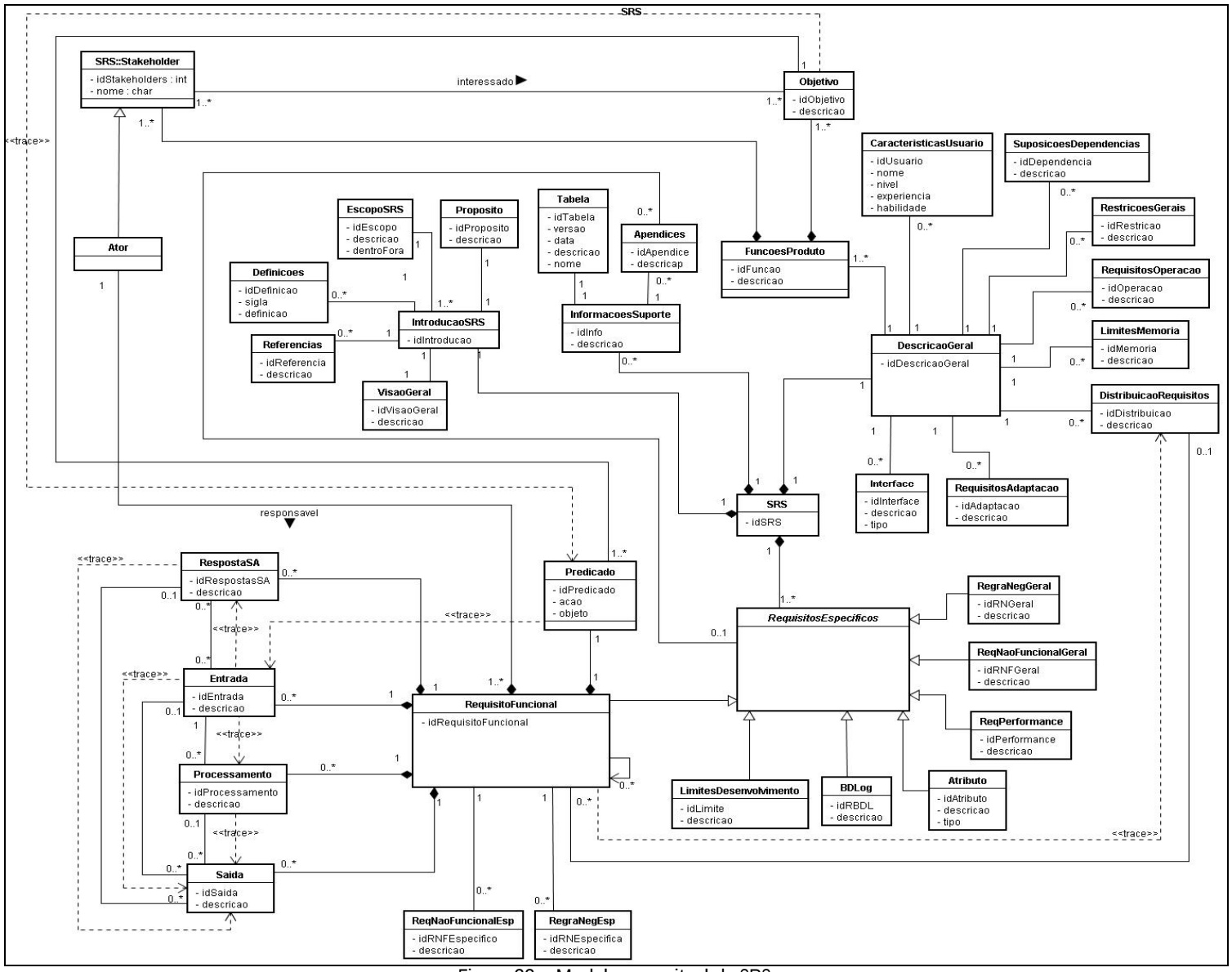


Figura 22 – Modelo conceitual da SRS.

6.1.1 Classes

As classes apresentadas no modelo conceitual da SRS representam os elementos indicados nas práticas recomendadas para especificação de requisitos de software, definidas pela IEEE [IEE84] e [IEE98], e as associações entre estes elementos representam como eles estão relacionados. Seus atributos foram identificados na descrição de cada elemento, encontradas em [IEE84], [IEE98] e [SCH00], e representam as informações referentes a cada um deles.

A seguir é apresentada uma descrição de cada uma das classes e de seus atributos. O nome das classes é representado por palavras em negrito e seus atributos por palavras em itálico.

- SRS: A classe SRS representa o documento em si, e contém um identificador idSRS que é único para cada SRS. Este documento mostra que uma SRS é composta de quatro seções maiores: Introdução, Descrição Geral, Requisitos Específicos e Informações de Suporte representadas, respectivamente, pelas classes: IntroducaoSRS, DescricaoGeral, RequisitosEspecificos e InformacoesSuporte.
- IntroducaoSRS: A classe IntroducaoSRS representa a seção Introdução da SRS e contém informações que dão uma visão geral sobre o documento. Mantém o atributo idIntroducao para armazenar o identificador da classe. Na SRS esta seção é composta de outras cinco seções: Propósito, Escopo, Definições, Referências e Visão Geral representadas, respectivamente, pelas classes Proposito, EscopoSRS, Definicoes, Referencias e VisaoGeral.
- Proposito: A classe Proposito representa a subseção Propósito da SRS que indica seu objetivo geral. Indica, também, a audiência pretendida para a SRS. Seus atributos são idProposito que mantém a identificação da classe e descricao que mantém as informações sobre a finalidade da SRS.
- EscopoSRS: A classe EscopoSRS representa a subseção Escopo da SRS e descreve os objetivos específicos, focando no software que está sendo especificado. Tem como atributos o idEscopo para manter a identificação da classe, descricao para identificar o produto de software a ser produzido e descrever, por exemplo, benefícios relevantes, objetivos e metas e o atributo inOut que fornece uma lista "dentro/fora"[4].
- Definicoes: A classe Definicoes representa a subseção Definições, Acrônimos e Abreviaturas que contém as definições de todos os termos, siglas e abreviaturas necessárias para interpretar apropriadamente a SRS. Seu atributo idDefinicao mantém a identificação do

[4] A lista dento/fora é apresentada por Cockburn [COC01] como uma opção de sucesso na definição do escopo do documento, visto que, com ela, fica facilitada a identificação do que está dentro e o que está fora do escopo de desenvolvimento.

elemento, sigla mantém a sigla a ser representada e descricao mantém a informação que identifica a sigla.

- Referencias: A classe Referencias representa a subseção Referências da SRS que identifica todos os documentos referenciados. Seus atributos idReferencia mantém o código da referência e descricao apresenta o título do documento e outras informações que se fizerem necessárias.
- VisaoGeral: A classe VisaoGeral representa a subseção Visão Geral da SRS que apresenta informações referentes à organização do documento, informação esta apresentada no atributo descricao. O atributo idVisaoGeral mantém a identificação da classe.
- DescricaoGeral: A classe DescricaoGeral representa a Seção Descrição Geral da SRS, responsável por descrever os fatores gerais que afetam o produto. Mantém o atributo idDescricaoGeral que referencia a classe. Esta seção contém outras nove subseções: Funções do Produto, Características do Usuário, Suposições e Dependências, Restrições Gerais, Requisitos de Operação, Limites de Memória, Distribuição dos Requisitos, Requisitos de Adaptação de Local e Interfaces representados, respectivamente, pelas classes FuncoesProduto, CaracteristicasUsuario, SuposicoesDependencias, RestricoesGerais, RequisitosOperacao, LimitesMemoria, DistribuicaoRequisitos, RequisitosAdaptacao e Interface.
- FuncoesProduto: A classe FuncoesProduto representa a subseção Funções do Produto da SRS que fornece uma relação das funções do sistema, a fim de informar os principais objetivos. Para isto é criado o atributo idFuncao que mantém a identificação da função e o atributo descricao que fornece uma descrição geral sobre a funcionalidade do produto. Esta classe é composta de outros dois elementos: Ator e Objetivo representados, respectivamente, pelas classes Ator e Objetivo.
- Ator: A classe Ator é responsável por manter todos os atores que farão parte do sistema. Relaciona os atributos idAtor para manter uma identificação única para cada um deles e nomeAtor que mantém seu nome.
- Objetivo: A classe Objetivo é responsável por manter todos os objetivos necessários para o desenvolvimento completo do sistema. Mantém o atributo idObjetivo que o identifica e descricao que mantém a informação sobre ele.
- Stakeholder: A classe Stakeholder armazena todos os interessados no sistema. Um ator, por exemplo, é um tipo de stakeholder. Mantém o atributo idStakeholder para sua identificação única e nome para manter seu nome.

- CaracteristicasUsuario: A classe CaracteristicasUsuario representa a subseção Características do Usuário da SRS que descreve as características gerais dos usuários do sistema. Para isto o atributo idUsuario é definido para manter a identificação do usuário, o atributo nome mantém o nome do usuário, o atributo nivel para manter o nível educacional do usuário, o atributo experiencia para manter informações sobre as experiências do usuário e o atributo habilidade para descrever suas habilidades.
- SuposicoesDependencias: A classe SuposicoesDependencias representa a subseção Suposições e Dependências da SRS que define os fatores que afetam os requisitos expressos na SRS como condições específicas de hardware. O atributo idDependencia foi definido para manter a identificação da classe e o atributo descricao foi definido para manter a descrição da informação.
- RestricoesGerais: A classe RestricoesGerais representa a subseção Restrições Gerais da SRS que fornece uma descrição geral de qualquer outro item que limite as opções dos desenvolvedores como normas reguladoras, limites de hardware, protocolos etc. Para isto o atributo idRestricao foi definido para manter a identificação da restrição e o atributo descricao para manter sua informação.
- RequisitosOperacao: A classe RequisitosOperacao representa a subseção Operação da SRS e descreve todas as operações normais e/ou especiais requisitadas pelo usuário, como rotinas de inicialização, processamento, backup's e restauração. É criado, para isto, o atributo idOperacao que mantém a identificação de cada operação e o atributo descricao que define a operação.
- LimitesMemoria: A classe LimitesMemoria representa a subseção Limites de Memória da SRS que especifica a memória (interna e externa) a ser, provavelmente, utilizada pelo software. É criado, para isto, o atributo idMemoria que mantém a identificação de cada limite e o atributo descricao que define esta especificação.
- DistribuicaoRequisitos: A classe DistribuicaoRequisitos representa a subseção Distribuição dos Requisitos na SRS que identifica os requisitos que podem ser adiados até versões futuras do sistema. Para isto foi criado o atributo idDistribuicao que mantém a identificação da classe e descricao que deve definir os motivos pelos quais tal requisito pode ser adiado.
- RequisitosAdaptacao: A classe RequisitosAdaptacao representa a subseção Requisitos de Adaptação do Local da SRS que contém a especificação das situações em que o software deverá ser adaptado antes da instalação. Para isto, é criado o atributo idAdaptacao que identifica o requisito e o atributo descricao que mantém a informação sobre ele.

- Interface: A classe Interface representa as informações referentes as interfaces do sistema. Para isto, foi definido o atributo idInterface que armazena a identificação da interface, descricao que mantém a informação sobre ela e tipo que indica o tipo de interface dentre os seguintes: Interfaces de Usuário, Interfaces do Hardware, Interfaces do Software e Interfaces de Comunicação.

- RequisitosEspecificos: A classe RequisitosEspecificos é uma classe abstrata que representa a Seção Requisitos Específicos da SRS que descreve todas as necessidades, em nível de detalhe suficiente, para que os desenvolvedores estejam aptos a satisfazer essas necessidades no desenvolvimento do sistema. Os requisitos específicos podem ser dos tipos: Regra de Negócio Geral, Requisito Não-funcional Geral, Requisito de Performance, Atributo, Banco de Dados Lógico, Limites de Desenvolvimento e Requisito Funcional representados, respectivamente, pelas classes RegraNegGeral, ReqNaoFuncionalGeral, ReqPerformance, Atributo, BDLog, LimitesDesenvolvimento e RequisitoFuncional.

- RegraNegGeral: A classe RegraNegGeral representa a seção Regra de Negócio Geral, identificada através do estudo de caso para armazenar as regras de negócio que se relacionam ao sistema como um todo. Para esta classe o atributo idRNGeral mantém sua identificação e o atributo descricao mantém a informação referente a regra.

- ReqNaoFuncionalGeral: A classe ReqNaoFuncionalGeral representa a seção Requisito Não-funcional Geral identificada no estudo de caso para armazenar os requisitos não-funcionais referentes ao sistema como um todo. Para ela, são definidos os atributos idRNFGeral que mantém sua identificação e descricao que mantém sua informação.

- ReqPerformance: A classe ReqPerformance representa a seção Requisitos de Performance da SRS que é responsável por especificar os requisitos numéricos estáticos e dinâmicos existentes no software. Todos estes requisitos devem ser expressos em termos mensuráveis. O atributo idPerformance foi criado para armazenar sua identificação e para a especificação deste requisito é utilizado o atributo descricao.

- Atributo: A classe Atributo representa a seção Atributos do Sistema de Software da SRS, que especifica os atributos do software. Eles podem ser especificados desde que sejam verificáveis. É definido o atributo idAtributo para manter sua identificação, descricao para manter a informação e o atributo tipo para classificar o tipo do atributo dentre os seguintes: confiabilidade, disponibilidade, segurança, manutenibilidade, portabilidade, reusabilidade, testabilidade, eficiência, flexibilidade, interoperabilidade, robustez e usabilidade.

- BDLog: A classe BDLog representa a seção Requisitos de Banco de Dados Lógico (BDL) da SRS, responsável por especificar os requisitos lógicos das informações armazenadas em um banco de dados. Para esta classe são definidos os atributos idRBDL que identifica o requisito e descricao que mantém a especificação do requisito de BDL.
- LimitesDesenvolvimento: A classe LimitesDesenvolvimento representa a seção Limites de Desenvolvimento da SRS que informa as restrições impostas por fatores como limitações de hardware e software. Define o atributo idLimite que identifica cada informação e descricao que mantém estas informações.
- RequisitoFuncional: A classe RequisitoFuncional representa a seção Requisitos Funcionais da SRS que especifica o processamento e as saídas do software para determinadas entradas. Para esta classe é definido o atributo idRequisitoFuncional que mantém a identificação única de cada requisito. Esta classe é composta por outras seis subclasses: Ator, Predicado, RespostasSA, Entrada, Processamento e Saida. Além destas, outras duas classes se relacionam com ela: ReqNaoFuncionalEsp e RegraNegEsp. A classe Ator relacionada é a mesma já especificada anteriormente, também ligada a classe FuncoesProduto. Entretanto, neste caso, sua instância é responsável por representar o ator do requisito funcional.
- Predicado: A classe Predicado representa o predicado para o requisito funcional, que junto com o ator forma a descrição completa do requisito. É composta dos atributos idPredicado que identifica a classe, acao que descreve a ação do predicado e objeto que descreve o objeto em questão.
- RespostasSA: A classe RespostasSA é encontrada na seção Requisitos Funcionais e define todas as possibilidades de exceção na execução de determinado requisito. O atributo idRespostasSA mantém a identificação da classe e descricao é o atributo criado para manter tais possibilidades.
- Entrada: A classe Entrada é encontrada na definição da seção Requisitos Funcionais da SRS e identifica todas as entradas possíveis para determinado requisito. Para ela é definido o atributo idEntrada que mantém sua identificação e descricao utilizado para especificar e descrever as entradas.
- Processamento: A classe Processamento é encontrada na definição da seção Requisitos Funcionais na SRS e descreve todas as funções realizadas pelo sistema em resposta a uma entrada ou em suporte a uma saída. É definido o atributo idProcessamento que mantém a identificação da classe e descricao responsável pela especificação do processamento.

- Saida: A classe Saida faz parte da definição da seção Requisitos Funcionais na SRS, é responsável por manter todas as saídas possíveis para o requisito. Para isto é definido o atributo idSaida que mantém sua identificação e descricao para manter as determinadas saídas para o requisito.
- ReqNaoFuncionalEsp: A classe ReqNaoFuncionalEsp representa a seção Requisitos Não-funcionais Específicos, identificada através do estudo de caso. É responsável por manter as informações sobre os requisitos não-funcionais específicos de determinado requisito funcional. Para esta classe é definido o atributo idRNFEspecifico que mantém sua identificação e o atributo descricao que mantém sua informação.
- RegraNegEsp: A classe RegraNegEsp representa a seção Regra de Negócio Específica, identificada através do estudo de caso. É responsável por manter as informações sobre as regras de negócio específicas de determinado requisito funcional. Para esta classe é definido o atributo idRNEspecifica que mantém sua identificação e o atributo descricao que mantém sua informação.
- InformacoesSuporte: A classe InformacoesSuporte define a seção Informações do Suporte da SRS responsável por tornar a SRS mais fácil de ser utilizada. Define o atributo idInfo para manter sua identificação e o atributo descricao para manter informações necessárias para a compreensão da SRS. Ainda são definidas duas subseções: Tabela de Conteúdo e Índice e Apêndices representadas, respectivamente, pelas classes Tabela e Apendices.
- Tabela: A classe Tabela representa a subseção Tabela de Conteúdo e Índice da SRS seguindo as práticas convencionais. Para isto é definido o atributo idTabela para manter a identificação da referência, versao que mantém a versão atual da SRS, data que mantém a data da atualização, descricao que mantém a informação sobre a referência e nome que mantém o responsável pela atualização.
- Apendices: A classe Apendices representa a subseção Apêndices da SRS que os especifica, se necessário. Caso sejam identificados, deve ser informado se eles são ou não parte dos requisitos. Para isto é definido o atributo idApendice que identifica o apêndice e o atributo descricao que informa se ele faz parte ou não dos requisitos.

6.2 Modelo conceitual do modelo de casos de uso

A pesquisa apresentada em [SMI05] mostra uma forma eficiente de estruturação para os cenários dos casos de uso. Este trabalho contribuiu para parte da adaptação do modelo conceitual do MCU. As relações identificadas são descritas a seguir.

- SPO(PO) Sentence / Passo: esta relação foi identificada pois a classe Passo do MCU tem o mesmo objetivo da classe SPO(PO) Sentence do metamodelo apresentado em [SMI05]. Ambas buscam apresentar uma ação na descrição do caso de uso. Sendo assim, a classe Passo foi adaptada conforme a classe SPO(PO) Sentence.
- Sentence Subject / Participante: esta relação foi identificada visto que a classe Sentence Subject tem por objetivo representar o sujeito da frase, ou seja, o participante de cada passo da descrição do caso de uso, representado pela classe Participante no MCU.
- Conditional sentence / FluxoAlternativo: esta relação foi identificada visto que ambas as classes representam as possibilidades de exceção na descrição do caso de uso.

A Figura 23 apresenta as relações identificadas entre os modelos.

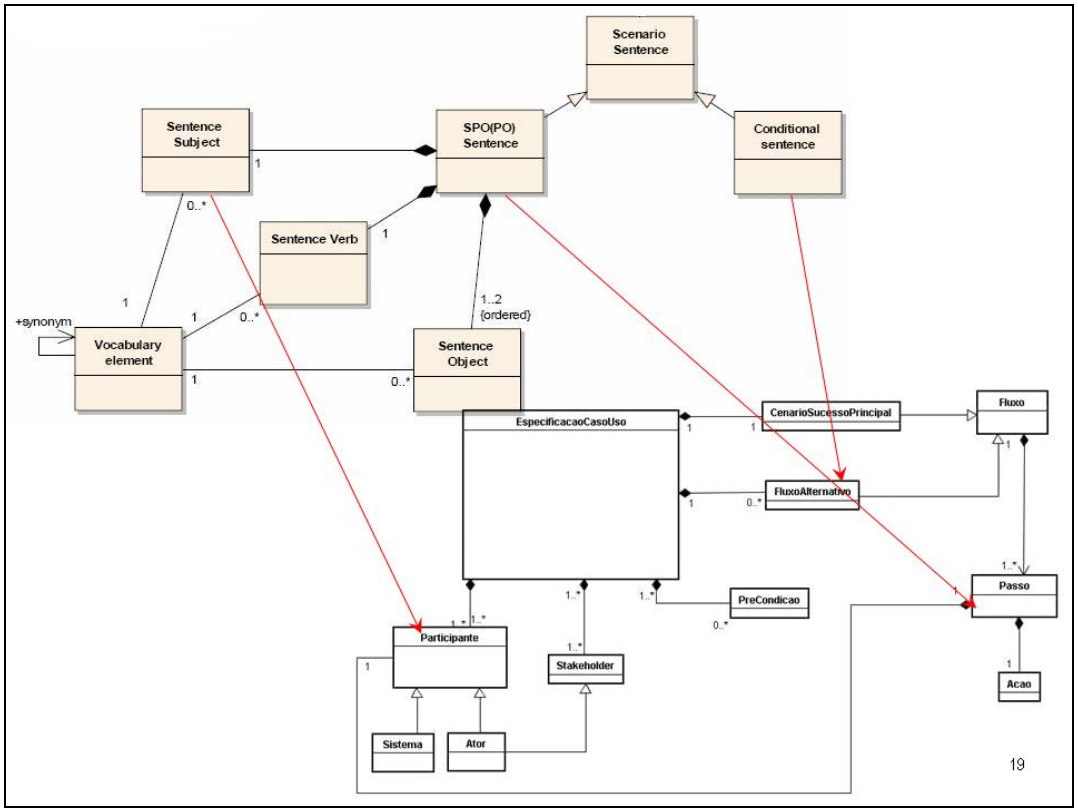


Figura 23 – Relação do metamodelo de [SMI05] com o modelo conceitual do MCU.

Estas relações foram consideradas importantes, visto que tendem a reduzir a ambigüidade da descrição dos casos de uso e tendem a facilitar, também, que esta descrição seja compreendida da mesma maneira por qualquer pessoa que a leia.

A partir da adaptação apresentada acima, chegou-se ao modelo conceitual do MCU que será utilizado como ponto de partida para o modelo de rastreabilidade. Este modelo pode ser visto na Figura 24.

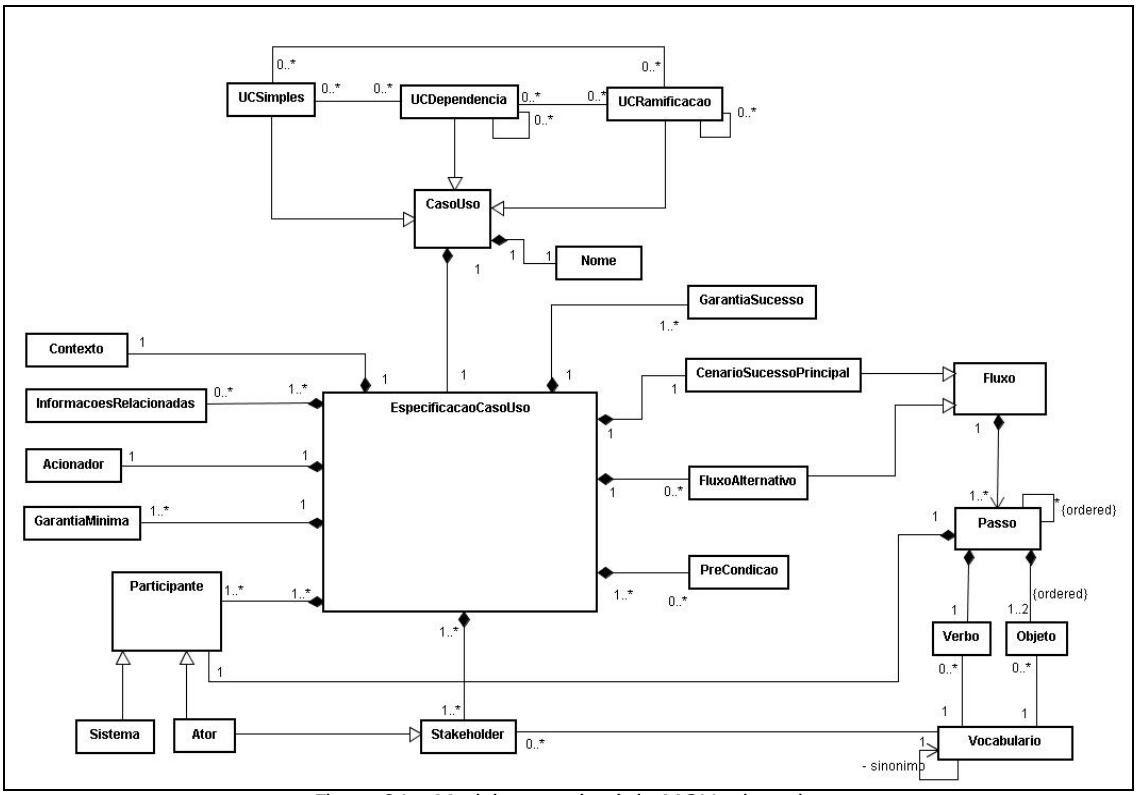


Figura 24 – Modelo conceitual do MCU adaptado.

Sendo assim, a classe Passo continua a ser relacionada com a classe Participante e passa a se relacionar, também, com as classes Verbo e Objeto. As classes Verbo e Objeto passam a se relacionar com a classe Vocabulario e, ao invés de esta classe ser relacionada com a classe Participante, optou-se por realizar este relacionamento com a classe Stakeholders. Esta opção se deu pelo fato de que todos os atores são stakeholders, de acordo com o modelo. Ou seja, todos os atores estarão relacionados com o vocabulário.

A seguir será apresentada a descrição de todas as classes do modelo.

6.2.1 Classes

As classes apresentadas no modelo conceitual do MCU representam os elementos indicados na literatura [COC01], [RUP06] e [SMI05] e as associações entre estes elementos representam como eles estão relacionados. Com relação aos atributos, todas as classes possuem um identificador e uma descrição responsável por armazenar suas informações.

- CasoUso: A classe CasoUso define que os casos de uso podem ser dos tipos: simples, dependência e ramificação. É composta das classes Nome e EspecificacaoCasoUso.
- UCSimples: A classe UCSimples é definida para indicar se o caso de uso é um caso de uso simples.

- UCDependencia: A classe UCDependencia é definida para indicar se o caso de uso em questão é um caso de uso do tipo <include> ou do tipo <extend>.
- UCRamificacao: A classe UCRamificacao é definida para indicar se o caso de uso é um caso de uso de generalização.
- Nome: A classe Nome é responsável por manter o nome do caso de uso, representa o elemento Nome do caso de uso na especificação apresentada no template utilizado neste trabalho [COC01].
- EspecificacaoCasoUso: A classe EspecificacaoCasoUso é composta por todas as outras classes que fazem parte da especificação dos casos de uso, ou seja, de sua descrição. São elas: Contexto, InformacoesRelacionadas, Acionador, GarantiaMinima, Participante, Stakeholder, GarantiaSucesso, PreCondicao, CenarioSucessoPrincipal e FluxoAlternativo.
- Contexto: A classe Contexto mantém uma sentença maior do objetivo e representa o Contexto no template utilizado [COC01].
- InformacoesRelacionadas: A classe InformacoesRelacionadas relaciona todas as informações adicionais necessárias. Elas podem ser requisitos não-funcionais ou regras de negócio relacionadas ao caso de uso, bem como qualquer outra informação que se fizer necessária para a especificação.
- Acionador: A classe Acionador mantém a informação referente ao que dá inicio ao caso de uso.
- GarantiaMinima: A classe GarantiaMinima define como os interesses são protegidos sob todas as saídas, ou seja, se o caso de uso falhar quais são as garantias dadas aos interessados.
- Participante: A classe Participante armazena as informações do participante do caso de uso, que pode ser o sistema ou o ator.
- Sistema: A classe Sistema representa um participante do caso de uso.
- Ator: A classe Ator armazena o ator primário do caso de uso, representa um participante.
- Stakeholder: A classe Stakeholder relaciona todos os interessados em determinado caso de uso.
- GarantiaSucesso: A classe GarantiaSucesso define quais os interesses dos stakeholders são satisfeitos depois de uma conclusão bem-sucedida do caso de uso.
- PreCondicao: A classe PreCondicao define o que já deve ser verdadeiro para que seja possível a execução do caso de uso.
- CenarioSucessoPrincipal: A classe CenarioSucessoPrincipal é um tipo de fluxo, responsável por manter uma execução de sucesso para o caso de uso.

- FluxoAlternativo: A classe FluxoAlternativo é um tipo de fluxo, ou seja, define as mesmas características da classe fluxo. Representa todas as formas alternativas de o caso de uso ser executado.
- Fluxo: A classe Fluxo é responsável por armazenar um conjunto de passos que serão executados para a realização de determinada funcionalidade. Estes passos são representados através da classe Passo.
- Passo: A classe Passo é responsável por descrever cada passo a ser executado dentro de um fluxo. Os passos devem estar ordenados. Cada um deles é composto de um participante (representado no modelo através da classe Participante) , de um verbo (representado no modelo através da classe Verbo) e de um ou dois objetos (representado no modelo através da classe Objeto).
- Verbo: A classe Verbo é responsável por manter o verbo representado no passo do caso de uso.
- Objeto: A classe Objeto mantém os objetos que participam do passo do caso de uso.
- Vocabulario: A classe Vocabulario mantém os verbos, objetos e, também, os stakeholders relacionados ao caso de uso. Nesta classe é indicado, também, os sinônimos para cada item utilizado.

6.3 Utilização do modelo de integração

O modelo de integração, apresentado em [CER06a] e no Apêndice I, foi utilizado como ponto de partida para o desenvolvimento do modelo de rastreabilidade. A partir dos relacionamentos identificados neste modelo e no estudo de caso, foi possível chegar a um conjunto de relacionamentos entre os elementos dos modelos da SRS e do MCU.

Assim, com base inicial nestes relacionamentos, partiu-se para o desenvolvimento do modelo de rastreabilidade, que será apresentado a seguir.

6.4 Proposta de um modelo de rastreabilidade

Realizadas as adaptações nos modelos conceituais, partiu-se para o desenvolvimento do modelo de rastreabilidade (MR). O objetivo deste modelo é demonstrar as relações entre os documentos da SRS e do MCU, considerando os impactos[5] que por ventura possam vir a acontecer caso alguma alteração seja realizada em um dos dois documentos.

[5] No sentido desta dissertação, impactos acontecem quando uma alteração em um elemento afeta outro. Neste caso, diz-se que o elemento afetado por esta alteração foi impactado.

As relações foram identificadas pelos estudos bibliográficos e, também, pelo estudo de caso realizado, tendo como ponto de partida o modelo conceitual de integração, desenvolvido previamente.

Uma das principais dúvidas relacionadas ao modelo de rastreabilidade surgiu devido à correspondência de um requisito com um ou mais casos de uso. [LEF00] afirma que uma feature pode estar relacionada a vários casos de uso, sabendo-se que uma feature pode ser satisfeita por um ou vários requisitos e cada um deles irá representar um objetivo do software. Em [SMI05] é relatado que um caso de uso é um serviço único (com um único objetivo) e, ainda, explica que cada caso de uso pode ser composto de vários cenários sendo que todos eles buscam satisfazer ao objetivo do caso de uso em questão. Já em [BEL05] e [BEL05b] considera-se que cada objetivo seja transformado em um caso de uso, sabendo-se que cada um destes objetivos corresponde a uma grande funcionalidade do sistema.

Pelo exposto em [LEF00], [SMI05], [BEL05] e [BEL05b], com base no estudo de caso realizado e, também, pelo padrão de escrita de requisitos adotado neste trabalho, foi definido que um requisito dará origem a um caso de uso.

O modelo de rastreabilidade e os impactos entre as classes são apresentados, respectivamente, pelo diagrama de classes e pelas relações de dependência trace[6] propostas pela UML [UML05].

A fim de facilitar a leitura e a visibilidade do processo, este modelo não compreende todas as classes dos modelos conceituais da SRS e do MCU, pois pela quantidade de classes, relacionamentos e pela dificuldade de leitura ficaria inviável representar todas as dependências em um único modelo. Porém, isto não dificulta a visibilidade, visto que as classes são apresentadas com a identificação de seu diagrama de origem para que possam ser facilmente identificadas.

Os impactos podem ocorrer de três maneiras:

- Elementos da SRS impactando em elementos da SRS;
- Elementos da SRS impactando em elementos do MCU;
- Elementos do MCU impactando em elementos do MCU;

Sendo assim, as dependências entre os elementos da SRS são apresentadas no modelo conceitual da SRS; as dependências entre os elementos da SRS e os elementos do MCU e as dependências que ocorrem por propagação entre os elementos do MCU são apresentadas no modelo de rastreabilidade.

[6] As relações de dependência trace especificam os relacionamentos entre elementos do modelo ou conjunto de elementos que representam o mesmo conceito em diferentes modelos. Sabendo-se que as mudanças podem ser realizadas em ambas as direções, geralmente a direção das dependências pode ser ignorada [UMO05].

Com isso, são definidos os artefatos que fazem parte do processo de rastreabilidade: o modelo conceitual da SRS, o modelo de rastreabilidade e os procedimentos de rastreabilidade para cada elemento.

A Figura 25 apresenta o modelo de rastreabilidade proposto, que também pode ser conferido em [CER06b].

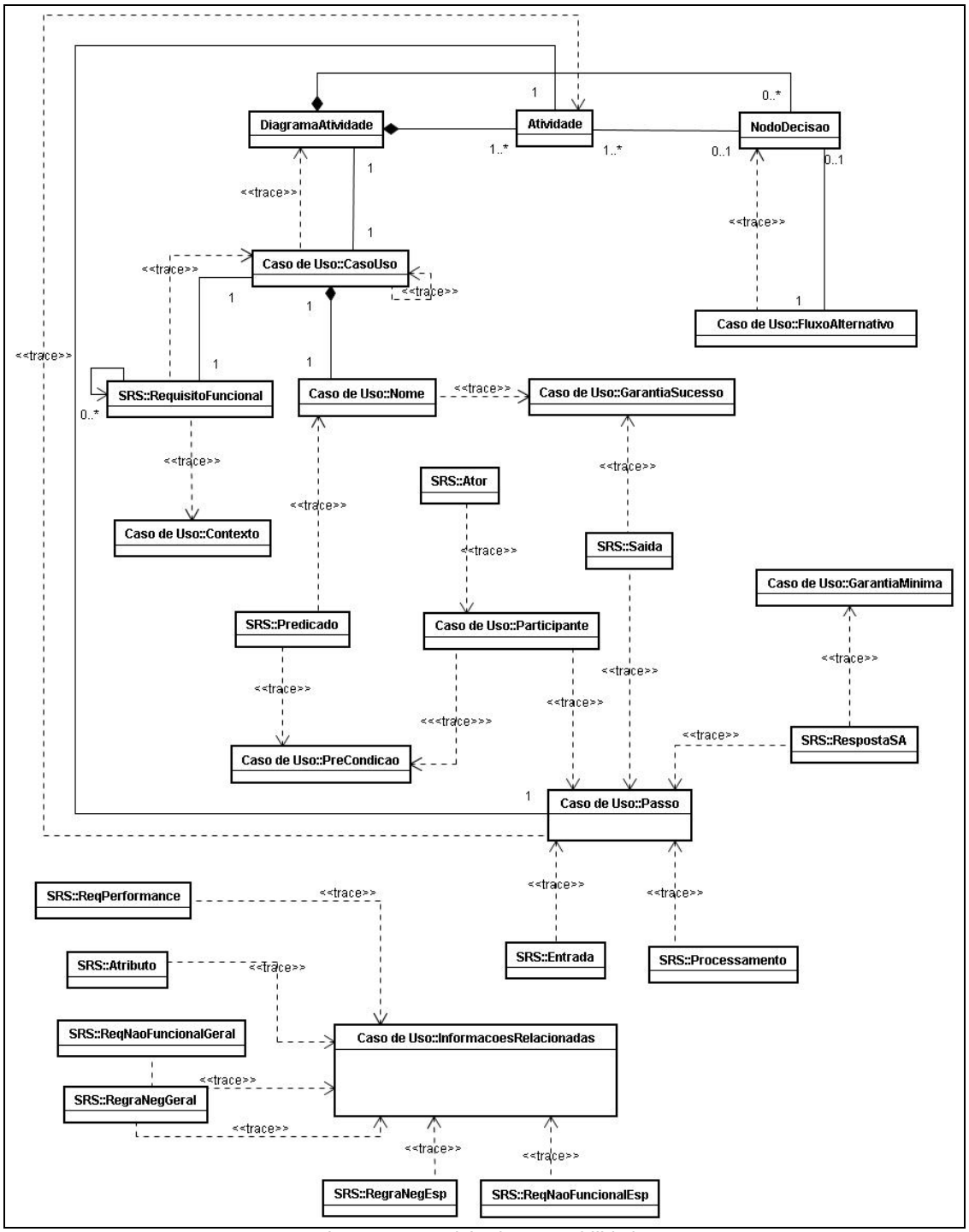


Figura 25 – Modelo de rastreabilidade.

