THAÍS MARA SILVA

UMA PROPOSTA DE NOVOS ESTUDOS PARA MELHORIA DO PROCESSO DE ELICITAÇÃO DE REQUISITOS PARA UMA COOPERATIVA DE SOFTWARE LIVRE

Monografia de graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do Curso de Ciência da Computação para obtenção do título de Bacharel em Ciência da Computação.

Orientador
Prof. D.Sc José Monserrat Neto

LAVRAS
MINAS GERAIS – BRASIL
2009

Ficha Catalográfica Preparada pela Divisão de Processos Técnicos
da Biblioteca Central da UFLA

Silva, Thaís Mara.
Uma Proposta de Novos Estudos para Melhoria do Processo de Elicitação de Requisitos para uma Cooperativa de Software Livre / Thaís Mara Silva. – Lavras: UFLA, 2009.
51 p.: il.

Monografia (graduação) – Universidade Federal de Lavras, 2009.
Orientador: José Monserrat Neto.
Bibliografia.

1. Engenharia de Software. 2. Engenharia de Requisitos. 3. Padrões. I. Universidade Federal de Lavras. II. Título.

THAÍS MARA SILVA

UMA PROPOSTA DE NOVOS ESTUDOS PARA MELHORIA DO PROCESSO DE ELICITAÇÃO DE REQUISITOS PARA UMA COOPERATIVA DE SOFTWARE LIVRE

Monografia de graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do Curso de Ciência da Computação para obtenção do título de Bacharel em Ciência da Computação.

APROVADA em ____ de _____ de _____.

Prof. Antônio Maria Pereira de Resende

Prof. André Luiz Zambalde

Prof. José Monserrat Neto
UFLA
(Orientador)

LAVRAS
MINAS GERAIS – BRASIL

Dedico este trabalho aos meus pais,
João Bosco (in memorian) e Eliana, as minhas irmãs
Laís e Thaíla, por fazerem parte dessa
família que eu amo.

AGRADECIMENTOS

A Deus, meu guia, conforto e proteção durante toda minha vida.

A toda minha família, por ter acreditado em mim e sempre me incentivado para nunca desistir.

Aos meus pais, pelo amor, carinho e apoio em todos os momentos da minha vida. Obrigado por todos os ensinamentos passados, lições e pela confiança depositada em mim.

As minhas irmãs, pelo companheirismo, nos momentos de alegria e tristeza estiveram sempre do meu lado.

Aos meus avós, tios, primos e padrasto por sempre me confortar com palavras positivas.

Aos meus amigos, de todas as etapas da minha vida, todos contribuíram de alguma forma para me ajudar a crescer como pessoa. Amizades verdadeiras são pra sempre.

As companheiras de república, que nos momentos mais difíceis dessa etapa, estiveram sempre do meu lado.

Aos professores, do primário à universidade, que formaram a base do meu conhecimento para o lado acadêmico e para as outras etapas da vida. Obrigado, aos professores do Departamento de Ciência da Computação da UFLA e em especial, ao Professor Monserrat, que confiou na minha capacidade e ajudou na concretização desse trabalho.

Aos amigos da TecnoLivre, pelo aprendizado, pelas oportunidades, pela prestatividade e pela iniciativa de cada um de vocês para tornar o sonho da TecnoLivre uma realidade.

UMA PROPOSTA DE NOVOS ESTUDOS PARA MELHORIA DO PROCESSO DE ELICITAÇÃO DE REQUISITOS PARA UMA COOPERATIVA DE SOFTWARE LIVRE

RESUMO

Um processo de elicitação de requisitos bem definido e estruturado pode auxiliar na busca de requisitos detalhados claramente, facilitando a fase de desenvolvimento do *software*. Este trabalho apresenta um estudo dos conceitos da Engenharia de Requisitos, tais como as técnicas de elicitação, e também sobre os padrões de desenvolvimento de *software* existentes na literatura. Por meio de questionário e análise da atividade de elicitação de requisitos, foi elaborada uma proposta de melhoria do processo de elicitação para uma Cooperativa de *Software* Livre.

Palavras-chave: Engenharia de Requisitos, Elicitação de Requisitos, Técnicas de Elicitação de Requisitos, Padrão.

A PROPOSAL FOR A NEW STUDY TO IMPROVE THE PROCESS ELICITATION REQUIREMENTS FOR COOPERATIVE SOFTWARE FREE

ABSTRACT

A process of requirements elicitation well defined and structured can help the search for detailed requirements clearly, facilitating the development phase of the software. This paper presents a study of the concepts of requirements engineering, such as elicitation techniques, and also on the patterns of software development in the literature. Through the questionnaire and analysis of the activity of requirements elicitation, a legislative proposal to improve the elicitation process for a Cooperative Software.

Keywords: Requirements Engineering, Requirements Elicitation Techniques, Requirements elicitation, Patterns.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

DCC – Departamento de Ciência da Computação

ER – Engenharia de Requisitos

JAD - *Joint Application Development*

UFLA – Universidade Federal de Lavras

PDF - Portable *Document Format*

SGBD – Sistema Gerenciador de Banco de Dados

SQL - *Structured Query Language*

WIS - *Web - based Information Systems*

1 INTRODUÇÃO

1.1 Contextualização

Desde 1950, com o avanço da tecnologia e a construção de computadores cada vez mais poderosos, um grande desafio surgiu ao desenvolvimento de *software*. O declínio dos preços de *hardware* transformou aquela máquina milionária no agente da nova era: a era da informação. Apesar de todas as promessas, elas não poderiam realizar-se, a menos que *software* fosse construído para materializá-la. Assim, visando tornar possível o uso das, cada vez mais, novas e poderosas máquinas, *software* passou a absorver mais complexidade do mundo real (LOPES, 2002).

Ao mencionar-se *Software*, verifica-se a presença deste em diversos aparelhos: de fornos a celulares, de carros a máquinas de diagnóstico computadorizadas, de computadores pessoais a sistemas industriais de controle. Conforme Lopes (2002), o *software* transformou-se na parte mais cara dos sistemas computadorizados complexos. Na medida em que, coloca em risco o avanço da tecnologia de *hardware,* quando desenvolvido com baixa qualidade, com pouca previsibilidade de custo e recursos.

Segundo Pressman (2002), a falta de adoções de métodos, ferramentas e procedimentos no desenvolvimento de *software*, resultam em números expressivos de projetos não concluídos, e de projetos concluídos que não atendem às necessidades do cliente. Para solucionar este problema, existe uma área do conhecimento da computação denominada engenharia de *software,* que surgiu a partir de meados dos anos 60, propondo a criação e a utilização de sólidos princípios de engenharia, a fim de construir *software* de maneira econômica e confiável.

Contudo, apesar dos inegáveis avanços ainda existem pontos pouco explorados, demandando atenção da comunidade acadêmica. Um desses pontos é certamente, o de requisitos de *software*. Assim, qualquer esforço no sentido de minimizar o tempo de familiarização de novos estudantes ou pesquisadores com os conceitos da área é bem-vindo.

1.2 Motivação e Objetivo

Para que o processo de desenvolvimento de *software* se realize, é necessário fazer alguns levantamentos para identificar e definir as necessidades do projeto que começará. Esses levantamentos estão compreendidos na elicitação[1] de requisitos, que não é uma tarefa simples de ser realizada, mas de suma importância para a conclusão do projeto (LOPES, 2007).

Tendo como base as definições da Engenharia de Requisitos e os seus propósitos especificados, o trabalho em questão tem como objetivo geral propor, a partir de a) estudo das técnicas de elicitação de requisitos; b) dos padrões no desenvolvimento de *software;* e c) da análise de como é realizada atualmente a elicitação de requisitos pela Cooperativa*,* novos estudos para melhoria do processo de elicitação de requisitos para uma Cooperativa de *Software* Livre.

Como objetivos específicos deste estudo estão:

- Exploração do conhecimento sobre elicitação de requisitos na literatura;
- Análise de técnicas de elicitação de requisitos existentes;
- Busca pelas melhores práticas destas técnicas.

1 A palavra elicitação não existe no português e tem sido empregada como a tradução da palavra inglesa *elicitation*. Elicitar, no português, pode ser entendido como extrair, descobrir, obter, levantar informações. Será usada daqui para frente para se referir à extração e coleta de requisitos.

- Descoberta do conhecimento sobre padrões no desenvolvimento de *software* existentes na literatura;

A cooperativa em questão é a TecnoLivre[2] – Cooperativa de Tecnologia e Soluções Livres. Esta organização está sediada na cidade de Lavras – MG. Foi originada a partir do Departamento de Ciência da Computação (DCC) e formada a partir de 2006, com apoio da Universidade Federal de Lavras (UFLA). A organização é composta por professores, graduados e estudantes de computação da UFLA. A TecnoLivre atua na elaboração, desenvolvimento e implementação de soluções tecnológicas livres para os mais variados setores da indústria, comércio e serviços.

Tem-se como principal motivação para este trabalho a necessidade de melhorar a atividade de elicitação de requisitos para os projetos da cooperativa. A partir da proposta de novos estudos para a melhoria do processo de elicitação, poderá ser uma forma de conseguir qualidade.

A proposta de novos estudos para a melhoria do processo de elicitação de requisitos poderá envolver mudança nas atividades de elicitação de requisitos, adaptadas no ciclo do processo de desenvolvimento, de tal forma que ela traga o maior impacto positivo possível para a organização.

1.3 Metodologia

1.3.1 Tipo de Pesquisa

Para classificar o tipo de pesquisa adotado, deve-se considerar o enfoque abordado, os interesses, as áreas, as condições, os objetivos, as metodologias entre outros. É responsabilidade do pesquisador, escolher qual método melhor se aplica a sua pesquisa.

2 <http://www.tecnolivre.com.br>

O tipo de pesquisa a ser utilizada neste trabalho se classifica como aplicada quanto à natureza, uma vez que terá como finalidade a propor novos estudos para uma melhora do processo de elicitação de requisitos, e que o mesmo possa ser aplicado na cooperativa em questão. Segundo Jung (2004), uma pesquisa aplicada ou tecnológica se caracteriza por uma pesquisa que gera produtos ou processos além de conhecimento com finalidade imediata.

Quanto aos seus objetivos, é uma pesquisa de caráter exploratório, pois visa propor um novo ou, a formalização de uma melhora do processo de elicitação de requisitos, a partir dos padrões existentes. Ainda conforme Jung (2004), uma pesquisa de caráter exploratório busca a descoberta de novas teorias e práticas que modifique as existentes.

A pesquisa baseia-se em referências bibliográficas e em materiais digitais disponíveis na Internet. Foram referenciados livros, artigos, páginas da Internet e notas de aulas, contendo padrões existentes e técnicas de elicitação de requisitos, além de conceitos em torno dessa área.

1.3.2 Procedimento Metodológico

O estudo realizado seguiu os seguintes procedimentos metodológicos. Inicialmente, foi realizado o referencial teórico da pesquisa, a partir das referências de livros, artigos, notas de aula e outros, para adquirir o conhecimento sobre a área, tais como engenharia de requisitos, técnicas de elicitação e padrões existentes no desenvolvimento de *software*.

Em seguida, foi realizada uma abordagem qualitativa e exploratória, utilizando um questionário, a fim de descobrir informações sobre o processo de coleta de requisitos dos projetos do empreendimento. Para que, por meio destes, sejam levantados os problemas provenientes desta atividade, assim como as técnicas mais utilizadas. E por fim, propor uma melhora do

processo de elicitação de requisitos, para que a mesma possa se aplicada na cooperativa.

1.4 Estrutura do Trabalho

O presente trabalho apresentou primeiramente conceitos sobre a Engenharia de Requisitos, onde foram abordadas algumas técnicas de elicitação de requisitos no Capítulo 2. Em seguida no Capítulo 3, foram apresentados alguns conceitos sobre arcabouços de desenvolvimento - *framework* e padrões existentes no desenvolvimento de *software*. No Capítulo 4, foi apresentada uma proposta de melhoria para o processo de elicitação de requisitos para uma Cooperativa de *Software* Livre. Por fim, apresentam-se as considerações finais e as referências bibliográficas.

2 ELICITAÇÃO DE REQUISITOS

Neste capítulo, são abordados os conceitos envolvidos na elicitação de requisitos, bem como a sua importância e as normas adotadas neste processo. Primeiramente, é definido o que é requisito – o objeto principal de estudo. Em seguida, faz-se uma abordagem geral sobre o estado da arte da engenharia de requisitos, explora as suas atividades, apresenta as dificuldades que envolvem o processo de elicitação de requisitos e uma visão detalhada das técnicas de elicitação de requisitos, mostrando a definição de alguns autores da área.

2.1 Conceitos Básicos

2.1.1 Requisitos

Os requisitos de um sistema de *software* estabelecem o que o sistema deve fazer e definem restrições sobre sua operação e sua implementação (SOMMERVILLE, 2003).

Segundo Robertson (1999 *apud* LOPES, 2002), requisito é definido como "*alguma coisa que o produto tenha que fazer ou uma qualidade que precise estar presente*".

Os requisitos podem ser classificados em funcionais e não funcionais. De acordo com Sommerville (2003), requisitos funcionais são declarações de funções que o sistema deve ser capaz de executar ou fornecer. Os requisitos de domínio são requisitos funcionais, provenientes de características do domínio de aplicação. O autor descreve requisitos não funcionais como requisitos do produto, que restringem o sistema a ser desenvolvido, os requisitos de processo que se aplicam ao processo de desenvolvimento do sistema e, portanto, são aplicados ao sistema como um todo.

A norma IEEE Std 1233 IEEE *Guide for Developing System Requirements Specifications* (IEEE 1233, 1998), em sua seção de definições, em seu item 3.15, define requisitos como: i) uma condição ou capacidade necessária para o usuário resolver um problema ou atingir um objetivo; ii) uma condição ou uma capacidade que precise ser atendida ou estar presente em um sistema ou em um componente, para satisfazer um contrato, uma norma, uma especificação ou outro documento imposto formalmente; iii) enfim uma representação documentada de uma condição ou capacidade, tal como definidas anteriormente.

Pode-se perceber que há diversos autores conceituando requisitos. Há outros que não foram mencionados, contudo as definições estão muito próximas. Assim, no contexto deste trabalho, requisitos são considerados como as descrições das necessidades as quais o sistema deve atender, tendo como objetivo orientar os desenvolvedores e tornar clara a visão do sistema para o cliente. Assim, é importante que estejam escritos de forma clara aos interessados. Os requisitos funcionais definem o comportamento do sistema, o que o *software* fará, especificando as funções que ele ou seus componentes deverão ser capazes de implementar, usando as entradas dos processos para obter as saídas esperadas. Os requisitos não funcionais expressam de que forma o sistema fará aquilo que foi definido pelos requisitos funcionais, apontando as restrições de qualidade, abrangência e operação que o sistema deverá satisfazer.

2.1.2 Engenharia de Requisitos

A Engenharia de Requisitos (ER), dentro da área de Engenharia de *Software*, trata uma parte fundamental do processo de produção de *software*, que é a definição do que se quer produzir. Cabe à ER propor métodos, técnicas e ferramentas que facilitem o trabalho de definição do que se quer

de um *software*. A sistematização da etapa de definição de requisitos é necessária porque a complexidade dos sistemas atuais exige maior atenção ao correto entendimento do problema antes do comprometimento de uma solução. Para que a definição de requisitos seja a mais eficaz possível, cabe aos engenheiros de *software* entender o ambiente no qual o *software* irá funcionar e escolher o modelo ou os modelos que melhor representam o ambiente (BORTOLI, 2000 *apud* SOARES, 2007).

Uma das principais medidas do sucesso de um *software* é o grau no qual ele atende aos objetivos e requisitos para os quais foi projetado. De forma geral, a ER lida com o processo de identificar todos os envolvidos, descobrir seus objetivos e necessidades, e documentá-los de forma apropriada para análise, comunicação e posterior implementação (FALBO, 2002).

Conforme Sommerville (2003), as descrições das funções que um sistema deve incorporar e das restrições que devem ser satisfeitas são os requisitos para o sistema. Isto é, os requisitos definem o que o sistema deve fazer e as circunstâncias sob as quais deve operar. Ou seja, o processo de descobrir, analisar, documentar e verificar/validar requisitos é chamado de processo de engenharia de requisitos.

Para Sommerville (2003), o processo de engenharia de requisitos inclui:

- **Estudo de viabilidade:** Nesta atividade, é realizado um estudo breve e direcionado, sobre as contribuições do sistema para a empresa, a tecnologia utilizada e a possível integração com outros sistemas, que tem como entrada uma descrição geral do sistema e de como ele será utilizado dentro de uma organização. Sua saída será um relatório que decidirá se o

sistema proposto é viável do ponto de vista comercial e se poderá ser desenvolvido;

- **Levantamento e análise de requisitos:** Nesta atividade, os membros da equipe técnica de desenvolvimento de *software* trabalham com o cliente e os usuários finais do sistema para descobrir mais informações sobre o domínio da aplicação, quais serviços o sistema deve fornecer, desempenho exigido do sistema, as restrições de *hardware* e assim por diante;
- **Especificação de requisitos:** Nesta atividade, são traduzidas as informações coletadas durante a atividade de análise em um documento que define um conjunto de requisitos;
- **Validação de requisitos:** Nesta atividade, são mostrados os requisitos que definem o sistema desejado pelo cliente. Durante esse processo, inevitavelmente, são descobertos erros na documentação de requisitos. Assim, estes requisitos devem ser modificados a fim de corrigir esses problemas.

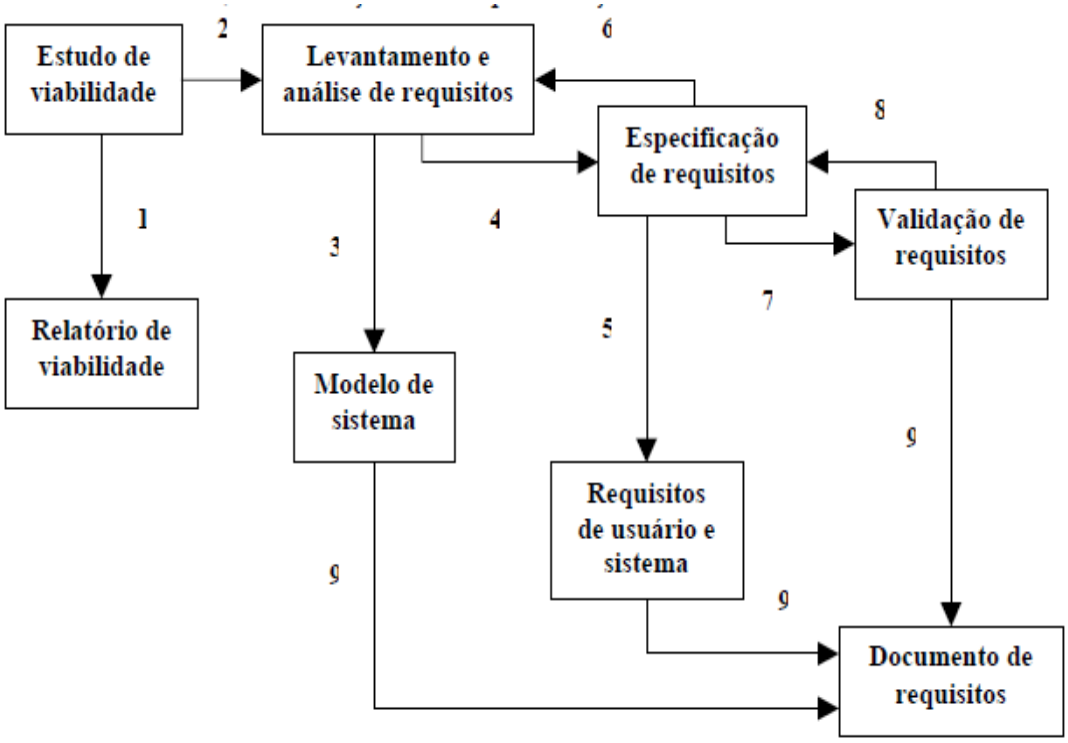

Figura 2.1 – Processo de Engenharia de Requisitos (Sommerville, 2003)

A Figura 2.1 apresenta o processo de engenharia de requisitos segundo Sommerville (2003). Esse processo começa com a fase de estudo da viabilidade gerando um relatório de viabilidade (1), este artefato determina se a próxima fase será realizada. Caso o relatório indique que o sistema é viável, o levantamento e análise de requisitos devem ser feitos (2), gerando o modelo do sistema (3). Com as informações levantadas na etapa anterior, inicia-se a especificação de requisitos (4), construindo o documento com os requisitos do usuário e do sistema (5). Caso haja necessidade, volta-se para a fase anterior para complementar os requisitos (6). Esse documento com os requisitos é validado, corrigindo na fase de validação de requisitos (7).