6.4.1 Classes

As classes utilizadas no modelo são classes derivadas dos modelos conceituais da SRS, do MCU, do modelo de integração e de pesquisas bibliográficas para complementação.

Do modelo conceitual da SRS, as classes participantes do processo são: Ator, Stakeholder, Predicado, RequisitoFuncional, Saida, RespostasSA, Entrada, Processamento, Atributo, Performance, ReqNaoFuncionalGeral, RegraNegGeral, RegraNegEsp, ReqNaoFuncionalEsp, Objetivo e DistribuicaoRequisitos.

Do modelo conceitual do MCU, as classes participantes são: CasoUso, Nome, PreCondicao, Participante, Stakeholder, GarantiaSucesso, Contexto, FluxoAlternativo, Passo, InformacoesRelacionadas e GarantiaMinima.

As classes adicionadas ao modelo para complementação, com base em [SMI05], foram: DiagramaAtividade, Atividade, NodoDecisao.

A descrição das classes derivadas dos modelos da SRS e do modelo de casos de uso já foi apresentada. A descrição das classes adicionais é apresentada a seguir:

- DiagramaAtividade: a classe DiagramaAtividade corresponde ao diagrama de atividades representado em [OMG05] e é composta de duas subclasses: Atividade e NodoDecisao, também oriundas de [OMG05].
- Atividade: a classe Atividade representa uma ação realizada no diagrama de atividades.
- NodoDecisao: a classe NodoDecisao representa um nodo de decisão do diagrama de atividades.

A próxima seção apresenta os relacionamentos que, junto com as classes, compõem o modelo de rastreabilidade.

6.4.2 Relacionamentos

Esta seção descreve os relacionamentos existentes entre as classes do modelo de rastreabilidade proposto. Cada um dos relacionamentos é descrito devido a sua importância em processos de especificação de requisitos e casos de uso e, assim, devem ser seguidos a fim de garantir que a rastreabilidade entre os elementos possa ser realizada durante o processo.

Cada um dos relacionamentos apresentados no modelo foi identificado a partir de análises detalhadas no modelo conceitual da SRS, do MCU e de Integração, embora o estudo de caso tenha contribuído para a confirmação, adaptação e inclusão de novos relacionamentos.

A fim de representar as dependências entre os elementos, faz-se uso da relação de dependência trace prevista na UML 2.0 [UML05].

Abaixo são apresentadas as descrições de todos os relacionamentos de dependência apresentados nos modelos e que fazem parte do processo de rastreabilidade.

- Objetivo/Predicado: este relacionamento indica que uma instância da classe Objetivo estará representada na instância de Predicado da descrição de um requisito funcional, ou seja, caso este objetivo seja alterado sua instância relacionada de Predicado deverá ser, também, atualizada.
- RequisitoFuncional/DistribuicaoRequisitos: este relacionamento demonstra que caso uma instância da classe RequisitoFuncional seja alterada, a classe DistribuicaoRequisitos deverá ser analisada a fim de identificar a presença, ou não, desta instância.
- Predicado/Entrada: este relacionamento indica que se houver alteração na instância de Predicado da descrição do requisito funcional, as instâncias da classe Entrada poderão ser impactadas e, por isso, devem ser analisadas.
- Entrada/RespostaSA: quando existir a possibilidade de falha em alguma entrada, haverá uma instância de Entrada relacionada a uma instância de RespostasSA. Este relacionamento indica que caso haja alteração em uma instância de Entrada, as instâncias correspondentes de RespostasSA devem ser analisadas a fim de verificar a necessidade de alteração.
- Entrada/Processamento: quando existir a necessidade de que uma entrada seja testada, avaliada ou qualquer outro tipo de processamento necessário, haverá uma instância da classe Processamento ligada a uma instância da classe Entrada. Este relacionamento indica que caso haja alteração em uma instância de Entrada, as instâncias correspondentes de Processamento devem ser analisadas a fim de verificar a necessidade de alteração.
- Entrada/Saida: quando uma interface de saída estiver diretamente relacionada a uma interface de entrada, haverá uma instância de Saida ligada a uma instância de Entrada. Este relacionamento indica que caso haja alteração em uma instância de Entrada, as instâncias correspondentes de Saida devem ser analisadas a fim de verificar a necessidade de alteração.
- RespostaSA/Saida: quando uma resposta a situação anormal originar uma saída, existirá uma instância de RespostasSA ligada a uma instância de Saida. Este relacionamento indica que caso haja alteração em uma instância de RespostasSA, as

instâncias correspondentes de Saida devem ser analisadas a fim de verificar a necessidade de alteração.

- Processamento/Saida: quando uma interface de saída estiver diretamente relacionada a execução de determinado processamento, haverá uma instância da classe Saida ligada a uma instância da classe Processamento. Este relacionamento indica que caso haja alteração em uma instância de Processamento, as instâncias correspondentes de Saida devem ser analisadas a fim de verificar a necessidade de alteração.
- RequisitoFuncional/CasoUso: um caso de uso será desenvolvido a partir da descrição de um requisito funcional. Sendo assim, qualquer alteração na instância de RequisitoFuncional pode afetar a instância relacionada de CasoUso. Além do relacionamento de dependência entre os dois elementos, tem-se, também, uma associação que garante que um requisito estará sempre relacionado a um caso de uso.
- CasoUso/CasoUso: um caso de uso pode estar diretamente relacionado a outros casos de uso, através de relacionamentos de include, extend e generalização. Assim, este relacionamento indica que uma alteração em uma instância da CasoUso pode afetar outras instâncias de CasoUso que estejam ligados a ele.
- Ator/Participante: o ator do requisito funcional será o participante do caso de uso. Sendo assim, caso uma instância de Ator do requisito seja alterada, a instância correspondente de Participante do caso de uso deverá ser analisada conforme esta alteração.
- Predicado/Nome: o nome do caso de uso é retirado do predicado do requisito funcional. Assim, caso haja alteração na funcionalidade do requisito, alterando sua instância de Predicado, a instância de Nome do caso de uso também deverá ser alterada.
- Predicado/PreCondicao: a pré-condição do caso de uso pode estar referenciando a funcionalidade do requisito (predicado), então, caso haja uma alteração na instância de Predicado, a instância de PreCondicao do caso de uso correspondente deve ser analisada a fim de identificar a necessidade de alteração.
- Participante/PreCondicao: muitas vezes a pré-condição pode estar relacionando o participante do caso de uso. Assim, caso haja alteração neste participante, a pré-condição também deve ser revista.
- Participante/Passo: o participante estará nos passos do cenário de sucesso principal e/ou do fluxo alternativo. Assim, caso a instância de Participante seja alterada é

necessário que as instâncias de Passo ligadas a este participante sejam identificadas e modificadas.

- Nome/GarantiaSucesso: o nome do caso de uso apresenta um resumo de sua funcionalidade, ou seja, o que se busca neste caso de uso. Sendo assim, uma instância de GarantiaSucesso será sempre relacionada a uma instância de Nome, já que uma garantia de sucesso é ter a funcionalidade do caso de uso realizada e qualquer alteração nele efetuada afetará a garantia de sucesso dali retirada.
- Saida/GarantiaSucesso: das saídas de um requisito podem ser retiradas outras garantias de sucesso para um caso de uso. Com isso, caso seja realizada alguma alteração nas instâncias de Saida dos requisitos, as instâncias de GarantiasSucesso relacionadas devem ser revistas e atualizadas.
- RespostasSA/Passo: informações para o cenário de sucesso principal e/ou fluxo alternativo podem ser retiradas das respostas às situações anormais visto que podem conter informações que indiquem os passos para os fluxos. Assim, caso uma instância de RepostasSA seja alterada, as instâncias correspondentes de Passo devem ser revistas.
- Processamento/Passo: informações para o cenário de sucesso principal e/ou fluxo alternativo podem ser retiradas dos processamentos visto que podem conter informações que indiquem os passos para os fluxos. Assim, caso uma instância de Processamento seja alterada, as instâncias correspondentes de Passo devem ser revistas.
- Entrada/Passo: informações para o cenário de sucesso principal e/ou fluxo alternativo podem ser retiradas das interfaces de entrada dos requisitos funcionais visto que as mesmas poderão ser transformadas em passos para os casos de uso. Com isso, alterações nas instâncias de Entrada podem impactar em instâncias de Passo tanto do cenário de sucesso principal quanto dos fluxos alternativos do caso de uso.
- Saida/Passo: informações para o cenário de sucesso principal e/ou fluxo alternativo podem ser retiradas das interfaces de saída visto que podem conter informações que indiquem os passos para os fluxos. Assim, caso uma instância de Saida seja alterada, as instâncias correspondentes de Passo devem ser revistas.
- RequisitoFuncional/Contexto: o contexto do caso de uso é representado pela descrição do requisito funcional. Com isso, qualquer alteração realizada em uma instância de RequisitoFuncional, que deu origem a um caso de uso, impactará em alterações na instância correspondente de Contexto.

- RespostasSA/GarantiaMinimas: as garantias mínimas podem ser retiradas das respostas às situações anormais dos requisitos funcionais, pois ali são identificados pontos de falha para o requisito e assim eles podem ser tratados. Sendo assim, se alterações forem efetuadas nas instâncias de RespostasSA, as instâncias de GarantiaMinima do caso de uso correspondente devem ser revistas.
- Atributo, ReqPerformance, ReqNaoFuncionalGeral, ReqNaoFuncionalEspecifico, RegraNegGeral, RegraNegEspecifica/Informações Relacionadas: uma maneira de identificar as informações relacionadas do caso de uso, é analisar as informações de atributos, requisitos de performance, requisito não-funcional geral, requisito não-funcional específico, regra de negócio geral e regra de negócio específica, descritos na SRS. Sendo assim, caso as instâncias de Atributo, ReqPerformance, ReqNaoFuncionalGeral, ReqNaoFuncionalEsp, RegraNegGeral e RegraNegEsp sejam alteradas, as instâncias correspondentes de InformacoesRelacionadas também devem ser atualizadas.
- CasoUso/DiagramaAtividade: este relacionamento indica que um caso de uso será representado em um diagrama de atividades, com isso uma instância de CasoUso estará ligada a uma instância de DiagramaAtividade.
- DiagramaAtividade/Atividade: este relacionamento de composição representa que um diagrama de atividade é composto de várias atividades [SMI05].
- DiagramaAtividade/NodoDecisao: este relacionamento de composição representa que um diagrama de atividade pode ter vários nodos de decisão [SMI05].
- Atividade/NodoDecisao: esta associação indica que uma atividade pode estar relacionada ou não a um nodo de decisão e que, se o nodo de decisão existir, este estará relacionado à apenas uma atividade [SMI05].
- FluxoAlternativo/NodoDecisao: este relacionamento indica que uma instância de NodoDecisao poderá estar relacionada a uma instância de FluxoAlternativo, conforme proposto em [SMI05].
- Passo/Atividade: este relacionamento indica que uma instância de Passo será representada através de uma instância de Atividade [SMI05].

Os relacionamentos, acima descritos, são de ampla importância para que a rastreabilidade entre os documentos da SRS e do MCU seja mantida. A próxima seção apresenta o processo de rastreabilidade, explicando suas regras e os procedimentos a serem seguidos para que a execução do processo seja realizada com eficiência.

6.5 Processo de rastreabilidade

O processo de rastreabilidade demonstra como o rastreamento entre as instâncias das classes será realizado. Para sua concretização, serão necessários três artefatos: o modelo conceitual da SRS, com dependências, o modelo de rastreabilidade e o conjunto de procedimentos para a aplicação do processo.

Os procedimentos estabelecem como os elementos e relacionamentos, apontados entre os modelos, devem ser tratados durante o processo, a fim de garantir que os documentos permaneçam consistentes ao final de sua aplicação.

Este processo é realizado no sentido SRS à MCU (documento de especificação de requisitos impactando no modelo de casos de uso) e é dividido em três fases:

- 1ª FASE: AnalisarRastro (ElementoInicial, SRS) à conjunto de SubElementos impactados da SRS;
- 2ª FASE: AnalisarRastro (ElementoInicial, MR) à conjunto de elementos impactados do MCU;
- 3ª FASE: AnalisarRastro (SubElemento, MR) à conjunto de elementos impactados do MCU;

A primeira fase identifica como as instâncias das classes da SRS são impactadas por uma alteração em outra instância da SRS (SRS ß SRS). Esta fase utiliza como recurso a instância alterada (ElementoInicial) e o modelo conceitual da SRS (SRS). Ao término da primeira fase, ter-se-á, como resultado, um conjunto de SubElementos impactados, que serão instâncias de determinadas classes afetadas pelo impacto.

Na segunda fase serão identificados, no MCU, as instâncias das classes impactadoas pelas alterações nos SubElementos (SubElemento), encontrados na primeira fase (MCU ß SRS). Esta fase é realizada sob o modelo de rastreabilidade (MR).

A terceira fase corresponde à identificação das instâncias das classes do MCU impactadas pelas alterações realizadas no ElementoInicial (MCU ß SRS). Esta fase utiliza como recurso a instância alterada inicialmente (ElementoInicial) e o modelo de rastreabilidade (MR).

A Tabela 8 apresenta as regras a serem seguidas em cada uma das três fases apresentadas.

Tabela 8 – Regras a serem realizadas em cada fase do processo de rastreabilidade.

PRIMEIRA FASE: AnalisarRastro (ElementoInicial, SRS)
1. Analisar as relações de impacto de 1º nível.
2. Analisar as relações de impacto de 2º nível, partindo dos SubElementos encontrados no passo 1.
3. O passo 2 se repete até que não existam mais impactos.
SEGUNDA FASE: AnalisarRastro (SubElemento, MR)
* Esta fase deve ser realizada para todos os SubElementos resultantes da 1ª FASE *
1. Analisar as relações de impacto de 1º nível.
2. Analisar as relações de impacto de 2º nível, partindo dos SubElementos encontrados no passo 1.
3. O passo 2 se repete até que não existam mais impactos.
TERCEIRA FASE: AnalisarRastro (ElementoInicial, MR)
1. Analisar as relações de impacto de 1º nível.
2. Analisar as relações de impacto de 2º nível, partindo dos SubElementos encontrados no passo 1.
3. O passo 2 se repete até que não existam mais impactos.

As regras apresentadas acima são gerais e compatíveis com todas as classes que compõem o modelo conceitual da SRS e o modelo de rastreabilidade. Seguindo-as, juntamente com os modelos, pode-se chegar aos procedimentos individuais de alteração em cada instância destas classes, apresentados na próxima seção. Nota-se que tais procedimentos foram derivados do modelo seguindo as regras apresentadas na Tabela 8.

Estas regras são facilmente visualizadas nos procedimentos de substituição, pois eles são detalhados passo a passo e divididos conforme as fases apresentadas. Já os procedimentos de exclusão e inclusão não são apresentados por fases, visto que impactam diretamente em uma determinada instância que, por sua vez, impacta por propagação em outras.

6.5.1 Procedimentos individuais de alteração dos elementos

Para este trabalho são consideradas alterações de substituição (S), inclusão (I) e exclusão (E) de instâncias. Os impactos ocasionados dependem do tipo de alteração realizada. Foram consideradas as principais alterações para as instâncias das classes. Outras também podem ser realizadas, porém não trazem impactos consideráveis no restante do processo. Sendo assim, a Tabela 9 mostra as classes cujas instâncias podem ser substituídas, incluídas e/ou excluídas.

Tabela 9 – Classes passíveis de alteração no processo de rastreabilidade.

Substituição (S)	Inclusão (I)	Exclusão (E)
Saida	Saida	Saida
RegraNegGeral	RegraNegGeral	RegraNegGeral
RegraNegEspecifica	RegraNegEspecifica	RegraNegEspecifica
ReqNaoFuncionalGeral	ReqNaoFuncionalGeral	ReqNaoFuncionalGeral
ReqNaoFuncionalEspecifico	ReqNaoFuncionalEspecifico	ReqNaoFuncionalEspecifico
ReqPerformance	ReqPerformance	ReqPerformance
Atributo	Atributo	Atributo
Processamento	Processamento	Processamento
Entrada	Entrada	Entrada
Stakeholder	RespostasSA	RespostasSA
Objetivo	Stakeholder	Stakeholder
	Objetivo	Objetivo

Além das classes apresentadas acima, ainda pode ser incluído um relacionamento entre as instâncias das classes Stakeholder e Objetivo.

Abaixo são apresentados todos os passos para a realização das alterações, mostradas na Tabela 9. Inicialmente serão apresentados os procedimentos de Substituição (S), logo após de Inclusão (I) e por fim os procedimentos de Exclusão (E) para cada elemento.

6.5.1.1 Procedimentos de substituição

Os procedimentos para substituição serão aplicados quando a instância de uma classe for substituída por outra. Dependendo da instância substituída, existirão impactos em outras instâncias da SRS e/ou do MCU. Assim, abaixo são apresentados os procedimentos que representam os impactos ocasionados caso uma substituição seja realizada.

As figuras procuram facilitar a visualização do impacto ocasionado em casa fase quando da substituição de determinada instância. Servem apenas como uma representação visual e não são uma notação formal para as classes.

- Classe: Saida

A Figura 26 apresenta todas as classes cujas instâncias serão impactadas quando uma instância de Saida for substituída na SRS. Já a Tabela 10 mostra os passos detalhados para a alteração destas instâncias, utilizando como referência as regras apresentadas na Tabela 8.

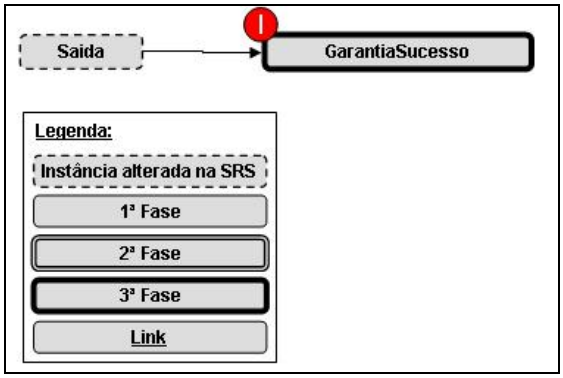


Figura 26 – Classes impactadas quando da substituição de uma instância de Saida.

Tabela 10 – Procedimentos para substituição de uma instância de Saida.

1ª FASE
Não se aplica.
2ª FASE
Não se aplica.
3ª FASE
1. Analisar se a substituição realizada na instância de Saida, de determinado requisito, afeta alguma (s) instância de GarantiaSucesso do caso de uso correspondente a este requisito.

û Classes: RegraNegGeral, RegraNegEspecifica, ReqNaoFuncionalGeral, ReqNaoFuncionalEsp, ReqPerformance, Atributo

As substituições realizadas nas instâncias destas classes impactarão da mesma maneira e na mesma classe, conforme Figura 27. A Tabela 11 mostra os procedimentos detalhados para a substituição nestas instâncias.

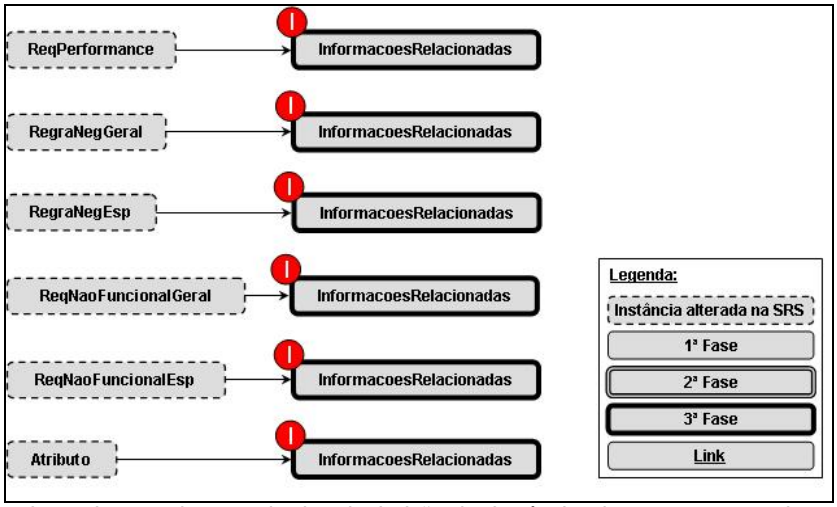


Figura 27 – Classes impactadas quando da substituição das instâncias de RegraNegGeral, RegraNegEsp, ReqNaoFuncionalGeral, ReqNaoFuncionalEsp, ReqPerformance, Atributo.

Tabela 11 – Procedimentos de substituição para as instâncias de: RegraNegGeral, RegraNegEsp, ReqNaoFuncionalGeral, ReqNaoFuncionalEsp, ReqPerformance, Atributo.

1ª FASE
Não se aplica.
2ª FASE
Não se aplica.
3ª FASE
1. Analisar se a substituição em uma instância de RegNegGeral, RegNegEsp, ReqNaoFuncionalGeral, ReqNaoFuncionalEsp, ReqPerformance ou Atributo afeta alguma instância de InformacoesRelacionadas descritas no caso de uso.

û Classe: Processamento

A Figura 28 apresenta as classes cujas instâncias podem ser impactadas quando uma instância de Processamento for substituída na SRS. Já a Tabela 12 mostra os passos detalhados para a substituição realizada nas instâncias desta classe.

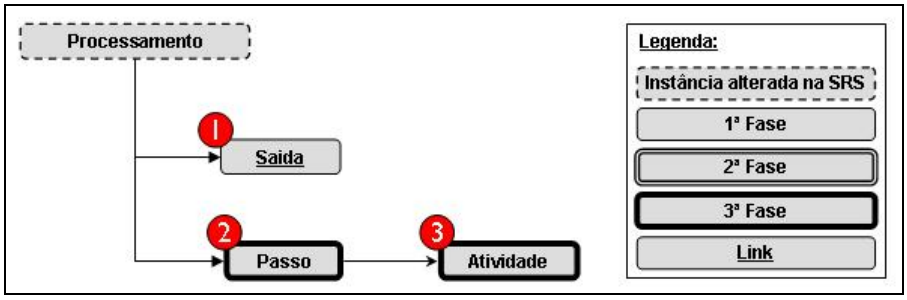


Figura 28 – Classes impactadas quando da substituição de uma instância de Processamento.

Tabela 12 – Procedimentos de substituição para uma instância de Processamento.

1ª FASE
1. Analisar se há instâncias de Saida relacionadas à instância de Processamento alterada e verificar necessidade de alteração. (Ir para procedimentos de substituição da Saida)
2ª FASE
Não se aplica.
3ª FASE
2. Analisar as instâncias da classe Passo a serem alteradas conforme substituição da instância de Processamento.
3. Analisar a necessidade de alteração nas instâncias da classe Atividade, correspondentes às instâncias alteradas de Passo.

û Classe: Entrada

A Figura 29 apresenta todas as classes impactados quando uma instância de Entrada for substituída na SRS. Já a Tabela 13 mostra os passos detalhados para a substituição destas instâncias.

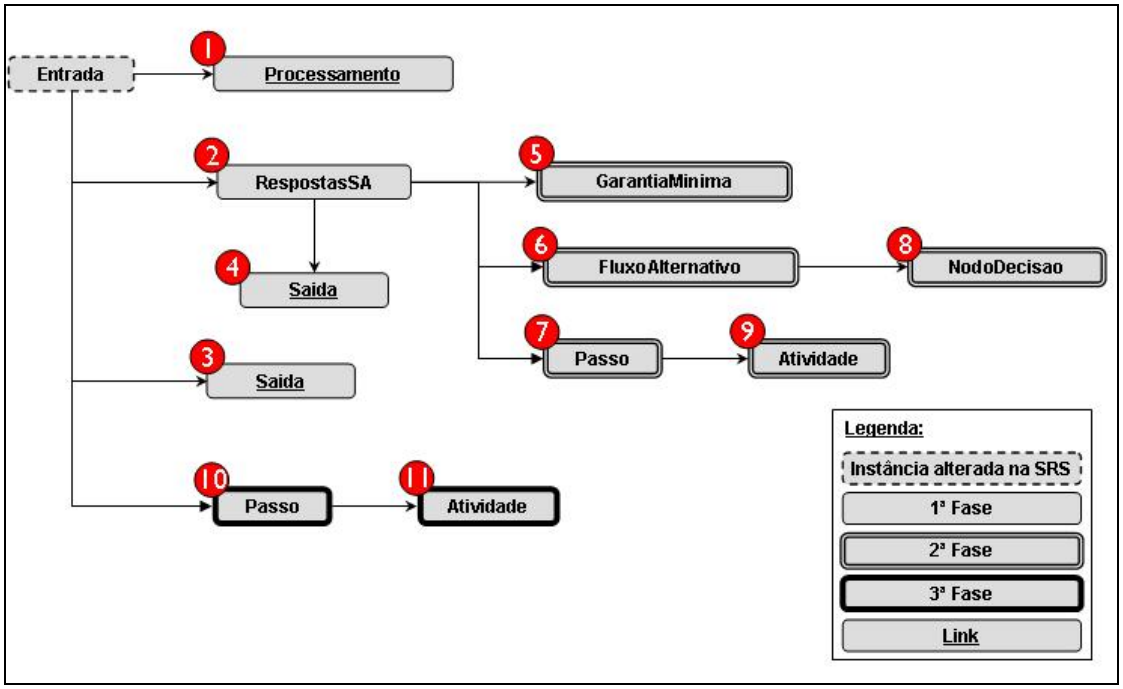


Figura 29 – Classes impactadas quando da substituição de uma instância de Entrada.

Tabela 13 – Procedimentos de substituição para uma instância de Entrada.

1ª FASE
1. Analisar se há necessidade de alteração nas instâncias de Processamento relacionadas à instância substituída de Entrada. (Ir para procedimentos de substituição das instâncias de Processamento)
2. Analisar se há necessidade de alteração nas instâncias de RespostasSA relacionadas à instância substituída de Entrada.
3. Analisar se há necessidade de alteração nas instâncias de Saida relacionadas à instância substituída de Entrada. (Ir para procedimentos de substituição das instâncias de Saida)
4. Analisar se há necessidade de alteração nas instâncias de Saida relacionadas às instâncias substituídas de RespostasSA. (Ir para procedimentos de substituição das instâncias de Saida)
2ª FASE
5. Analisar se há necessidade de alteração nas instâncias de GarantiaMinima relacionadas a instância substituída de RespostasSA.
6. Analisar se há impacto na classe FluxoAlternativo relacionada à instância de RepostasSA substituídas.
7. Analisar as instâncias de Passo a serem alteradas conforme substituição na instância de RespostasSA.
8. Analisar a necessidade de alteração nas instâncias de Atividade, correspondentes a instância substituída de Passo.
9. Analisar se há impacto na classe NodoDecisao correspondente a alteração realizada na classe FluxoAlternativo.
3ª FASE
10. Analisar as instâncias Passo a serem alteradas conforme substituição da instância de Entrada.
11. Analisar a necessidade de alteração nas instâncias de Atividade, correspondentes a alteração realizada na instância de Passo.

û Classe: Stakeholder

A Figura 30 apresenta todas as classes que serão impactadas quando uma instância de Stakeholder for substituída na SRS. Já a Tabela 14 mostra os passos detalhados para esta substituição.

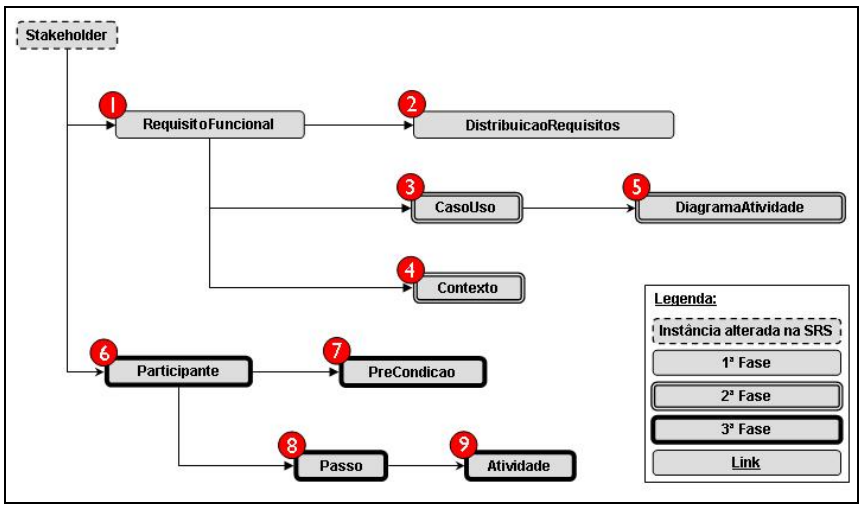


Figura 30 – Classes impactadas quando da substituição do stakeholder.

Tabela 14 – Procedimentos para substituição de uma instância de Stakeholder.

Se for substituída a instância de stakeholder relacionada a um objetivo específico:
1ª FASE
1. Todas as instâncias relacionadas de RequisitoFuncional serão automaticamente modificadas.
2. Analisar se as instâncias de RequisitoFuncional substituídas estão relacionadas na classe DistribuicaoRequisitos.
Os passos seguintes devem ser realizados para cada instância substituída de RequisitoFuncional
2ª FASE
3. Identificar a instância de CasoUso que corresponde a instância substituída de RequisitoFuncional.
4. Atualizar a instância de Contexto, de acordo com a descrição alterada de RequisitoFuncional.
5. Identificar a instância de DiagramaAtividade correspondente a instância alterada de CasoUso.
3ª FASE
6. Atualizar a instância de Participante, do caso de uso, de acordo com a instância de Stakeholder substituída no requisito funcional.
7. Analisar se a instância de PreCondicao do caso de uso foi impactada pela substituição realizada na instância Participante.
8. Analisar as instâncias de Passo a serem alteradas conforme substituição realizada na instância de Participante.
9. Analisar a necessidade de alteração nas instâncias da classe Atividade, correspondentes as instâncias alteradas Passo.
** Se for substituída a instância de Stakeholder relacionada ao contexto geral do sistema, ou seja, para todos os objetivos **
1. Os procedimentos acima deverão ser executados para cada objetivo impactado pela substituição da instância de Stakeholder.

û Classe: Objetivo

A Figura 31 apresenta todas as classes que serão impactadas quando uma instância de Objetivo for substituída na SRS. A Tabela 15 mostra os passos detalhados para a substituição destas instâncias.

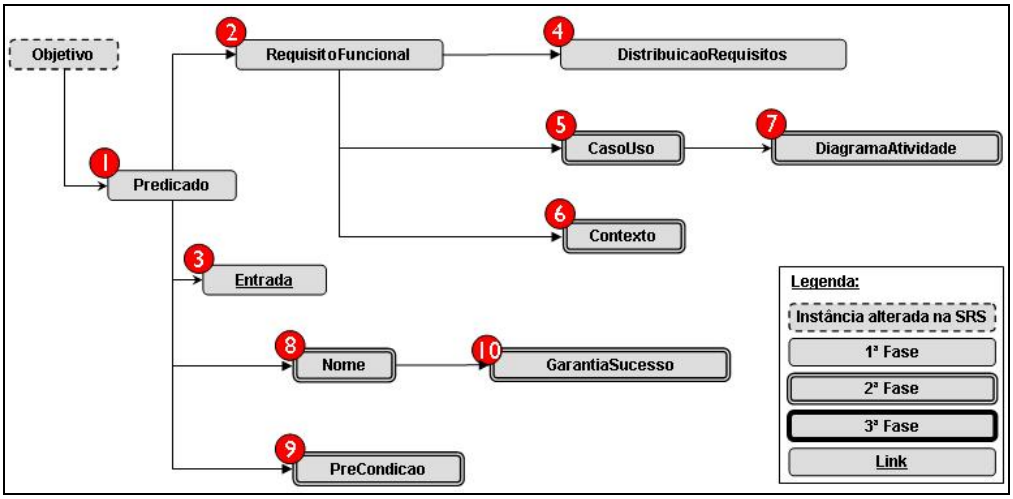


Figura 31 – Classes impactadas quando da substituição de uma instância de Objetivo.

Tabela 15 – Regras para substituição de uma instância de Objetivo.

1ª FASE
1. Todas as instâncias de Predicado relacionadas a instância substituía de Objetivo devem ser revisadas.
2. Todas as instâncias de RequisitoFuncional, relacionadas a instância substituída de Predicado, serão atualizadas.
3. Analisar todas as instâncias de Entrada relacionadas a instância substituída de Predicado e verificar necessidade de alteração. (Ir para procedimentos de substituição de Entrada).
4. Analisar se as instâncias alteradas de RequisitoFuncional estão relacionadas na classe DistribuicaoRequisitos.
Os passos seguintes devem ser realizados para cada instância alterada de requisito funcional (RequisitoFuncional)
2ª FASE
5. Identificar a instância de CasoUso que corresponde a instância alterada de RequisitoFuncional.
6. Atualizar a instância de Contexto, de acordo com a descrição da instância alterada de RequisitoFuncional.
7. Identificar a instância de DiagramaAtividade correspondente ao CasoUso identificado.
8. Alterar a instância de Nome, do caso de uso, conforme substituição realizada na instância de Predicado.
9. Analisar a necessidade de alteração nas instâncias de PreCondicao, correspondentes a instância alterada de Predicado.
10. Analisar a necessidade de alteração nas instâncias de GarantiaSucesso, correspondentes a instância alterada de Nome, do caso de uso.
3ª FASE
Não se aplica.

6.5.1.2 Procedimentos de exclusão

Os procedimentos para exclusão serão aplicados quando a instância de uma classe for excluída. Dependendo da instância excluída, existirão impactos em outras instâncias da SRS e/ou do MCU. Assim, abaixo são apresentados os procedimentos que representam os impactos ocasionados caso uma exclusão seja realizada. Para facilitar sua visualização, são utilizados os diagramas de atividade da UML [UML05].

- Classe: Saida

O diagrama de atividades, apresentado na Figura 32 demonstra que quando houver a exclusão de uma instância da classe Saida, suas instâncias relacionadas de GarantiaSucesso e de Passo também devem ser excluídas.

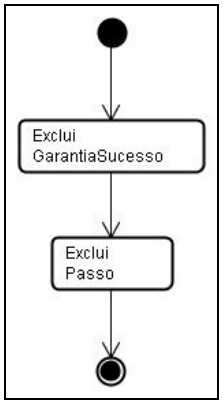


Figura 32 – Diagrama de atividades: exclusão da instância de Saida.

- Classe: RegraNegGeral, RegraNegEspecifica, ReqNaoFuncionalGeral, ReqNaoFuncionalEsp, ReqPerformance, Atributo

A Figura 33 apresenta o diagrama de atividades que demonstra os impactos ocasionados no momento da exclusão de uma instância de algumas destas classes. Ou seja, caso uma delas seja excluída, sua instância relacionada de InformacoesRelacionadas também deve ser excluída.

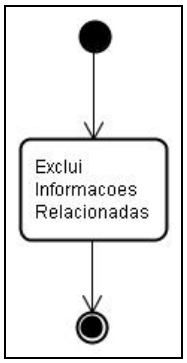


Figura 33 – Diagrama de atividades: exclusão de uma instância de RegraNegGeral, RegraNegEspecifica, ReqNaoFuncionalGeral, ReqNaoFuncionalEsp, ReqPerformance, Atributo.

û Classe: Processamento

A Figura 34 apresenta o diagrama de atividades que demonstra os impactos ocasionados no momento da exclusão de uma instância de Processamento.

Quando uma instância desta classe for excluída, sua instância relacionada de Saida também deve ser excluída. Nota-se, na Figura 34, que a atividade Exclui Saida possui um sub-diagrama, o que indica que os procedimentos desta alteração também devem ser considerados. Logo após, a instância de Passo, relacionada à instância excluída de Processamento, deve ser excluída e com ela sua instância relacionada de Atividade.

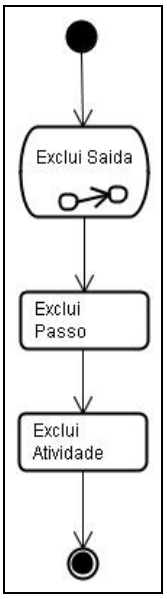


Figura 34 – Diagrama de atividades: exclusão da instância de Processamento.