Havendo necessidade, volta-se para a fase de elicitação de requisitos (8). A fim do processo, uma documentação de requisitos é produzida, ou seja, tem-se a especificação do sistema (9).

Os requisitos usualmente são apresentados em dois níveis de detalhes. Os usuários e os clientes necessitam de uma especificação de alto nível dos requisitos; os desenvolvedores do sistema precisam de uma especificação mais detalhada. A análise de requisitos continua durante a definição e a especificação de requisitos. Portanto, as atividades de análise, de definição e de especificação são intercaladas.

Segundo Viana (2004), a figura 2.2 abaixo também pode sintetizar o conjunto de atividades da Engenharia de Requisitos.

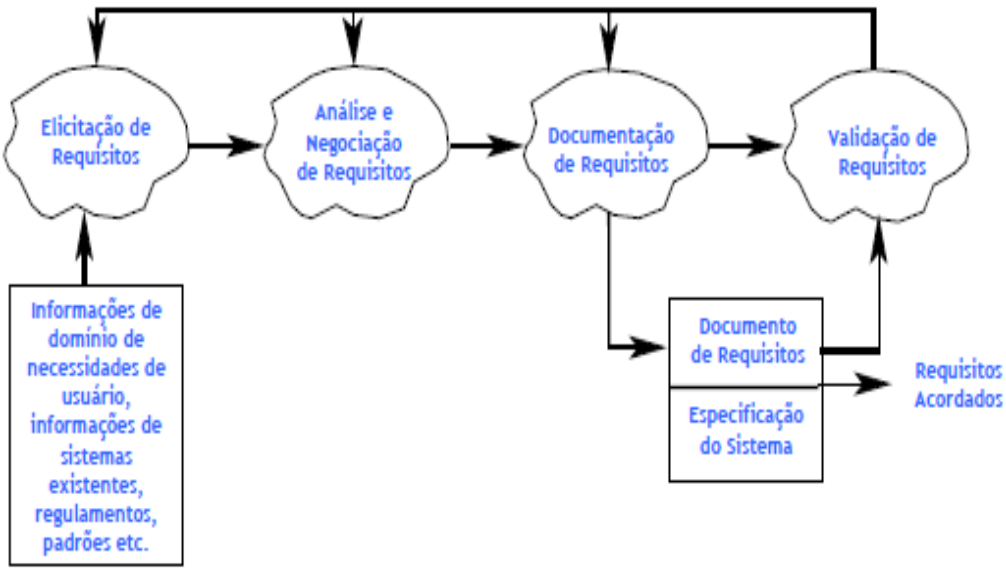

Figura 2.1 – Modelo de Atividades de Requisitos (Vianna, 2004)

As etapas que constituem o modelo ilustrado na Figura 2.1 estão descritas a seguir:

- **Elicitação de Requisitos**: Compreende o conjunto de atividades que a equipe de desenvolvimento emprega para elicitar ou descobrir as solicitações dos usuários, determinando as reais necessidades que estão subjacentes às solicitações (OBERG, 2000 *apud* VIANNA, 2004). É quando todas as necessidades, expectativas e recursos esperados do sistema serão levantados pelo Analista de Requisitos, junto a cada *stakeholder*[3]. Esta atividade normalmente requer uma forte comunicação entre o Analista e os *stakeholders*. Esta interação pode ocorrer através dos mais diversos meios, como entrevistas, dinâmicas participativas, questionários, entre outros. Os requisitos devem se levantados pela equipe do projeto, em conjunto com representantes do cliente, usuários-chave e outros especialistas da área de aplicação.
- **Análise e Negociação de Requisitos**: Nesta etapa está incluída a análise de todas as necessidades e expectativas levantadas na atividade anterior, que agora serão compiladas e ponderadas pelo Analista de Requisitos. Através da utilização de critérios de análise, como prioridade, estabilidade ou benefício do requisito, ter-se-á como resultado a definição dos requisitos que fazem parte do escopo do sistema, e que devem ser devidamente registrados e aprimorados no processo de desenvolvimento. A determinação de quais requisitos serão tratados no desenvolvimento deve ser estabelecida por meio de um

3 Qualquer envolvido que afeta ou é afetado pelo projeto. Exemplos de *stakeholders* são: usuário final do sistema patrocinador, clientes ou outros sistemas (sistemas legados, por exemplo, ou sistemas com os quais o novo *software* deva estabelecer uma interface).

consenso entre a equipe de desenvolvimento e os *stakeholders*, como consequência da negociação.

- **Documentação de Requisitos**: Compreende a atividade de especificação dos requisitos do sistema, ou seja, a "tradução" dos desejos e necessidades dos *stakeholders* para uma forma mais estruturada, que possa ser compreendida pelos demais participantes da equipe de desenvolvimento. Tal entendimento comum serve como base para o estabelecimento de um acordo entre os usuários e a equipe de desenvolvimento e como tal, a especificação é o documento central do processo, que orientará as fases subsequentes do desenvolvimento.
- **Validação de Requisitos**: Através desta atividade os requisitos previamente documentados ou especificados são validados pelos *stakeholders*, sob o ponto de vista de sua consistência e completude. O resultado perseguido é a aprovação dos requisitos pelos *stakeholders*, sinalizando sua continuidade dentro do processo de desenvolvimento, porém pode-se ter o eventual retrabalho de determinados requisitos.

Complementando a figura anterior, podemos dizer que existe uma outra atividade que permeia todo o modelo de requisitos, conhecida como a **Gestão de Requisitos** que, de acordo como (Oberg, 2000 *apud* Vianna, 2004), pode ser definida como:

- Uma abordagem sistemática para elicitar, organizar e documentar os requisitos do sistema;
- Um processo que estabelece e mantém um acordo entre o cliente e o time do projeto sobre as mudanças de requisitos no sistema.

A especificação de requisitos do *software* é tradada como o documento central no projeto – um elemento que possui relacionamentos com outros elementos do projeto. Através da definição dos relacionamentos entre os requisitos e os demais componentes do projeto é possível, por exemplo, conhecer o impacto no projeto causado pela alteração de um determinado requisito. A Gestão de Requisitos possibilita o controle dos requisitos, bem como o controle de suas mudanças, servindo como uma poderosa ferramenta na precaução de mudanças de escopo do sistema.

Algumas dificuldades podem ocorrer durante o processo de extração de requisitos e Digiampietri (2004) ilustra algumas dessas dificuldades:

Falta de Conhecimento

- O usuário não conhece suas reais necessidade e o que o produto de *software* pode lhe oferecer.
- Desenvolvedor não conhecem o domínio do problema.
- Diferenças entre o que os usuários querem e o que precisam.
- Escolhas quando dois requisitos não podem ser totalmente satisfeitos.

Problemas de Comportamento

- Domínio do processo de extração de requisitos pelos desenvolvedores.
- Conflitos e ambiguidades nos papéis – clima de insatisfação e participação menos efetiva.
- RESULTADO: custo maior, atraso no planejamento e projetos cancelados.

Problemas de Comunicação

- Usuários podem ser incapazes de expressar suas necessidades apropriadamente.
- Mal-entendidos entre usuários e desenvolvedores.
- Ambiguidade da língua natural.

Problemas Técnicos

- Avanço tecnológico muito rápido – problemas cada vez mais complexos e conhecimentos cada vez mais detalhados; requisitos caros ou complexos podem se tornar viáveis.
- Alteração nos requisitos.
- Muitas fontes de requisitos.

Contudo, a engenharia de requisitos propõe um conjunto de métodos, técnicas e ferramentas para dar apoio ao processo de definição de requisitos. Em suma, o conjunto de técnicas de levantamento, documentação e análise de requisitos forma a engenharia de requisitos.

2.2 Técnicas de Elicitação de Requisitos

Nesta seção, será apresentada uma visão detalhada de algumas técnicas de elicitação de requisitos. Serão consideradas a princípio as seguintes técnicas: i) observação; ii) entrevista; iii) questionário iv) JAD (*Joint Application Development*); v) prototipação; vi) *Brainstorming* e vii) Análise de Documentos.

Para uma elicitação de requisitos eficaz, existem técnicas para o melhor entendimento e comunicação entre clientes e analistas, para que problemas não ocorram ou, se ocorrerem, que sejam mais facilmente resolvidos. Segundo Jackson (1995 *apud* Belgamo, 2000), uma técnica deve explorar as características específicas do problema. Como as características do problema variam muito, é necessário um repertório de métodos para cada

classe de problema. O problema da elicitação de requisitos não pode ser resolvido com uma abordagem puramente tecnológica, porque nesta fase o contexto social é crucial, e mais importante do que na fase de programação, especificação e projeto.

2.2.1 Observação

A técnica de observação consiste em observar o usuário final executando uma tarefa em particular em seu local de trabalho e anotar os elementos e comportamentos envolvidos em cada tarefa executada. A técnica mostra como os usuários interagem com os sistemas e com os companheiros de trabalho (BATISTA, 2003).

Conforme Belgamo (2000), a observação é útil para "descobrir" aspectos novos de um problema. Isto se torna crucial nas situações em que não existe uma base teórica sólida que oriente a coleta de dados. Ela pode ser também usada quando não é possível a análise de documentos, isto é, para sistemas sem documentação. Aplica-se tanto para tarefas manuais, automatizadas ou para as tarefas complexas de serem descritas (BATISTA, 2003).

Ao mesmo tempo em que o contato direto e prolongado do pesquisador com a situação pesquisada apresenta as vantagens mencionadas, envolve também uma série de problemas. Algumas críticas são feitas ao método de observação. Primeiramente por provocar alterações no ambiente ou no comportamento das pessoas observadas. Outra crítica é a de que este método se baseia muito na interpretação pessoal. Além disso, há críticas no sentido de que o grande envolvimento do pesquisador leve a uma visão distorcida do fenômeno ou a uma representação parcial da realidade (BELGAMO, 2000).

2.2.2 Entrevista

Entrevista é a técnica de elicitação mais utilizada. Entrevistas são praticamente obrigatórias em qualquer desenvolvimento, já que é uma das formas de comunicação mais natural entre as pessoas (TORO, 2000). Segundo Belgamo (2000), o engenheiro de requisitos, ou analista, discute o sistema com diferentes usuários, a partir da qual elabora um entendimento de seus requisitos. Entrevistas podem ser efetivas para desenvolver uma compreensão detalhada do problema e para elicitar muitos requisitos gerais do sistema.

Na fase inicial da elicitação, o desenvolvedor pode entrevistar os usuários para obter informações sobre o trabalho que eles executam. Além disso, nos primeiros encontros as questões tendem a ser mais genéricas e tornam-se mais específicas nos encontros posteriores, conforme o desenvolvedor vai gradualmente compreendendo o domínio e fique claro que tipo de informações ele realmente necessita obter dos usuários (BATISTA, 2003).

Batista (2003) afirma que as entrevistas podem ser divididas em três tipos:

- Entrevistas não estruturadas: dependem, primeiramente, da geração espontânea das perguntas no fluxo natural de uma interação. Este tipo de entrevista é apropriado quando o entrevistador quer manter o máximo de flexibilidade na entrevista para direcionar o questionamento no sentido que lhe parecer apropriado, explorando as informações contidas nas respostas ou nas conversas com os indivíduos participantes.
- Entrevistas semi-estruturadas: envolvem a preparação de um guia para a entrevista, na qual é listado um conjunto pré-determinado

de perguntas ou de itens que devem ser explorados durante a entrevista. Este guia serve como uma lista de verificação (*checklist*) durante a entrevista e assegura que a mesma informação seja obtida de diferentes pessoas.

- Entrevistas estruturadas: consistem em um conjunto de perguntas fechadas, antecipada e cuidadosamente elaboradas. É útil quando é desejável ter a mesma informação de cada entrevistado em momentos diferentes ou quando há limitação de tempo para o levantamento de dados e análise.

Podem ser utilizadas durante todo o processo de ER tanto em projetos pequenos como em grandes, em situações em que as fontes de informações são os usuários. O tipo, o detalhe e a validade dos dados coletados variam com o tipo de entrevista e com a experiência do entrevistador (MAGUIRE, 1998 *apud* BATISTA, 2003).

2.2.3 Questionário

Bastos Júnior (2005 *apud* Soares, 2007) afirma que os questionários são um dos possíveis meios a ser utilizado para a aquisição de dados na elicitação de requisitos. Um questionário é um documento usado para guiar uma ou mais pessoas a responder uma ou mais perguntas. A elaboração de um questionário é um processo mais complexo do que possa aparentar inicialmente, devido às informações buscadas serem implícitas. Por isso, aplicar tempo e esforço adequados para a construção do questionário é uma necessidade e um fator de diferenciação favorável.

Os questionários podem ser aplicados das seguintes maneiras: formulários impressos para posterior retorno, pessoalmente (respondido na hora, como numa entrevista estruturada), respondido pelo telefone, por

formulários disponíveis na internet ou intranet ou por correio eletrônico (BATISTA, 2003).

Um questionário abrange a aplicação de várias perguntas escritas para uma ampla amostragem de usuários. Existem dois tipos de questionários: **fechados**, quando os pesquisados são solicitados a selecionar respostas a partir de uma lista de opções múltipla-escolha; e **abertos**, quando os pesquisados ficam à vontade para responder as perguntas (KIRAKOWSKI, 1997 *apud* BATISTA, 2003).

Num questionário fechado cada pergunta é apresentada sem nenhuma possibilidade de detalhamento ou de desvio. Conseqüentemente, é importante que as perguntas sejam escritas com cuidado a fim de ampliar a compreensão e restringir a ambiguidade. No entanto, em comparação com entrevistas, os questionários são menos ricos em informações e mais baratos. Eles não fornecem uma oportunidade de julgar a exatidão das respostas e não é possível detectar respostas tendenciosas.

De acordo com Camacho (2005), as principais vantagens da utilização de questionário são a padronização das perguntas e a possibilidade de tratamento estatístico das respostas. Como desvantagens, o autor aponta a limitação do universo de respostas e a pouca interação, visto a característica da impessoalidade da técnica.

2.2.4 JAD (Joint Application Development)

Conforme Belgamo (2000), JAD é uma marca registrada da IBM. O tema principal do JAD é colocar autoridades representativas e gerenciais juntas dentro de um *workshop* estruturado para promover decisões. Ou seja, visa criar sessões de trabalho estruturadas, através de uma dinâmica de grupo e recursos visuais, em que analistas e usuários trabalham juntos para projetar

um sistema, desde os requisitos básicos até o *layout* de telas e relatórios, dinâmica esta que promove a cooperação e o entendimento.

Segundo Batista (2003), JAD baseia-se em quatro princípios básicos:

- Dinâmica de grupo, com a utilização de sessões de grupo facilitadas para aumentar a capacidade dos indivíduos;
- Uso de técnicas audiovisuais para aumentar a comunicação e o entendimento;
- Manutenção do processo organizado e racional;
- Utilização de documentação padrão, preenchida e assinada pelos participantes de uma sessão.

Sessões JAD permitem aos analistas coletar simultânea e eficientemente uma grande quantidade de requisitos do sistema, junto a uma gama de usuários-chave. Essas sessões são úteis por considerar necessidades específicas dos usuários. JAD também pode ser usada em conjunto com outras técnicas de elicitação, por exemplo, a prototipação. À medida que os requisitos são obtidos nas sessões, pode-se construir um protótipo que demonstre alguma funcionalidade destes requisitos.

Em comparação com as entrevistas individuais, apresenta as seguintes vantagens (TORO, 2000):

- Economia de tempo ao evitar que as opiniões dos participantes divirjam;
- Todo o grupo, incluindo os clientes e futuros usuários, revisa a documentação gerada, não somente os desenvolvedores; e
- Envolve mais os clientes e usuários no desenvolvimento.

Segundo Batista (2003), JAD tem um ponto fraco e este relaciona-se ao fato de utilizar a dinâmica de grupo, em função da qual os membros mais quietos ou menos experientes do grupo podem ficar relutantes em discutir

abertamente certos assuntos, especialmente se envolver pontos de vista contrários àqueles de seu chefe ou de alguns dos membros mais comunicativos do grupo. Uma alternativa para resolver este problema, poderia ser a condução das sessões de maneira *on-line* usando um *login* anônimo para cada participante.

2.2.5 Prototipação

A prototipação é uma técnica que consiste basicamente em implementar uma parte do sistema para que o usuário possa dar um retorno útil ao processo elicitação de requisitos. Essa implementação é feita a partir de uma especificação preliminar com objetivo de simular a aparência e a funcionalidade do *software*, mesmo que de forma incompleta. A prototipação é utilizada no processo de levantamento de requisitos quando existe nível elevado de incerteza ou quando é preciso obter um rápido retorno dos usuários. O protótipo desenvolvido, depois de utilizado, pode proporcionar aprovação, interação, avaliação e alterações nas características de usabilidade e de funcionalidade do *software* proposto na especificação (SOARES, 2007). Com o objetivo maior de esclarecer os requisitos do usuário e conseguir defini-los precisamente, recomenda-se o uso da técnica.

Na construção do protótipo, geralmente, não precisam ser observados requisitos não-funcionais tais como desempenho, usabilidade, aspectos de segurança. Funcionalidades e flexibilidade podem ser deixadas de lado e mecanismos de gerenciamento e garantias de qualidade podem ser ignorados (KOTONYA, 1998 *apud* BATISTA, 2003).

Segundo Kotonya (1998) *apud (*Batista, 2003) existem dois tipos principais de prototipação:

Prototipação rápida/descartável *(throw-away)*: a prototipação rápida serve para ajudar a elicitar os requisitos do sistema e serve também

como mecanismo para validação dos requisitos. O protótipo rápido, ou descartável, pode ser usado como um meio para explorar os requisitos que causam mais dificuldades aos usuários e/ou que são mais difíceis de se compreender, fornecendo meios, assim, de controlar a sua constante evolução. Neste sentido, o protótipo pode ser produzido para facilitar o entendimento dos requisitos e descartado em favor da construção do sistema definitivo. Requisitos bem compreendidos não necessitam ser implementados no protótipo.

Prototipação evolutiva: são protótipos construídos através de interações do ciclo de vida do *software* em direção à sua versão definitiva. Esta abordagem propõe entregar rapidamente ao usuário uma versão executável do sistema. Portanto, os requisitos que deveriam ser completados pelas versões iniciais do sistema são aqueles que são bem compreendidos e que podem ter suas funcionalidades implementadas. Os requisitos mal compreendidos somente deverão ser implementados após extensivo uso do protótipo.

As desvantagens da prototipação enfatizadas por Soares (2007) são: a) normalmente, a gerência do projeto por serem necessárias várias interações para aprimorar um protótipo, surgindo a dificuldade sobre o término da prototipação; e b) a possibilidade do protótipo ser considerado como o sistema final, visto que a qualidade pode não ter sido devidamente analisada.

Para Paula Filho (2003), as vantagens da utilização dessa técnica são: a) a redução dos riscos na construção; b) aumento da manutenibilidade; c) aumento da estabilidade dos requisitos e d) oportunidade para treinamento dos programadores menos experientes.

2.2.6 Brainstorming

Brainstorming, também chamada "tempestade de idéias", é uma técnica muito eficiente para grupos desenvolverem idéias criativas. Essa técnica cria idéias novas, resolve problemas e motiva equipes. Motiva porque envolve os membros da equipe em uma discussão gerencial maior e eles trabalham juntos. Entretanto, o *brainstorming* não é simplesmente uma atividade aleatória, a técnica necessita ser estruturada e seguir regras (MAGUIRE, 1998 *apud* SOARES, 2007).

De acordo com Kirawowshi (1997), o *brainstorming* é uma das várias técnicas de reuniões de grupo, possivelmente a mais conhecida e a mais antiga, utilizada para originar idéias. A essência básica dessa técnica é reunir um grupo de especialistas no sistema e no negócio para que cada participante possa estimular o outro a criar idéias que auxiliem a encontrar a solução do problema em uma ou mais reuniões. Ainda, segundo Kirawowski (1997), as idéias que surgirem e forem discutidas nestas reuniões não podem ser recriminadas ou julgadas. A técnica pode ser aplicada no começo da fase do desenvolvimento onde pouco do projeto é conhecido e são necessárias idéias novas.