û Classe: RespostasSA

A Figura 35 apresenta o diagrama de atividades que demonstra os impactos ocasionados no momento da exclusão de uma instância de RespostasSA.

Quando uma instância desta classe for excluída, suas instâncias relacionadas de GarantiaMinima e de Saida devem ser excluídas, bem como todas as instâncias impactadas por estas exclusões. Logo após, a instância de Passo, relacionada à instância excluída de Saida, deve ser excluída e com ela sua instância relacionada de Atividade.

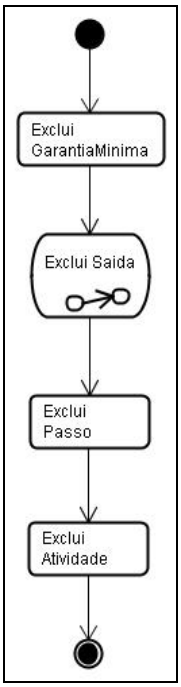


Figura 35 – Diagrama de atividades: exclusão da instância de RespostasSA.

û Classe: Entrada

A Figura 36 apresenta o diagrama de atividades que demonstra os impactos ocasionados no momento da exclusão de uma instância de Entrada.

Quando uma instância desta classe for excluída, devem ser excluídas suas instâncias relacionadas de Processamento, RespostasSA e Saida, bem como as instâncias impactadas pela realização destas exclusões. Em seguida, devem ser excluídas as instâncias de Passo e de Atividade, relacionadas à instância excluída de Entrada.

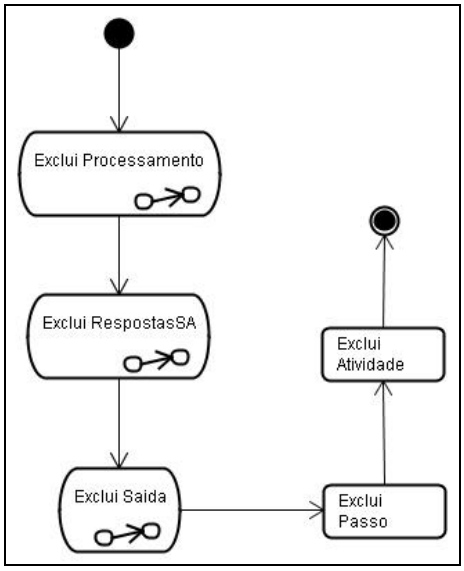


Figura 36 – Diagrama de atividades: exclusão da instância de Entrada.

û Classe: Stakeholder

A Figura 37 apresenta o diagrama de atividades que demonstra os impactos ocasionados no momento da exclusão de uma instância de Stakeholder.

Quando houver a necessidade de excluir totalmente uma instância desta classe, deverão ser excluídos todos os relacionamentos dos quais ela participa, onde relacionamentos são as ligações entre as instâncias de Stakeholder e as instâncias de Objetivo.

Caso a opção seja excluir uma instância de Stakeholder do relacionamento com uma instância de um Objetivo específico, ou seja, fazer com que este stakeholder não tenha mais interesse por determinado objetivo, apenas o relacionamento em questão deve ser excluído. A exclusão de relacionamentos é mostrada no decorrer desta seção.

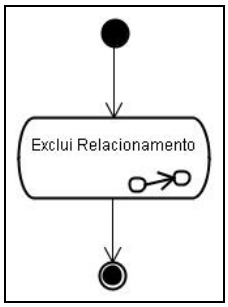


Figura 37 – Diagrama de atividades: exclusão da instância de Stakeholder.

û Classe: Objetivo

A Figura 38 apresenta o diagrama de atividades que demonstra os impactos ocasionados no momento da exclusão de uma instância de Objetivo.

O procedimento é o mesmo mostrado para a classe anterior. Caso se deseje excluir totalmente uma instância de Objetivo, todos os seus relacionamentos com as instâncias de Stakeholder devem ser excluídos.

Caso a opção seja apenas a exclusão do relacionamento de uma instância de Objetivo com uma instância de Stakeholder, ou seja, fazer com que um objetivo não faça mais parte dos interesses de determinado stakeholder, apenas o relacionamento em questão deve ser excluído.

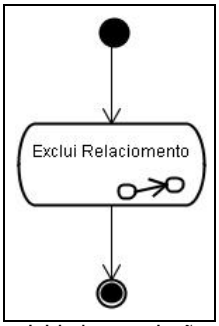


Figura 38 – Diagrama de atividades: exclusão da instância de Objetivo.

û Exclusão de um relacionamento

A Figura 38 apresenta o diagrama de atividades que demonstra os impactos ocasionados no momento da exclusão de um relacionamento entre uma instância de Stakeholder e uma instância de Objetivo.

Como explicado anteriormente, a exclusão de um relacionamento se dá quando há a necessidade de excluir, apenas, a participação da instância de uma classe do relacionamento com a instância da outra classe. Entretanto, neste caso há vários pontos a serem considerados.

As instâncias de Stakeholder fazem o papel tanto de ator principal do requisito funcional (aquele que interage com o sistema) como de ator interessado no requisito funcional (aquele que não interage com o sistema).

Sendo assim, no momento da exclusão de um relacionamento, inicialmente deve ser analisado se a instância de Stakeholder participante interage com o sistema, chamado doravante de stakeholder ator, ou de ator que não interage com o sistema, chamado doravante de stakeholder interessado.

Se for um stakeholder ator, devem ser excluídas as instâncias relacionadas de RequisitoFuncional, CasoUso e DiagramaAtividade. Se for um stakeholder interessado, devem ser identificadas as instâncias relacionadas de RequisitoFuncional e CasoUso para, depois, ser excluída a instância de Stakeholder participante deste caso de uso.

Em seguida deve ser analisado se o stakeholder possui ou não interesse em outros objetivos, ou seja, outros relacionamentos com instâncias de Objetivo. Caso não possua, sua instância deve ser excluída. Se possuir, nenhuma atividade é realizada.

O próximo passo é verificar se o objetivo possui outros stakeholders interessados. Caso não possua, sua instância é excluída. Se possuir, e estes forem apenas stakeholders interessados, é executada a atividade que chama o diagrama de atividades Exclui Objetivo e, assim, suas atividades são realizadas.

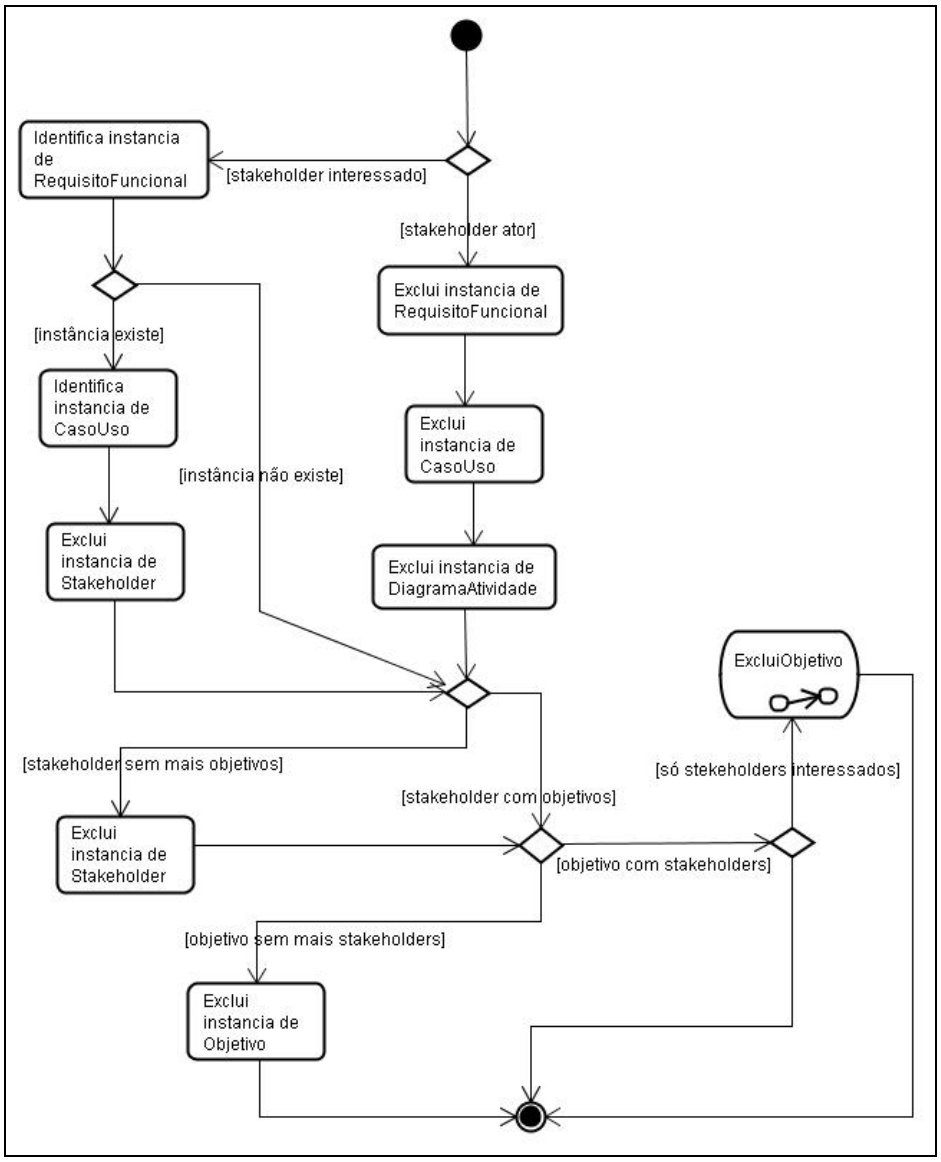
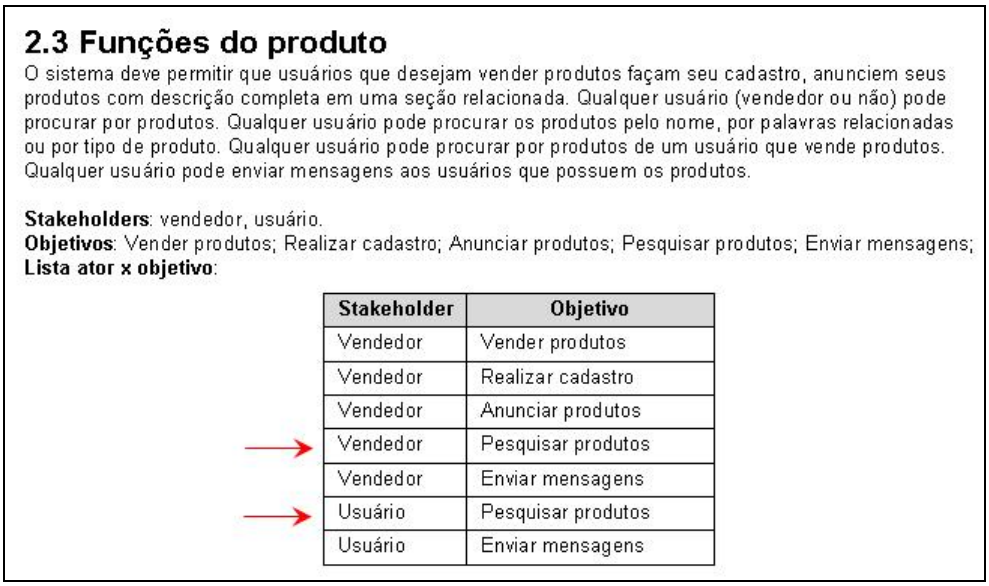


Figura 39 – Diagrama de atividades: exclusão de um relacionamento.

Por exemplo, a Figura 40 mostra um trecho da seção Funções do Produto da SRS. O objetivo Pesquisar produtos tem um relacionamento com o stakeholder ator "Vendedor" e com o stakeholder interessado "Usuário". Supondo que o primeiro relacionamento seja excluído (Vendedor – Pesquisar produtos).



2.3 Funções do produto

O sistema deve permitir que usuários que desejam vender produtos façam seu cadastro, anunciem seus produtos com descrição completa em uma seção relacionada. Qualquer usuário (vendedor ou não) pode procurar por produtos. Qualquer usuário pode procurar os produtos pelo nome, por palavras relacionadas ou por tipo de produto. Qualquer usuário pode procurar por produtos de um usuário que vende produtos. Qualquer usuário pode enviar mensagens aos usuários que possuem os produtos.

Stakeholders: vendedor, usuário.
Objetivos: Vender produtos; Realizar cadastro; Anunciar produtos; Pesquisar produtos; Enviar mensagens;
Lista ator x objetivo:

Stakeholder	Objetivo
Vendedor	Vender produtos
Vendedor	Realizar cadastro
Vendedor	Anunciar produtos
Vendedor	Pesquisar produtos
Vendedor	Enviar mensagens
Usuário	Pesquisar produtos
Usuário	Enviar mensagens

Figura 40 – Trecho da seção Funções do Produto.

Seguindo o diagrama de atividades, a primeira verificação indica que o stakeholder representa o stakeholder ator, então devem ser excluídas suas instâncias relacionadas de RequisitoFuncional, CasoUso e DiagramaAtividade.

Em seguida verifica-se, na mesma figura, que o stakeholder ainda tem interesse em outros objetivos, então nenhuma atividade é realizada. O próximo passo indica que o objetivo ainda tem o interesse de outros stakeholders (nota-se na Figura 40 que ainda há o interesse do ator Usuário).

Como o ator Usuário não é um stakeholder ator, e não há nenhum um outro relacionado, é realizada a atividade que contém o sub-diagrama Exclui Objetivo.

O diagrama Exclui Objetivo indica, conforme Figura 38, a exclusão do relacionamento (neste caso o relacionamento Usuário – Pesquisar Produtos).

Assim, no primeiro passo comprova-se que o stakeholder é um stakeholder interessado e, com isso, verifica-se a inexistência da instância de RequisitoFuncional. Este passo é realizado pois se o relacionamento principal do objetivo já foi excluído, como é o caso, sua instância relacionada de RequisitoFuncional também já terá sido excluída. Logo após verifica-se que o ator ainda possui objetivos Figura 40). Logo, como o objetivo não tem mais nenhum ator interessado, sua instância é excluída e o diagrama é finalizado.

6.5.1.3 Procedimentos de inclusão

Os procedimentos para inclusão serão aplicados quando a instância de uma classe for incluída. Dependendo da instância incluída, existirão impactos em outras instâncias da SRS e/ou do MCU. Assim, abaixo são apresentados os procedimentos que representam os impactos ocasionados com esta alteração. Para facilitar sua visualização, os diagramas de atividade da UML [UML05] foram

utilizados e serão explicados conforme forem sendo apresentados.

û Classe: Saida

O diagrama de atividades, apresentado na Figura 41 demonstra que quando houver a inclusão de uma instância da classe Saida, suas instâncias relacionadas de GarantiaSucesso e de Passo também devem ser incluídas.

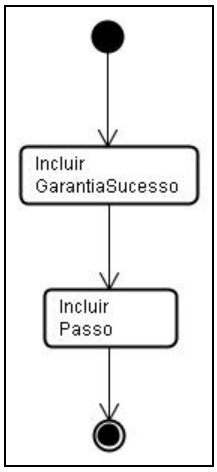


Figura 41 – Diagrama de atividades: inclusão da instância de Saida.

û Classe: RegraNegGeral, RegraNegEspecifica, ReqNaoFuncionalGeral, ReqNaoFuncionalEsp, ReqPerformance, Atributo

A Figura 42 apresenta o diagrama de atividades que demonstra os impactos ocasionados no momento da inclusão de uma instância de alguma destas classes. Ou seja, caso uma delas seja incluída, sua instância relacionada de InformacoesRelacionadas também deve ser excluída.

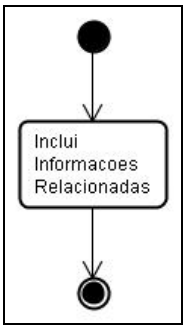


Figura 42 – Diagrama de atividades: inclusão de uma instância de RegraNegGeral, RegraNegEspecifica, ReqNaoFuncionalGeral, ReqNaoFuncionalEsp, ReqPerformance, Atributo.

û Classe: Processamento

A Figura 43 apresenta o diagrama de atividades que demonstra os impactos ocasionados no momento da inclusão de uma instância de Processamento.

Quando uma instância desta classe for incluída, sua instância relacionada de Passo também deve ser incluída.

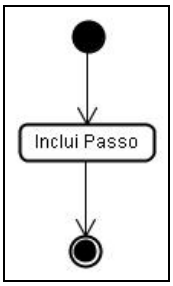


Figura 43 – Diagrama de atividades: inclusão da instância de Processamento.

û Classe: RespostasSA

A Figura 44 apresenta o diagrama de atividades que demonstra os impactos ocasionados no momento da inclusão de uma instância de RespostasSA.

Quando uma instância desta classe for excluída, suas instâncias relacionadas de Passo e de GarantiaMinima devem ser incluídas.

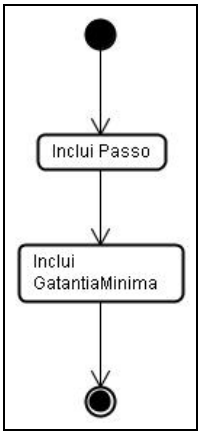


Figura 44 – Diagrama de atividades: inclusão da instância de RespostasSA.

û Classe: Entrada

A Figura 45 apresenta o diagrama de atividades que demonstra os impactos ocasionados no momento da inclusão de uma instância de Entrada.

Quando uma instância desta classe for incluída, sua instância relacionada de Passo, também deve ser incluída.

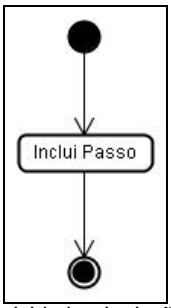


Figura 45 – Diagrama de atividades: inclusão da instância de Entrada.

û Classe: Stakeholder

A Figura 46 apresenta o diagrama de atividades que demonstra os impactos ocasionados no momento da inclusão de uma instância de Stakeholder, considerando que para que um stakeholder possa ser incluído, ele deve estar relacionado a um objetivo.

Caso a instância incluída seja de um stakeholder interessado em um objetivo, devem ser identificadas as instâncias relacionadas de RequisitoFuncional e CasoUso e, logo após, a instância relacionada de Stakeholder deve ser incluída no caso de uso em questão.

Caso a instância incluída seja de um stakeholder ator, devem ser incluídas suas instâncias relacionadas de RequisitoFuncional, CasoUso e DiagramaAtividade.

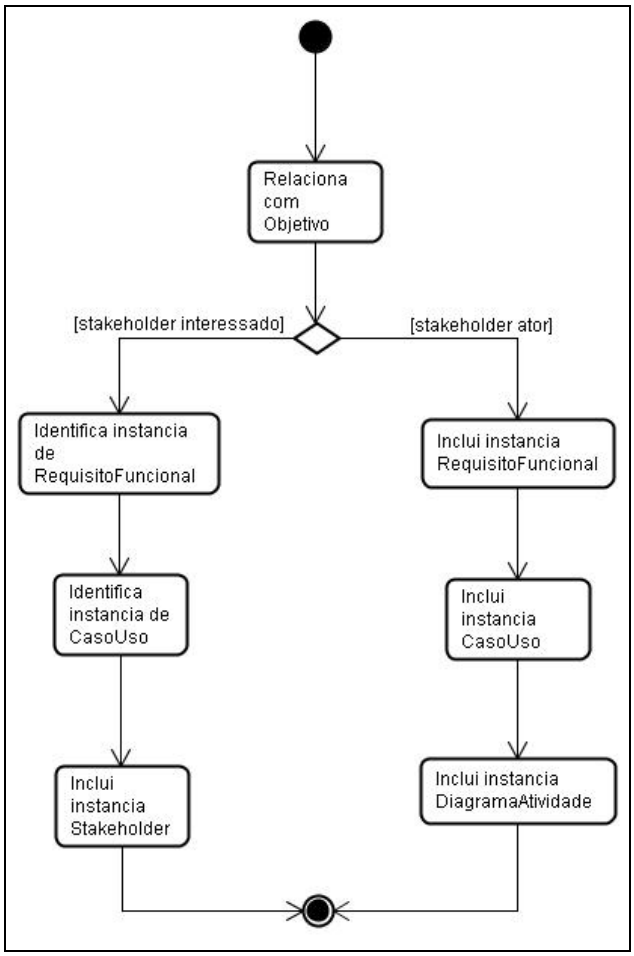


Figura 46 – Diagrama de atividades: inclusão da instância de Stakeholder.

û Classe: Objetivo

A Figura 47 apresenta o diagrama de atividades que demonstra os impactos ocasionados no momento da inclusão de uma instância de Objetivo, que pode ser incluída individualmente, sem a necessidade de um relacionamento com uma instância de Stakeholder.

Neste procedimento, considera-se que quando um objetivo for relacionado pela primeira vez com um stakeholder, este deve ser um stakeholder ator. Não é possível inserir um stakeholder interessado para um objetivo que não tenha sido relacionado anteriormente com um stakeholder ator.

Caso a instância incluída tenha um stakeholder interessado, o que significa que este objetivo já tem outro stakeholder ator relacionado, é incluído o relacionamento com a instância de Stakeholder. Em seguida, suas instâncias de RequisitoFuncional, CasoUso e DiagramaAtividade.

Se a instância não for relacionada com nenhuma instância de Stakeholder, ela é apenas incluída.

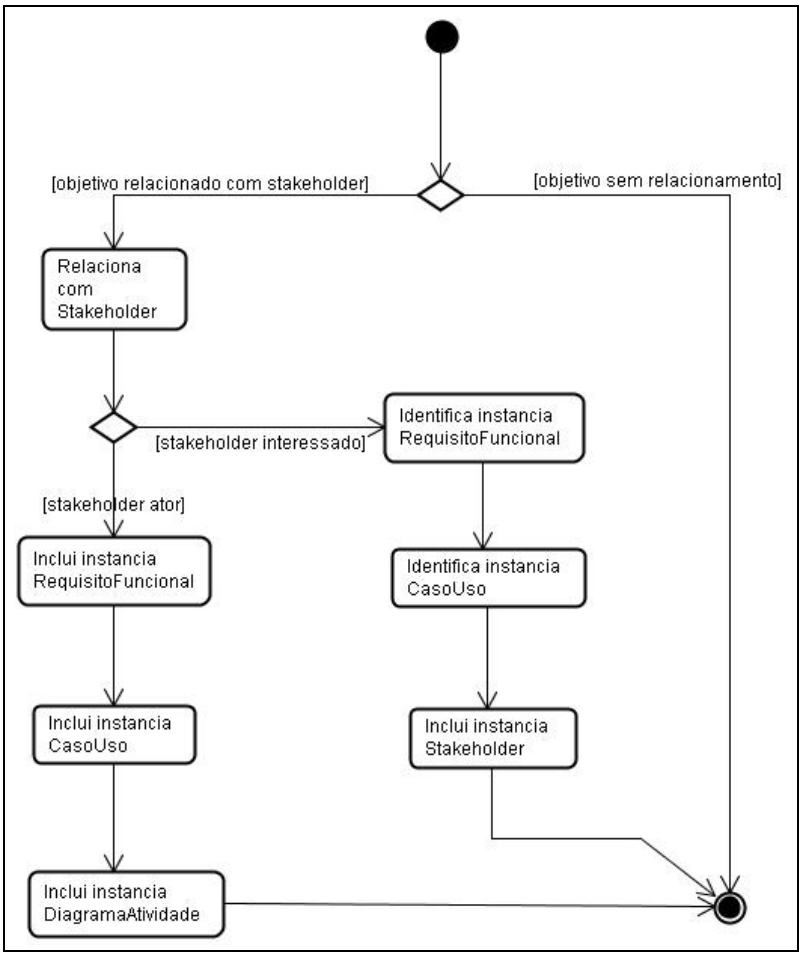


Figura 47 – Diagrama de atividades: inclusão da instância de Objetivo.

6.6 Considerações finais

Este capítulo apresentou, com detalhes, o modelo e o processo de rastreabilidade propostos neste trabalho. Foram apresentados, ainda, os modelos conceituais da SRS e do MCU utilizados em conjunto com o modelo de rastreabilidade.

O próximo capítulo apresenta a avaliação do modelo proposto, através de um exemplo utilizando uma SRS desenvolvida por um grupo de alunos da Universidade de Passo Fundo.

7 AVALIAÇÃO DO MODELO PROPOSTO

Para avaliar a consistência e a viabilidade do modelo e do processo propostos neste trabalho, será utilizado um cenário exemplo criado a partir de uma documentação desenvolvida pelos alunos da disciplina de Engenharia de Software do Curso de Ciência da Computação da Universidade de Passo Fundo – UPF. O cenário consiste da utilização de um documento de especificação de requisitos (SRS) desenvolvido com base no modelo proposto pela IEEE [IEE98] e de um modelo de casos de uso composto dos casos de uso correspondentes aos requisitos da SRS e de suas especificações.

Cada grupo da disciplina desenvolveu uma SRS, sendo que cada um deles trabalhou com dois requisitos e com o modelo de caso de uso correspondente. Aleatoriamente a SRS de um dos grupos foi escolhida para a aplicação do modelo proposto. O Apêndice III apresenta a documentação utilizada nesta avaliação. Esta documentação é escrita fielmente para manter a fidelidade do resultado.

A fim de demonstrar seu funcionamento, serão apresentados três exemplos da aplicação da proposta sendo uma alteração de substituição, uma alteração de exclusão e uma alteração de inclusão.

Nas seções seguintes, a abordagem utilizada para a aplicação do modelo é descrita em maiores detalhes, assim como seu processo de condução e os resultados encontrados.

7.1 Abordagem utilizada

A SRS utilizada para a aplicação do modelo envolve os requisitos e seu respectivo modelo de casos de uso, descritos no Apêndice III.

Embora os requisitos da SRS em questão não estejam escritos utilizando o padrão proposto nesta dissertação, a aplicação do modelo ainda é possível, visto que o documento possui grande parte dos campos sugeridos para que seja viável alcançar um resultado eficiente utilizando o modelo e processo de rastreabilidade propostos.

No momento em que os exemplos forem sendo apresentados, será mostrada a maneira correta, de acordo com o padrão proposto, para a escrita do requisito. A versão adaptada, conforme os padrões propostos nesta dissertação, pode ser conferida no Apêndice IV.

A avaliação é composta das seguintes alterações:

- Alteração 1: Substituição do ator “Vendedor” pelo ator “Administrador” com relação ao objetivo “Anunciar produtos”
- Alteração 2: Exclusão do ator “Vendedor” referente ao objetivo “Pesquisar Produtos”.
- Alteração 3: Inclusão do objetivo “Anunciar produtos” para o stakeholder interessado “Administrador”.

7.2 Processo de condução

O processo de avaliação foi realizado através das alterações indicadas na seção anterior e detalhadas a seguir.

As alterações demonstradas nos exemplos não são acumulativas, ou seja, a realização de todas as alterações serão efetuadas sob o documento inicial e não sob o documento modificado pelas alterações anteriores.

Como neste exemplo têm-se somente os requisitos relacionados a dois objetivos, os procedimentos apresentados são relacionados apenas a eles.

7.2.1 Alteração 1: Substituição do ator “Vendedor” pelo ator “Administrador” com relação ao objetivo “Anunciar produtos”

No documento em questão, o “ator” Vendedor representa o “stakeholder ator” proposto nos procedimentos deste trabalho. Neste exemplo, este stakeholder deverá ser substituído pelo Administrador. Nota-se que esta substituição afetará apenas o objetivo Anunciar produtos.

Para esta substituição serão utilizados os procedimentos para substituição de instâncias da classe Stakeholder, apresentados na seção 6.5.

De acordo com a Tabela 14, para a primeira fase tem-se os seguintes procedimentos:

- Procedimento 1: Todas as instâncias relacionadas de RequisitoFuncional serão modificadas.

Como a substituição do stakeholder é apenas para o objetivo Anunciar produtos o requisito funcional relacionado Anúncio de produtos será o alterado, conforme mostrado na Figura 48.

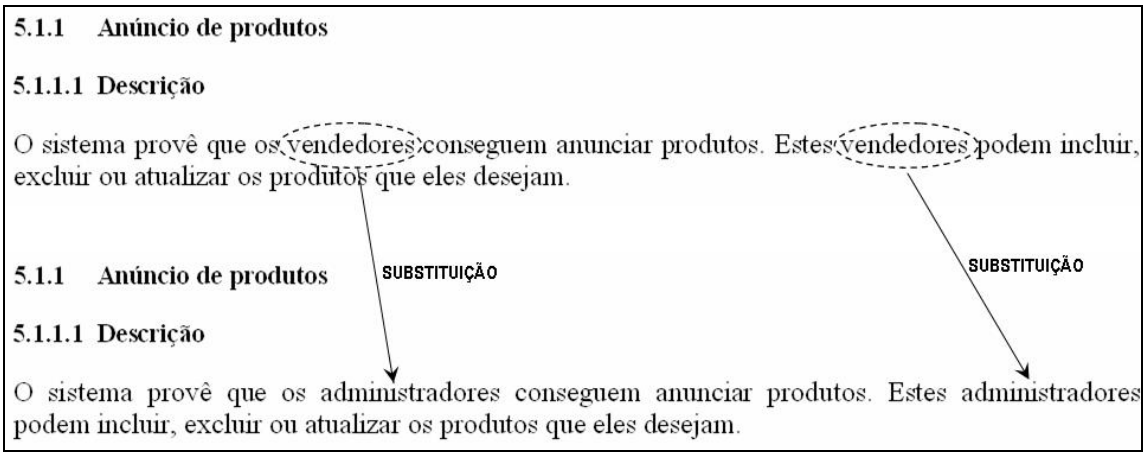


Figura 48 – Substituição de stakeholder.

A escrita do requisito fora do padrão estipulado dificultaria na execução automatizada deste procedimento. No protótipo, descrito no Apêndice VI, a entrada do requisito funcional é padronizada.

Para estar dentro do padrão o requisito deveria ser escrito da maneira mostrada na Figura 49.

Figura 49 – Padrão utilizado para a escrita do Requisito Funcional.

û Procedimento 2: Analisar se as instâncias de RequisitoFuncional substituídas estão relacionadas na classe DistribuicaoRequisitos.

Nota-se pelo documento que a instância alterada não está relacionada nesta classe.

De acordo com a Tabela 14, para a segunda fase tem-se os seguintes procedimentos:

û Procedimento 3: Identificar a instância de CasoUso que corresponde à instância substituída de RequisitoFuncional.

Conforme o Apêndice III, o caso de uso em questão é o “Anúncio de produtos”, como mostra Figura 50

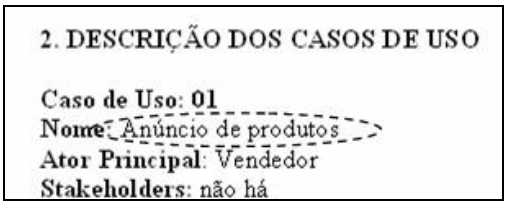


Figura 50 – Caso de uso relacionado ao requisito funcional Anúncio de produtos.

û Procedimento 4: Atualizar a instância de Contexto de acordo com a descrição alterada de requisito funcional.

Como pode ser percebido no Apêndice III, o campo Contexto não é utilizado. Sendo assim, essa alteração não se aplica.

Se o documento estivesse no padrão proposto, o campo Contexto conteria a descrição do requisito funcional relacionado, apresentado na SRS.

û Procedimento 5: Identificar a instância de DiagramaAtividade correspondente à instância alterada de CasoUso.

Da mesma forma que o procedimento anterior, este não se aplica, pois o documento não faz uso do Diagrama de Atividades.

De acordo com a Tabela 14, para a terceira fase tem-se os seguintes procedimentos:

û Procedimento 6: Atualizar a instância de Participante do caso de uso de acordo com a instância de Ator substituída no requisito funcional.

Na instância de RequisitoFuncional foi substituído o ator[7] Vendedor pelo ator Administrador.

Assim, o participante do caso de uso será também substituído, como mostra Figura 51.

û Procedimento 7: Analisar se a instância de PreCondicao do caso de uso foi impactada pela substituição realizada na instância de Participante.

A pré-condição também referencia o ator principal do caso de uso (no caso o participante), sendo assim, a Figura 51 mostra esta alteração.

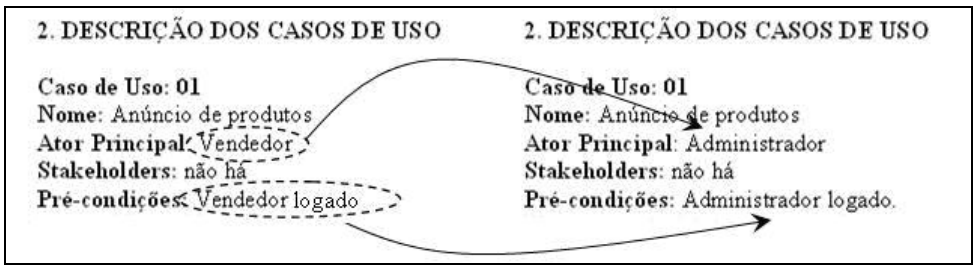


Figura 51 – Substituição do participante e da pré-condição do caso de uso Anúncio de Produtos.

[7] Neste caso fala-se ator ao invés de stakeholder pois o requisito funcional é uma composição de ator e predicado, onde o ator é uma especialização de stakeholder.

û Procedimento 8: Analisar as instâncias de Passo a serem alteradas conforme substituição realizada na instância de Participante.

Este procedimento indica que todos os passos que tenham o Vendedor como participante deverão ser alterados, conforme Figura 52.

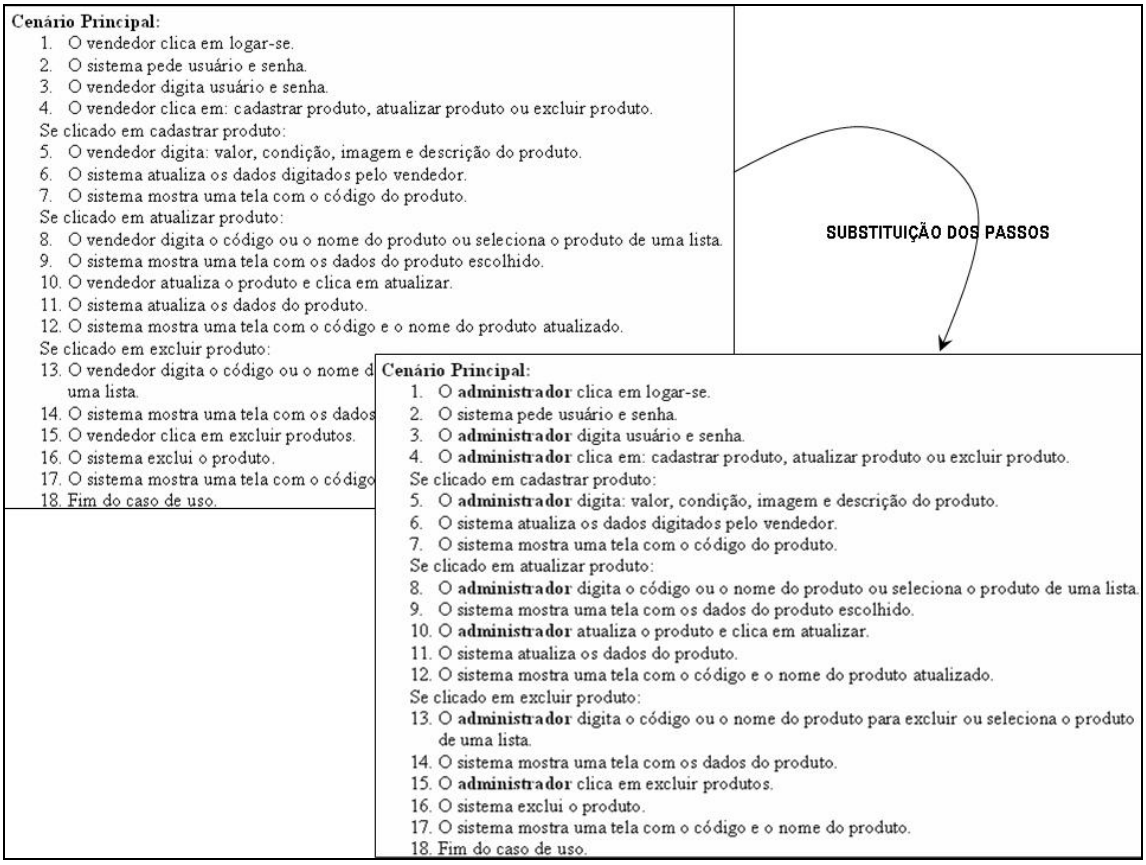


Figura 52 – Alteração dos passos conforme substituição do participante.

û Procedimento 9: Analisar a necessidade de alteração nas instâncias da classe Atividade, correspondentes às instâncias alteradas de Passo.

Como o documento não faz uso do diagrama de atividades, este procedimento não se aplica.

Com a realização deste procedimento, a alteração é finalizada.

7.2.2 Alteração 2: Exclusão do ator "Vendedor" referente ao objetivo "Pesquisar produtos"

Nesta alteração, o ator Vendedor será excluído apenas para o objetivo Pesquisar produtos. Ou seja, o ator Vendedor não terá mais interesse no objetivo Pesquisar produtos.

Para esta exclusão serão utilizados os procedimentos para exclusão de instâncias da classe Stakeholder, apresentados na seção 6.5.

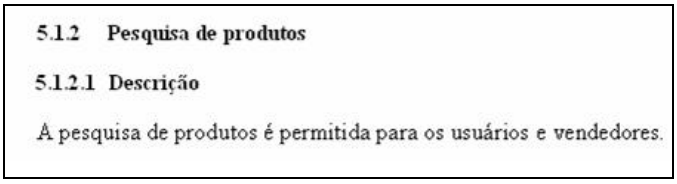
De acordo com a Figura 37, a atividade a ser realizada é a exclusão do relacionamento. Ou seja, o relacionamento entre o Vendedor e o objetivo Pesquisar produtos será excluído. Ainda na Figura 37, nota-se que esta atividade contém um diagrama de atividades relacionado, apresentado na Figura 39, que corresponde às atividades a serem executadas para a exclusão deste relacionamento.

- Procedimento 1: Verificar se o stakeholder que compõe o relacionamento a ser excluído é um stakeholder interessado ou um stakeholder ator.

Pela SRS pode-se notar que para este objetivo o Vendedor faz o papel de stakeholder ator. Assim, a atividade a ser executada é: Exclui instância de RequisitoFuncional.

- Procedimento 2: Excluir instância de RequisitoFuncional.

A instância de RequisitoFuncional relacionada ao objetivo Pesquisar produtos é Pesquisa de produtos, como mostrado na Figura 53.


5.1.2 **Pesquisa de produtos**

5.1.2.1 **Descrição**

A pesquisa de produtos é permitida para os usuários e vendedores.

Figura 53 – Exclusão do requisito funcional Pesquisa de produtos.

A exclusão deste requisito implica na exclusão por propagação de todas as instâncias das classes a ele relacionadas. Ou seja, serão excluídas todas as interfaces de entrada, processamento, interfaces de saída etc.

- Procedimento 3: Excluir instância de CasoUso.

A instância de CasoUso relacionada à instância excluída de RequisitoFuncional deverá ser excluída. Conforme o documento, o caso de uso a ser excluído é Pesquisa de produtos, como mostra Figura 54.

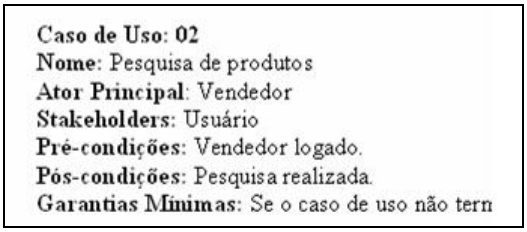

Caso de Uso: 02
Nome: Pesquisa de produtos
Ator Principal: Vendedor
Stakeholders: Usuário
Pré-condições: Vendedor logado.
Pós-condições: Pesquisa realizada.
Garantias Mínimas: Se o caso de uso não tern

Figura 54 – Exclusão do caso de uso Pesquisa de produtos.

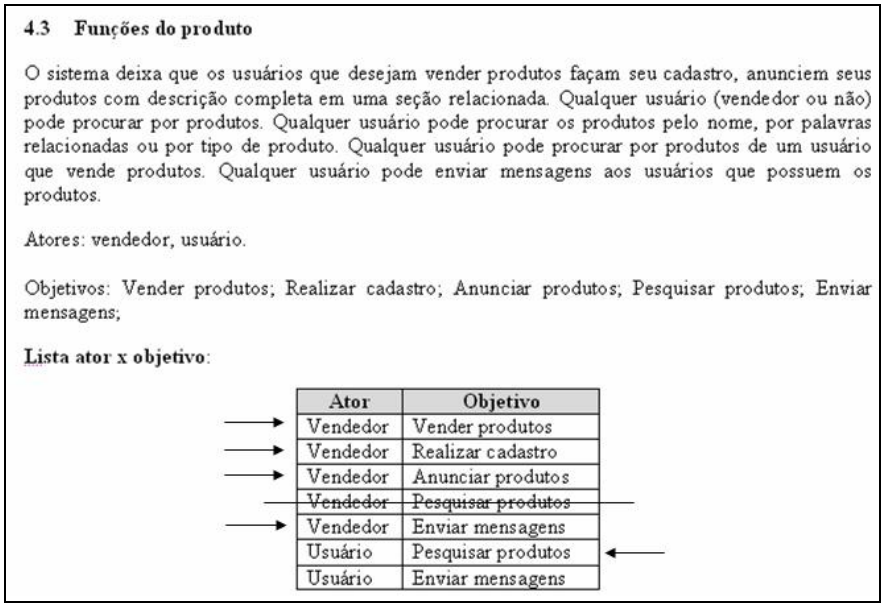
Nota-se que neste caso também há a exclusão por propagação de todas as instâncias relacionadas a este caso de uso. Por exemplo, serão excluídas junto com ele suas pré-condições, pós-condições e etc.

û Procedimento 4: Excluir instância de DiagramaAtividade

A instância de DiagramaAtividade relacionada à instância excluída de CasoUso deverá ser excluída. Como o documento não faz uso deste tipo de diagrama, este procedimento não se aplica.

û Procedimento 5: Verificar se o stakeholder interessado do relacionamento ainda possui interesse em outros objetivos.

Pela tabela apresentada na Figura 55, retirada do no Apêndice III, o stakeholder em questão ainda possui relacionamentos com outros objetivos, conforme setas do lado esquerdo da lista.



4.3 Funções do produto

O sistema deixa que os usuários que desejam vender produtos façam seu cadastro, anunciem seus produtos com descrição completa em uma seção relacionada. Qualquer usuário (vendedor ou não) pode procurar por produtos. Qualquer usuário pode procurar os produtos pelo nome, por palavras relacionadas ou por tipo de produto. Qualquer usuário pode procurar por produtos de um usuário que vende produtos. Qualquer usuário pode enviar mensagens aos usuários que possuem os produtos.

Atores: vendedor, usuário.

Objetivos: Vender produtos; Realizar cadastro; Anunciar produtos; Pesquisar produtos; Enviar mensagens;

Lista ator x objetivo:

Ator	Objetivo
Vendedor	Vender produtos
Vendedor	Realizar cadastro
Vendedor	Anunciar produtos
Vendedor	Pesquisar produtos
Vendedor	Enviar mensagens
Usuário	Pesquisar produtos
Usuário	Enviar mensagens

Figura 55 – Lista de ator x objetivo.

û Procedimento 6: Verificar se o stakeholder participante do relacionamento ainda possui interesse em outros objetivos.

Neste passo deve-se verificar se o objetivo em questão ainda possui relacionamentos com outros stakeholders, o que se confirma também na Figura 55, pela seta mostrada no lado direito da lista.

Com a realização deste procedimento, a alteração é finalizada.

7.2.3 Alteração 3: Inclusão do objetivo "Anunciar produtos" para o stakeholder interessado "Administrador"

Nesta etapa o objetivo Anunciar produtos será inserido para o stakeholder Administrador. Ou seja, este objetivo passa a ter o interesse do Administrador.

Para esta inclusão serão utilizados os procedimentos para inclusão de instâncias da classe Stakeholder, apresentados na seção 6.5.

Nota-se que não há procedimentos para a inclusão de um relacionamento, pois aqueles correspondentes à inclusão de um stakeholder ou de um objetivo já suprem esta necessidade. Sendo assim, neste caso está sendo incluído um objetivo.

De acordo com a Figura 47, deve-se verificar se o objetivo será relacionado com algum stakeholder. Abaixo são apresentados os procedimentos desta alteração.

- Procedimento 1: Verificar se o objetivo será incluído para um stakeholder específico ou será incluído, por enquanto, sem nenhum interessado.

Como o foco é a inclusão do objetivo para o stakeholder Administrador, o próximo passo é relacionar o objetivo incluído com este stakeholder.

- Procedimento 2: Relacionar o objetivo incluído com um stakeholder.

Neste passo será incluído o relacionamento entre o objetivo Anunciar produtos e o stakeholder Administrador. A execução deste procedimento pode ser verificada na Figura 56.

4.3 Funções do produto

O sistema deixa que os usuários que desejam vender produtos façam seu cadastro, anunciem seus produtos com descrição completa em uma seção relacionada. Qualquer usuário (vendedor ou não) pode procurar por produtos. Qualquer usuário pode procurar os produtos pelo nome, por palavras relacionadas ou por tipo de produto. Qualquer usuário pode procurar por produtos de um usuário que vende produtos. Qualquer usuário pode enviar mensagens aos usuários que possuem os produtos.

Atores: vendedor, usuário.

Objetivos: Vender produtos; Realizar cadastro; Anunciar produtos; Pesquisar produtos; Enviar mensagens;

Lista ator x objetivo:

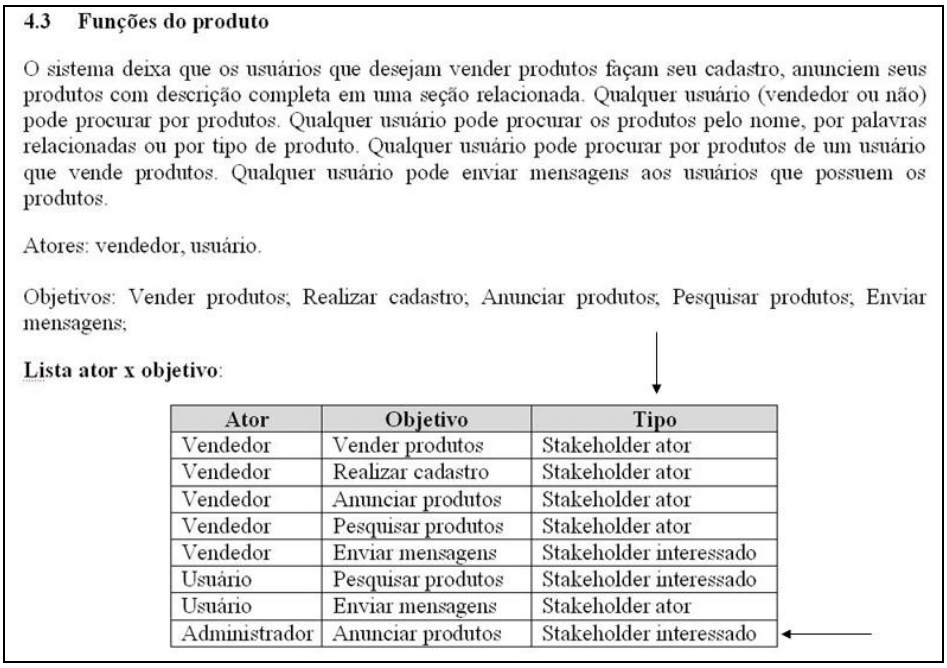
Ator	Objetivo
Vendedor	Vender produtos
Vendedor	Realizar cadastro
Vendedor	Anunciar produtos
Vendedor	Pesquisar produtos
Vendedor	Enviar mensagens
Usuário	Pesquisar produtos
Usuário	Enviar mensagens
Administrador	Anunciar produtos

Figura 56 – Inclusão do objetivo Anunciar produtos relacionado com o stakeholder Administrador.

Após ter sido relacionado, deve-se verificar se o stakeholder participante do relacionamento é o stakeholder ator para o objetivo ou apenas o stakeholder interessado.

- Procedimento 3: Verificar se o stakeholder é um ator ou um interessado.

O stakeholder participante do relacionamento incluído é um interessado. Para que isto fique mais claro, sugere-se que seja incluída mais uma coluna na tabela de ator x objetivo apresentada na SRS indicando o tipo de stakeholder, conforme Figura 57.



4.3 Funções do produto

O sistema deixa que os usuários que desejam vender produtos façam seu cadastro, anunciem seus produtos com descrição completa em uma seção relacionada. Qualquer usuário (vendedor ou não) pode procurar por produtos. Qualquer usuário pode procurar os produtos pelo nome, por palavras relacionadas ou por tipo de produto. Qualquer usuário pode procurar por produtos de um usuário que vende produtos. Qualquer usuário pode enviar mensagens aos usuários que possuem os produtos.

Atores: vendedor, usuário.

Objetivos: Vender produtos; Realizar cadastro; Anunciar produtos; Pesquisar produtos; Enviar mensagens;

Lista ator x objetivo:

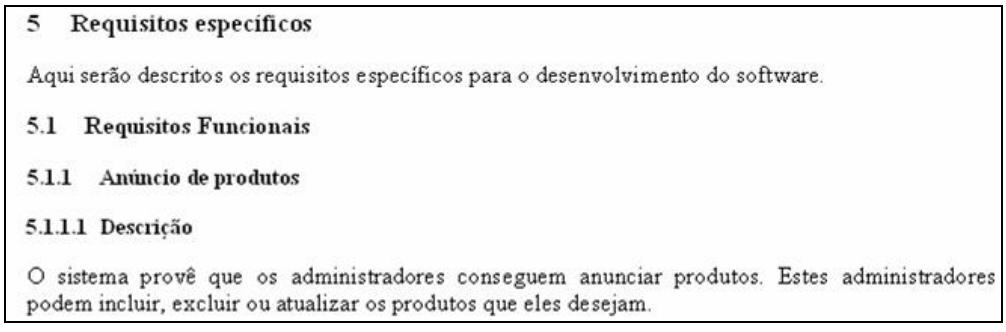
Ator	Objetivo	Tipo
Vendedor	Vender produtos	Stakeholder ator
Vendedor	Realizar cadastro	Stakeholder ator
Vendedor	Anunciar produtos	Stakeholder ator
Vendedor	Pesquisar produtos	Stakeholder ator
Vendedor	Enviar mensagens	Stakeholder interessado
Usuário	Pesquisar produtos	Stakeholder interessado
Usuário	Enviar mensagens	Stakeholder ator
Administrador	Anunciar produtos	Stakeholder interessado

Figura 57 – Inclusão de nova coluna na tabela de ator x objetivo.

Como o stakeholder Administrador é um interessado, deve-se identificar a instância relacionada de RequisitoFuncional.

û Procedimento 4: Identificar a instância relacionada de RequisitoFuncional.

Como o Administrador é apenas um stakeholder interessado no objetivo, subentende-se que o objetivo em questão já possui um stakeholder ator relacionado a ele e, com isso uma instância relacionada de RequisitoFuncional. Sendo assim, esta deve ser identificada, como mostra a Figura 58.



5 Requisitos específicos

Aqui serão descritos os requisitos específicos para o desenvolvimento do software.

5.1 Requisitos Funcionais

5.1.1 Anúncio de produtos

5.1.1.1 Descrição

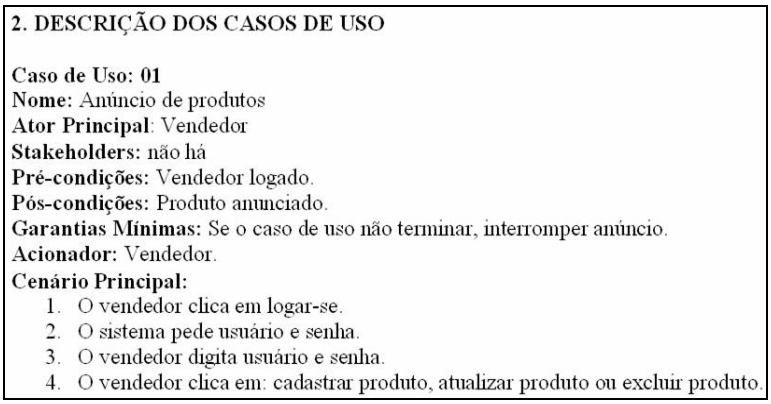
O sistema provê que os administradores conseguem anunciar produtos. Estes administradores podem incluir, excluir ou atualizar os produtos que eles desejam.

Figura 58 – Identificação do requisito funcional correspondente ao objetivo Anunciar produtos.

O próximo passo é identificar a instância relacionada de CasoUso.

û Procedimento 5: Identificar a instância relacionada de CasoUso

Para poder incluir um stakeholder em um caso de uso, este deve ser identificado. Para isto mostra-se, na Figura 59, o caso de uso relacionado ao requisito funcional Anúncio de produtos.


2. DESCRIÇÃO DOS CASOS DE USO

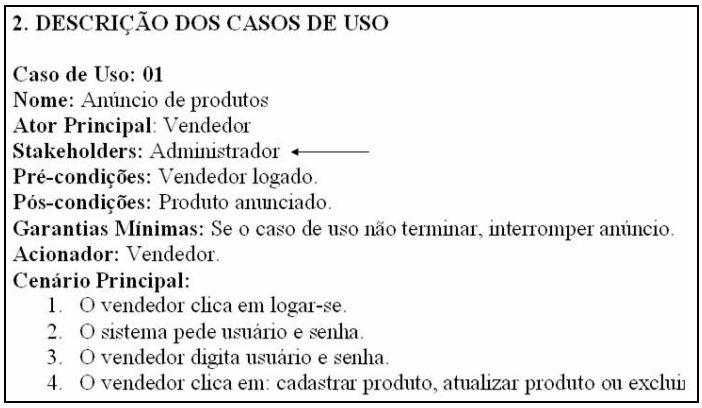
Caso de Uso: 01
Nome: Anúncio de produtos
Ator Principal: Vendedor
Stakeholders: não há
Pré-condições: Vendedor logado.
Pós-condições: Produto anunciado.
Garantias Mínimas: Se o caso de uso não terminar, interromper anúncio.
Acionador: Vendedor.
Cenário Principal:
1. O vendedor clica em logar-se.
2. O sistema pede usuário e senha.
3. O vendedor digita usuário e senha.
4. O vendedor clica em: cadastrar produto, atualizar produto ou excluir produto.

Figura 59 – Caso de uso correspondente ao requisito funcional Anúncio de produtos.

Em seguida, deve ser incluída a instância de Stakeholder.

û Procedimento 6: Incluir instância relacionada de Stakeholder.

Feita a identificação do caso de uso correspondente ao requisito funcional, deve-se incluir a instância Administrador como um stakeholder deste caso de uso, sendo assim, ele passa a ficar como mostrado na Figura 60.


2. DESCRIÇÃO DOS CASOS DE USO

Caso de Uso: 01
Nome: Anúncio de produtos
Ator Principal: Vendedor
Stakeholders: Administrador ←
Pré-condições: Vendedor logado.
Pós-condições: Produto anunciado.
Garantias Mínimas: Se o caso de uso não terminar, interromper anúncio.
Acionador: Vendedor.
Cenário Principal:
1. O vendedor clica em logar-se.
2. O sistema pede usuário e senha.
3. O vendedor digita usuário e senha.
4. O vendedor clica em: cadastrar produto, atualizar produto ou exclui

Figura 60 – Inclusão do stakeholder Administrador no caso de uso Anúncio de produtos.

Com a realização deste procedimento, a alteração é finalizada.

7.3 Avaliação dos resultados

Embora não tenham sido apresentados todos os exemplos possíveis da aplicação do modelo e do processo de rastreabilidade propostos neste trabalho, os apresentados contribuíram facilitando a visualização da proposta e comprovando seu funcionamento diante dos objetivos inicialmente propostos.

É importante considerar que como o documento utilizado não contém todos os campos e todas as características propostas, algumas avaliações ficaram incompletas. Entretanto, isto não

significa que tenham ficado comprometidas visto que o padrão facilita a aplicação mas não a restringe.

De qualquer forma, o documento apresentado no Apêndice III foi reescrito no padrão proposto, Apêndice IV, a fim de demonstrar a facilidade e a organização que ele propõe. Este padrão também pode ser considerado uma contribuição deste trabalho, já que sua utilização para a escrita do documento de especificação de requisitos permite que todos os interessados compreendam as funcionalidades da mesma maneira, mesmo sem ter participado da construção do documento.

Os exemplos apresentados foram escolhidos aleatoriamente, dentre os mais complexos, diante da grande possibilidade de alterações que podem ser realizadas com esta proposta.

Durante sua efetivação, pôde-se verificar que a rastreabilidade e a consistência entre os documentos era constantemente mantida visto que a realização de uma alteração implicava diretamente na alteração de todos os elementos impactados por ela. Isto fez com que os elementos permanecessem sempre consistentes um com o outro, evitando que ocorressem discrepâncias ocasionadas pela falta de gerenciamento dos requisitos.

O modelo de rastreabilidade pôde ser avaliado durante esta aplicação, visto que as relações nele identificadas foram seguidas e comprovadas no decorrer do exemplo. Não houve a necessidade de incluir, excluir e nem adaptar nenhum dos relacionamentos propostos.

Com relação aos procedimentos de alteração dos elementos, comprovou-se, também, sua efetividade. A utilização destes procedimentos facilita a realização da rastreabilidade entre as instâncias das classes dos modelos, visto que são didáticos e de fácil compreensão por qualquer pessoa da área ou não, o que contribuiu, também, para a construção do Protótipo que dá suporte ao modelo.

7.4 Considerações do capítulo

Neste capítulo foi apresentada a avaliação do modelo e do processo propostos. Para isto, foi utilizada uma SRS juntamente com os casos de uso correspondentes. Foram aplicados os procedimentos para a realização de três exemplos de alteração e, logo após, os resultados obtidos foram descritos.

O próximo capítulo apresenta as considerações finais obtidas com o desenvolvimento desta dissertação.

8 CONSIDERAÇÕES FINAIS

A engenharia de requisitos é uma das atividades críticas e mais importantes do processo de desenvolvimento de software, visto que é a partir dela que o sistema final será desenvolvido. Atualmente, é tida como um dos principais desafios na engenharia de software e como a principal razão de muitas das falhas ocorridas nos sistemas, já que se os requisitos não forem eficientemente elicitados, informações podem ser perdidas e, com elas, funcionalidades importantes do sistema.

A utilização, durante todo o processo de desenvolvimento, dos documentos de especificação de requisitos em conjunto com o modelo de casos de uso pode contribuir para que o resultado alcançado ao final deste processo seja considerado satisfatório por todos os interessados. Se estes documentos estiverem sempre alinhados entre si há uma possibilidade maior de que o processo de desenvolvimento siga sempre pelo caminho ideal, chegando ao resultado esperado.

Neste contexto, o tema abordado nesta dissertação busca auxiliar a atividade de engenharia de requisitos, através da definição de um modelo e de um processo de rastreabilidade entre o documento de especificação de requisitos do sistema (SRS) e o modelo de casos de uso do sistema (MCU). Partindo de uma documentação definida, consistente e completa, aumentam-se as possibilidades de que o processo de desenvolvimento seja realizado adequadamente e de que as necessidades dos usuários sejam atendidas. Através da utilização deste modelo, torna-se possível seguir os elos definidos entre as instâncias de suas classes, fazendo com que mudanças realizadas nos requisitos não sejam desconsideradas e possibilitando que o documento da SRS esteja sempre em conformidade com seu MCU.

Outro ponto considerado durante o desenvolvimento desta dissertação, foi a facilidade de leitura por todos os stakeholders. Sabendo-se que os requisitos devem estar escritos de forma com que todos os interessados compreendam da mesma maneira suas funcionalidades, optou-se pela utilização de um padrão de escrita, já que este procedimento é citado na literatura como uma possibilidade de diminuir este tipo de problema.

Conforme apresentado no Capítulo 1, o objetivo geral desta pesquisa foi atingido. Foi apresentado um modelo de rastreabilidade entre o documento de especificação de requisitos e o modelo de casos de uso, que possibilita que seus artefatos estejam sempre consistentes entre si. Da mesma forma, os objetivos específicos também foram atingidos, como mostrado durante o decorrer desta dissertação.

O referencial teórico, apresentado no Capítulo 2, surgiu devido à pesquisa bibliográfica

realizada sobre o tema, onde os principais pontos foram considerados e apresentados. O estudo de caso realizado permitiu avaliar a documentação utilizada e criada pela empresa, comparando-a com a proposta deste trabalho. Neste caso, a proposta pôde ser adaptada a fim de que satisfizesse tanto os fundamentos levantados na literatura como as necessidades apresentadas pela empresa participante. Por fim, a aplicação do modelo, através do exemplo apresentado no Capítulo 7 e o protótipo desenvolvido, permitiram avaliá-lo para a confirmação do alcance dos objetivos desta pesquisa.

8.1 Contribuições

A proposta de um modelo de rastreabilidade entre a SRS e o MCU visa contribuir para a área de engenharia de software ao preencher uma lacuna existente na área de engenharia de requisitos.

A partir de um documento de requisitos consistente com as necessidades do negócio e completo de acordo com estas necessidades é possível de serem gerados, também, casos de uso consistentes e completos.

Neste sentido, como contribuição deste trabalho tem-se:

- Um modelo de rastreabilidade entre a SRS e o MCU, que mantém a consistência entre seus artefatos, tornando possível, mesmo quando requisitos são alterados, que o resultado seja satisfatório de acordo com as necessidades dos usuários.
- Um processo de rastreabilidade, que facilita a realização do rastreamento entre as instâncias das classes do modelo.
- Um protótipo de uma ferramenta, detalhado no Apêndice VI, que suporta a proposta apresentada.

Com isto, o estudo visa contribuir com a comunidade científica e com a prática na indústria ao atender uma demanda organizacional crescente por qualidade do produto final.

8.2 Limitações do Estudo

A principal limitação do estudo está associada aos padrões utilizados nos documentos. Apesar de ser confirmado pela literatura que documentos padronizados melhoram a qualidade e a organização do projeto, nem sempre eles são utilizados.

Embora a proposta possa ser utilizada para documentos fora do padrão estabelecido, os resultados, neste caso, serão limitados, visto que, assim, não há subsídios de que os documentos contenham todas as informações necessárias para a satisfação dos usuários e nem que estas

informações permaneçam consistentes entre si quando da mudança nos requisitos. Isto se dá pelo fato de que é proposta uma rastreabilidade a nível estrutural e não a nível de conteúdo, ou seja, as informações descritas no documento não são analisadas semanticamente.

8.3 Trabalhos Futuros

O objetivo inicial da pesquisa apresentada nesta dissertação era propor um processo de geração do modelo de casos de uso do sistema consistente com os requisitos do negócio, estudados em detalhes em [CER05b]. Após a primeira etapa, com o amadurecimento da pesquisa, já com mais enfoque nos trabalhos existentes no meio acadêmico e, também, devido a sugestões e estudos realizados no grupo de pesquisas de engenharia de requisitos, optou-se por adaptar o foco do trabalho.

A adaptação se deu na restrição do foco para os requisitos do sistema e para os modelos de casos de uso do sistema. Inicialmente partir-se-ia dos requisitos do negócio e dos modelos de casos de uso do negócio com o objetivo de chegar aos requisitos do sistema e aos modelos de casos de uso do sistema. O foco, neste contexto, estava muito abrangente e ficaria complicado evoluir adequadamente o desenvolvimento do trabalho. Porém, sabendo-se da importância em considerar o contexto do negócio durante todo o processo de desenvolvimento, optou-se por partir de um documento de requisitos do sistema que, teoricamente, estivesse completo, ou seja, abrangendo todas as funcionalidades desejadas pelos clientes e por todos os stakeholders envolvidos no projeto.

Com base nisto, como trabalho futuro pretende-se realizar a rastreabilidade, também, entre os artefatos propostos no modelo de negócio. Assim, partindo da documentação do negócio e mantendo o restante da documentação do processo sempre consistente com ela, as chances de ter-se um produto final dentro das expectativas dos usuários aumentarão consideravelmente.

Outro trabalho futuro diz respeito a formalização, em OCL, dos procedimentos, apresentados inicialmente em linguagem natural. Ainda, pretende-se realizar análises semânticas entre os elementos fazendo uso, provavelmente, de processamento da linguagem natural. Com isso, além de um rastreamento a nível estrutural ter-se-á, também, um rastreamento a nível de conteúdo.

REFERÊNCIAS BIBLIOGRÁFICAS

[AND05] ANDERSON, B. “Formalism, technique and rigour in Use Case Modeling”. Journal of Object Technology, vol. 4, No. 6, Special Issue: Use Case Modeling at UML-2004, Aug. 2005, pp. 15-28.

[BEL05] BELGAMO, A.; FABBRI, S. “TUCCA: Técnica de Leitura para apoiar a Construção de Modelos de Casos de uso e a Análise de Documentos de Requisitos”. In: XIX Simpósio Brasileiro de Engenharia de Software, 2005. Capturado em: http://www.sbbd-sbes2005.ufu.br/arquivos/04-%209625.pdf, Março 2006. 16p.

[BEL05b] BELGAMO, A.; FABBRI, S.; MALDONADO, J. “TUCCA: Improving the Effectiveness of Use Case Construction and Requirement Analysis”. In: IEEE International Symposium on Empirical Software Engineering, Los Alamitos, California: IEEE Computer Society Press, 2005, pp. 257-266.

[BER98] BERRY, D.; LAWRENCE, B. “Guest Editor’s Introduction – Requirements Engineering”. IEEE Software, 1998, pp. 26-29

[BLA05] BLASCHEK, J. R. “Gerência de Requisitos, o principal problema dos projetos de software”. Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2003. Capturado em: http://www.bfpug.com.br/islig-rio/Downloads/Gerência%20de%20 Requisitos-o%20Principal%20Problema%20dos%20Projetos%20de%20SW.pdf, Setembro 2005. 4p.

[CER05a] CERRI, E. “A importância da especificação de requisitos de software e a investigação de seus métodos informais e formais”. Trabalho Individual I. Programa de Pós Graduação em Ciência da Computação, PUCRS, 2005, 70p.

[CER05b] CERRI, E. “A importância da modelagem de negócios para o desenvolvimento dos sistemas”. Trabalho Individual II. Programa de Pós Graduação em Ciência da Computação, PUCRS, 2005, 55p.

[CER06a] CERRI, E. “Uma proposta para geração do modelo de casos de uso do sistema a partir do documento de especificação de requisitos”. Seminário de Andamento. Programa de Pós Graduação em Ciência da Computação, PUCRS, 2006, 10p.

[CER06b] CERRI, E.; ROCHA, F.; BASTOS, R. “Um modelo de rastreabilidade entre o documento de especificação de requisitos e o modelo de casos de uso”. In: III Simpósio Brasileiro de Sistemas de Informação, 2006, [CDROM]. 8p.

[CES02] CESPEDES M. “Uma proposta para melhorar o rastreamento de requisitos de software”. 2002. Dissertação de Mestrado, Centro de Informática. Universidade Federal de Pernambuco. Recife, 2002, 190p.

[COC01] COCKBURN, A. “Writing Effective Use Cases”. Addison-Wesley, 2001, 246p.

[COC97] COCKBURN, A. “Structuring use cases with goals”. Journal of Object-Oriented Programming, vol. 5, no. 10, 1997, pp. 56-62.

[COX01] COX, K.; PHALP, K.; SHEPPERD, M. “Comparing Use Case Writing Guidelines”. In: 7th Int. Workshop on Requirements Engineering: Foundation for Software Quality, REFSQ’01, June 2001. Capturado em: http://dec.bournemouth.ac.uk/ESERG/kphalp/refsq01.pdf, April 2005. 12p.

[CRU03] CRUZ, C.; RIBEIRO, U. “Metodologia cientifica: teoria e pratica”. Rio de Janeiro: Axcel Books, 2003, 218 p.

[DAV93] DAVIS, A. M. “Software Requirements: Objects, Functions and Status”. Englewood Cliffs, New Jersey: Prentice Hall, 1993, 521p.

[FIO98] FIORINI, S.; LEITE, J.; LUCENA, C. “Organizado Processos de Requisitos”. In: Anais do WER98 - Workshop em Engenharia de Requisitos, 1998, pp. 1-8.

[FRA98] FRANCH, X.; BOTELLA, P. “Putting Non-Functional Requirements into Software Architecture”. In: 9th International Workshop on Software Specification. IEEE Software, 1998, pp. 60-67.

[FRE84] FREEMAN, R. “Strategic Management: A Stakeholder Approach”. Boston: Pitman, 1984, 276p.

[GOG96] GOGUEN, J. “Formality and Informality in Requirements Engineering”. In: Proceedings of the Second International Conference on Requirements Engineering, April 1996, pp. 102-108.

[GOT06] GOTTESDIENER, E. “Use Case: Best Practices”. Capturado em: http://www.ebgconsulting.com/pubs/Articles/UseCaseBestPractices-ottesdiener.pdf, Junho 2006. 19p.

[GOT97] GOTEL, O.; FINKELSTEIN, A. “Extended RequirementsTraceability: Results of an Industrial Case Study”. In: Proceedings of 3rd International Symposium on RequirementsEngineering RE97, 1997, pp. 169-178.

[HAZ03] HAZAN, C.; LEITE, J.C.S.P. “Definição de Indicadores Gerenciais de Requisitos”. In: Proceedings of the Sixth International Workshop on Requirements Engineering (WER2003), 2003, pp. 285-301.

[HEU01] HEUMANN, J. "What does "No time for Requirements" mean?". The Rational Edge, Nov. 2001, 7p.

[IEE84] IEEE Std 830-1984. “IEEE Guide to Software Requirements Specifications”. New York: Institute of Electrical and Electronic Engineers, 1984, 26p.

[IEE98] IEEE Std 830-1998. “IEEE Recommended Practice for Software Requirements Specifications”. New York: Institute of Electrical and Electronic Engineers, 1998, 38p.

[JAC92] JACOBSON, I.; CHRISTERSON, M.; JONSSON, P.; ÖVERGAARD, G. “Object-Oriented Software Engineering: A Use Case Driven Approach”. Addison-Wesley, 1992, 528p.

[JAR98] JARKE, M. “Requirements Tracing”. Communications of the ACM, vol. 41, 1998, pp. 32-36.

[KOT98] KOTONYA, G.; SOMMERVILLE, I. “Requirements Engineering: process and techniques”. New York: John Wiley & Sons Ltd, 1998, 282p.

[KRU00] KRUCHTEN, P. “The Rational Unified Process: An Introduction”. USA: Addison Wesley, 2000, 320p.

[KRU01] KRUCHTEN, P. “The Rational Unified Process: An Introduction”. Addison-Wesley, 2001, 298p.

[KUL03] KULAK, D.; GUINEY, E. “Use Cases: Requirements in Context”. Addison-Wesley, 2003, 272p.

[LEF00] LEFFINGWELL, D.; WIDRIG, D. "Managing Software Requirements - A Unified Approach”. Addison-Wesley, 2000, 492p.

[LOP04] LOPES, L.; PRIKLADNICKI, R.; AUDY, J.; MAJDENBAUM, A. “Distributed Requirements Specification: Minimizing the Effect of Geographic Dispersion”. In: Proceedings of 6th International Conference on Enterprise Information Systems – ICEIS, 2004, pp. 531-534.

[MCC96] MCCONNELL, S. “Rapid Development”. Microsoft Press, 1996, 647p.

[MED04] MEDEIROS, E. “Desenvolvendo software com UML 2.0”. São Paulo: Makron Books, 2004, 264p.

[NAD02] NADDEO, P. “Uma Taxonomia da Pesquisa na Área de Engenharia de Requisitos”. Dissertação de Mestrado. Instituto de Matemática e Estatística da Universidade de São Paulo, São Paulo, Mar. 2002, 92p.

[NUS00] NUSEIBEH, B.; EASTERBROOK, S. “Requirements Engineering: A Roadmap”. ACM - Future of Software Engineering, 2000, pp 37-45.

[OMG05] Object Management Group. “UML Superstructure Specification, v2.0”. OMG, 2005, 710 p.