Bellino & Case (2005) afirmam que o *brainstorming* é parte essencial do processo de pensamento de um negócio ou nova idéia em toda e qualquer organização. Quando esta técnica é adequadamente executada, ela pode conduzir a uma dinâmica e valiosa troca de idéias e inciativas. Porém, quando mal conduzida, ela pode promover ressentimento, tensão interna e ser contra produtivo. Os autores destacam algumas boas prática para conduzir uma sessão *brainstorming* para que seja ao mesmo tempo criativa e bem-sucedida:

- Estabelecer o papel do líder;
- Definir a tarefa e manter o controle na sua mão;

- Indicar o objetivo da sessão de *brainstorming*;
- Criar uma atmosfera positiva;
- Registrar as idéias;
- Incentivar o fluxo de idéias;
- Monitorar com cuidado uma comunicação verbal e não-verbal.

Os pontos fortes do *brainstorming* são: a) evitar a tendência a limitar o problema muito cedo; e b) fornecer uma interação social mais confortável do que algumas técnicas de grupo mais estruturadas; e c) pode ser aprendida com muito pouco investimento. A desvantagem é, por ser um processo relativamente não estruturado, não produzir a mesma qualidade ou nível de detalhes de outros processos (DIGIAMPIETRI, 2004).

2.2.7 Análise de Documentos

A análise de documentos é uma técnica comumente utilizada na elicitação de requisitos e contempla a pesquisa e a coleta de informação da documentação existente, podendo envolver ou não a interação com um especialista. Ela permite que se explore todo o conhecimento escrito encontrado no domínio da aplicação. Nesta análise, incluem-se todos os formulários, relatórios, manuais de sistema, manuais de política e diretrizes, base de dados, contratos, documentos fiscais. De acordo com Wessels (2002), a técnica ajuda a dar uma idéia clara sobre os aspectos formais do sistema atual.

A análise de documentos é usada para compreender o sistema atual. Pode ser usada tanto para determinar o tipo de informação processada no sistema, como para identificar pontos a serem melhorados no sistema atual (WESSELS, 2002).

Conforme Batista (2003), a análise dos documentos permite um contato com o vocabulário utilizado no domínio do problema. Normalmente, os documentos estão facilmente acessíveis e fornecem uma grande quantidade de informações.

Os documentos podem exprimir melhor as tarefas dinâmicas do que a análise do trabalho efetivo feita por observação.

A análise dos documentos impõe uma grande carga de trabalho aos analistas, visto que, pelas próprias características dos documentos, as informações estão espalhadas e nem todos os documentos disponíveis apresentam uma clara e completa visão do sistema atual (BRAY, 2002 *apud* BATISTA, 2003).

Outro problema apontado por Bray (2002 *apud* Batista, 2003) é que a maneira tradicional de analisar os resultados da análise dos documentos, geralmente através de fluxos, tende a culminar na elaboração de um modelo estrutural do sistema existente. Por si só isto não representa um problema, porém existe o perigo desta velha estrutura, geralmente com defeitos, ser transmitida para a especificação e, pior, para o novo sistema, sem as devidas críticas e alterações.

Os documentos refletem situações do passado e, potencialmente, existem muitos documentos para ser analisados.

2.3 Considerações Finais

Neste capítulo, pode-se observar a existência de várias técnicas cada uma com suas peculiaridades que as capacitam para aplicações distintas, de acordo com a maturidade da contratante e da contratada.

Também observa-se a importância da elicitação de requisitos, devido aos problemas que surgem caso esse processo não seja bem desenvolvido. Foram citados vários autores que mostram como realizar o levantamento de

requisitos através de técnicas de elicitação, esclarecendo como cada técnica é aplicada e como otimizar essa aplicação para que se possa obter um conjunto de requisitos que defina corretamente o sistema.

3 ARCABOUÇOS DE DESENVOLVIMENTO E PADRÕES

Neste capítulo serão examinados e discutidos os conceitos básicos em relação aos arcabouços de desenvolvimento, padrões de desenvolvimento de *software*, sobre sua tipologia, ou seja, os tipos de padrões e seus componentes essenciais presentes na literatura.

3.1 Arcabouços de Desenvolvimento

A exigência do mercado sobre o desenvolvimento de *software* de qualidade, a um preço acessível e com prazos de entrega reduzidos, requer uma reutilização sistemática de modelos, padrões e implementações aplicadas e testadas (SCHMIDT *and* BUSCHAMANN, 2003). Neste contexto, estão inseridos os arcabouços de desenvolvimento, também conhecidos por *frameworks*. Arcabouços de desenvolvimento são descritos na literatura sob diferentes perspectivas, pois representam um conceito abstrato e, portanto, podem apresentar características diferentes, embora todos visem o mesmo objetivo.

Um *Framework* é o projeto de um conjunto de objetos que colaboram entre si para execução de um conjunto de funcionalidades. Um *framework* reusa análise, projeto e código. Ele reusa análise porque descreve os tipos de objetos importantes e como um problema maior pode ser dividido em problemas menores. Ele reusa projeto porque contém algoritmos abstratos e descreve a interface que o programador deve implementar e as restrições a serem satisfeitas pela implementação. Ele reusa código porque torna mais fácil desenvolver uma biblioteca de componentes compatíveis e porque a implementação de novos componentes pode herdar parte de seu código das superclasses abstratas. Apesar de todos os tipos de reuso serem importantes, o reuso de análise e de projeto são os que mais compensam a longo prazo (JOHNSON, 1991).

Da perspectiva do desenvolvedor de aplicações, a maior diferença entre um *framework* e uma biblioteca de classes está no conhecimento necessário para utilizá-los. Os usuários da biblioteca de classes precisam somente entender a interface externa das classes e definir toda estrutura de sua aplicação. Em contraste, usuários de *frameworks* devem entender o projeto abstrato do *framework* bem como a estrutura de suas classes, de forma a adaptá-las ou estendê-las (CARNEIRO, 2003).

Em geral, é mais difícil aprender a utilizar um *framework* do que bibliotecas de classes, contudo, seu potencial para reutilização excede em muito o potencial de reutilização de biblioteca de classes, contribuindo mais significadamente para o aumento de produtividade no desenvolvimento de um *software*.

Conforme Ribeiro & Monserrat (2008), um arcabouço é um conjunto de artefatos de *software* (por exemplo, classes reutilizáveis, motores de controle de fluxo, modelos, padrões de desenvolvimento e documentação) que auxiliam no desenvolvimento de aplicações de um determinado domínio, por exemplo, aplicações médicas, sistemas de informação (SI), aplicações multimídia, sistemas de roteamento, *ecommerce*, sistemas *Web*, etc.

Uma característica importante dos *frameworks* é que os métodos definidos pelo usuário para especializá-lo são chamados a partir de dentro do próprio *framework*, ao invés de serem chamados a partir do código de aplicação do usuário (JOHNSON, 88 *apud* MALDONADO). O *framework* geralmente desempenha o papel do programa principal, como seu esqueleto principal, coordenando e sequenciando as funcionalidades da aplicação. Essa inversão de controle permite que o *framewor*k funcione como um esqueleto extensível. Ou seja, os métodos desenvolvidos pelo usuário especializam os

algoritmos genéricos, definidos no *framework*, para uma dada aplicação específica.

3.2 Padrões

A origem dos padrões de projeto está no trabalho realizado pelo arquiteto Christopher Alexander, no final dos anos 70. Ele escreveu dois livros: "A Pattern Language" e "A Timeless Way of Building" que além de exemplificarem, descrevem seu método para documentação de padrões. O trabalho de Alexander, apesar de ser voltado para a arquitetura, possui uma fundamentação básica que pode ser abstraída para a área de *software*.

Os padrões permaneceram esquecidos por um tempo, até que ressurgiram na conferência sobre programação orientada a objetos (OOPSLA) de 1987, mais especificamente, no Workshop sobre Especificação de Projeto para Programação Orientada a Objetos (Beck, 87). A partir de então, muitos artigos, revistas e livros têm aparecido abordando padrões de *software*, que descrevem soluções para problemas que ocorrem frequentemente no desenvolvimento de *software* e que podem ser reusadas por outros desenvolvedores. Um dos primeiros trabalhos estabelecendo padrões foi o de Coad, no qual são descritos sete padrões de análise. Neste mesmo ano Coplien publicou um livro definindo inúmeros "idiomas", que são padrões de programação específicos para a linguagem C++. Em 1993, Gamma e outros introduzem os primeiros de seus vinte e três padrões de projeto, que seriam publicados em 1995. Esses foram os trabalhos pioneiros na área de padrões, seguidos por muitos outros nos anos seguintes (MALDONADO, 2001).

Um padrão descreve uma solução para um problema que ocorre com frequência durante o desenvolvimento de *software*, podendo ser considerado como um par "problema/solução". Um padrão é um conjunto de

informações instrutivas que possui um nome e que capta a estrutura essencial e o raciocínio de uma família de soluções comprovadamente bem sucedidas, para um problema repetido, que ocorre sob um determinado contexto e um conjunto de repercussões.

Apesar de existirem diversos padrões, estes têm sido descritos em diferentes formatos. Porém, existem componentes essenciais que devem ser claramente identificáveis ao se ler um padrão (APPLETON, 97):

Nome: Todo padrão deve ter um nome significativo. Pode ser uma única palavra ou frase curta que se refira ao padrão e ao conhecimento ou estrutura descritos por ele. Se o padrão possuir mais do que um nome comumente usado ou reconhecível na literatura, subseções *"Aliases"* ou *"Also know as"* devem ser criadas.

Problema: Estabelece o problema a ser resolvido pelo padrão, descreve a intenção e objetivos do padrão perante o contexto e forças específicas.

Contexto: Pré-condições dentro das quais o problema e sua solução costumam ocorrer e para as quais a solução é desejável, o que reflete a aplicabilidade do padrão. Pode também ser considerado como a configuração inicial do sistema antes da aplicação do padrão.

Forças: Descrição dos impactos, influências e restrições relevantes para o problema e de como eles interagem ou são conflitantes entre si e com os objetivos a alcançar. Um cenário concreto que serve como motivação para o padrão.