[PAL97] PALMER, J.D. “Traceability”. In: Software Requirements Engineering, R.H. Thayer and M. Dorfman (Eds), 1997, pp. 364-374.

[POH96] POHL, K. “PRO-ART: Enabling Requirements Pre-Traceability”. In: Proceedings of the Second IEEE International Conference on Requirements Engineering, Colorado Springs, April, 1996, pp. 76-84.

[PRE01] PRESSMAN, R. “Engenharia de Software”. Rio de Janeiro: Mc Graw Hill, 2001, ed. 5, 888p.

[RAM01] RAMESH, B., JARKE, M. “Towards Reference Models for Requirements Traceability”. IEEE Trans. Software Engineering, vol. 27 (1), 2001, pp. 58-93.

[RAM95] RAMESH, B.; STUBBS, C.; POWERS, T.; EDWARDS, M. “Implementing Requirements Traceability: A Case Study”. In: Proceedings of the Second IEEE International Symposium on Requirements Engineering, March, 1995, pp. 89-95.

[ROC06a] ROCHA, F. “Modelo para avaliação da qualidade entre requisitos e casos de uso“. Seminário de Andamento. Programa de Pós Graduação em Ciência da Computação, PUCRS, 2006, 11p.

[ROC06b] ROCHA, F.; CERRI, E.; BASTOS, R.; YAMAGUTI, M. “Uma Proposta de Modelo para Avaliação da Qualidade da Tradução de Requisitos para Casos de Uso”. In: III Simpósio Brasileiro de Sistemas de Informação – III SBSI, 2006, [CDROM]. 8p.

[RUP06] RUP. RATIONAL Software Corporation. “Rational Unified Process Win Evaluation Version 7.0”. Capturado em: https://www14.software.ibm.com/webapp/iwm/web reg/ download.do?pkgid=&S_SRCID=RATLe-RUPWIN-EVAL&source= RATLe-UPWIN-EVAL&S_TACT= 105AGX28&S_CMP=DLMAIN&S_PKG= CR3DGNA& cp=UTF-8#, Junho 2006.

[SAY05] SAYÃO, M.; LEITE, J. C. S. P. “Rastreabilidade de Requisitos”. Relatório Técnico 2005, série Monografias em Ciência da Computação, DI/PUC-Rio, 2005, 30p.

[SCH00] SCHMIDT, M. “Implementing the IEEE Software Engineering Standards”. SAMS Publishing, USA, 2000, 255p.

[SCH01] SCHNEIDER, G.; WINTERS, J. “Applying Use Cases, A Practical Guide”. Addison-Wesley, 2001, 245p.

[SHA99] SHARP, H.; FILKENSTEIN, A.; GALAL, G. “Stakeholder Identification in the Requirements Engineering Process”. In: 10th International Workshop on Database & Expert Systems (REP’99), 1999, pp. 387-391.

[SMI05] •MIA•EK, M. “Accommodating informality with necessary precision in use case scenarios”. Journal of Object Technology, vol. 4, No. 6, Special Issue: Use Case Modeling at UML-2004, Aug. 2005, pp. 59-67.

[SOM97] SOMMERVILLE, I.; SAWYER, P. “Requirement Engineering – A good pratice guide”. EUA: Wiley, 1997, 391p.

[SOM03] SOMMERVILLE, I. “Engenharia de Software”. Addison Wesley, 2003, 606p.

[SOM05] SOMMERVILLE, I. “Integrated Requirements Engineering: A Tutorial”. IEEE Software, vol. 22, 2005, pp. 16-23.

[SOM06] SOMÉ, S. S. “Supporting use case based requirements engineering”. Information and Software Technology, vol. 48, 2006, pp. 43-58.

[SPA02] SPANOUDAKIS, G. “Plausible and adaptive requirement traceability structures”. In: 14th International Conference on Software Engineering and Knowledge Engineering, Ischia, Italy, Jul, 2002, pp. 135-142.

[TOR02] TORANZO, M.; CASTRO, J.; MELLO, E. “Uma proposta para melhorar o rastreamento de requisitos”. In: WER02 – Workshop em Engenharia de Requisitos, Valencia, Espanha, Novembro, 2002, pp. 194-209.

[STA95] STANDISH G. (1995) “CHAOS Report”. Capturado em: http://www.standishgroup. Com, Janeiro 2006. 9p.

[TOR99] TORANZO, M.; CASTRO, J. “The Multiview++ Environment: Requirements Traceability from perspective of stakeholders”. In: WER’99 – Workshop em Engenharia de Requisitos, Buenos Aires, Argentina, 1999, pp. 198-216.

[UML03] UML. “Unified Modeling Language: Superstructure”. Version 2.0, Final Adopted Specification, ptc/03-08-02, Object Management Group, 2003, 640p.

[UML05] UML. “Unified Modeling Language: Superstructure”. Version 2.0, Final Adopted Specification, ptc/05-07-04, Object Management Group, 2005, 710p.

[YIN01] YIN, R. “Estudo de Caso: planejamento e métodos”. São Paulo: Bookman, 2001, 212p.

APÊNDICE I – SEMINÁRIO DE ANDAMENTO [CER06a]

Uma proposta para geração do modelo de casos de uso do sistema a partir do documento de especificação de requisitos

Elisa Cerri e Cerri e Ricardo Melo Bastos

Resumo – ***O principal objetivo deste trabalho é gerar o modelo de casos de uso do sistema a partir de um documento de requisitos, bem escrito, que compreenda todas as necessidades do negócio. Será descrito um modelo conceitual da SRS e outro do Modelo de Casos de Uso. O primeiro tem como base as informações contidas na SRS da IEEE e o segundo baseia-se no Modelo de Casos de Uso do RUP e no template de Cockburn. É apresentado, também, um modelo de integração, que identifica a relação entre seus elementos. A partir disto, será definido um processo para rastreabilidade entre a SRS e o Modelo de Casos de Uso e uma ferramenta para apoiar a execução do processo. A principal contribuição desta proposta é auxiliar o processo de requisitos, propondo mecanismos que permitam que o resultado do processo, o modelo de casos de uso, seja rastreável com o documento de especificação de requisitos do sistema.***

Palavras-chave – Especificação de Requisitos, Modelos de Casos de Uso, Rastreabilidade, SRS.

I. Introdução

Os sistemas de software estão cada vez mais presentes no cotidiano das pessoas nos mais diferenciados contextos, que envolvem desde aplicações triviais até sistemas críticos. Uma questão fundamental no desenvolvimento de um software é garantir que ele represente, realmente, uma solução para o problema proposto. Estudos demonstram que a maioria dos projetos são cancelados ou fracassam por não atenderem completamente às necessidades dos clientes e excederem os prazos e os orçamentos estimados [1].

Sabendo-se que um documento de requisitos mal especificado pode gerar um produto incompleto ou inconsistente com as necessidades dos clientes, foram realizados estudos sobre a importância da especificação de requisitos para o desenvolvimento do software, buscando compreender padrões de uma especificação com qualidade.

Os modelos de casos de uso, atualmente, vêm sendo muito considerados pela comunidade científica, visto que facilitam a representação dos requisitos do sistema. Várias abordagens de desenvolvimento de software, incluindo o Processo Unificado (PU) [3], recomendam os casos de uso para a descrição dos requisitos.

É importante considerar que apenas descrever bem os requisitos não traz garantias suficientes para um produto completo, de qualidade e que considere todas as necessidades organizacionais. Para que isto seja possível, é imprescindível a realização detalhada e cautelosa de um processo de elicitação e negociação com os clientes para, então, chegar a um documento de especificação de requisitos válido e que represente tudo o que eles desejam. Tendo em mãos um documento de requisitos bem escrito e completo pode-se, então, partir para a descrição dos requisitos, através dos casos de uso.

Com isso, este projeto tem como objetivo desenvolver um processo e uma ferramenta de apoio para geração consistente do modelo de casos de uso conforme definido no RUP (*Rational Unified Process*) [4] a partir do Documento de Especificação de Requisitos do Sistema (*SRS – Software Requirements Specification*) proposto pela IEEE [5]. O processo permitirá a rastreabilidade entre a SRS e os elementos componentes do modelo de casos de uso, incluindo a especificação textual e diagrama de atividades bem como o diagrama de casos de uso para o sistema. A ferramenta a ser desenvolvida como protótipo neste projeto será utilizada para apoiar a execução do processo proposto.

Este documento está organizado da seguinte forma: a seção I corresponde a esta introdução. A seção II apresenta o referencial teórico para este trabalho. A seção III descreve os modelos conceituais da SRS e do Modelo de Casos de Uso e o modelo de integração entre eles, apresentando seus elementos e relacionamentos. Ainda nesta seção são apresentadas as idéias principais do processo proposto e da ferramenta de apoio. E, por fim, a seção IV apresenta os resultados parciais, os objetivos, as contribuições deste trabalho e as próximas atividades.

II. Referencial teórico

Esta seção descreve o referencial teórico deste trabalho apresentando os principais conceitos envolvidos com o tema de pesquisa. Serão apresentados, ainda, alguns

trabalhos relacionados que estão servindo como referência para o desenvolvimento do processo.

A. Processo de Engenharia de Requisitos

Os requisitos são um componente fundamental no processo de software. Não utilizar requisitos é equivalente a pensar “não há tempo para descobrir o que se deve construir, devemos começar a construir agora” [6]. Apesar de a literatura apresentar diversas definições para requisitos, neste projeto optou-se pela utilização da definição de Gougen [7] que diz que os requisitos são propriedades que um software deve possuir para funcionar com sucesso no ambiente para o qual foi proposto, considerando tanto o contexto social quanto o contexto técnico do ambiente, o que tem se mostrado importante na utilização de requisitos, pois muitas das informações utilizadas por eles estão ligadas ao ambiente social dos usuários e gerentes e podem ser informais ou tácitas.

Os requisitos estão associados aos principais problemas do desenvolvimento de software. O sucesso de um sistema de software é determinado pela maneira com que ele satisfaz as necessidades do cliente, onde esta satisfação é determinada pela forma com que a solução proposta está relacionada com as necessidades apontadas pelos *stakeholders* [8].

Na tentativa de evitar este tipo de problema é importante que as organizações disponham de um processo de requisitos bem definido, controlado, medido e aprimorado para guiar os desenvolvedores na árdua tarefa de definição de requisitos de um sistema [9].

O processo fundamental de Engenharia de Requisitos (RE) varia dependendo do tipo de aplicação que será desenvolvida, do tamanho e da cultura das companhias envolvidas e do processo de aquisição de software utilizado, porém, independente do tipo de processo realizado, algumas atividades são fundamentais: o reconhecimento do problema, a interpretação das necessidades que demandam o sistema, a modelagem, a especificação e a validação dos requisitos que atendem estas necessidades [10].

Como em qualquer processo, o processo de RE pode apresentar várias dificuldades. A maior delas, certamente, é a definição precisa sobre o que construir. Este é um problema antigo e que está presente até hoje no processo de software. Nenhuma outra parte é mais difícil de ser corrigida tardiamente [11].

B. Especificação de Requisitos

Geralmente, os requisitos são escritos em linguagem natural e freqüentemente são descrições vagas do que se deseja em lugar de especificações detalhadas. Em situações onde os requisitos mudam constantemente essa pode ser a solução aconselhável, pois os custos de manutenibilidade de uma especificação detalhada são injustificáveis. Em outras situações, porém, uma falha em definir precisamente o que é requerido pode resultar em conflitos infinitos entre o cliente e os desenvolvedores [12].

As especificações podem ser classificadas em: informais, semi-formais e formais. As informais são as representações em linguagem natural, as semi-formais podem ser representadas, por exemplo, por linguagens gráficas e as formais são representações matemáticas para o sistema [13].

Neste trabalho, serão utilizadas as representações informais, fazendo uso de linguagem natural e as representações semi-formais, através dos modelos e descrições de casos de uso.

As necessidades organizacionais devem ser consideradas, durante a especificação, fazendo uso de técnicas de elicitação como reuniões, entrevistas, questionários e de análises dos documentos da empresa, em busca de garantir que o resultado do processo seja o esperado. Com isso, busca-se uma definição consistente do documento de requisitos para que, a partir dele, possa ser gerado um modelo de casos de uso que garanta a inclusão de todas as funcionalidades esperadas pelos clientes.

A linguagem natural será utilizada na construção da SRS, através da adaptação de uma estrutura, previamente definida em pesquisas anteriores, para garantir um padrão de escrita para os requisitos. A modelagem e a especificação de casos de uso serão utilizadas para descrever os requisitos representados em linguagem natural na SRS.

C. Documento de Especificação de Requisitos

O documento de especificação de requisitos, também chamado de *SRS (Software Requirements Specification)*, é uma especificação para um produto de software específico, um programa ou um conjunto de programas que executam certas funções em um ambiente específico. Deve ser escrito por um ou mais representantes do fornecedor, um ou mais representantes dos clientes ou por ambos [5].

Alguns pontos como documentar as necessidades dos usuários, evitar deduções prematuras de design, apresentar requisitos não conflitantes e não redundantes, devem sempre ser considerados para alcançar a qualidade de uma SRS. Uma maneira de garantir tais aspectos é permitir uma rastreabilidade entre os requisitos [14]. Para isto, é fundamental que todos os passos do projeto sejam documentados relatando, por exemplo, a qual requisito uma classe específica se refere, qual caso de teste reflete um requisito específico e assim por diante. Tais cuidados

podem garantir que uma mudança em determinado ponto não se torne uma catástrofe para o projeto final, já que se terá documentado quais artefatos serão atingidos com tal procedimento.

De acordo com a IEEE [5] para que possa ser considerada boa, uma SRS deve ser: correta, não ambígua, completa, consistente, superior pela importância e/ou estabilidade, verificável, modificável e rastreável.

Existem diversas maneiras de escrever uma SRS, geralmente elas são adaptadas à realidade do projeto e/ou da empresa. Infelizmente, nem sempre a escrita segue um padrão ideal e nem sempre considera todos os aspectos necessários para que o documento possa ser considerado completo. Com isso, o restante do processo fica debilitado, considerando que se determinadas características foram esquecidas ou expressas de maneira inadequada o resultado pode não ser o esperado.

Para evitar este tipo de problema uma das propostas deste trabalho está em adaptar um padrão de escrita para este documento garantindo, assim, que os documentos sejam padronizados mesmo quando escritos por pessoas em diferentes contextos.

D. Modelos de Casos de Uso

Um modelo de caso de uso é um modelo das funções a serem realizadas pelo sistema e seus limites. Funcionam como um contrato entre os *stakeholders* [4]. Os casos de uso (*Use Case – UC*) servem como uma linha base a ser seguida durante todo o desenvolvimento do sistema e são um resultado obtido através da análise da especificação dos requisitos. Segundo [15] um UC é uma seqüência de ações que um sistema realiza para gerar um resultado observável de valor para um ator em particular, onde um ator é alguém ou alguma coisa de fora do sistema que interage com ele.

Existem muitas maneiras de modelar um sistema, cada qual servindo para uma finalidade diferente. Entretanto, o propósito mais importante de um modelo de casos de uso é comunicar o comportamento do sistema ao cliente ou ao usuário final [4], ou seja, é necessária a utilização de texto simples e de fácil compreensão.

A modelagem de requisitos funcionais, através da especificação dos casos de uso é, atualmente, considerada como uma abordagem extremamente adequada, visto que facilita o entendimento e possibilita uma melhor comunicação entre a equipe de desenvolvimento e os usuários [16]. Porém, independente de os requisitos estarem bem representados e/ou bem descritos, é preciso que eles sejam válidos, ou seja, que eles representem as reais necessidades dos usuários. Para isso, é fundamental que eles sejam rastreáveis.

E. Rastreabilidade

O rastreamento dos requisitos [17][18] refere-se à habilidade para descrever e seguir a vida de um requisito em ambas as direções, para frente e para trás. De acordo com [19], os requisitos não podem ser efetivamente gerenciados sem rastreabilidade.

A rastreabilidade é uma técnica fundamental no apoio às diversas atividades do projeto, assegurando que sistemas e software estão em conformidade às mudanças dos requisitos [17][18]. Entretanto, é citada como uma área problema pelos desenvolvedores.

Apesar de se saber que o esforço necessário para a aplicação da rastreabilidade pode ser elevado, como aumento de custo ou tempo, sem a utilização de mecanismos deste tipo podem surgir inconsistências durante o processo de adição, remoção ou modificação de requisitos.

A rastreabilidade, neste trabalho, será usada para garantir que todos os requisitos expressos na SRS estejam englobados nos casos de uso. O inverso também é verdadeiro, já que se, por ventura, forem encontradas dúvidas na especificação de determinado caso de uso estas podem ser sanadas na descrição de seu requisito de origem.

F. Trabalhos Relacionados

Foram estudados trabalhos encontrados na literatura que apresentam características correspondentes ao contexto deste trabalho. Suas idéias básicas são apresentadas a seguir.

O trabalho desenvolvido por [21] apresenta uma técnica para construção do modelo de casos de uso e para análise do documento de requisitos. TUCCA (*Technique for Use Case model construction and Construction-based requirements document Analysis*) é composta por duas técnicas de leitura: : AGRT (*Actor Goal Reading Technique*) e UCRT (*Use Case Reading Technique*) cujos propósitos são, respectivamente, determinar os atores do sistema e seus objetivos e determinar o Modelo de Casos de Uso. Ambas as técnicas são utilizadas por meio de passos que dão suporte à construção de Modelos de Casos de Uso e incorporam, também, uma revisão do Documento de Requisitos. TUCCA foi definida com o objetivo de reduzir o grau de subjetividade e a necessidade de experiência do projetista.

TUCCA utiliza por padrão a SRS da IEEE [5] como artefato básico para ambas as técnicas. Porém, nem todas as seções da SRS são necessárias, a premissa é de que se a SRS incluir as seções Definições, Funções do Produto, Características do Usuário, Requisitos Funcionais e Requisitos Não-Funcionais já é possível a aplicação da técnica. Porém o campo “Requisitos Não-Funcionais” não

faz parte do modelo apresentado em [5]. Aparentemente este campo foi definido para facilitar a aplicação da técnica, entretanto, não se pode dizer que ele é proposto na SRS da IEEE.

O trabalho apresenta considerações importantes para identificação de atores e objetivos, em busca de encontrar os casos de uso do sistema, porém apenas cita que existem sub-passos a serem aplicados para a especificação detalhada dos casos de uso. Outro aspecto identificado nesta pesquisa foi a maneira como o requisito está descrito. O trabalho não se preocupa em definir um padrão de escrita para os requisitos, o que pode fazer com que a técnica tenha um desempenho diferente se os requisitos não forem escritos de maneira semelhante aos do estudo de caso apresentado em [21].

Embora tais limitações tenham sido identificadas, este trabalho contribui para esta pesquisa no sentido de definir alguns relacionamentos e também na definição de algumas diretrizes para utilização do processo proposto.

O trabalho de Somé [16] apresenta uma abordagem para dar suporte aos casos de uso, baseando-se na elicitação, clarificação, composição e simulação dos requisitos. Oferece uma forma restrita de linguagem natural, dependente de contexto, para casos de uso e uma derivação automatizada da especificação. O modelo de domínio é usado para capturar os conceitos do domínio relevantes ao contexto, bem como as definições das operações de pré e pós-condições em paralelo com os casos de uso.

Este trabalho considera como base o *template* proposto por Cockburn [21], onde os casos de uso são descritos usando um texto estruturado. Não faz referências ao uso de documentos de requisitos, porém apresenta interessantes maneiras de representar condições e operações fazendo uso de uma estrutura de linguagem natural buscando, com isso, evitar alguns problemas referentes desta área. Como o objetivo principal desta pesquisa não é simplesmente a descrição dos casos de uso, alguns detalhes importantes como a representação dos requisitos não-funcionais, são deixados de lado.

Como se tem o propósito de utilizar uma maneira estruturada de escrever os requisitos em linguagem natural, o trabalho de Somé [16] poderá contribuir para melhorar e enriquecer a estrutura, definida em pesquisas anteriores.

III. Processo de Geração dos Modelos de Casos de Uso

Neste trabalho é proposto um processo para geração do modelo de casos de uso a partir do documento de especificação de requisitos do sistema.

Para o documento de especificação de requisitos, optou-se pela utilização da SRS da IEEE. Isso se deu pelo fato de este documento ser utilizado pela indústria e, também, amplamente referenciado e aceito pela comunidade acadêmica.

Para o modelo de casos de uso, optou-se por utilizar como referência o modelo proposto pelo RUP, devido a sua relevância como processo de engenharia de software e sua ampla utilização no mercado. O *template* para especificação de requisitos será o proposto por Cockburn [22], visto que apresenta informações abrangentes, organizadas e de fácil entendimento pelos *stakeholders*.

Para facilitar a leitura e compreensão da SRS, foi estabelecido um padrão de escrita para os requisitos visto que sem isso, tratando-se de linguagem natural, ficaria praticamente impossível definir qualquer processo de extração de informação dos documentos.

A. Estrutura para especificação de requisitos em linguagem natural

Este modelo é resultado de uma pesquisa desenvolvida no Centro de Desenvolvimento e Pesquisa DELL/PUCRS (CDPe).

A idéia de seu desenvolvimento surgiu devido a diversos desafios identificados na engenharia de requisitos em ambientes distribuídos de software, que podem ser conferidos em [23].

O padrão de escrita é apresentado em forma de uma estrutura de frase, conforme Fig.1.

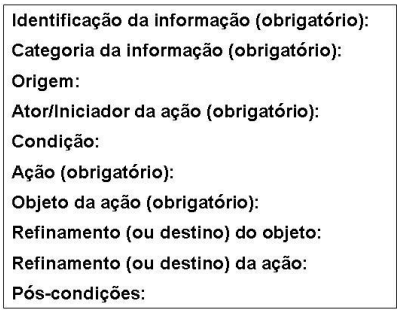


Fig. 1 – Estrutura de frases em linguagem natural

Esta estrutura é composta de campos padrão que devem ser preenchidos de acordo com regras para cada categoria de informação. Quando a estrutura de frase é preenchida, a frase final é obtida concatenando-se os campos em uma ordem pré-definida. O resultado final representa o requisito em si e é escrito na seção Requisitos Funcionais da SRS.

Para este trabalho, o padrão de escrita será utilizado não apenas na seção Requisitos Funcionais da SRS, mas também nas seções de Requisitos de Performance, Requisitos de Banco de Dados Lógico, Atributos do

Sistema, Requisitos de Interface e Limites de Desenvolvimento, cada seção com o padrão adaptado para suas características.

Com isso existirão mais possibilidades de que a técnica para extração dos casos de uso possa ser aplicada efetivamente.

B. Modelo Conceitual da SRS

A partir dos campos e características apresentadas na documentação da SRS da IEEE de 1998 [5] foi construído um modelo conceitual, que pode ser visto na 0. Este modelo foi definido em conjunto a partir de pesquisas realizadas no grupo de Engenharia de Requisitos do Centro de Desenvolvimento e Pesquisas DELL/PUCRS (CDPe) e será utilizado para o propósito desta pesquisa.

As classes apresentadas representam os elementos indicados nas práticas recomendadas para especificação de requisitos de software, definidas pela IEEE [5], e as associações entre estes elementos representam como eles estão relacionados. Seus atributos foram identificados na descrição de cada elemento, encontradas em [5][24][25], e representam as informações referentes a cada um deles.

Schimidt [25] coloca que [5] fornece *guidelines* detalhadas das informações que devem estar contidas em cada uma das seções da SRS, porém afirma que, ainda assim, existem dúvidas sobre quais informações detalhar em determinadas seção.

Um ponto comum de confusão, de acordo com [25], é a diferença entre a seção “Propósito” e a seção “Escopo”, subseções da “Introdução”. Segundo ele, a primeira deve delinear a finalidade da SRS e identificar sua audiência enquanto a segunda deve identificar o software em questão, explicando o que ele fará, descrevendo sua aplicação, seus benefícios e objetivos.

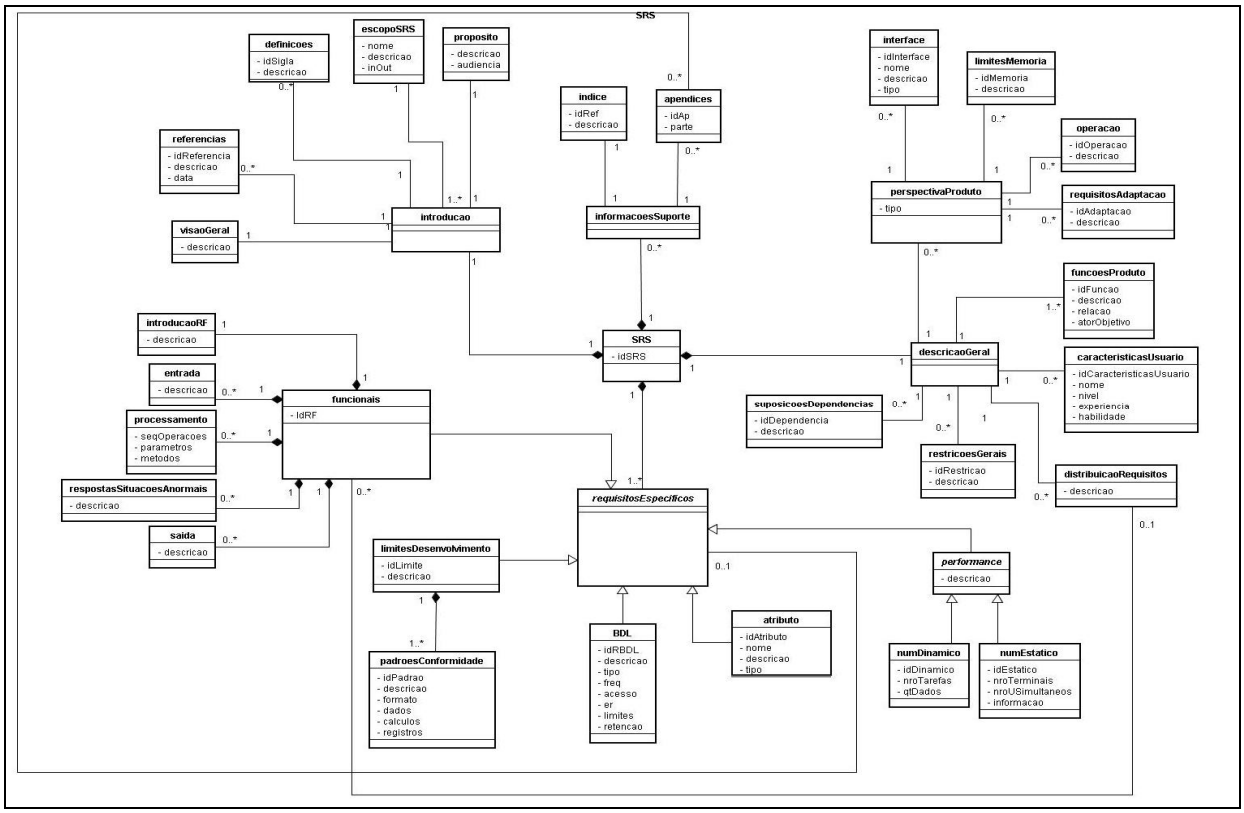


Fig. 2 – Modelo Conceitual da SRS.

Outra possível confusão geralmente ocorre em relação às informações presentes nas seções 2 (Descrição Geral) e 3 (Requisitos Específicos), já que ambas devem incluir informações sobre as interfaces, funcionalidades e restrições do software. Entretanto a seção 2 não especifica os requisitos, apenas dá uma base destes que serão detalhados na seção 3 da SRS.

Como o motivo deste artigo é apresentar o andamento das pesquisas, serão descritas, apenas, as classes que serão diretamente utilizadas no modelo de integração da SRS com os Casos de Uso. As classes serão apresentadas em negrito e os atributos em itálico. São elas: escopoSRS**, introducaoRF, entrada, processamento**, **respostasSituacoesAnormais, saidas, performance, atributo, suposicoesDependencias, funcoesProduto.**

A classe **escopoSRS** representa a subseção Escopo da SRS e descreve os objetivos específicos, focando no software que está sendo especificado. Tem como atributos o nome para identificar o produto de software a ser produzido, o atributo *inOut* que fornece uma lista "dentro/fora" e o atributo *descricao* que indica, por exemplo, benefícios relevantes, objetivos e metas.

A classe **introducaoRF** representa o requisito em si e contém, para isto, o atributo *descricao* que mantém a especificação do requisito. O formato padrão para esta descrição é o apresentado na seção anterior.

A classe **entrada** é encontrada na definição da seção Requisitos Funcionais da SRS e identifica todas as entradas possíveis para determinado requisito. Para ela é definido o atributo *descricao* utilizado para especificar as entradas.

A classe **processamento** é encontrada na definição da seção Requisitos Funcionais na SRS e descreve todas as funções realizadas pelo sistema em resposta a uma entrada ou em suporte a uma saída. É definido o atributo *seqOperacoes* que mantém a seqüência exata das operações, incluindo o tempo dos eventos, o atributo *parametros* que informa os parâmetros afetados pelas operações, ou seja, quais as conseqüências que elas trarão e o atributo *metodos* responsável por representar como as entradas serão transformadas nas saídas, apresenta, por exemplo: equações, algoritmos e operações lógicas.

A classe **respostasSituacoesAnormais** é encontrada na seção Requisitos Funcionais e define todas as possibilidades de exceção na execução de determinado requisito. O atributo *Descricao* é criado para manter tais possibilidades.

A classe **saida** faz parte da definição da seção Requisitos Funcionais na SRS, é responsável por manter todas as saídas possíveis para o requisito. Para isto é definido o atributo *descricao.*

A classe **performance** representa a seção Requisitos de Performance da SRS que é responsável por especificar os requisitos numéricos estáticos e dinâmicos existentes no software. Todos estes requisitos devem ser expressos em termos mensuráveis. Para armazenar a especificação deste requisito é criado o atributo *descricao*. Além disso, são criadas duas subclasses para representar os requisitos de performance numéricos estáticos e os requisitos de performance numéricos dinâmicos, o que significa dizer que os requisitos de performance podem ser destes dois tipos.

A classe **atributo** representa a seção Atributos do Sistema de Software da SRS, que especifica atributos de confiabilidade, disponibilidade, segurança, manutenabilidade e portabilidade, que podem ser especificados desde que sejam verificáveis. São definidos os atributos *idAtributo* que mantém sua identificação, *nome* que dá a idéia principal do requisito, *descricao* que mantém sua informação detalhada e *tipo* para definir o tipo de cada atributo.

A classe **suposicoesDependencias** representa a subseção Suposições e Dependências da SRS que define os fatores que afetam os requisitos expressos na SRS como condições específicas de hardware. O atributo *idDependencia* foi definido para manter a identificação e o atributo *descricao* foi definido para manter a descrição da informação.

A classe **funcoesProduto** representa a subseção Funções do Produto da SRS que fornece uma relação das funções do sistema, a fim de informar os principais objetivos. Para isto é criado o atributo *idFuncao* que mantém a identificação da função, o atributo *descricao* que descreve o objetivo e *relacao* que identifica as relações entre as funções. Contém, também, um atributo que armazena a relação entre os atores e os objetivos chamado *atorObjetivo.*

C. Modelo Conceitual dos Casos de Uso

Da mesma forma que no modelo anterior, este foi desenvolvido em conjunto pelo grupo de Engenharia de Requisitos do Centro de Desenvolvimento e Pesquisas DELL/PUCRS (CDPe). Este modelo teve como base o *template* de casos de uso de Cockburn [22], onde para cada campo foi definida uma classe, responsável por armazenar as devidas informações dos casos de uso. Estas informações são representadas pelos atributos das classes, que foram identificados em pesquisas realizadas em [4][16][22], que também faz uso do mesmo *template*. Considerando que existem vários *templates* para descrição dos casos de uso e que estes contêm vários campos correspondentes, os elementos explicados para cada seção foram sendo complementados e optou-se pela representação no *template* em questão.

O modelo conceitual dos Casos de Uso pode ser visto na Fig. 3.

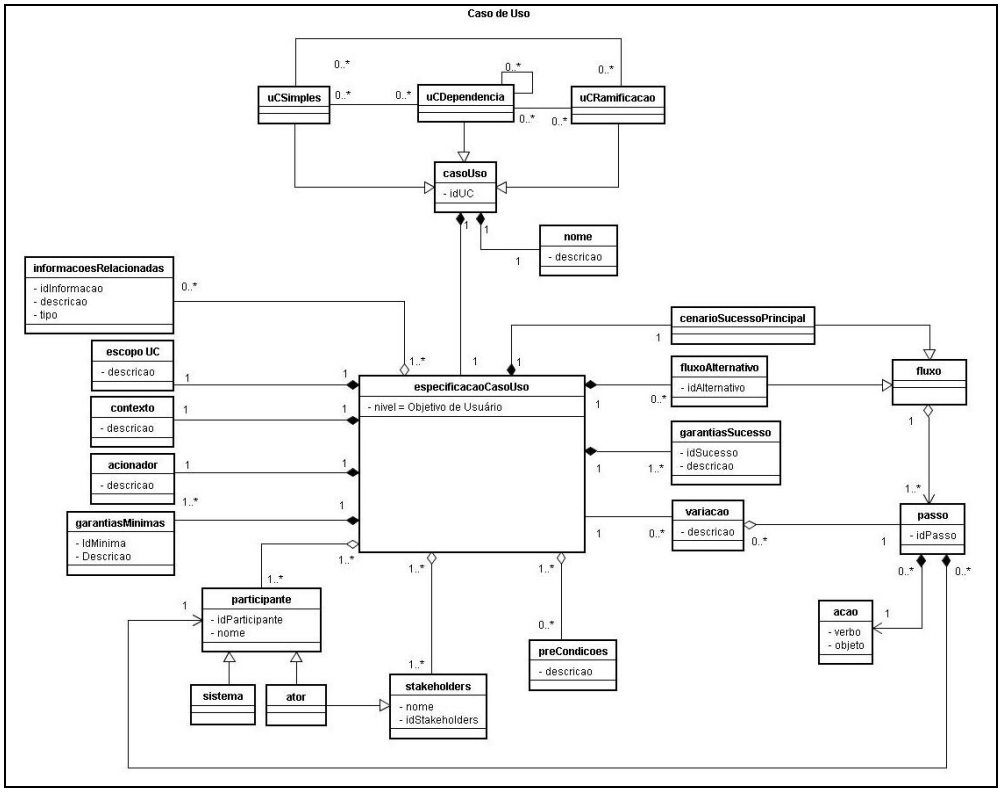


Fig. 3 - Modelo Conceitual dos Casos de Uso

A classe **escopoUC** define qual a funcionalidade está sendo considerada neste caso de uso. Define o atributo *descricao* que mantém a informação deste escopo.

A classe **contexto** mantém uma sentença maior do objetivo. Para isto, define o atributo *descricao.*

A classe **participante** armazena as informações do participante do caso de uso, que pode ser o sistema ou o ator. Para armazenar uma identificação única para cada participante, define-se o atributo *idParticipante.* Para manter o nome, é definido o atributo *nome.*

A classe **sistema** representa um participante sistema.

A classe **ator** armazena o ator primário do caso de uso.

A classe **acionador** mantém a informação referente ao que dá inicio ao caso de uso. Pode ser um ator executando uma ação ou mesmo um chamado vindo de outro caso de uso. Para armazenar estas informações é criado o atributo *descricao.*

A classe **stakeholders** relaciona todos os interessados em determinado caso de uso. Para isto cria os atributos *idStakeholder* que mantém a identificação do tipo do *stakeholder* (cliente, desenvolvedor etc.) e *nome* para manter a descrição do mesmo.

A classe **preCondicoes** define o que já deve ser verdadeiro para seja possível a execução do caso de uso. Para isto define o atributo *descricao.*

A classe **garantiasMinimas** define como os interesses são protegidos sob todas as saídas, ou seja, se o caso de uso falhar quais são as garantias dadas aos interessados. Para esta classe são definidos os atributos *idMinima* que identifica a possível falha e *descricao* que explica a garantia em si.

A classe **garantiasSucesso** define quais os interesses dos *stakeholders* são satisfeitos depois de uma conclusão bem-sucedida do caso de uso. São definidos os atributos *idSucesso* que identifica o interesse satisfeito e *descricao* que o descreve.

A classe **variacoes** armazena a lista das variações tecnológicas e de dados, que é responsável por indicar as varias maneiras, se existirem, de determinada funcionalidade ser executada. Será referenciado o passo de origem de cada variação. Para a definição das variações é criado o atributo *descricao.*

A classe **informacoesRelacionadas** relaciona todas as informações adicionais necessárias. Para esta classe definido atributo *idInformacao* para manter sua identificação, o atributo *descricao* que mantém a informação e o atributo *tipo* que identifica se a informação é uma especificação não-funcional ou uma regra de negócio.

A classe **fluxo** é responsável por armazenar um conjunto de passos que serão executados para a realização de determinada funcionalidade.

A classe **fluxoAlternativo** é um tipo de fluxo, ou seja, define as mesmas características da classe **fluxo.** Mantém o atributo *idAlternativo* para identificar o passo que deu origem a este fluxo.

A classe **cenarioSucessoPrincipal** também é um tipo de fluxo. É responsável por manter uma execução de sucesso para o caso de uso.

A classe **passos** é responsável por descrever cada passo a ser executado dentro de um fluxo. Os passos são identificados por números, através do atributo *idPasso*. Cada passo é formado pelo conjunto de um ator e uma ação.

A classe **acao** mantém um atributo *verbo* e um atributo *objeto* que interagem com um ator em questão, formando assim um passo do caso de uso.

A classe **casoUso** apenas define que os casos de uso podem ser dos tipos: simples, dependência e ramificação. Define o atributo *idUC* para manter a identificação única do caso de uso.

A classe **uCSimples** é definida para indicar se o caso de uso é um caso de uso simples.

A classe **uCDependencia** édefinida para indicar se o caso de uso em questão é um caso de uso do tipo <*include*> ou do tipo <*extend*>.

A classe **uCRamificacao** é definida para indicar se o caso de uso é um caso de uso de generalização.

A seção Cenário de Sucesso Principal do *template*, é onde é contada a história sobre o que o sistema entrega como resultado. Antes de tentar identificar todas as exceções ou condições de falha, deve ser mostrado como o sistema funciona em caso de sucesso. Este tipo de cenário pode gerar exceções, o que ocasiona a criação de um Fluxo Alternativo para seu tratamento. Tanto o cenário de sucesso quanto o fluxo alternativo são compostos de passos que explicam sua execução.

D. Modelo de Integração

O modelo de integração é resultado de uma análise realizada sob os modelos conceituais da SRS e dos Casos de Uso. Através de estudos e pesquisas realizadas em artigos, relatórios e documentos [4][5][22][24][25], foram identificadas as relações entre os elementos de ambos os modelos para que fosse possível definir de onde poderiam ser retiradas, na SRS, as informações para as descrições dos casos de uso.

Assim, foram definidas as relações mostradas na Fig. 4, que são explicadas a seguir.

CLASSES CASOS DE USO	CLASSES SRS
nome	introducaoRF
escopo	escopo
contexto	funcoesProduto
ator	introducaoRF
stakeholders	funcoesProduto
preCondicoes	processamento
garantiasMinimas	suposicoesDependencias
garantiasMinimas	respostasSituacoesAnormais
garantiasSucesso	introducaoRF
garantiasSucesso	saida
cenarioSucessoPrincipal	entrada
cenarioSucessoPrincipal	funcoesProduto
fluxoAlternativo	respostasSituacoesAnormais
variacoes	processamento
variacoes	funcoesProduto
informacoesRelacionadas	atributo
informacoesRelacionadas	performance

Fig.4 – Integração entre as classes dos modelos

Os motivos para criação dos relacionamentos apresentados na figura acima são explicados abaixo. É bom considerar que as regras para a extração dos elementos ainda estão em desenvolvimento e por este motivo ainda não serão apresentadas.

nome / introducaoRF: o nome do caso de uso será retirado da introdução do requisito funcional, que será estruturada da maneira apresentada na subseção A da seção III.

escopo / escopo: a informação para o escopo da descrição dos casos de uso pode ser retirada das informações apresentadas na seção escopo da SRS, fazendo a análise dos atributos *inOut* e *descricao*, a fim de comparar tais informações e encontrar suas ligações com os objetivos que estão sendo abrangidos no caso de uso.

contexto / funcoesProduto: o contexto do caso de uso pode ser retirado da seção funções do produto, na SRS, analisando os mesmos aspectos analisados no momento da retirada do escopo. Entretanto, a informação deve ser mais detalhada, já que o contexto deve ser uma sentença maior do objetivo.

ator / introducaoRF: o ator do caso de uso deve ser retirado da descrição do requisito funcional, que deve informar o ator principal responsável pela execução da ação.

stakeholders / funcoesProduto: os *stakeholders* do caso de uso são retirados de uma análise realizada no atributo *atorObjetivo*, da classe **funcoesProduto**, que relaciona os interesses dos atores nos objetivos especificados.

precondicoes / processamento: as precondições para os casos de uso podem ser encontradas na classe **processamento** dos requisitos funcionais. Além disso,

algumas precondições podem não estar expressas na SRS, como informações de mais baixo nível como usuário estar logado, etc.

garantiasMinimas / suposicoesDependencias: garantias mínimas podem ser encontradas na seção suposições e dependências da SRS, procurando identificar fatores que possam afetar a execução de determinado requisito, relacionado com o caso de uso em questão.

garantiasMinimas / respostasSituacoesAnormais: as garantias mínimas podem, ainda, ser retiradas das respostas às situações anormais dos requisitos funcionais, pois ali são identificados pontos de falha para o requisito e assim eles podem ser tratados.

garantiasSucesso / introducaoRF: uma garantia de sucesso será retirada do nome do caso de uso, obtido através da descrição do requisito funcional, já que uma pós-condição é ter a funcionalidade do caso de uso realizada.

garantiasSucesso / saidas: das saídas dos requisitos funcionais podem ser retiradas outras garantias de sucesso para os casos de uso.

cenarioSucessoPrincipal / entradas: informações para o cenário de sucesso principal podem ser retiradas da classe **entradas** dos requisitos funcionais visto que as mesmas poderão ser transformadas em passos para os casos de uso.

cenarioSucessoPrincipal / funcoesProduto: informações adicionais para o cenário de sucesso principal podem ser encontradas na seção funções do produto da SRS, visto que lá os objetivos do sistema podem conter mais detalhes do que os explicitados na parte dos requisitos funcionais.

fluxoAlternativo / respostasSituacoesAnormais: o fluxo alternativo pode ser identificado fazendo uma análise na parte de respostas às situações anormais dos requisitos funcionais. Lá podem ser encontradas quais as funcionalidade passíveis de ocasionar falhas e assim os passos que indicarão os fluxos alternativos para os casos de uso.

variacoes / processamento: as variações tecnológicas e de dados podem ser retiradas da classe **processamento** dos requisitos funcionais, analisando as entradas (na classe **entrada**) e verificando se existe alguma característica que indique uma ou várias maneiras possíveis da funcionalidade ser executada.

variacoes / funcoesProduto: as variações tecnológicas e de dados podem ser retiradas, também, realizando uma análise nas informações presentes na classe **funcoesProduto**, buscando indicações das possíveis maneiras das funcionalidades serem executadas.

informacoesRelacionadas / atributo: uma das possibilidades de identificar as informações relacionadas para os casos de uso é verificar se na classe **atributo** existe algum requisito relacionado com o objetivo em questão.

informacoesRelacionadas / performance: outra maneira de encontrar informações relacionadas é analisar, da mesma forma, a classe que representa os requisitos de performance da SRS.

Outro ponto que deve ser exposto é que durante o processo de geração do modelo de casos de uso, podem surgir dúvidas e, com elas, a descoberta de novas informações [22]. Por exemplo, durante a especificação de um caso de uso, podem ser encontradas outras funcionalidades que também possam gerar outros casos de uso. A idéia principal é de que a SRS em questão esteja completa, porém, não pode ser desconsiderada a hipótese de o cliente esquecer de fornecer algum dado que, para ele, possa parecer menos importante, mas que em um dado momento seja imprescindível, ou seja, pode acontecer de surgirem novas informações e novas dúvidas, o que seria o caso de uma nova negociação com o cliente para sanar tais pontos.

E. Processo Proposto

Conforme dito anteriormente, o processo proposto está em definir a rastreabilidade entre os requisitos do sistema e o Modelo de Casos de Uso. Para isto, está sendo estabelecido um conjunto de procedimentos a serem realizados que, além de suportar o processo de rastreabilidade, define diretrizes para derivar as informações dos casos de uso a partir da SRS.

Serão artefatos deste processo:

- Estrutura de Frases em Linguagem Natural;
- Modelo Conceitual da SRS;
- Modelo Conceitual do Modelo de Casos de Uso;
- Modelo Conceitual de Integração;
- Diretrizes para definição dos elementos;
- Diretrizes para rastreabilidade;

A estrutura de frases em linguagem natural, mostrada na seção III, será utilizada como um padrão de escrita para os requisitos. Esta padronização define o conjunto de elementos, componentes da especificação de um requisito, que devem ser preenchidos conforme as regras propostas pelo método, facilitando a geração dos casos de uso.

Os modelos conceituais da SRS e dos Casos de Uso foram definidos a fim de que os relacionamentos entre seus elementos fossem identificados. Através da análise das informações presentes em suas classes, foi possível definir-se um modelo de integração, que mostra como

estas classes estão relacionadas, ou seja, qual classe em um modelo influi e/ou é influenciada por outra classe do outro modelo.

O modelo de integração servirá como ponto de partida para a definição do processo de geração dos casos de uso e de sua rastreabilidade com o documento de especificação de requisitos que conterá diretrizes para facilitar a identificação dos elementos que devem estar presentes nos documentos (SRS e Casos de Uso).

Com isso, no momento em que houver a necessidade de alteração em um determinado requisito, será possível apontar quais casos de uso serão afetados por esta mudança, permitindo, assim, a rastreabilidade.

Sendo assim, o primeiro propósito deste trabalho é fornecer um instrumento que permita verificar se todas as informações presentes na SRS estão englobadas no modelo de casos de uso. Como segundo propósito tem-se que realizada determinada manutenção em qualquer ponto de um dos dois documentos, será indicado o ponto em que o outro documento será afetado. Como terceiro propósito chega-se ao objetivo da ferramenta, que será desenvolvida com a finalidade de auxiliar no processo de visualização de tais pontos.

F. Ferramenta de Apoio à execução do processo

Visando apoiar automaticamente e dar auxílio à avaliação do processo proposto, será desenvolvido um protótipo de uma ferramenta.

O escopo desta ferramenta está em fase de desenvolvimento, mas o objetivo é que ela dê apoio a rastreabilidade entre os casos de uso e o documento de especificação de requisitos do sistema.

O uso desta ferramenta permitirá auxiliar o usuário a perceber visualmente a rastreabilidade entre os documentos, fazendo com que os impactos ocasionados pelas mudanças ocorridas em um sejam mais facilmente identificados no outro.

IV. Resultados, Objetivos e Contribuições

Esta seção descreve os resultados obtidos até o momento, visando verificar o andamento do trabalho, os resultados desejados, os objetivos previamente definidos e identificar as reais contribuições deste trabalho.

A. Resultados Parciais

No Plano de Estudo e Pesquisa (PEP) foram definidos objetivos e atividades. Conforme o andamento das pesquisas, estes objetivos foram sendo adaptados e as atividades alteradas. Com isso, tem-se como resultados parciais:

- Estudo da base teórica;
- Avaliação de propostas de geração de casos de uso e suas relações com os documentos de requisitos;
- Identificação dos elementos a serem considerados para geração dos casos de uso a partir do documento de requisitos do sistema;
- Definição do modelo conceitual da SRS;
- Definição do modelo conceitual dos Casos de Uso;
- Definição do modelo preliminar, chamado modelo de integração, para geração dos casos de uso a partir dos requisitos do sistema;

Inicialmente foram definidos os modelos conceituais da SRS e dos Casos de Uso. Analisando estes modelos, estabeleceu-se um modelo de integração, responsável por definir onde as informações contidas em um podem ser identificadas em outro.

B. Objetivos e contribuições

Os resultados deste processo visam fornecer apoio para um processo adequado de engenharia de requisitos a fim de definir padrões para uma descrição eficiente e consistente dos casos de uso do sistema, buscando facilitar a comunicação e entendimento entre os *stakeholders* para, com isso, garantir a conformidade com as necessidades dos clientes. Isto é de ampla importância já que, atualmente, é cada vez mais comum as organizações estarem insatisfeitas ao final de um processo de desenvolvimento de software, visto que nem sempre o sistema resultante é o sistema esperado.

Para conclusão deste trabalho, algumas atividades estão em desenvolvimento. São elas:

- Estudo de caso;
- Definição de um *guideline* de boa formação, com a finalidade de estabelecer diretrizes para derivar as informações dos casos de uso a partir da SRS;
- Desenvolvimento do processo de rastreabilidade entre os modelos;
- Especificaçao da ferramenta de apoio ao processo proposto;
- Redação da dissertação;

O estudo de caso, em andamento, tem como propósito identificar os elementos presentes nas SRS e a completude de suas informações, buscando avaliar a aplicabilidade do modelo de integração definido e analisar vários casos de uso, juntamente com suas SRS, em busca de informações incompletas ou presentes em um e não presentes em outro, visando encontrar subsídios que indiquem se o caso de uso está realmente de acordo com sua SRS ou não.

Como resultado deste estudo de caso ter-se-á a avaliação do modelo de integração que apoiará a definição do processo, para, portanto, realizar um experimento em uma ou mais aplicações desta empresa a fim de identificar pontos problemáticos na integração dos

modelos. Este experimento servirá como base para a definição do processo de rastreabilidade.

Definido o processo de rastreabilidade, considerando os resultados dos estudos de caso, será desenvolvido o protótipo de uma ferramenta. A avaliação do processo se dará através de outros experimentos utilizando como apoio a ferramenta desenvolvida.

REFERÊNCIAS

[1] STANDISH Group. "CHAOS Report". Capturado em: http://www.standishgroup.com , 1995.

[2] CERRI, Elisa Cerri e. "A importância da especificação de requisitos de software e a investigação de seus métodos informais e formais". Trabalho Individual I. Pontifícia Universidade Católica do Rio Grande do Sul. 2005. 70p.

[3] JACOBSON, I.; BOOCH, G.; RUMBAUGH, J. "The Unified Software Development Process". Addison Wesley, New York, 1998.

[4] RATIONAL Software Corporation. "Rational Unified Process Win Evaluation". Version 7.0. Capturado em: https://www14.software.ibm. com/webapp/iwm/web/reg/download.do?pkgid=&S_SRCID=RATLe-RUPWIN-EVAL&source=RATLe-RUPWIN-EVAL&S_TACT=105AGX28&S_CMP=DLMAIN&S_PKG=CR3DGNA& cp=UTF-8#. Junho, 2006.

[5] IEEE, Institute. "IEEE Std 830-1998. IEEE Recommended Practice for Software Requirements Specifications". New York: Institute of Electrical and Electronic Engineers. Inc,. 1998.

[6] HEUMANN, Jim. "What does "No time for Requirements" mean?". The Rational Edge. Nov. 2001. 7p.

[7] GOGUEN, Joseph. "Formality and Informality in Requirements Engineering". In: International Conference on Requirements Engineering (ICRE '96), 2., 1996, Colorado Springs, EUA. Proceedings..., IEEE Computer Society. 1996. p. 102-109.

[8] NUSEIBEH, B., EASTERBROOK, S.. "Requirements Engineering: A Roadmap". ACM - Future of Software Engineering. 2000. pp 37-45

[9] BLASCHEK, J. R. "Gerência de Requisitos, o principal problema dos projetos de software". Artigo Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2003. Capturado em: http://www.bfpug.com. br/isligrio/Downloads/Gerência%20de%20Requisitos-o%20Principal%20Problema%20dos%20Projetos%20de%20SW.pdf. Setembro, 2005.

[10] PRESSMAN, Roger S. "Engenharia de Software". McGrawHill. 2001.

[11] BERRY, Daniel; LAWRENCE, Brian. "Guest Editor's Introduction – Requirements Engineering". IEEE Software. 1998. pp. 26-29

[12] SOMMERVILLE, Ian. "Integrated Requirements Engineering: A Tutorial". IEEE Software, 22 (1), 16-23, January/February 2005.

[13] SOMMERVILLE, Ian. "Engenharia de Software". 6ª edição. Addison Wesley. 2003.

[14] KULAK, D.; GUINEY, E. "Use Cases: Requirements in Context, Second Edition." Addison-Wesley, 2003.

[15] KRUCHTEN, Philippe. "The Rational Unified Process: An Introduction". USA: Addison-Wesley, 2000. 320p.

[16] SOMÉ, Stéphane S. "Supporting use case based requirements engineering". In: Information and Software Technology 48. January 2006, pp. 43-58.

[17] GOTEL, Orlena; FINKELSTEIN, Anthony. "Extended RequirementsTraceability: Results of an Industrial Case Study". ISRE'97 Third International Symposium on Requirements Engineering. IEEE. Proceedings,1997, pp 169-178.

[18] RAMESH, Bala; STUBBS, Curtis; POWERS, Tomothy; EDWARDS, Michael. Implementing Requirements Traceability: A Case Study ISRE'95 Second International Symposium on Requirements Engineering 1ed. USA . IEEE Proceedings. 1995. 12 p.

[19] KOTONYA, G.; SOMMERVILLE, I. "Requirements Engineering: process and techniques". New York: John Wiley & Sons Ltd, 1998, 282p.

[20] BELGAMO, A.; FABBRI, S; MALDONADO, J.C. "TUCCA Improving the Effectiveness of Use Case Construction and Requirements Analysis". In: 4TH International Symposium on Empirical Software Engineering. 2005.

[21] BELGAMO, A.; FABBRI, S.. "TUCCA: Técnica de Leitura para apoiar a Construção de Modelos de Casos de uso e a Análise de Documentos de Requisitos". In: XIX Simpósio Brasileiro de Engenharia de Software - Uberlândia, MG, Brasil – 2005.

[22] COCKBURN, Alistair. "Writing Effective Use Cases". Addison-Wesley, 2001. 246p

[23] LOPES, Leandro; PRIKLADNICKI, Rafael; AUDY, Jorge Luis Nicolas; MAJDENBAUM, Azriel. "Distributed Requirements Specification: Minimizing the Effect of Geographic Dispersion". Proceedings of 6th International Conference on Enterprise Information Systems - ICEIS. 2004.

[24] IEEE, Institute. "IEEE Std 830-1984. IEEE Guide to Software Requirements Specifications". New York: Institute of Electrical and Electronic Engineers. Inc,. 1984.

[25] SCHMIDT, M. "Implementing the IEEE Software Engineering Standards". SAMS Publishing, USA, 2000. 255p.

APÊNDICE II – QUESTÕES UTILIZADAS NO ESTUDO DE CASO

Este apêndice apresenta as questões utilizadas no estudo de caso para a análise individual das SRS bem como para a análise geral, realizada posteriormente.

1. Análise Individual das SRS:

Com relação às próximas 12 questões, as repostas devem ser consideradas caso as informações estejam identificadas.

1. Número de requisitos funcionais:
2. Número de requisitos não-funcionais específicos:
3. Número de requisitos não-funcionais gerais:
4. Número de requisitos de interface de entrada:
5. Número de requisitos de interface de saída:
6. Número de regras de negócio específicas:
7. Número de regras de negócio gerais:
8. Número de requisitos de performance:
9. Número de requisitos de banco de dados lógico:
10. Número de requisitos de restrição de memória:
11. Número de requisitos de operação:
12. Número de requisitos de adaptação de local:
13. Número de requisitos de documentação:
14. Número de atributos de sistema de software:
15. A SRS está no padrão utilizado pela organização?
16. A SRS apresenta requisitos distintos em níveis diferenciados?
17. Os requisitos estavam escritos de maneira simples de entender?
18. Os requisitos estavam escritos utilizando um padrão?
19. Para cada requisito funcional, quantos requisitos não-funcionais são relacionados?
20. Para cada requisito funcional, quantos requisitos de interface de entrada são relacionados?
21. Para cada requisito funcional, quantos requisitos de interface de saída são relacionados?
22. Para cada requisito funcional, quantas regras de negócio são apresentadas?
23. As relações entre requisitos funcionais são apresentadas?
24. A SRS apresenta um modelo de caso de uso?
25. Cada caso de uso representa quantos requisitos funcionais?
26. Todos os requisitos funcionais possuem um caso de uso relacionado?

27. Além dos requisitos funcionais, são englobados, também, na descrição dos casos de uso os requisitos não-funcionais, de performance entre outros?
28. A descrição dos casos de uso é apresentada sempre com o mesmo padrão?
29. Quais os elementos descritos na descrição dos casos de uso?
30. Existe um padrão de escrita para os cenários dos casos de uso?
31. O caso de uso apresenta, primeiramente, um cenário de sucesso principal?
32. O caso de uso diferencia um cenário de sucesso principal de um fluxo alternativo?
33. Como são apresentadas as “exceções” na descrição de um caso de uso?
34. Como são apresentadas as diversas maneiras de uma ação ser executada, na descrição de um caso de uso?
35. Como são apresentados os acionadores dos casos de uso, tem relação direta com a execução de outros casos de uso?
36. Como são apresentadas as pré-condições nos casos de uso?
37. Como são apresentadas as pós-condições nos casos de uso?
38. É possível identificar a origem do ator de um caso de uso?
39. São apresentados casos de uso de include, extend e generalização?
40. Quando existem casos de uso de include, extend e generalização são listados os casos de uso que se relacionam com eles?
41. O caso de uso apresenta uma relação de todos os interessados em sua execução?
42. Existe algum lugar, na SRS, em que são explicadas as informações que estão sendo apresentadas na descrição dos casos de uso?

2. Análise Geral das SRS:

1. Número de SRS analisados:
2. Todas as SRS estavam no mesmo padrão?
3. Todas as SRS apresentavam os requisitos distintos em locais diferenciados?
4. Os modelos de casos de uso são apresentados para todas as SRS?
5. Considerando SRS diferentes, as descrições de casos de uso são apresentadas com o mesmo padrão?

APÊNDICE III – ESPECIFICAÇÃO DE REQUISITOS E MODELO DE CASO DE USO DO PROJETO ML

Especificação de Requisitos do Projeto ML

SRS

Tabela de Conteúdo

1. Histórico de aprovação do documento

Nome	Data
José da Silva	01/01/2006
Pedro Paulo Souza	10/01/2006
Angelo Soares	17/03/2006

2. Histórico de revisão do documento

Data	Descrição
01/01/2006	Primeira versão.

3. Introdução

3.1 Propósito do documento

Os propósitos deste documento de requisitos são:

- Desenvolver e especificar as necessidade da comunidade que devem ser atendidas pelo projeto ML, bem como definir para os desenvolvedores e colaboradores o produto a ser feito.

3.2 Escopo do documento

Esta SRS contém os requisitos para o projeto ML, baseado nas necessidades do cliente.

3.3 Acrônimos e abreviaturas

Os seguintes acrônimos e termos serão usados neste documento e durante todo o projeto ML.

Sigla	Definição
ML	Minha Loja
LI	Loja na Internet

3.4 Referências

Referência	Descrição
[IEE98]	IEEE, Institute. “IEEE Std 830-1998. IEEE Recommended Practice for Software Requirements Specifications”. New York: Institute of Electrical and Electronic Engineers. Inc,. 1998.

3.5 Visão Geral

Este documento fornece informação sobre os requisitos a serem implementados no projeto ML, definidos pelo grupo de alunos da cadeira de Engenharia de Software.

4. Descrição Geral

4.1 Características do usuário

Esta seção descreve as características gerais dos usuários do sistema.

Vendedor:

Nome: Vendedor
Nível: Secundário
Experiência: 3 anos
Habilidade: vendas, computador.

Usuário:

Nome: Usuário
Nível: Variado
Experiência:
Habilidade: internet, compras, computador

4.2 Interfaces

4.2.1 Interfaces de usuário

1. Cadastro de usuário;
2. Cadastro de produtos;
3. Anúncio de produtos;
4. Tela fale conosco;
5. Tela logon;
6. Tela esqueceu a senha;

4.2.2 Interfaces de hardware

Não há interfaces de hardware.

4.2.3 Interfaces de software

1. O sistema faz interface com o banco de dados MySql;

4.2.4 Interfaces de comunicação

1. Os usuários usarão navegadores para fazer a interface com a aplicação;

4.3 Funções do produto

O sistema deixa que os usuários que desejam vender produtos façam seu cadastro, anunciem seus produtos com descrição completa em uma seção relacionada. Qualquer usuário (vendedor ou não) pode procurar por produtos. Qualquer usuário pode procurar os produtos pelo nome, por palavras relacionadas ou por tipo de produto. Qualquer usuário pode procurar por produtos de um usuário que vende produtos. Qualquer usuário pode enviar mensagens aos usuários que possuem os produtos.
Atores: vendedor, usuário.
Objetivos: Vender produtos; Realizar cadastro; Anunciar produtos; Pesquisar produtos; Enviar mensagens;

Lista ator x objetivo:

Ator	Objetivo
Vendedor	Vender produtos
Vendedor	Realizar cadastro
Vendedor	Anunciar produtos
Vendedor	Pesquisar produtos
Vendedor	Enviar mensagens
Usuário	Pesquisar produtos
Usuário	Enviar mensagens

4.4 Suposições e dependências

1. O anúncio do produto deve ter uma data de expiração.

4.5 Requisitos de adaptação de local

1. Possuir um navegador instalado.
2. Possuir um mouse ou outro dispositivo apontador.
3. Possuir uma impressora.

4.6 Requisitos de operação

Não há requisitos de operação.

4.7 Limites de memória

1. O sistema será executado com, no mínimo, 32Mb de memória.

4.8 Distribuição dos requisitos

1. Todas as funcionalidades listadas serão implementadas na primeira versão do sistema.

4.9 Restrições Gerais

1. Não cobre deficiências motoras;
2. Navegadores com versões iguais ou superiores a: Firefox 1.0 e Internet Explorer 5.0

5. Requisitos específicos

Aqui serão descritos os requisitos específicos para o desenvolvimento do software.

5.1 Requisitos Funcionais

5.1.1 Anúncio de produtos

5.1.1.1 Descrição

O sistema provê que os usuários conseguem anunciar produtos. Estes usuários podem incluir, excluir ou atualizar os produtos que eles desejam.

5.1.1.2 Requisitos de interface de entrada

1. Nome do produto;
2. Valor do produto;
3. Condição;
4. Imagem;
5. Descrição geral;

5.1.1.3 Processamento

1. Valor não pode ser negativo.

5.1.1.4 Respostas a situações anormais

1. Esgotado o tempo, não realiza a transação.

5.1.1.5 Requisitos de interface de saída

1. Código do produto;
2. Apresentar tela de erro;
3. Apresentar tela de que a transação não foi realizada, caso não seja.
4. Apresentar tela com dados do produto quando for atualização;
5. Apresentar uma tela informando sucesso quando finaliza uma transação;

5.1.1.6 Regras de negócio específicas

Não tem regras de negócio.

5.1.1.7 Requisitos não-funcionais específicos

1. Um código pode apenas ser de um produto.

5.1.2 Pesquisa de produtos

5.1.2.1 Descrição

A pesquisa de produtos é permitida para os usuários e vendedores.

5.1.2.2 Requisitos de interface de entrada

1. Nome do produto.
2. Apresentar lista com código e nome do produto para o usuário selecionar;

5.1.2.3 Processamento

1. Para pesquisar, o produto tem que existir.

5.1.2.4 Respostas a situações anormais

Não há respostas a situações anormais.

5.1.2.5 Requisitos de interface de saída

1. Lista de produtos;

2. Apresentar tela com dados do produto: valor, condição, descrição.
3. Produto não existe = apresenta tela dando erro e volta pra pesquisa.

5.1.2.6 Regras de negócio específicas

Não há regras de negócio específicas.

5.1.2.7 Requisitos não-funcionais específicos

Não há requisitos não-funcionais específicos.

5.2 Requisitos de performance

1. 20 segundos para consulta.

5.3 Limites de desenvolvimento

1. Compatibilidade com ML versão 0.5;
2. Aproveitar funções do ML versão 0.5;
3. Utilizar PHP e MySql;

5.4 Requisitos de banco de dados lógico

Não há requisitos de banco de dados lógico.

5.5 Regras de negócio do sistema

Não há regras de negócio para o sistema.

5.6 Requisitos não funcionais do sistema

Não há requisitos não funcionais do sistema.

5.7 Atributos do sistema de software

Não há atributos do sistema.

5.7.1 Segurança

1. Autenticação usando: “ML SECURITY”.

5.7.2 Manutenibilidade

Não há requisitos de manutenibilidade.

5.7.3 Confiabilidade

Não há requisitos de confiabilidade.

5.7.4 Portabilidade

Não há requisitos de portabilidade.

5.7.5 Reusabilidade

Não há requisitos de reusabilidade.

5.7.6 Testabilidade

Não há requisitos de testabilidade.

5.7.7 Disponibilidade

Não há requisitos de disponibilidade.

5.7.8 Eficiência

Não há requisitos de eficiência.

5.7.9 Flexibilidade

1. Há 100% de flexibilidade.

5.7.10 Interoperabilidade

Não há requisitos de interoperabilidade.

5.7.11 Robustez

1. Há suporte a falhas.

5.7.12 Usabilidade

Não há requisitos de usabilidade.

6. Informações de suporte

Indica a versão do template e outras informações necessárias.

6.1 Tabela de conteúdo

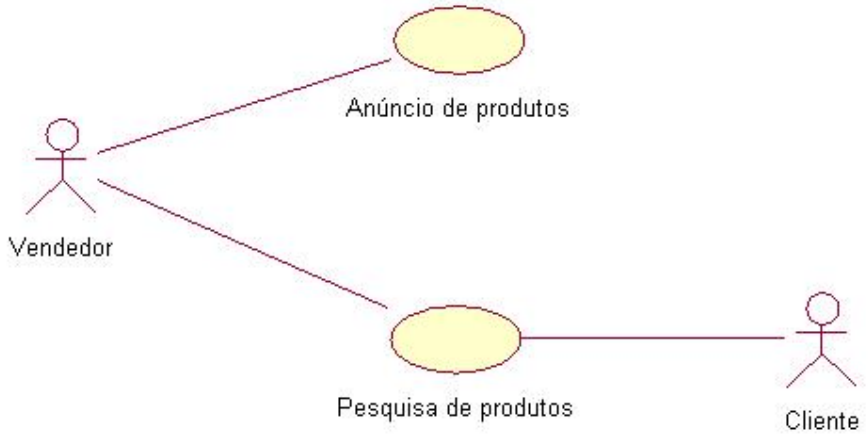
Versão	Data	Descrição	Nome
1	01/05/2006	O *template* da SRS foi adaptado como o solicitado na disciplina	Carla Dias.

6.2 Apêndices

Não há necessidade de apêndices.

MODELO DE CASOS DE USO DO PROJETO ML

1. DIAGRAMA DE CASOS DE USO DO SISTEMA



2. DESCRIÇÃO DOS CASOS DE USO

Caso de Uso: 01
Nome: Anúncio de produtos
Ator Principal: Vendedor
Stakeholders: não há
Pré-condições: Vendedor logado.
Pós-condições: Produto anunciado.
Garantias Mínimas: Se o caso de uso não terminar, interromper anúncio.
Acionador: Vendedor.
Cenário Principal:

1. O vendedor clica em logar-se.
2. O sistema pede usuário e senha.
3. O vendedor digita usuário e senha.
4. O vendedor clica em: cadastrar produto, atualizar produto ou excluir produto.

Se clicado em cadastrar produto:

5. O vendedor digita: valor, condição, imagem e descrição do produto.
6. O sistema atualiza os dados digitados pelo vendedor.
7. O sistema mostra uma tela com o código do produto.

Se clicado em atualizar produto:

8. O vendedor digita o código ou o nome do produto ou seleciona o produto de uma lista.
9. O sistema mostra uma tela com os dados do produto escolhido.
10. O vendedor atualiza o produto e clica em atualizar.
11. O sistema atualiza os dados do produto.
12. O sistema mostra uma tela com o código e o nome do produto atualizado.

Se clicado em excluir produto:

13. O vendedor digita o código ou o nome do produto para excluir ou seleciona o produto de uma lista.
14. O sistema mostra uma tela com os dados do produto.
15. O vendedor clica em excluir produtos.
16. O sistema exclui o produto.
17. O sistema mostra uma tela com o código e o nome do produto.
18. Fim do caso de uso.

Exceções:

1. Se o valor do produto for negativo, pedir outro valor.
2. Se der erro no momento da inclusão, não incluir.
3. Se der erro no momento da atualização, não atualizar.
4. Se der erro no momento da exclusão, não excluir.

Informações Relacionadas:

1. O tempo de realização das consultas é de no máximo 20 segundos.

Caso de Uso: 02
Nome: Pesquisa de produtos
Ator Principal: Vendedor
Stakeholders: Usuário
Pré-condições: Vendedor logado.
Pós-condições: Pesquisa realizada.
Garantias Mínimas: Se o caso de uso não terminar, interromper pesquisa.
Acionador: Vendedor.
Cenário Principal:

1. O vendedor clica em logar-se.
2. O sistema pede usuário e senha.
3. O vendedor digita usuário e senha.
4. O vendedor clica em: pesquisar produtos.
5. O vendedor digita código ou nome do produto;
6. O sistema mostra uma tela com a lista dos produtos.
7. O vendedor seleciona o produto.
8. O sistema mostra uma tela com os dados do produto.
9. Fim do caso de uso.

Exceções:

1. Se o produto não existir o sistema mostra uma tela de erro.

Informações Relacionadas:

2. O tempo de realização das consultas é de no máximo 20 segundos.

APÊNDICE IV – ESPECIFICAÇÃO DE REQUISITOS E MODELO DE CASO DE USO DO PROJETO ML, REESCRITO NO PADRÃO PROPOSTO NESTA DISSERTAÇÃO

Especificação de Requisitos do Projeto ML

SRS

Tabela de Conteúdo

1. Histórico de aprovação do documento

Nome	Data
José da Silva	01/01/2006
Pedro Paulo Souza	10/01/2006
Angelo Soares	17/03/2006

2. Histórico de revisão do documento

Data	Descrição
01/01/2006	Primeira versão.

3. Introdução

3.1 Propósito do documento

Os propósitos deste documento de requisitos são:

- Desenvolver e especificar as necessidades da comunidade que devem ser atendidas pelo projeto ML, bem como definir para os desenvolvedores e colaboradores o produto a ser feito.

3.2 Escopo do documento

Esta SRS contém os requisitos para o projeto ML, baseado nas necessidades do cliente.

3.3 Acrônimos e abreviaturas

Os seguintes acrônimos e termos serão usados neste documento e durante todo o projeto ML.

Sigla	Definição
ML	Minha Loja
LI	Loja na Internet

3.4 Referências

Referência	Descrição
[IEE98]	IEEE, Institute. “IEEE Std 830-1998. IEEE Recommended Practice for Software Requirements Specifications”. New York: Institute of Electrical and Electronic Engineers. Inc,. 1998.

3.5 Visão Geral

Este documento fornece informação sobre os requisitos a serem implementados no projeto ML, definidos pelo grupo de alunos da cadeira de Engenharia de Software.

4. Descrição Geral

4.1 Características do usuário

Esta seção descreve as características gerais dos usuários do sistema.

Vendedor
Nome: Vendedor **Nível:** Secundário **Experiência:** 3 anos **Habilidade:** Vendas, computador.

Usuário
Nome: Usuário **Nível:** Variado **Experiência:** - **Habilidade:** Internet, compras, computador

4.2 Interfaces

4.2.1 Interfaces de usuário

1. O sistema deve conter a tela “Cadastro de usuário”;
2. O sistema deve conter a tela “Cadastro de produtos”;
3. O sistema deve conter a tela “Anúncio de produtos”;
4. O sistema deve conter a tela “Fale conosco”;
5. O sistema deve conter a tela “Logar-se”;
6. O sistema deve conter a tela “Esqueci minha senha”;

4.2.2 Interfaces de hardware

<não aplicável>

4.2.3 Interfaces de software

2. O sistema faz interface com o banco de dados MySql;

4.2.4 Interfaces de comunicação

3. Os usuários usarão navegadores para fazer a interface com a aplicação;

4.3 Funções do produto

O sistema deixa que os usuários que desejam vender produtos façam seu cadastro, anunciem seus produtos com descrição completa em uma seção relacionada. Qualquer usuário (vendedor ou não) pode procurar por produtos. Qualquer usuário pode procurar os produtos pelo nome, por palavras

relacionadas ou por tipo de produto. Qualquer usuário pode procurar por produtos de um usuário que vende produtos. Qualquer usuário pode enviar mensagens aos usuários que possuem os produtos.