Solução: Relacionamentos estáticos e regras dinâmicas descrevendo como obter o resultado desejado. Equivale a dar instruções que descrevem como o problema é resolvido, podendo para isso utilizar texto, diagramas e figuras.

Exemplos: Uma ou mais aplicações do padrão que ilustram, num contexto inicial específico, como o padrão é aplicado e transforma aquele contexto em um contexto final.

Contexto Resultante: O estado ou configuração do sistema após a aplicação do padrão, podendo ter uma subseção "*Conseqüências*" (tanto boas quanto ruins). Descreve as pós-condições e efeitos colaterais do padrão.

Fundamentação: Uma explicação das regras ou passos do padrão que explicam como e porque ele trata suas influências contrárias, definidas em 'Forces', para alcançar os objetivos, princípios e filosofia propostos. Isso nos diz realmente como o padrão funciona, porque funciona e porque ele é bom.

Padrões Relacionados: Os relacionamentos estáticos e dinâmicos desse padrão com outros dentro da mesma linguagem ou sistema de padrões. Padrões relacionados geralmente compartilham as mesmas influências.

Usos Conhecidos: Descreve ocorrências conhecidas do padrão e sua aplicação em sistemas existentes. Isso ajuda a validar o padrão, verificando se ele é realmente uma *solução provada para um problema recorrente*.

Conforme já mencionado, padrões de *software* abrangem diferentes níveis de abstração, podendo, portanto ser classificados em diversas categorias de modo a facilitar sua recuperação e uso. Porém, essa

classificação não é rigorosa, podendo haver padrões que se encaixam em mais de uma categoria. A seguir resumem-se algumas categorias importantes de padrões (MALDONADO, 2001).

- **Padrões de processo:** definem soluções para os problemas encontrados nos processos envolvidos na engenharia de *software*: desenvolvimento, controle de configuração, testes, etc.
- **Padrões arquiteturais:** expressam o esquema ou organização estrutural fundamental de sistemas de *software* ou *hardware*.
- **Padrões de padrão:** (em inglês, *patterns on patterns*): são padrões descrevendo como um padrão deve ser escrito, ou seja, que padronizam a forma com que os padrões são apresentados aos usuários.
- **Padrões de análise:** descrevem soluções para problemas de análise de sistemas, embutindo conhecimento sobre o domínio de aplicação específico.
- **Padrões de projeto:** definem soluções para problemas de projeto de *software*.
- **Padrões de interface:** definem soluções para problemas comuns no projeto da interface de sistemas. É um caso particular dos padrões de projeto.
- **Padrões de programação:** descrevem soluções de programação particulares de uma determinada linguagem ou regras gerais de estilo de programação.
- **Padrões de Persistência:** descrevem soluções para problemas de armazenamento de informações em arquivos ou bancos de dados.
- **Padrões para Hipertexto:** descrevem soluções para problemas encontrados no projeto de hipertextos.

- **Padrões para Hipermídia:** descrevem soluções para problemas encontrados no desenvolvimento de aplicações hipermídia.

3.3 Relacionamento entre *Frameworks* e Padrões de Projeto

Frameworks são frequentemente entendidos como sendo padrões de larga escala. Embora isto não seja verdade, o relacionamento entre padrões de projeto e *frameworks* é importante: padrões tipicamente têm o foco nos aspectos de flexibilidade de *software* e flexibilidade é exatamente o que é necessário aos *frameworks* para que possibilitem sua especialização em aplicações reais.

Carneiro (2003) apresenta uma discussão sobre a diferença entre padrões e *frameworks*:

- *Framework*: é um projeto reutilizável em soluções de problemas em algum domínio específico.
- Padrão: Cada padrão descreve como resolver uma pequena parte de um grande problema de projeto. Padrões são perfeitamente indicados para ensinar como usar um *framework*.

Gamma *et al. apud* Carneiro (2003) apresentam a seguinte diferenciação entre padrões e *frameworks*:

- Padrões de projeto são mais abstratos que *frameworks*: *frameworks* podem ser incorporados em código, enquanto que apenas exemplos de padrões podem ser incorporados em códigos.
- Padrões de projetos são elementos arquiteturais menores do que *frameworks*: um típico *framework* contém vários padrões de projeto, mas o inverso não é verdade.
- Padrões de projeto são menores especializados do que *freameworks*: *frameworks* sempre pertencem a um domínio em

particular, enquanto que os padrões de projeto poderiam, em princípio, ser usados em qualquer tipo de aplicação.

3.4 Considerações Finais

Este capítulo teve a finalidade de esclarecer os conceitos básicos em relação aos *frameworks,* sobre suas características e sua importância para o processo de desenvolvimento de *software* e padrões de desenvolvimento. Por fim, foram apresentados alguns relacionamentos entre *frameworks* e os padrões de projeto.

4 PROPOSTA DE ELICITAÇÃO DE REQUISITOS

Neste capítulo, serão apresentadas as observações provenientes do questionário aplicado e uma proposta de melhoria para o processo de elicitação de requisitos, que deverá ser utilizada pela Cooperativa.

4.1 Apresentação das Observações

Para realizar a coleta de informações sobre a forma que atualmente é realizada a elicitação de requisitos na organização, aplicou-se um questionário aos cooperados que já participaram da atividade de elicitação de requisitos. O questionário utilizado está descrito no Anexo A. As seguintes observações foram obtidas a partir do questionário:

Utiliza-se com mais frequência a técnica de entrevista informal e realiza-se a análise de documentos, como principal fonte de coleta de requisitos;

As técnicas de *Brainstorming* (reuniões de grupos), JAD (sessões estruturadas) e questionários não são muito utilizadas, porém nas poucas vezes que foram empregadas, elas ocorreram informalmente;

A cada contato com o cliente é em geral utilizada a mesma técnica, ou seja, a entrevista é a principal fonte de dados e informações para a realização da elicitação de requisitos;

A observação foi empregada em pelo menos um projeto da Cooperativa. No entanto, a restrição para a utilização desta técnica, segundo um cooperado, é quando não se tem acesso ao ambiente de trabalho do usuário final;

A Cooperativa em questão já utilizou-se de protótipos como forma de coletar requisitos. No entanto, ocorreu um problema: o cliente acreditou que o protótipo já era o sistema, revelando não possuir um conhecimento de informática para compreender do que se tratava.

Quanto aos perfis mencionados pelos cooperados cita-se:

Perfis do Processo: podem apresentar um processo estável, em mudança/construção, ou simplesmente não existe;

Perfis do Usuário: os usuários podem possuir habilidade em informática ou não;

Perfis do Cliente: os clientes podem saber o que querem, ou saber parcialmente o que querem, ou não saber o que querem;

Perfis do projeto: projetos podem ser de longo, médio ou curto prazo, e ter um custo de desenvolvimento abundante, satisfatório ou escasso;

Embora sejam citados os perfis acima, o presente trabalho focaliza em particular os perfis de processo e do cliente.

Com base nestas observações, apresentam-se abaixo alguns cenários comuns na elicitação de requisitos, capaz de exemplificar os problemas enfrentados na área.

4.2 Estudo de Cenários em Formato Padrão de Elicitação de Requisitos

A proposta aqui apresentada baseia-se a) no estudo das técnicas de elicitação de requisitos; b) nos padrões de desenvolvimento de software existentes na literatura; c) nas dificuldades da extração dos requisitos mencionados por Digiampietri (2004) e d) nas informações adquiridas pelos cooperados, através de questionário, referentes à elicitação de requisitos de projetos da Cooperativa.

A proposta de melhoria do processo de elicitação de requisitos, que descreve uma solução para um problema em um dado cenário, utiliza o formato de padrão de Appleton (1997), e é composto pelos seguintes componentes: Nome, Contexto, Problema, Solução, Forças e outros. A seguir é apresentada uma sequência de cenários, em que problemas na elicitação de requisitos podem ocorrer, bem como, respectivamente, as

soluções para cada um deles, tendo em vista que padrões correspondem ao par 'problema/solução'.

O passo inicial do processo de elicitação de requisitos é aplicar a técnica de entrevista não-estruturada, a fim de estabelecer um primeiro contato com os *stakeholders* e inicar a primeira sessão de elicitação de requisitos. É desta primeira sessão que se poderá verificar em qual cenário-problema o cliente se enquadra, para que o processo de elicitação de requisitos prossiga com qualidade. É importante ressaltar que a elicitação de requisitos é um processo que, em alguns casos, pode ser longo. Por isso, as situações apresentadas a seguir, com suas soluções, poderão ou não ocorrer. Além disso, elas poderão ocorrer de forma alternada, ora um problema, ora outro. É importante que a proposta de melhoria do processo de elicitação seja visto como um processo flexível, não como um elemento de engessamento e burocratização da atividade de elicitação de requisitos.

Cenário 1:

Nome do Cenário: Filtragem do nível de detalhes dos requisitos da aplicação por parte do cliente

Contexto: Um cliente necessita de um sistema e ele conhece suas reais necessidades, e o que o produto de software pode lhe oferecer, além de possuir conhecimento detalhado de seu processo de trabalho, caracterizando-o como estável.

Problema: O problema tem origem no fato do cliente possuir um alto domínio da aplicação, podendo dificultar o processo de elicitação. Uma vez que, o cliente ou usuário apresentará aos analistas um nível de detalhes

muito grande sobre a aplicação, o que pode às vezes dificultar a análise dos requisitos.

Solução: Propõe-se como solução as técnicas de entrevista semi-estruturada e a análise de documentos. A primeira técnica recomenda-se, pois ela possibilita entrevistar diferentes pessoas de uma maneira mais completa, através da limitação dos assuntos tratados na entrevista. Sugere-se também a análise de documentos, pois a procura pelo conhecimento escrito, encontrado no domínio de aplicação, possibilitará o conhecimento dos detalhes que realmente são pertinentes para que ocorra a análise de requisitos.

Forças: A entrevista semi-estruturada se caracteriza pela flexibilidade, pois dentro da lista de tópicos ou de assuntos, o entrevistador está livre para explorar determinadas perguntas com maior profundidade. Enquanto a desvantagem é que ela não permite ao entrevistador explorar tópicos ou assuntos de interesse que surgem no processo, na medida que não foram contemplados na elaboração do guia da entrevista. A análise de documentos tem como vantagem permitir um conhecimento detalhado do vocabulário utilizado no domínio do problema. A limitação da técnica é que ela impõe uma grande carga de trabalho aos analistas, visto que, pelas próprias características dos documentos, as informações estão espalhadas e nem todos os documentos disponíveis apresentam uma clara e completa visão do sistema atual.

Cenário 2:
Nome do Cenário: Problema do domínio debilitado da aplicação

Contexto: Um cliente necessita de um sistema e não conhece suas reais necessidades, mas possui conhecimento detalhado de seu processo de trabalho. Caracterizando o processo do cliente como estável.

Problema: O problema tem origem no fato do cliente não possuir o domínio do problema, podendo dificultar o processo de elicitação, na medida em que o cliente ou usuário não transmitirá aos analistas os detalhes necessários sobre o seu problema, de modo que o processo de elicitação ocorra efetivamente.

Solução: Propõe-se como solução as técnicas de observação e *brainstorming*. A primeira técnica recomenda-se, porque permite observar a interação entre o usuário e o sistema atual. Uma vez que o cliente possua um processo de trabalho estável, a observação possibilita verificar suas reais necessidades que não são identificadas por ele. E sugere-se o *brainstorming*, pois a partir da geração de novas idéias, o cliente poderá definir de forma mais clara quais são suas reais necessidades. Além disso, o envolvimento entre o grupo poderá estimular e possibilitar um melhor entrosamento entre clientes e usuários na solução do problema.

Forças: A observação tem como ponto forte o fato de ser uma técnica simples de ser empregada e não requerer muito treinamento e preparação. Sua limitação é a possibilidade de alterar o comportamento do usuário, devido à presença de um observador. Já o *brainstorming* tem como vantagem o fato de boas idéias poderem surgir da combinação de idéias aparentemente ruins, sugeridas por pessoas que foram encorajadas a propor idéias, muitas vezes incomuns. Sua desvantagem é a de que sessões de

brainstoming são relativamente caras, devendo ser usadas moderadamente, possivelmente uma sessão durante todo o processo.

Cenário 3:
Nome do Cenário: Problema do processo do cliente em modificação

Contexto: Um cliente necessita de um sistema e ele conhece suas reais necessidades. No entanto, seu processo de trabalho está em mudança ou em (re)construção, o que dificulta a determinação das atividades do processo do cliente, por não estarem bem definidas para que os analistas possam captar.

Problema: O problema tem origem no fato de que, embora o cliente tenha ciência de suas reais necessidades, seu processo de trabalho encontra-se em mudança. Assim, este fato pode ocasionar uma dificuldade, por parte dos analistas, para efetuar o processo de elicitação.

Solução: Propõe-se como solução as técnicas de entrevista não-estruturada e questionário. A primeira técnica recomenda-se, pois como o processo de trabalho do cliente se encontra em modificação, a entrevista não-estruturada será apropriada, uma vez que o entrevistador poderá manter o máximo de flexibilidade para direcionar o questionamento no sentido que lhe parecer apropriado, explorando as informações contidas nas respostas ou nas conversas com os indivíduos participantes. Já o questionário poderá ser uma boa maneira de confirmar informações levantadas por outras técnicas aplicadas.

Forças: A vantagem da entrevista não-estruturada é a sua aparente simplicidade, pois entrevistar parece ser uma habilidade que a maioria das

pessoas sentem que possuem, devido a atividade básica do ser humano de falar e se comunicar. Sua limitação é o fato de que ela pode gerar muitos dados, que são difíceis para classificar e analisar. Já o questionário tem como ponto forte ser uma boa maneira de confirmar informações levantadas por outras técnicas. Sua desvantagem é que os questionários não fornecem uma oportunidade de julgar a exatidão das respostas, além de permitir pouca ou nenhuma interação.

Cenário 4:

Nome do Cenário: Problema do domínio debilitado de aplicação e do processo em mudança do cliente

Contexto: Um cliente necessita de um sistema e ele não conhece suas reais necessidades além disso, seu processo de trabalho está em mudança ou em (re)construção, dificultando a determinação das atividades do processo do cliente, por não estarem bem definidas para que os analistas possam elicitá-las.

Problema: O problema origina-se do fato do cliente não possuir o domínio adequado do problema, podendo dificultar o processo de elicitação, na medida em que o cliente ou usuário não transmitirá aos analistas os detalhes pertinentes sobre a aplicação para que o processo de elicitação ocorra efetivamente. Além disso, caso o processo de trabalho do cliente esteja em modificação ou em mudança, isso pode ocasionar dificuldades para os analistas efetuarem o processo de elicitação.

Solução: Como solução propõe-se o uso das técnicas JAD (*Joint Application Development*) e a entrevista estruturada. A primeira técnica é

recomendada porque, possibilita que analistas e usuários trabalhem juntos para projetar o sistema, desde de os requisitos básicos até o *layout* de telas e relatórios. Isso porque JAD poderá ajudar a promover decisões importantes ao permitir que o cliente passe a conhecer as suas reais necessidades, bem como o seu processo de trabalho em mudança. Sugere-se a entrevista estruturada já que poderá ser útil quando se deseja ter a mesma informação de cada entrevistado em momentos diferentes, ou quando há limitação de tempo para o levantamento de dados e análise, possibilitando filtrar informações coletadas por outras técnicas.

Forças: A técnica JAD mostra-se vantajosa quando as sessões reunem especialistas que compartilham suas visões, permitindo compreendê-las e desenvolver o sentimento de posse do projeto. Sua limitação é que algumas pessoas podem não ter a chance de falar sobre suas visões do sistema ou podem sentir-se inibidas por outros membros do grupo. Já a entrevista estruturada tem como ponto forte a possibilidade de coletar sistematicamente dados detalhados, facilitando a comparação das respostas de todos os componentes. Enquanto sua limitação é a de não permitir ao entrevistador explorar os tópicos ou itens que não foram contemplados na elaboração do instrumento da entrevista.

Cenário 5:

Nome do Cenário: Problema do processo do cliente não estar definido

Contexto: Um cliente necessita de um sistema e ele conhece suas reais necessidades. No entanto, seu processo de trabalho não existe ou não está definido, o que dificulta o processo de elicitação de requisitos pelos

analistas. Embora o cliente conheça o domínio do problema, o fato do processo de trabalho não estar definido pode impactar na coleta de requisitos.

Problema: O problema origina-se do fato de que, embora o cliente possua o conhecimento das suas necessidades, seu processo de trabalho não existe ou não está bem definido. Neste cenário, a ausência de um processo, revela uma aparente desorganização das atividades desempenhadas no empreendimento, que também impacta no processo de extração de requisitos quando se deseja desenvolver um sistema.

Solução: Propõe-se como solução a aplicação das técnicas JAD (*Joint Application Development*) e a observação. A primeira técnica pode ser indicada com o propósito de envolver mais os clientes e os usuários no desenvolvimento, promovendo decisões importantes. Já a observação é recomendada na medida em que, como o cliente não possui um processo bem definido, a técnica permitirá que o observador visualize com mais clareza o que os usuários realmente fazem e concentrará a atenção em áreas específicas de interesse.

Forças: A técnica JAD apresenta como ponto forte a possibilidade dos analistas coletarem simultânea e eficientemente uma grande quantidade de requisitos do sistema, junto a uma gama de usuários-chave, além de permitir a consideração das necessidades específicas dos usuários ou opções de projeto. Porém, um ponto fraco do JAD está relacionado ao fato de utilizar uma dinâmica de grupo, em função da qual membros mais tímidos ou menos experientes do grupo poderem ficar relutantes em discutir abertamente

alguns assuntos, caso envolva aos da gerencia, ou de alguns membros mais comunicativos do grupo. Já a observação apresenta como vantagem a possibilidade do observador poder capturar os aspectos sociais e organizacionais, sem necessidade dos usuários falarem explicitamente, além de revelar detalhes que outras técnicas talvez não consigam fornecer. Como ponto fraco a técnica pode capturar uma grande quantidade de informações irrelevantes.

Cenário 6:

Nome do Cenário: Problema do domínio do cliente não ser definido e o processo do cliente estar em mudança.

Contexto: Um cliente necessita de um sistema e ele não conhece suas reais necessidades, porém além disso seu processo de trabalho não existe ou não está definido, dificultando ainda mais o processo de elicitação de requisitos pelos analistas. Uma vez que o cliente não conhece o domínio do problema e não possui um processo bem definido, isso ocasionará considerável lentidão ao processo de elicitação.

Problema: O problema origina-se do fato do cliente não possuir o domínio do problema nem sobre o processo de trabalho. Neste contexto, os analistas terão dificuldade extra em coletar as informações, já que a aparente desorganização das atividades desempenhadas no empreendimento impactarão negativamente no processo de extração de requisitos.

Solução: Propõe-se como solução as técnicas de *brainstorming* e questionário. A primeira possibilitará um processo de interação do grupo que

costuma ser recompensador criando um sentimento de posse ou propriedade do resultado. O questionário é recomendado por ser uma boa maneira de confirmar informações levantadas a partir de um grande número de pessoas, obtidas por outras técnicas.

Forças: O *brainstorming* tem como vantagem o fato de todos do grupo poderem levar crédito pelas boas idéias. Além disso, é uma técnica que não gasta muito tempo para gerar dados úteis, em geral a sessão não necessita durar mais de uma hora. Porém, deve-se cuidar para que as idéias geradas não sejam superficiais. O questionário tem como ponto positivo ser uma maneira propícia de confirmar informações levantadas por outras técnicas. No entanto, sua limitação é a de que os questionários podem ser difíceis de elaborar e realizar a análise de comentários interessantes que não são possíveis manter registros.

Após observar os cenários descritos, a autora deste trabalho sugere o estudo e a aplicação das técnicas abaixo, para as situações baseadas nas características, encontrada nos empreendimentos, sobre os perfis do cliente e do processo, em que se deseja promover o desenvolvimento de um sistema.

	Característica do Cliente	
Característica do Processo	**Sabe o que quer**	**Não sabe o que quer**
Processo Estável	**Cenário 1** Entrevista semi-estruturada e Análise de Documentos	**Cenário 2** Observação e *Brainstorming*
Processo em Modificação ou construção	**Cenário 3** Entrevista não-estruturada e Questionário	**Cenário 4** JAD e Entrevista estruturada
Processo não existente	**Cenário 5** JAD e Observação	**Cenário 6** *Brainstorming* e Questionário

Tabela 4.1 - Características do Cliente x Característica do Processo

4.3 Considerações Finais

Este capítulo teve a finalidade de mostrar as observações adquiridas a partir da análise do questionário aplicado na cooperativa de *Software* Livre e apresentar uma proposta de melhoria do processo de elicitação de requisitos. Para tanto, a proposta exibiu cenários em formato de padrão, na qual descreveu uma solução para um problema que frequentemente ocorre em organizações que prestam serviços de desenvolvimento de *software*.

5 CONCLUSÕES

Foi realizado um estudo sobre a engenharia de requisitos, especificadamente as técnicas de elicitação de requisitos, sobre os padrões existentes no desenvolvimento de *software*. Também foram examinadas as técnicas mais comumente utilizadas em uma cooperativa que desenvolve *software* livre. Após tais estudos foi proposta uma melhoria para o processo de elicitação de requisitos para solucionar os problemas mais comuns na atividade de extração de requisitos. Espera-se, com isso, que com os estudos de melhoria do processo de elicitação para a cooperativa, traga um impacto positivo no desenvolvimento de *software*.

Os requisitos descrevem o sistema e devem instruir os desenvolvedores para a construção do sistema, sendo necessário obtê-los com maior precisão possível. Dispensar esforço e tempo buscando requisitos de qualidade é extremamente importante para o processo de *software* de maneira geral, pois possibilita a economia de trabalho e custo neste processo. Os engenheiros de requisitos necessitam de meios de definir os requisitos, o estudo de melhoria proposto poderá ser visto como uma opção para alcançar requisitos com as características necessárias.

Os novos estudos para a melhoria do processo de elicitação de requisitos definido neste trabalho, por enquanto, é uma proposta, que necessita ser implantada e avaliada. A partir disso, a cooperativa dará seqüência em seu objetivo de melhorar a atividade de elicitação de requisitos e a produção de *software* de qualidade.

5.1 Trabalhos Futuros

A partir deste trabalho, alguns outros poderão ser desenvolvidos para chegar ao objetivo final que a cooperativa deseja. Alguns destes trabalhos são:

- Utilizar a proposta de melhorias do processo de elicitação em um projeto piloto para avaliá-lo;
- Propor melhoria para essa proposta de elicitação, a partir da sua avaliação;
- Adaptar a proposta de melhoria de processo de elicitação de requisitos a um processo de requisitos existente;
- Fazer um estudo sobre as ferramentas de gerenciamento e de automação em um processo de elicitação de requisitos e; como isso, escolher as melhores para serem aplicadas pela proposta definida neste trabalho.

6 REFERENCIAIS BIBLIOGRÁFICOS

APPLETON, Brad. *Patterns and Software: Essential Concepts and Terminology*, disponível em: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.41.2738&rep=rep1&type=pdf. Último acesso: 09/11/2009.

BATISTA, E. A. **Uma Taxonomia facetada para Técnicas de Elicitação de Requisitos**. Trabalho final (mestrado profissional) – Universidade Estadual de Campinas, Instituto de Computação. Campinas (SP), 2003. Disponível em: http://libdigi.unicamp.br/document/? code=vtls000316140. Último acesso em 09/09/2009.

BELLINO, R., CASE, S.; **Como Conduzir Uma Sessão Criativa E Bem-Sucedida de "*Brainstorming*"?.** Disponível em: http://www.catho.com.br/jcs/inputer_view.phtml?id=7252. Consulta em 01/10/2009.

BELGAMO, A. e MARTINS, L. E. G. **Um Estudo Comparativo sobre as Principais Técnicas de Elicitação de Requisitos do Software.** Publicado em outubro de 2000 no 8º Congresso de Iniciação Científica UNIMEP/CNPq.

CAMACHO, C. **Gerenciando Conflitos em Reuniões: Uma estratégia para a Elicitação de Requisitos de Software.** Rio de Janeiro: PUC-Rio, Departamento de Informática, 2005.

CARNEIRO, C. M. P. DA SILVA. **Frameworks de aplicações orientadas a objetos – Uma abordagem interativa e incremental.** Universidade Salvador – UNIFACS – Bahia. Dissertação de Mestrado, 2003.

DIGIAMPIETRI, L. A. **Extração de requisitos.** Notas de aula 2004. Disponível em: http://www.ic.unicamp.br/~luciano/mc426/modulo2-v.pdf. Consulta em 20/10/2009.

IEEE-SA STANDARDS BOARDS IEEE Std 1233-1998: **IEEE Guide for Developing System Requirements Specifications.** Dezembro 1998.

FALBO, R. A., **Engenharia de Software**. UFES – Universidade Federal do Espírito Santo, 2005.

GOMES, A. S. e WANDERLEY, E. G.: **Elicitando requisitos em projetos de software educativo**. WIE'2003, Campinas (SP);

JOHNSON, R. E.; RUSSO, V. **Reusing Object-Oriented Designs.** Relatório Técnico da Universidade de Illinois, UIUCDCS 91-1696, 1991.

JUNG, C. F. **Metodologia para pesquisa & desenvolvimento: aplicada a novas tecnologias, produtos e processos**. Rio de Janeiro/RJ. Axcel Books, 2004.

KIRAWOWSKI, J. **Requirements Engineering and Specification in Telematics: Methods for User-Orientated Requirements Specification.** Human Factors Research Group, Cork, Ireland,1997. Disponível em http://www.ucc.ie/hfrg/projects/respect/urmethod/brains.htm. Consulta em 01/10/2009.

LOPES, P. S. N. D. **Uma taxonomia da Pesquisa na Área de Engenharia de Requisitos**. Dissertação de Mestrado, Março de 2002, São Paulo (SP).

MALDONADO, J. C.; BRAGA, R. T. V.; GERMANO, R. S. R.; MASIERO, P. C. **Padrões e Frameworks de Software.** Notas Didáticas. Disponível em: http://scholar.google.com.br/scholar?q=Padr%C3%B5es+e+Frameworks+de+Software&hl=pt-BR&lr=&btnG=Pesquisar&lr=. Último acesso: 17/06/2009.

PAULA FILHO, W. de P. **Engenharia de software: fundamentos, métodos e padrões.** Wilson de Pádua Paula Filho. 2. ed. Rio de Janeiro: Livros Técnicos e Científicos, 2003.

RIBEIRO, R. T., MONSERRAT. N., J. **SIMP: *Framework* para o Desenvolvimento de Sistemas de Informação Modulares em PHP**. Departamento de Ciência da Computação – Universidade Federal de Lavras.

SCHMIDT, D. C., BUSCHAMANN, F. (2003) **Patterns, Frameworks, and Middleware: Their Synergistic Relationships**, In: Software Engineering, 25th International Conference on Software Engineering (ICSE'03), pages 694-704.

SILVA, RICARDO PEREIRA. **Suporte ao Desenvolvimento e Uso de Frameworks e Componentes**. 2000. 262 f. Tese (Doutorado em Ciência da

Computação) – Universidade Federal do Rio Grande do Sul, Instituto de Informática, Rio Grande do Sul.

SOARES, E. S. **SWREQUIREMENT: Uma Proposta de Integração de Técnicas de Elicitação de Requisitos ao Processo de Levantamento e Análise de Requisitos**. Monografia de Graduação. Universidade Federal de Lavras, Departamento de Ciência da Computação. Minas Gerais, 2007, 47p.

SOMMERVILLE, I. **Engenharia de Software**. São Paulo: Addison-Wesley, 6ª edição, 2003.

TECNOLIVRE – Cooperativa de Tecnologia e Soluções Livres, disponível em: <http://www.tecnolivre.com.br>, acessado em 18/05/2009.

TORO, A .D.; JIMÉNEZ, B. ***Metodología para la Elicitación de Requisitos de Sistemas de Software***. Informe Técnico LSI-2000-10. Faculdad de Informática y Estadística Universidad de Sevilha, Outubro, 2000. Disponível em: http://www.info-ab..uclm.es/42541/pdf/metodologia_elicitaction.pdf. Último acesso: 15/09/2009.

VIANNA, E. C. C. M. **Estudo e proposta de práticas participativas na Gestão de Requisitos.** Trabalho Final de Mestrado. Universidade Estadual de Campinas, Instituto de Computação. Campinas (SP), 2004.

WESSELS, David. ***Requirements and analysis. Lectures Notes for Systems Analysis*** – Computing Science Department, Faculty of Science and Techonology Malaspina University-College, 2002. Disponível em: http://csciun1.mala.bc.ca:8080/~wesselsd/csc375/notes.html. Consulta em 01/10/2009.

ANEXOS

Anexo A – Questionário Avaliativo das Técnicas de Elicitação de Requisitos

Questionário avaliativo das Técnicas de Elicitação de Requisitos
1- Como é realizada a primeira coleta de informações junto ao cliente para um projeto de desenvolvimento de sistema? Qual forma é realizada?
2- É utilizada reuniões de grupos, ou seja, geração de idéias, para coletar informações (requisitos) dos clientes? Esse tipo de técnica é muito utilizada?
3- O empreendimento já utilizou técnicas de sessões estruturadas, através de dinâmica de grupo e recursos visuais, em que analistas (cooperados) e usuários (cliente) trabalham juntos para projetar um sistema? Se já, que tipo de serviço foi utilizada esta técnica e com que freqüência?
4- Já utilizou-se de questionários para elicitar requisitos junto ao cliente?
5- A cada reunião com o cliente, são utilizados os mesmos critérios de coleta de requisitos, ou seja, a mesma técnica é empregada em diversos estágios da elicitação?
6- A observação das atividades dos clientes já foi empregada para coletar requisitos no desenvolvimento de sistemas? De que forma? Com que frequência foi utilizada?
7- A análise de documentos é muito utilizada na coleta de informações para o desenvolvimento de sistemas?
8- Já foram utilizados protótipos (modelo de software) como forma de elicitação de requisitos?
9- Quais os perfis de clientes? E como você os definiria? (Ex: clientes que sabem o que quer ou aqueles que não sabem o que quer e assim por diante)
10- Qual o problema que a equipe de elicitação de requisitos enfrenta de acordo com o tipo de cliente?