Atores: vendedor, usuário.

Objetivos: Vender produtos; Realizar cadastro; Anunciar produtos; Pesquisar produtos; Enviar mensagens;

Lista ator x objetivo:

Ator	Objetivo	Tipo
Vendedor	Vender produtos	Stakeholder ator
Vendedor	Realizar cadastro	Stakeholder ator
Vendedor	Anunciar produtos	Stakeholder ator
Vendedor	Pesquisar produtos	Stakeholder ator
Vendedor	Enviar mensagens	Stakeholder interessado
Usuário	Pesquisar produtos	Stakeholder interessado
Usuário	Enviar mensagens	Stakeholder ator
Administrador	Anunciar produtos	Stakeholder interessado

4.4 Suposições e dependências

2. O anúncio do produto deve ter uma data de expiração.

4.5 Requisitos de adaptação de local

3. O sistema do usuário deve possuir um navegador instalado.
4. O sistema do usuário deve possuir um mouse ou outro dispositivo apontador.
5. O sistema do usuário deve possuir uma impressora.

4.6 Requisitos de operação

<não se aplica>

4.7 Limites de memória

6. O sistema será executado com, no mínimo, 32Mb de memória.

4.8 Distribuição dos requisitos

2. Todas as funcionalidades listadas serão implementadas na primeira versão do sistema.

4.9 Restrições Gerais

3. O sistema não cobre deficiências motoras;
4. O sistema suporta navegadores com versões iguais ou superiores a: Firefox 1.0 e Internet Explorer 5.0

5. Requisitos específicos

Aqui serão descritos os requisitos específicos para o desenvolvimento do software.

5.1 Requisitos Funcionais

5.1.1 Anunciar Produtos

5.1.1.1 Descrição

O vendedor deve ser capaz de incluir, atualizar e excluir produtos de sua lista.

Requisitos Funcionais Relacionados:

1. O vendedor deve ser capaz de incluir produtos em sua lista.
2. O vendedor deve ser capaz de atualizar produtos em sua lista.
3. O vendedor deve ser capaz de excluir produtos de sua lista.

5.1.1.2 Requisitos de interface de entrada

6. O sistema deve solicitar o tipo de transação a ser realizada, permitindo ao usuário selecionar de uma lista de valores dentre os seguintes: inclusão, atualização e exclusão.

5.1.1.3 Processamento

<não se aplica>

5.1.1.4 Respostas a situações anormais

<não se aplica>

5.1.1.5 Requisitos de interface de saída

<não se aplica>

5.1.1.6 Regras de negócio específicas

<não se aplica>

5.1.1.7 Requisitos não-funcionais específicos

<não se aplica>

5.1.2 Incluir Produtos

5.1.2.1 Descrição

O vendedor deve ser capaz de incluir produtos em sua lista.

5.1.2.2 Requisitos de interface de entrada

1. O sistema deve obter o nome do produto através de um campo de texto. Este campo é obrigatório. O tamanho máximo do campo é de 30 caracteres.
2. O sistema deve obter o valor do produto através de um campo de texto. Este item é obrigatório.

3. O sistema deve obter a condição do produto, permitindo ao usuário selecionar de uma lista de valores existentes. Apenas um valor pode ser selecionado. Este item é obrigatório. Os valores de condição são: usado, novo.
4. O sistema deve obter uma imagem do produto. O tamanho do arquivo deve ser, no máximo, de 500kb.
5. O sistema deve obter uma descrição geral sobre o produto através de um campo de texto. Este item é obrigatório.

5.1.2.3 Processamento

2. Com relação a requisitos de interface de entrada (3.1.1.2), **item 2**:

2. O sistema deve verificar se o valor do produto é um valor acima de zero.

5.1.2.4 Respostas a situações anormais

3. Com relação a requisitos de interface de entrada (3.1.1.2), **item 2**:

2. TimeOut no momento de inserção no Banco de Dados:
   2.1. O sistema deve desfazer qualquer transação já realizada desde o início da inserção;

5.1.2.5 Requisitos de interface de saída

4. Com relação a requisitos de interface de entrada (3.1.1.2):

6. Se a inclusão for realizada com sucesso, o sistema deve retornar o código do produto cadastrado.

5. Com relação aos requisitos de processamento (3.1.1.3):

Com relação ao item **1**:

1. Se o valor informado for um valor abaixo de zero, o sistema deve retornar um erro e informá-lo ao usuário.

6. Com relação a respostas a situações anormais (3.1.1.4):

Com relação ao item **1**:

2. O sistema deve informar ao usuário que a transação não foi realizada.

5.1.2.6 Regras de negócio específicas

<não se aplica>

5.1.2.7 Requisitos não-funcionais específicos

2. O sistema deve possuir um código único para identificação dos produtos.

5.1.3 Atualizar Produtos

5.1.3.1 Descrição

O vendedor deve ser capaz de atualizar produtos de sua lista.

5.1.3.2 Requisitos de interface de entrada

1. O sistema deve obter o nome ou o código do produto, permitindo ao usuário selecionar através de uma lista de valores existentes.

5.1.3.3 Processamento

<não se aplica>

5.1.3.4 Respostas a situações anormais

1. TimeOut no momento de atualização no Banco de Dados:
   1.1. Desfaz qualquer transação já realizada desde o inicio da operação;

5.1.3.5 Requisitos de interface de saída

7. Com relação a requisitos de interface de entrada (3.1.1.2):

Com relação ao item **1**:

1. O sistema fornece os dados do produto selecionado.

2. Se a atualização for realizada com sucesso, o sistema deve retornar o código e o nome do produto atualizado.
3. Se a atualização for realizada com sucesso, o sistema deve retornar os campos atualizados.

8. Com relação a respostas a situações anormais (3.1.1.4):

Com relação ao item **1**:

1. O sistema deve informar ao usuário que a transação não foi realizada.

5.1.3.6 Regras de negócio específicas

<não se aplica>

5.1.3.7 Requisitos não-funcionais específicos

<não se aplica>

5.1.4 Excluir produtos

5.1.4.1 Descrição

O vendedor deve ser capaz de excluir produtos de sua lista.

5.1.4.2 Requisitos de interface de entrada

1. O sistema deve obter o nome ou o código do produto, permitindo ao usuário selecionar através de uma lista de valores existentes.

5.1.4.3 Processamento

<não se aplica>

5.1.4.4 Respostas a situações anormais

9. Com relação a requisitos de interface de entrada (3.1.1.2), **item 2**:

1. TimeOut no momento de atualização no Banco de Dados:
   1.1. Desfaz qualquer transação já realizada desde o inicio da operação;

5.1.4.5 Requisitos de interface de saída

10. Com relação a requisitos de interface de entrada (3.1.1.2):

1. Se a exclusão for realizada com sucesso, o sistema deve informar ao usuário o nome e o código do produto excluído.

11. Com relação a respostas a situações anormais (3.1.1.4):

Com relação ao item **1**:

1. O sistema deve informar ao usuário que a transação não foi realizada.

5.1.4.6 Regras de negócio específicas

<não se aplica>

5.1.4.7 Requisitos não-funcionais específicos

3. O sistema deve possuir um código único para identificação dos produtos.

5.1.5 Pesquisar produtos

5.1.5.1 Introdução

O vendedor deve ser capaz de pesquisar produtos.

5.1.5.2 Requisitos de interface de entrada
3. O sistema deve obter o nome do produto.
4. O sistema deve obter a escolha do produto desejado, permitindo ao usuário selecionar de uma lista de produtos.

5.1.5.3 Processamento

12. Com relação a requisitos de interface de entrada (3.1.1.2), **item 1**:

2. O sistema deve verificar se o produto é existente no sistema.

5.1.5.4 Respostas a situações anormais

<não se aplica>

5.1.5.5 Requisitos de interface de saída

13. Com relação a requisitos de interface de entrada (3.1.1.2):

4. O sistema deve fornecer uma lista com os produtos encontrados.
5. O sistema deve fornecer os dados do produto selecionado. Os dados são: valor do produto, condição do produto e descrição geral do produto.

14.Com relação aos requisitos de processamento (3.1.1.3):

Com relação ao item **1**:

1. Se o produto é inexistente, o sistema deve informar ao usuário e solicitar nova pesquisa.

5.1.5.6 Regras de negócio específicas

<não se aplica>

5.1.5.7 Requisitos não-funcionais específicos

<não se aplica>

5.2 Requisitos de performance

1. O tempo de realização de qualquer consulta é no máximo 20 segundos.

5.3 Limites de desenvolvimento

4. O sistema deve ser compatível com ML versão 0.5;
5. O sistema deve usar todas as funções do ML versão 0.5;
6. O sistema deve ser implementado em PHP e MySQL;

5.4 Requisitos de banco de dados lógico

<não se aplica>

5.5 Regras de negócio do sistema

<não se aplica>

5.6 Requisitos não funcionais do sistema

<não se aplica>

5.7 Atributos do sistema de software

<não se aplica>

5.7.1 Segurança

2. O sistema deve autenticar o usuário usando o sistema: “ML SECURITY”.

5.7.2 Manutenibilidade

<não se aplica>

5.7.3 Confiabilidade

<não se aplica>

5.7.4 Portabilidade

<não se aplica>

5.7.5 Reusabilidade

<não se aplica>

5.7.6 Testabilidade

<não se aplica>

5.7.7 Disponibilidade

<não se aplica>

5.7.8 Eficiência
<não se aplica>

5.7.9 Flexibilidade

1. O sistema fornece 100% de flexibilidade.

5.7.10 Interoperabilidade

<não se aplica>

5.7.11 Robustez

1. O sistema fornece suporte a falhas.

5.7.12 Usabilidade

<não se aplica>

6. Informações de suporte

Indica a versão do template e outras informações necessárias.

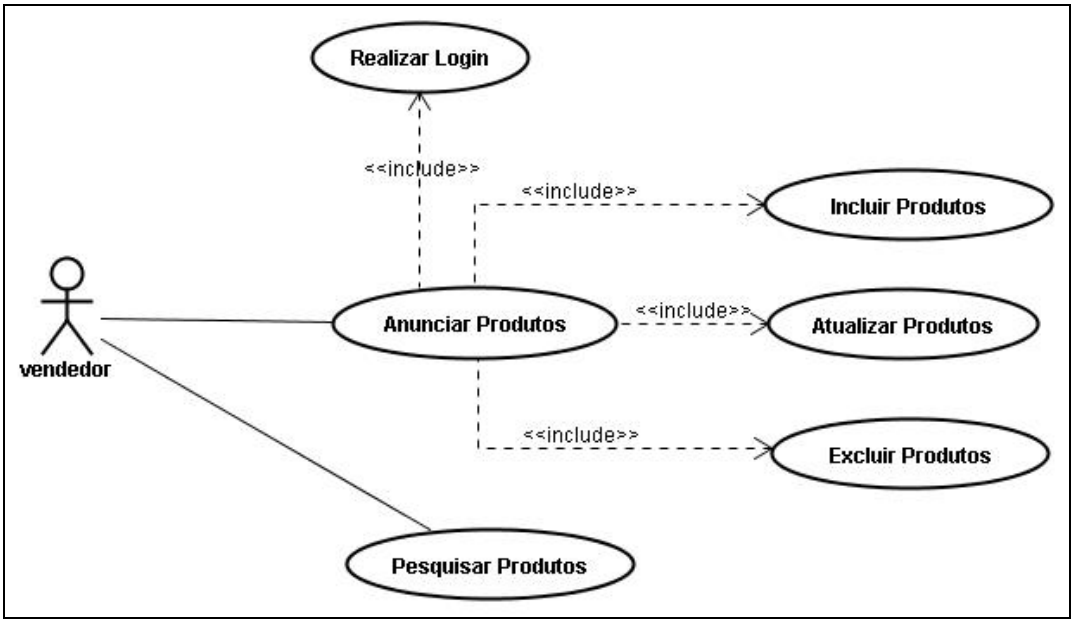
6.1 Tabela de conteúdo

Versão	Data	Descrição	Nome
1	01/05/2006	O *template* da SRS foi adaptado como o solicitado na disciplina	Carla Dias.

6.2 Apêndices
<não se aplica>

MODELO DE CASOS DE USO DO SISTEMA ML

1. DIAGRAMA DE CASOS DE USO DO SISTEMA



2. DESCRIÇÃO DOS CASOS DE USO

Caso de uso 01: Anunciar Produtos

NOME	IDUC
Anunciar produtos	01
CONTEXTO	
O vendedor deve ser capaz de incluir, atualizar e excluir produtos de sua lista.	
NÍVEL	
Objetivo de Usuário	
ATOR	
Vendedor	
STAKEHOLDERS	
-	
PRÉ-CONDIÇÕES	
1. O vendedor deve estar cadastrado no sistema.	
GARANTIAS MÍNIMAS	
Caso o caso de uso seja interrompido antes do término, qualquer transação realizada deve ser desfeita.	

GARANTIAS DE SUCESSO
Produto anunciado com sucesso.
ACIONADOR
Vendedor solicita o anúncio de produtos
CENÁRIO DE SUCESSO PRINCIPAL
PASSOS
1 Login (include Login) 2 O sistema solicita o tipo de transação a ser realizada dentre as seguintes: Incluir produtos (include Incluir Produtos), Atualizar produtos (include Atualizar Produtos), Excluir produtos (include Excluir Produtos); 3 O vendedor realiza transações até optar pelo término. 4 O sistema finaliza o anúncio de produtos.
FLUXO ALTERNATIVO
PASSOS
INFORMAÇÕES RELACIONADAS
1 O produto deve possuir um código único de identificação. 2. O tempo de realização de consultas é de no máximo 20 segundos.

Caso de uso 02: Realizar login

NOME	IDUC
Realizar login	02
CONTEXTO	
O usuário deve ser capaz de realizar login no sistema.	
NÍVEL	
Objetivo de Usuário	
ATOR	
Ator	
STAKEHOLDERS	
-	
PRÉ-CONDIÇÕES	
O usuário deve estar cadastrado no sistema.	
GARANTIAS MÍNIMAS	
GARANTIAS DE SUCESSO	
Usuário logado com sucesso.	

ACIONADOR
1. Usuário opta por logar no sistema; 2. O caso de uso é acionado pelo caso de uso base.

CENÁRIO DE SUCESSO PRINCIPAL	
PASSOS	
1	O sistema solicita o nome do usuário.
2	O sistema solicita a senha do usuário.
3	O sistema valida usuário e senha.
4	O caso de uso é finalizado.

FLUXO ALTERNATIVO	
PASSOS	
3.1	Senha inválida: sistema informa ao usuário e finaliza o caso de uso.

INFORMAÇÕES RELACIONADAS
1. O tempo de realização de consultas é de no máximo 20 segundos.

Caso de uso 03: Incluir Produtos

NOME	IDUC
Incluir Produtos	03

CONTEXTO
O vendedor deve ser capaz de incluir produtos em sua lista.

NÍVEL
Objetivo de Usuário

ATOR
Vendedor

STAKEHOLDERS
-

PRÉ-CONDIÇÕES
1. O vendedor deve estar logado no sistema.

GARANTIAS MÍNIMAS
Caso o caso de uso seja interrompido antes do término, qualquer transação realizada deve ser desfeita.

GARANTIAS DE SUCESSO
Produto incluído com sucesso.

ACIONADOR
1. Vendedor solicita a inclusão de produtos 2. O caso de uso é acionado pelo caso de uso base.

CENÁRIO DE SUCESSO PRINCIPAL

PASSOS

1	O sistema solicita o nome do produto.
2	O sistema solicita o valor do produto.
3	O sistema solicita a condição do produto.
4	O sistema solicita upload da imagem do produto.
5	O sistema solicita a descrição do produto.
6	O sistema atualiza os dados do produto.
7	O sistema fornece o código do produto cadastrado.
8	O caso de uso é finalizado.

FLUXO ALTERNATIVO

PASSOS

2.1	Valor negativo: o sistema informa ao usuário e solicita novo valor.
6.1	Se der erro no momento da atualização: o sistema desfaz qualquer transação realizada.
*	Caso o caso de uso seja encerrado antes do término, o sistema deve informar ao usuário que a transação não foi realizada.

INFORMAÇÕES RELACIONADAS

1 O produto deve possuir um código único de identificação.

Caso de uso 04: Atualizar Produtos

NOME	IDUC
Atualizar Produtos	04

CONTEXTO

O vendedor deve ser capaz de atualizar produtos de sua lista.

NÍVEL

Objetivo de Usuário

ATOR

Vendedor

STAKEHOLDERS

-

PRÉ-CONDIÇÕES

1. O vendedor deve estar logado no sistema.

GARANTIAS MÍNIMAS

Caso o caso de uso seja interrompido antes do término, qualquer transação realizada deve ser desfeita.

GARANTIAS DE SUCESSO

Produto atualizado com sucesso.

ACIONADOR

1. Vendedor solicita a atualização de produtos
2. O caso de uso é acionado pelo caso de uso base.

CENÁRIO DE SUCESSO PRINCIPAL	
PASSOS	
1	O sistema solicita o nome ou o código do produto.
2	O vendedor fornece o nome ou o código do produto.
3	O sistema apresenta os dados do produto escolhido.
4	O cliente realiza as atualizações necessárias.
5	O sistema atualiza o cadastro.
6	O sistema apresenta o código e o nome do produto atualizado.
7	O sistema apresenta os dados atualizados do produto.
8	O caso de uso é finalizado.
FLUXO ALTERNATIVO	
PASSOS	
5.1	TimeOut no momento da atualização: O sistema deve desfazer qualquer transação realizada.
*	Caso o caso de uso seja encerrado antes do término, o sistema deve informar ao usuário que a transação não foi realizada.
INFORMAÇÕES RELACIONADAS	
1 O tempo de realização de consultas é de no máximo 20 segundos.	

Caso de uso 05: Excluir Produtos

NOME	IDUC
Excluir Produtos	05
CONTEXTO	
O vendedor deve ser capaz de excluir produtos de sua lista.	
NÍVEL	
Objetivo de Usuário	
ATOR	
Vendedor	
STAKEHOLDERS	
-	
PRÉ-CONDIÇÕES	
1. O vendedor deve estar logado no sistema.	
GARANTIAS MÍNIMAS	
Caso o caso de uso seja interrompido antes do término, qualquer transação realizada deve ser desfeita.	
GARANTIAS DE SUCESSO	
Produto excluído com sucesso.	
ACIONADOR	
1. Vendedor solicita a exclusão de produtos	

2. O caso de uso é acionado pelo caso de uso base.

CENÁRIO DE SUCESSO PRINCIPAL	
PASSOS	
1	O sistema solicita o nome ou o código do produto a ser excluído.
2	O vendedor fornece o nome ou o código do produto a ser excluído.
3	O sistema apresenta os dados do produto escolhido.
4	O sistema solicita a confirmação da exclusão.
5	O vendedor confirma a exclusão.
6	O sistema realiza a exclusão do produto.
7	O sistema apresenta o código e nome do produto excluído.
8	O caso de uso é finalizado.

FLUXO ALTERNATIVO	
PASSOS	
6.1	TimeOut no momento da exclusão: O sistema deve desfazer qualquer transação realizada.
*	Caso o caso de uso seja encerrado antes do término, o sistema deve informar ao usuário que a transação não foi realizada.

INFORMAÇÕES RELACIONADAS
1 O tempo de realização de consultas é de no máximo 20 segundos.

APÊNDICE V – ARTIGOS PUBLICADOS

Com o objetivo de submeter os resultados desta pesquisa à apreciação da comunidade científica, foram desenvolvidos durante o ano de 2006 dois artigos para um evento relacionado à área de engenharia de software. Dos dois artigos submetidos, os dois foram aceitos e publicados.

CERRI, Elisa Cerri e; ROCHA, Fabiana Z. F.; BASTOS, Ricardo Melo. "Um modelo de rastreabilidade entre o documento de especificação de requisitos e o modelo de casos de uso". In: III Simpósio Brasileiro de Sistemas de Informação, 2006, Curitiba.

ROCHA, Fabiana Z. F.; CERRI, Elisa Cerri e; BASTOS, Ricardo Melo; YAMAGUTI, Marcelo H. "Uma Proposta de Modelo para Avaliação da Qualidade da Tradução de Requisitos para Casos de Uso". In: III Simpósio Brasileiro de Sistemas de Informação, 2006, Curitiba.

Esta pesquisa foi parcialmente financiada pelo Convênio Dell/PUCRS, através da Lei de Informática Brasileira (Lei nº. 8.248/91).

APÊNDICE VI – PROTÓTIPO PARA APOIO AUTOMATIZADO AO PROCESSO DE RASTREABILIDADE

Este apêndice apresenta a descrição de um protótipo para apoio automatizado ao processo de rastreabilidade desenvolvido no contexto desta pesquisa.

1 Descrição do Protótipo

Visando auxiliar a aplicabilidade do modelo e do processo de rastreabilidade propostos neste trabalho um protótipo de software foi desenvolvido.

A implementação iniciou com a escolha da arquitetura e das linguagens para a implementação do protótipo. Foi definido que a persistência seria feita em um banco de dados, com a entrada de dados feita através de uma interface gráfica. Assim, optou-se pela arquitetura em três camadas, no modelo MVC (Model View Control). Java foi a linguagem de implementação utilizada para as camadas de dados, controle e interface. Já para a camada de persistência, que dá suporte a camada de dados, foi utilizado o banco de dados Oracle 10g. Estas escolhas foram feitas por sua grande utilização tanto no meio acadêmico como no meio industrial, por experiências de sucesso alcançadas anteriormente, pela disponibilidade das ferramentas e, também, pela facilidade de integração entre o banco e a liguagem definida.

2 Detalhamento do Protótipo

Nesta seção serão descritos os procedimentos de entrada para as instâncias de algumas das classes dos modelos conceituais implementados na ferramenta. Este processo será guiado pela apresentação das interfaces gráficas e seu comportamento durante o procedimento em questão. Também serão discutidos os impactos dos procedimentos e seus relacionamentos.

É importante ressaltar que nesta ferramenta, um projeto é responsável por armazenar suas SRS's relacionadas a seus Casos de Uso.

2.1.1 Gerenciamento de projetos

Para a inclusão de um projeto deve-se acessar o menu Arquivo, seu sub-menu Incluir e em seguida selecionar a opção Projeto, conforme mostra Figura 1.

Figura 1 – Arquivo -> Incluir -> Projeto

Em seguida será apresentada uma tela requisitando a entrada do nome para o novo projeto, conforme Figura 2. Assim, o novo projeto será exibido juntamente com os demais, se eles existirem. Todos os projetos já cadastrados serão visualizados na aba projetos, presente no canto esquerdo da tela, conforme Figura 3.

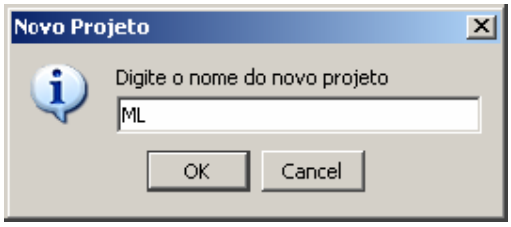


Figura 2 – Novo Projeto

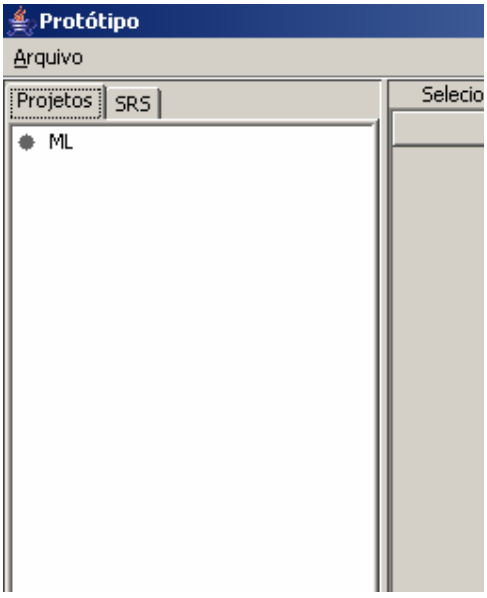


Figura 3 – A aba projetos

A alteração possível trata da modificação dos arquivos referentes a estes projetos. Pode-se visualizar as SRSs cadastradas para cada projeto, selecionando um deles na aba Projetos. Com isso a aba da SRS é acionada e mostrará, caso existam, os arquivos cadastrados para este projeto, como mostra Figura 4.

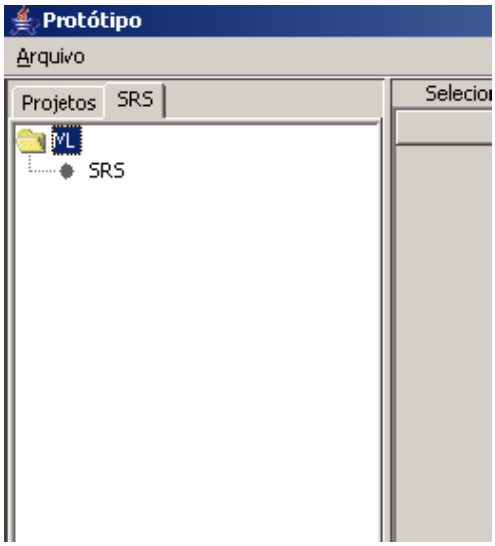


Figura 4 – Alteração de projetos

Um projeto pode ser excluído através do menu Arquivo e sub-menu Remover, conforme Figura 5. Assim, o sistema solicitará o nome do projeto a ser excluído, Figura 6. Após a entrada de dados, que segue o mesmo padrão da inclusão de projetos, será exibida uma tela de confirmação sobre a ação executada, conforme Figura 7.

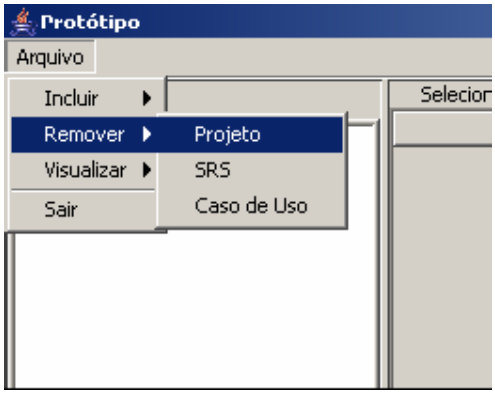


Figura 5 – Exclusão de projetos através do menu Arquivo

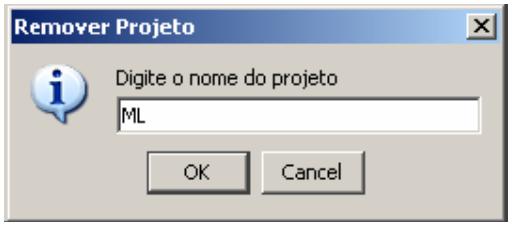


Figura 6 – Exluir projeto

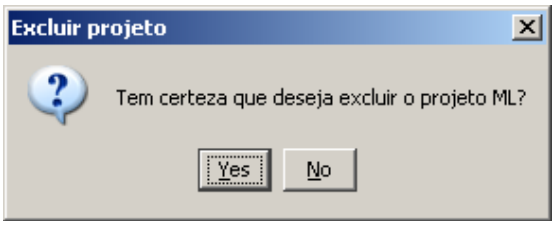


Figura 7 – Confirmação de exclusão de projetos

2.1.2 Gerenciamento de SRSs

Para que uma SRS possa ser criada e manipulada, ao menos um projeto deve estar previamente cadastrado, já que uma SRS estará sempre ligada a um projeto. Assim, tendo um projeto cadastrado pode-se realizar o gerenciamento dos arquivos.

Para a inclusão de SRSs deve-se acessar o menu Arquivo, seu sub-menu Incluir e, em seguida, a opção SRS, como mostra Figura 8.

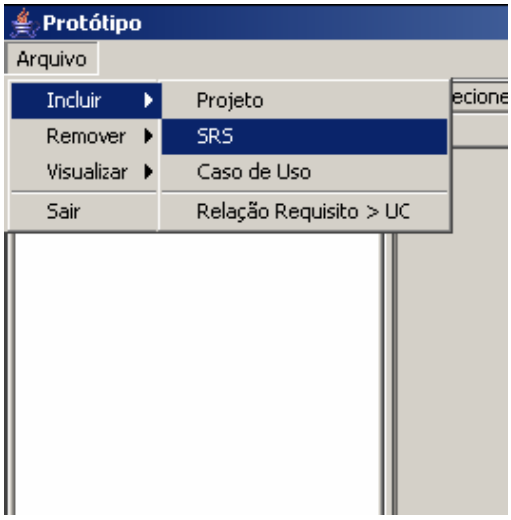


Figura 8 – Opção de inclusão de SRS

Logo após será exibida uma tela com a lista de todos os projetos cadastrados. A partir desta lista um dos projetos deve ser selecionado, a fim de incluir a SRS relacionada, e o campo “Nome” deve ser preenchido. É importante considerar que uma SRS só será cadastrada se ambos os procedimentos forem executados. Este procedimento pode ser visualizado na Figura 9.

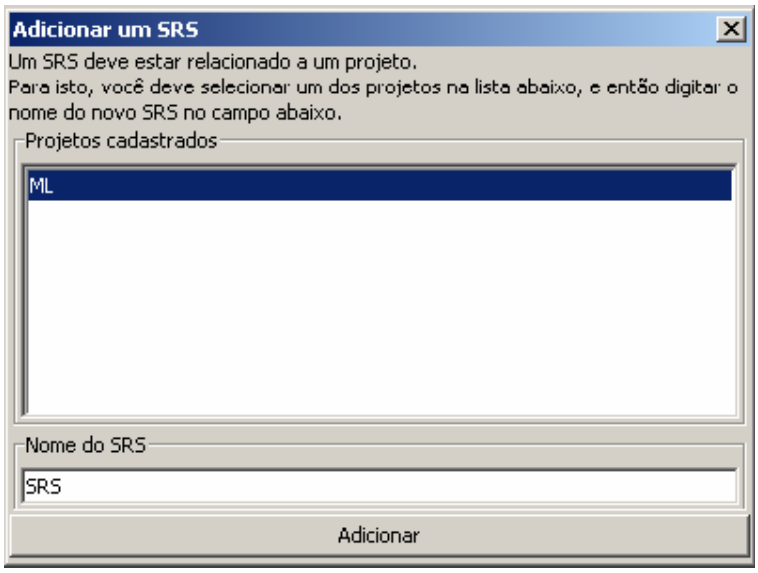


Figura 9 – Adicionar SRS

Para preencher os campos desta SRS, basta selecioná-la da mesma maneira como apresentado na Alteração de Projetos (Figura 4). Assim a tela, mostrada na Figura 10, será apresentada.

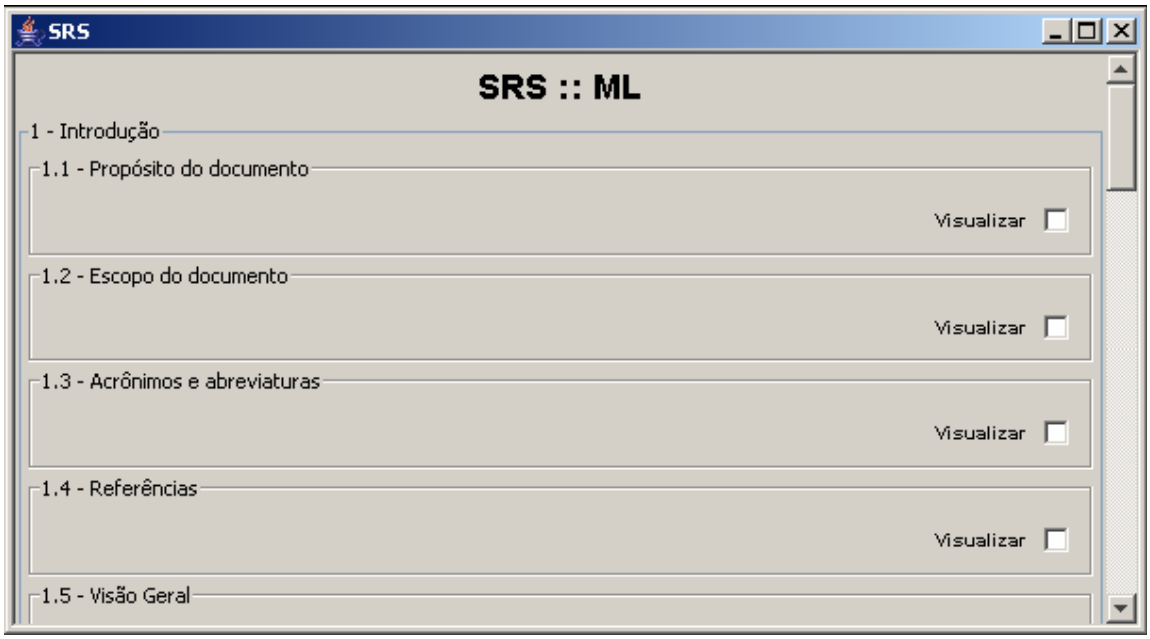


Figura 10 – Seções Introdução da SRS.

Para a edição dos elementos da SRS, seleciona-se a opção Visualizar e a seção será expandida, mostrando os itens a serem preenchidos. A Figura 11 mostra o exemplo da seção Referências e a Figura 12 mostra o exemplo desta mesma seção sendo editada (através do botão Editar).

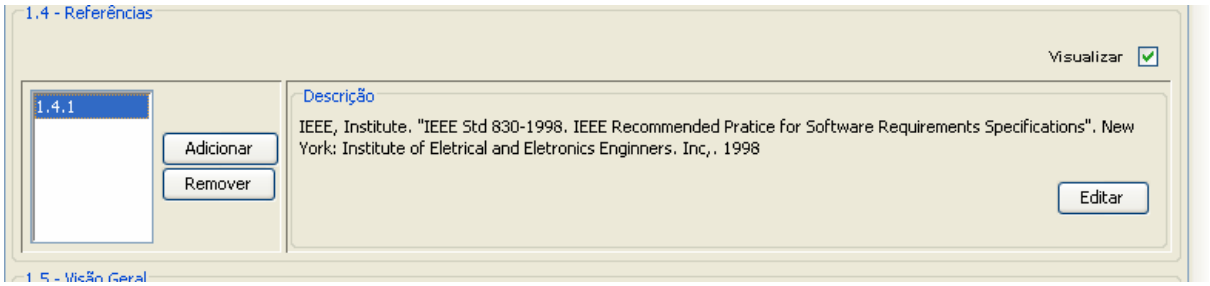


Figura 11 – Exemplo da seção Referências.

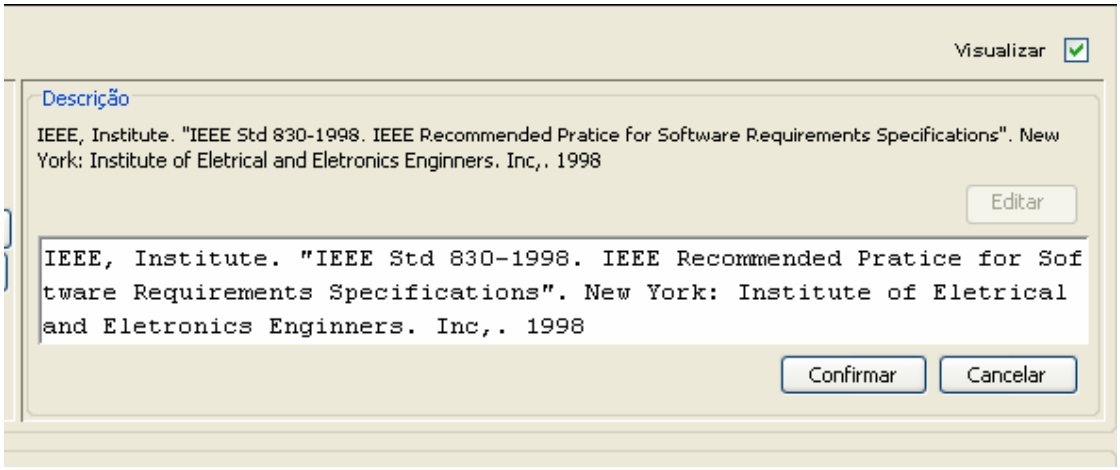


Figura 12 – Exemplo da edição na seção Referências.

Os procedimentos de exclusão de uma SRSs são idênticos aos procedimentos de exclusão de um projeto. A única diferença se refere a tela de confirmação de exclusão, como visto na Figura 13.

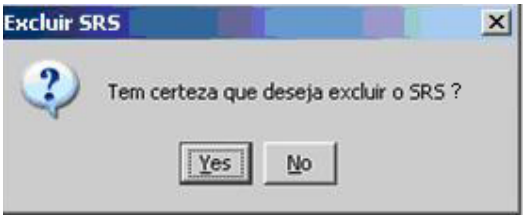


Figura 13 – Tela de confirmação de exclusão de uma SRS.

Após terem sido inseridas as SRSs do projeto, pode-se criar os casos de uso relacionados aos requisitos presentes nestes documentos. Um caso de uso sempre estará ligado a uma SRS, sendo assim, para que possamos criá-los e manipulá-los ao menos uma SRS deve ter sido previamente criada. Na próxima seção o gerencimento de casos de uso é detalhado.

2.1.3 Gerenciamento de Casos de Uso

Como explicado anteriormente, um caso de uso estará ligado a uma SRS. Para incluir um caso de uso segue-se o seguinte procedimento: acesso ao menu Arquivo, acesso ao sub-menu Incluir e em seguida acesso à opção Caso de Uso, conforme é mostrado na Figura 14.

Figura 14 – Inclusão de Casos de Uso

Como o objetivo é inserir um caso de uso, que deve sempre estar ligado a uma SRS, quando a operação acima for realizada será apresentada uma tela, contendo uma lista com todos os projetos cadastrados, conforme Figura 15. A partir desta lista seleciona-se um dos projetos e uma SRS. Um caso de uso apenas será inserido se ambas as operações forem executadas.

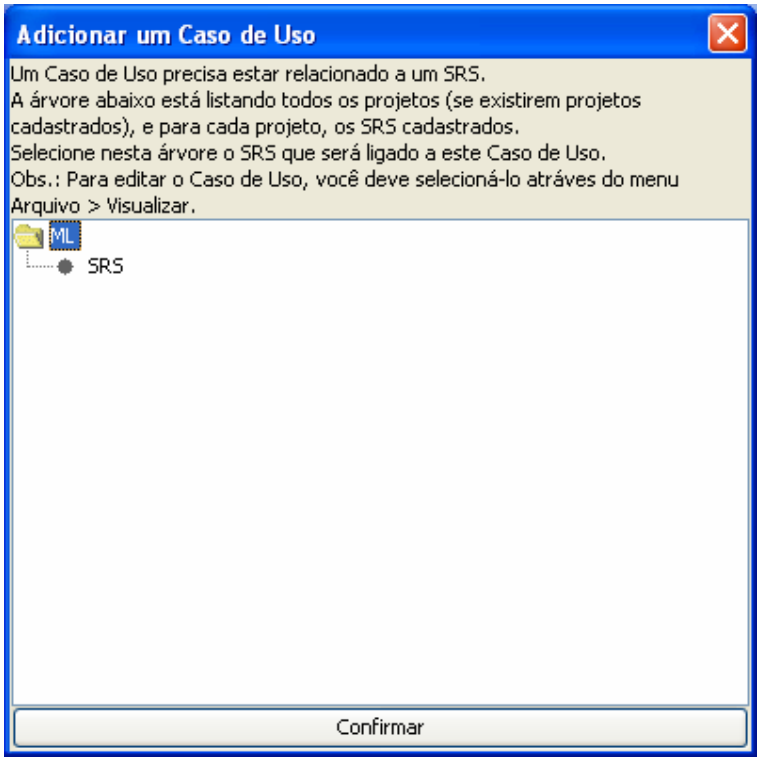


Figura 15 – Lista de projetos cadastrados.

Com relação aos casos de uso, os procedimentos de alteração tratam da modificação das seções internas do documento. Assim, deve-se selecionar o Caso de Uso a ser alterado, acessando o meno Arquivo, logo após o sub-menu Visualizar e a opção CasoUso. Em seguida será exibida uma tela (Figura 16) contendo três listas: Projetos, SRS e Caso de Uso. Na primeira, o usuário deve selecionar o projeto no qual a SRS está cadastrada, na segunda seleciona-se a SRS que está relacionada ao Caso de Uso e na terceira o Caso de Uso a ser alterado.

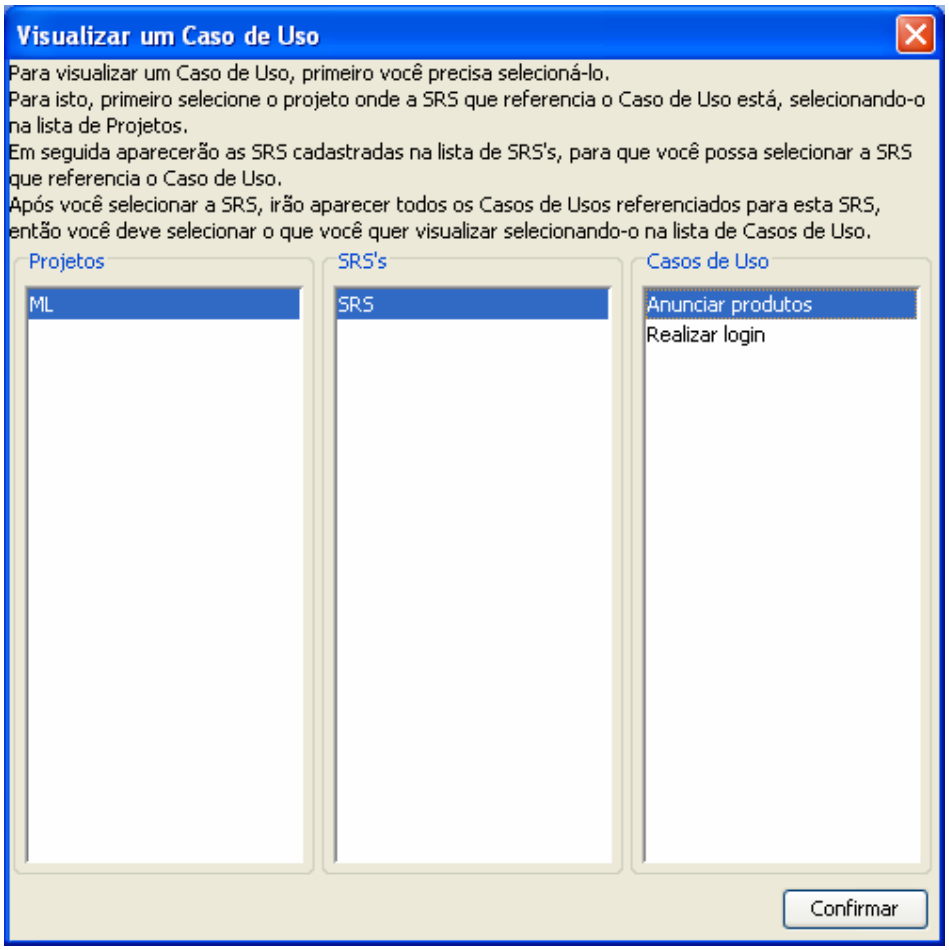


Figura 16 – Tela para visualização de um Caso de Uso

O caso de uso selecionado será apresentado em uma tela para que as devidas alterações sejam feitas. Devido ao tamanho, a Figura 17 mostra apenas os primeiros campos do caso de uso apresentado.

Figura 17 – Exemplo de caso de uso.

Para edição, deve-se marcar o item “Visualizar” e o campo será aberto com seus detalhes. A Figura 18 mostra um exemplo que detalha o campo 1.2 “Contexto”.

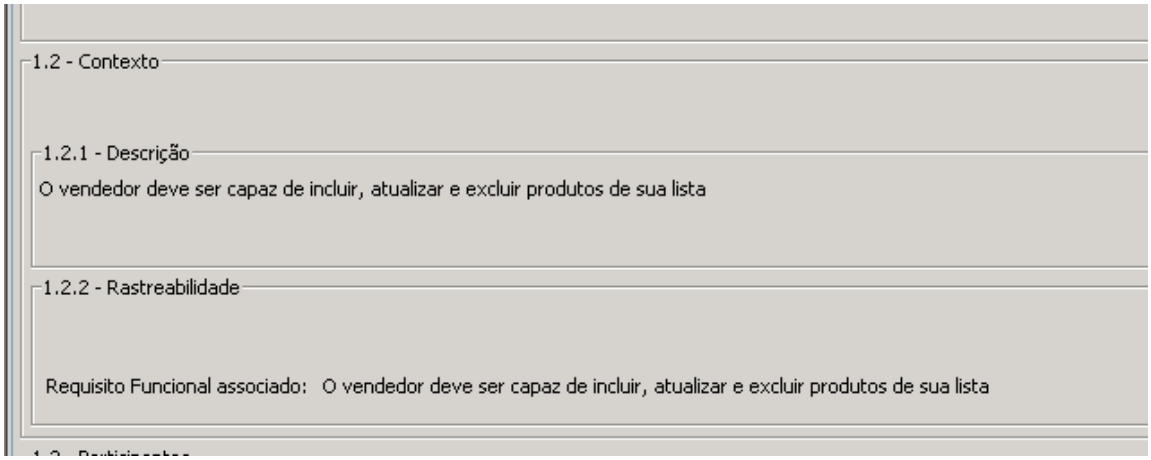


Figura 18 – Exemplo da seção Contexto detalhada.

Os procedimentos de exclusão de um Caso de Uso são idênticos aos da SRS.

2.1.4 Aplicação das relações de rastreabilidade

Esta seção descreve como as relações «trace» são construídas através da ferramenta, ou seja, como o processo de rastreabilidade é executado.

û Relacionamento: Requisito Funcional/Caso de Uso

A associação entre os requisitos funcionais e os casos de uso são feitas acessando os seguintes menus: Arquivo / Incluir / Relação Requisito > UC, conforme mostrado na Figura 19.

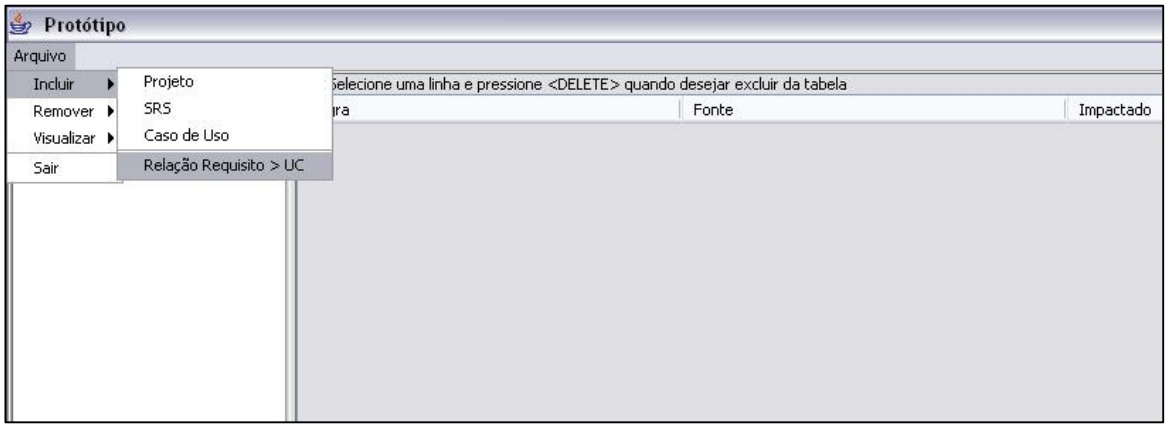


Figura 19 – Inclusão da relação Requisito X Caso de Uso.

Realizados estes procedimentos, será apresentada a tela mostrada na Figura 20, onde:

A seção Requisitos Funcionais exibe todos os projetos. A partir de um projeto é possível selecionar uma de suas SRSs e, então, selecionar um de seus requisitos.

A seção Caso de Uso exibe todas as SRSs do projeto. A partir de uma SRS será possível visualizar todos os casos de uso vinculados a ela.

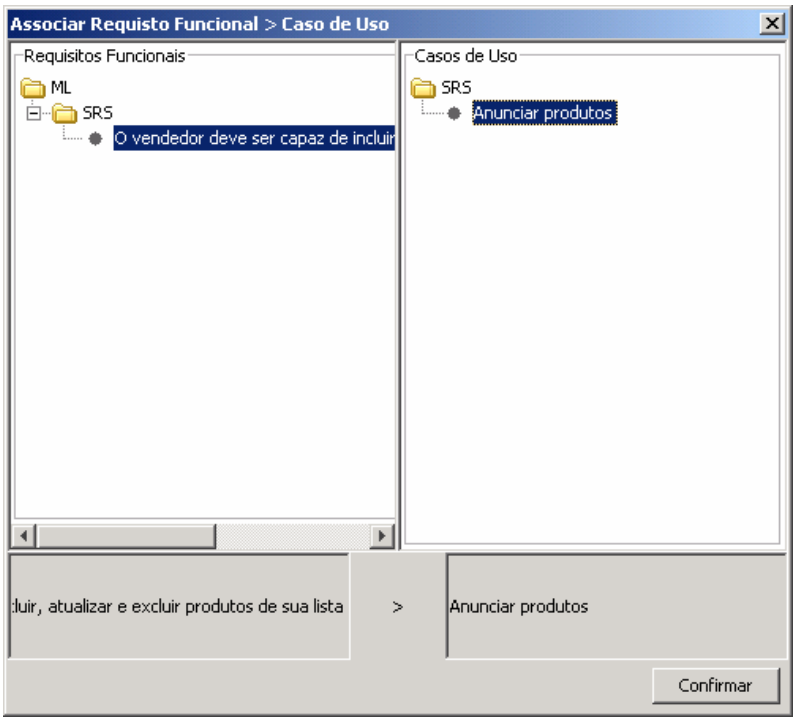


Figura 20 – Relacionamento dos requisitos com casos de uso.

No momento em que o relacionamento entre um requisito e um caso de uso for criado, vários outros serão criados automaticamente, de acordo com as regras do modelo. Todos os relacionamentos entre requisitos e casos de uso podem ser visualizados através dos seguintes menus: Arquivo / Visualizar / Relação Requisitos > UC. Através destes menus será apresentada uma tela contendo uma tabela com todos os itens, conforme mostra Figura 21.

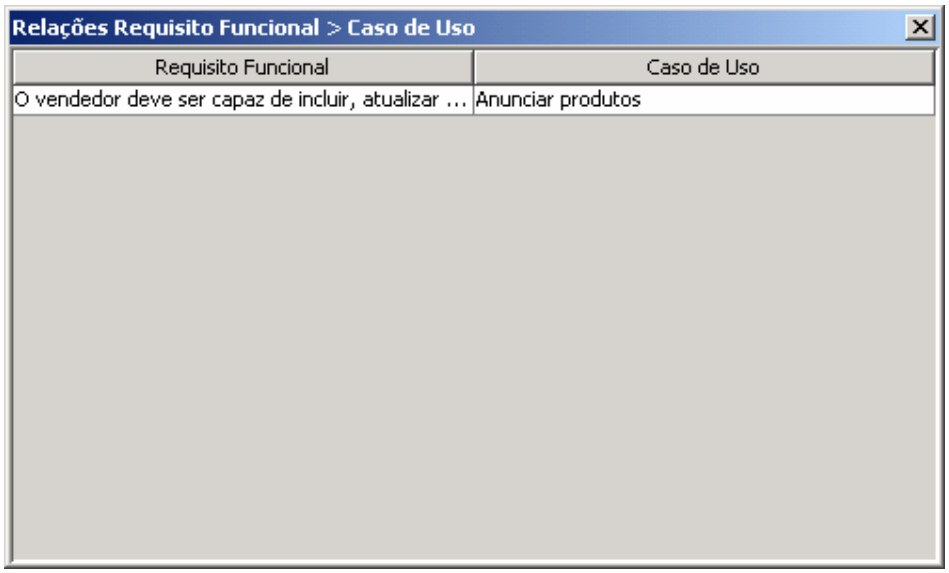


Figura 21 – Tabela de relacionamento entre requisitos e casos de uso.

Alguns elementos como Ator e Participante, por exemplo, podem ter seus relacionamentos construídos livremente. Este relacionamento é realizado através da tela do Caso de Uso da seguinte maneira:

- Selecionar o participante;
- Abrir a seção Rastreabilidade (todos os atores que estão relacionados ao requisito funcional serão listados);
- Selecionar um ator: o relacionamento é criado.

Como exemplo, a Figura 22 demonstra o participante “Vendedor” relacionado ao ator “Vendedor”.

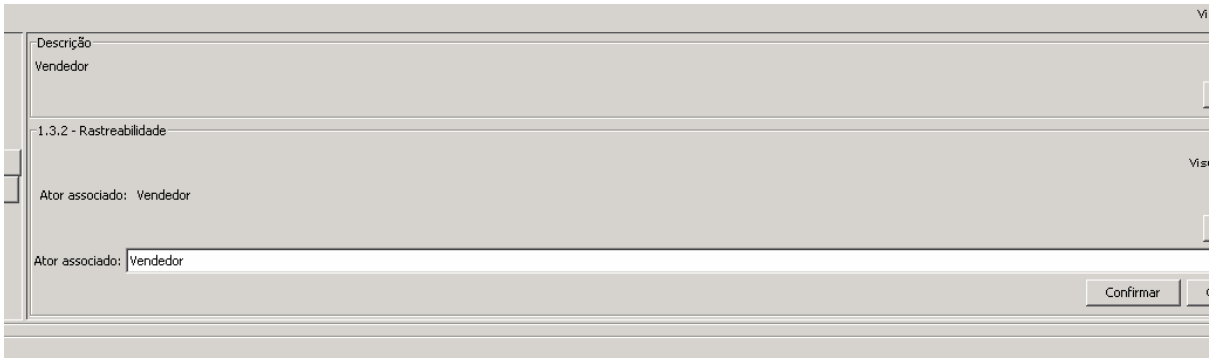


Figura 22 – Relacionamento entre participante e ator.

2.1.5 Exemplo de comportamento do protótipo durante a execução de uma regra

Neste momento será descrito o comportamento da ferramenta no momento da execução de uma regra. Como exemplo será utilizada a regra de substituição de stakeholders, já descrita na seção 6.5.1.1. Todas as outras regras, sejam de substituição, inclusão ou exclusão seguem os mesmos princípios, podendo ser facilmente simuladas seguindos os passos que serão aqui demonstrados.

Como documento base para o exemplo, será utilizada a SRS apresentada no Apêndice I. O exemplo é o mesmo apresentado na seção 7.2.1 e consiste na substituição do ator “Vendedor” pelo ator “Administrador”, com relação ao obetivo “Anunciar Produtos”. A descrição dos procedimentos de alteração será feita através dos passos apresentados na tabela Tabela 14, descrita na seção 6.5.1.1.

1. Todas as instâncias relacionadas de RequisitoFuncional serão automaticamente modificadas.

Os requisitos funcionais cadastrados na SRS são visualizados na seção 3.1 da tela da SRS. Neste caso, o requisito afetado será o 3.1.1, como mostra a Figura 23.

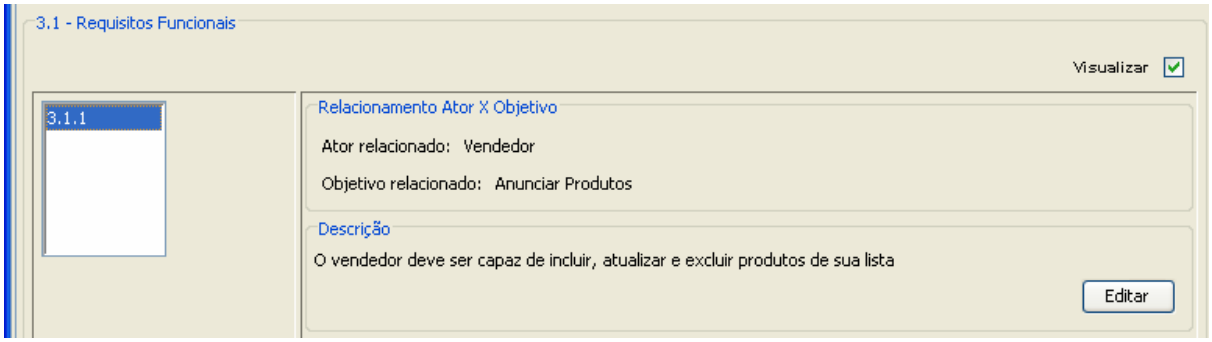


Figura 23 – Requisito modificado conforme execução da regra.

2. Analisar se as instâncias de RequisitoFuncional substituídas estão relacionadas na classe DistribuicaoRequisitos

Este procedimento é realizado automaticamente pela ferramenta. Para exemplificar, a seção Distribuição dos Requisitos pode ser acessada através da seção 2.8 na tela da SRS, como mostra Figura 24.

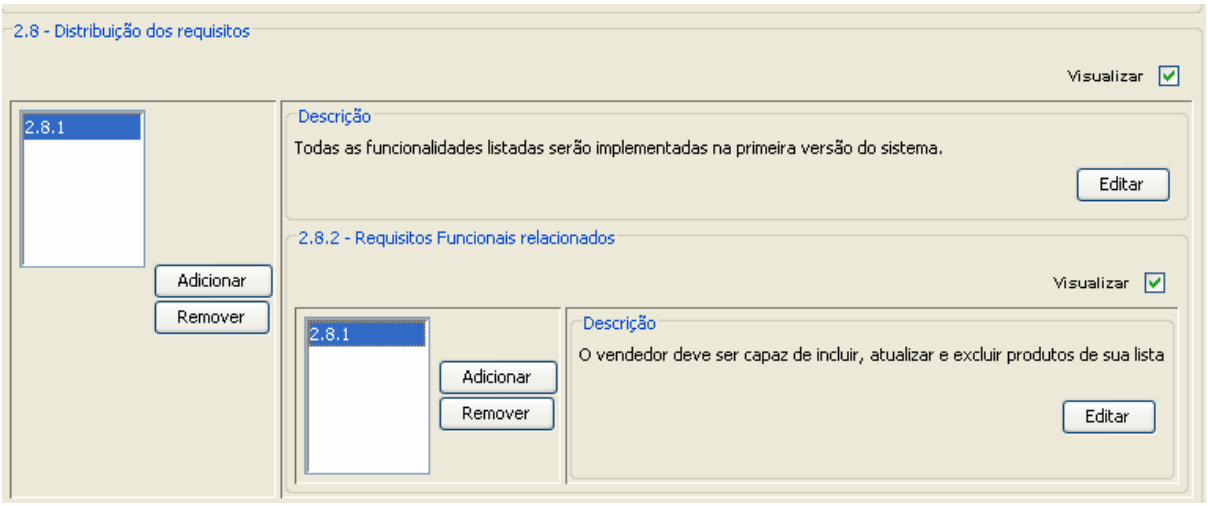


Figura 24 – Requisitos ligados à seção DistribuicaoRequisitos.

Ao abrir um dos itens desta tela, serão exibidas duas seções. A primeira contém a descrição deste item e a segunda a lista de requisitos que estão relacionados nesta seção.

3. Identificar a instância de CasoUso que corresponde a instância substituída de RequisitoFuncional

Como explicado anteriormente, as relações entre requisitos funcionais e casos de uso podem ser vistas através do seguinte caminho: Arquivo / Visualizar / Relação Requisito > UC. A Figura 25 mostra este relacionamento.

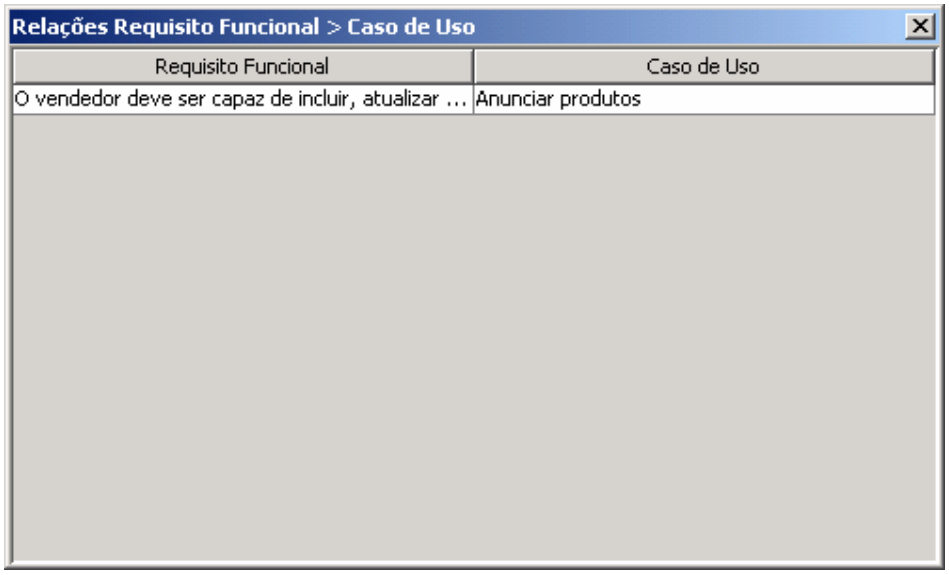


Figura 25 – Requisitos relacionados aos casos de uso.

4. Atualizar a instância de Contexto, de acordo com a descrição alterada de RequisitoFuncional

Os elementos da seção Contexto podem ser acessados através da seção 1.2 da tela do Caso de Uso. Estes elementos serão automaticamente relacionados aos requisitos funcionais, no momento em que for construída a relação entre o requisito e o caso de uso. Esta associação pode ser verificada na seção Rastreabilidade, na visualização do Contexto. A Figura 26 demonstra esta tela. Todos os itens afetados serão alterados.

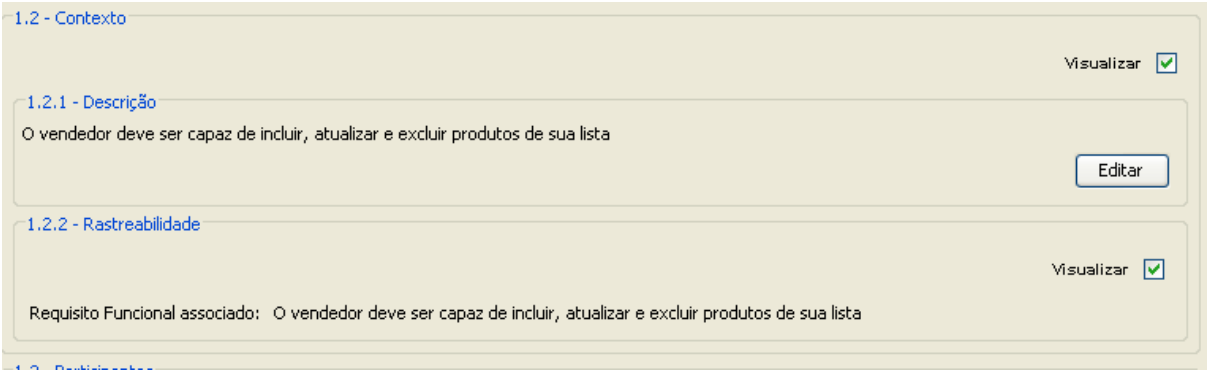


Figura 26 – Seção Contexto.

5. Identificar a instância de DiagramaAtividade correspondente a instância alterada de CasoUso

A classe DiagramaAtividade não foi implementada no modelo, será desenvolvida em versões futuras da ferramenta.

6. Atualizar a instância de Participante, do caso de uso, de acordo com a instância de Stakeholder substituída no requisito funcional

As instâncias de Participante podem ser visualizadas na seção 1.3 da tela do Caso de Uso. Os participantes a serem alterados são os relacionados com o stakeholder do requisito funcional em questão. Este relacionamento pode ser construído através da seção Rastreabilidade, disponível no momento em que um participante está sendo visualizado. Esta seção é demonstrada na Figura 27.

Figura 27 – Seção Participante do Caso de Uso.

7. Analisar se a instância de PreCondicao do caso de uso foi impactada pela substituição realizada na instância Participante

Os elementos cadastrados como Pre Condições podem ser visualizados através da seção 1.5 na tela do Caso de Uso. Para que uma pré-condição seja analisada, ela deve estar relacionada ao participante a ser substituído. Este relacionamento é construído através da seção Rastreabilidade, visualizada na Figura 28.

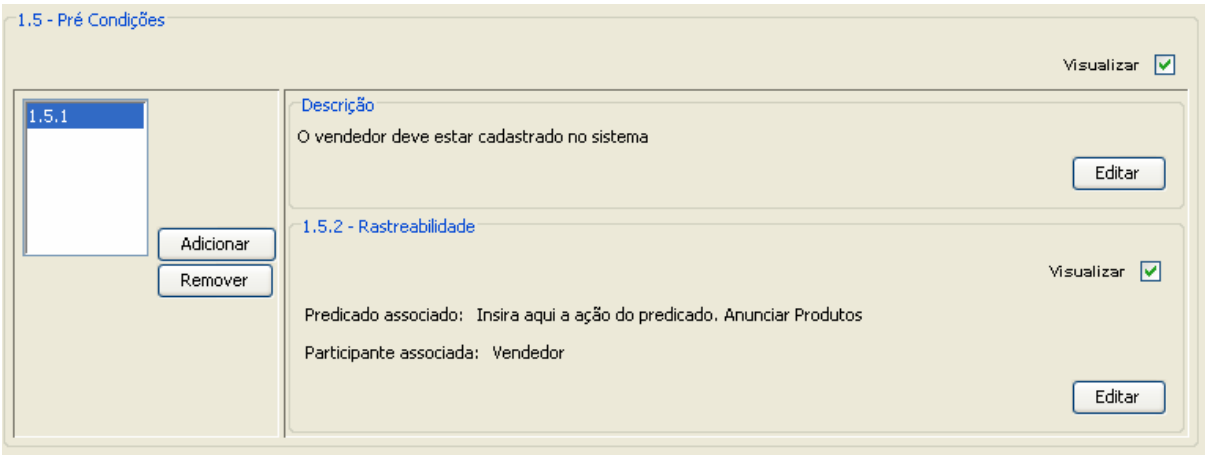


Figura 28 – Seção Pré Condições do Caso de Uso.

8. Analisar as instâncias de Passo a serem alterados conforme substituição realizada na instância de Participante

Os elementos da classe Passo podem ser visualizados em duas seções: Cenário de Sucesso Principal e Fluxo Alternativo. Ambas podem conter mais de um passo. As instâncias a serem analisadas são aquelas que possuem um relacionamento com o participaante a ser substituído. Este relacionamento pode ser visualizado através da seção Rastreabilidade, presente na tela do Caso de Uso em questão. A Figura 29 apresenta esta seção.

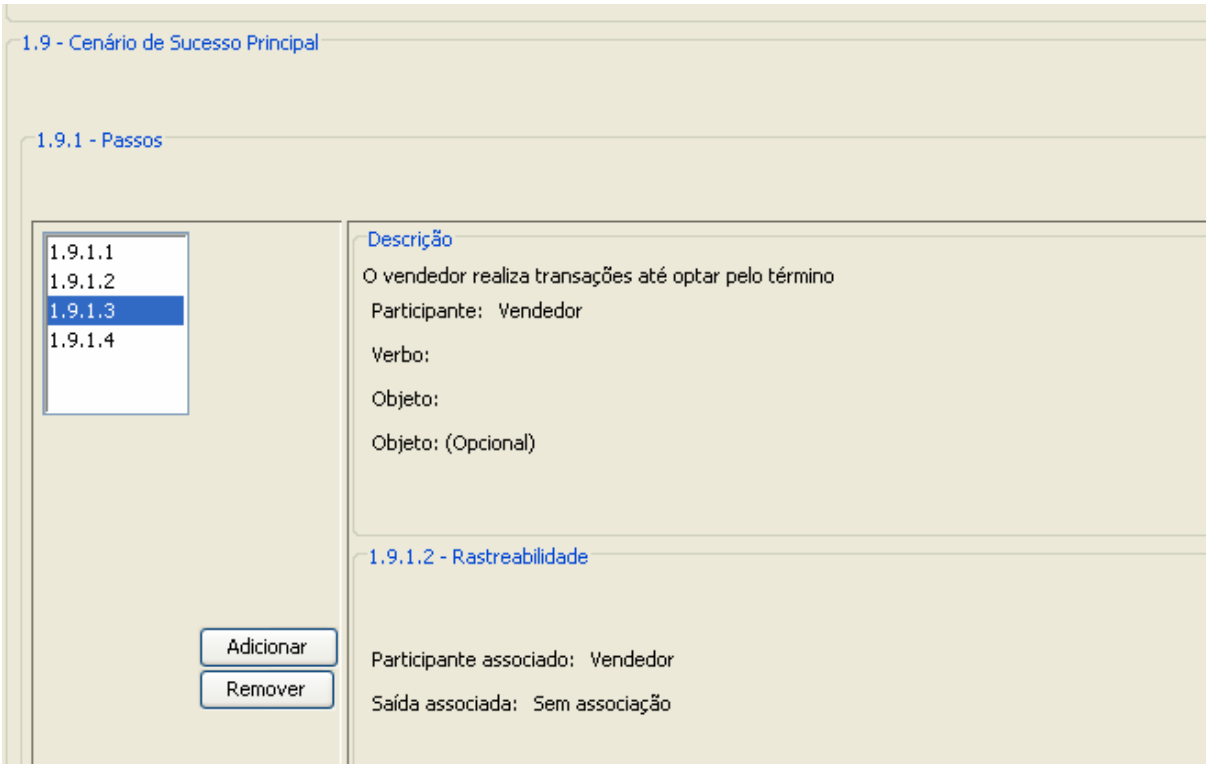
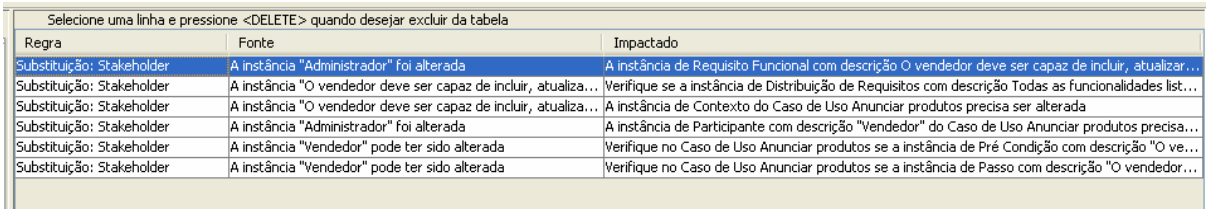


Figura 29 – Passos identificados.

9. Analisar a necessidade de alteração nas instâncias da classe Atividade, correspondentes as instâncias alteradas de Passo

As classes do tipo Atividade não foram implementadas nesta versão do protótipo. Sendo assim, este passo não será executado.

No momento da substituição deste ator, através dos passos citados acima, as mensagens mostradas na Figura 30 serão exibidas na interface principal da ferramenta:


Selecione uma linha e pressione <DELETE> quando desejar excluir da tabela

Regra	Fonte	Impactado
Substituição: Stakeholder	A instância "Administrador" foi alterada	A instância de Requisito Funcional com descrição O vendedor deve ser capaz de incluir, atualizar...
Substituição: Stakeholder	A instância "O vendedor deve ser capaz de incluir, atualiza...	Verifique se a instância de Distribuição de Requisitos com descrição Todas as funcionalidades list...
Substituição: Stakeholder	A instância "O vendedor deve ser capaz de incluir, atualiza...	A instância de Contexto do Caso de Uso Anunciar produtos precisa ser alterada
Substituição: Stakeholder	A instância "Administrador" foi alterada	A instância de Participante com descrição "Vendedor" do Caso de Uso Anunciar produtos precisa...
Substituição: Stakeholder	A instância "Vendedor" pode ter sido alterada	Verifique no Caso de Uso Anunciar produtos se a instância de Pré Condição com descrição "O ve...
Substituição: Stakeholder	A instância "Vendedor" pode ter sido alterada	Verifique no Caso de Uso Anunciar produtos se a instância de Passo com descrição "O vendedor...

Figura 30 – Mensagens dos impactos gerados pela alteração.

Na coluna Regra pode-se verificar que todas as mensagens foram geradas devido a execução da regra de susbstituição de um Stakeholder.

A coluna Fonte varia, pois, como foi visto, instâncias da classe RequisitoFuncional podem ter sido alteradas.

Já a coluna Impactado sempre reflete as classes que foram impactadas pelos passos demonstrados anteriormente. Pode-se verificar que todas as instâncias afetadas foram citadas nesta coluna.

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo