Uma Taxonomia Facetada para Técnicas de Elicitação de Requisitos

Edinelson Aparecido Batista

Trabalho Final de Mestrado Profissional

Instituto de Computação
Universidade Estadual de Campinas

Uma Taxonomia Facetada para Técnicas de Elicitação de Requisitos

Edinelson Aparecido Batista

29 de Agosto de 2003

Banca Examinadora:

- **Profa. Dra. Ariadne Maria Brito Rizzoni Carvalho (Orientadora)**
  IC - Unicamp
- **Prof. Dr. Luiz Eduardo Galvão Martins**
  Unimep
- **Profa. Dra. Cecília Mary Fischer Rubira**
  IC - Unicamp
- **Profa. Dra. Anamaria Gomide (Suplente)**
  IC- Unicamp

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA DO IMECC DA UNICAMP

Batista, Edinelson Aparecido

B32t Uma taxonomia facetada para técnicas de elicitação de requisitos / Edinelson Aparecido Batista -- Campinas, [S.P. :s.n.], 2003.

Orientador : Ariadne Maria Brito Rizzoni Carvalho

Trabalho final (mestrado profissional) - Universidade Estadual de Campinas, Instituto de Computação.

1. Engenharia de software – Metodologia. 2. Software – Desenvolvimento. 3. Análise de sistemas. I. Carvalho, Ariadne Maria Brito Rizzoni. II. Universidade Estadual de Campinas. Instituto de Computação. III. Título.

Uma Taxonomia Facetada para Técnicas de Elicitação de Requisitos

Este exemplar corresponde à redação final do Trabalho Final devidamente corrigido e defendido por Edinelson Aparecido Batista e aprovado pela Banca Examinadora.

Campinas, 29 de Agosto de 2003.

Ariadne M. B. R. Carvalho

Profa. Dra. Ariadne Maria Brito Rizzoni Carvalho
(Orientadora)

Trabalho Final apresentado ao Instituto de Computação, UNICAMP, como requisito parcial para a obtenção do título de Mestre em Computação na área de Engenharia de Computação

Aos meus pais,

com muita gratidão

Agradecimentos

A Deus pela minha vida, saúde e trabalho;

À Nossa Senhora Aparecida pela ajuda na superação dos momentos difíceis;

À Gisele pelo apoio e paciência;

À minha família (pais, irmãos, irmãs, cunhados, cunhadas, sobrinhas, sobrinho) que sempre soube compreender as minhas ausências nos churrascos e almoços dominicais;

À minha orientadora Ariadne, que sempre me recebeu com sorrisos e me deu bastante ânimo na condução deste trabalho;

Aos colegas da Secretaria de Economia e Planejamento do Estado de São Paulo que me apoiaram na fase inicial do MP;

À gerência do SSC-IT/CAS da Philips do Brasil que me liberou de horas preciosas de trabalho para participar das reuniões com a orientadora e para elaborar este trabalho;

Aos demais colegas do SSC-IT da Philips do Brasil que sempre me deram apoio e estímulo para não desistir e ouviram com paciência (nem sempre!) as minhas lamentações e preocupações;

A todos os usuários com quem já trabalhei nestes anos. Com certeza muito do que aprendi com eles está neste trabalho.

Resumo

O processo de engenharia de requisitos é peça-chave para o sucesso ou fracasso de um sistema. Dentre as diversas fases que compõem a engenharia de requisitos, as principais são: elicitação, análise, especificação e validação. A elicitação de requisitos é a fase em que o desenvolvedor se preocupa com a descoberta dos requisitos do sistema. É a mais subjetiva das fases e, por ser basicamente dependente de seres humanos, dá margem a interpretações diferentes e ambíguas. A definição dos requisitos não é um processo matemático e há fatores organizacionais, técnicos e sociais envolvidos.

Desde que a engenharia de requisitos surgiu no início dos anos 90, os desenvolvedores de software têm se defrontado com a questão de encontrar a melhor forma para a identificação dos requisitos do sistema e diversas técnicas e métodos têm sido propostos.

Durante o processo de elicitação de requisitos várias fontes de informação são consultadas pelos desenvolvedores, mas as pessoas são as mais importantes. Apesar de todo avanço tecnológico, o que se percebe é que a elicitação é feita de maneira bastante informal, muitas vezes com aplicação de técnicas inadequadas; isso pode ensejar erros que se propagam para as fases seguintes do ciclo de vida do software, gerando um produto com falhas, não confiável e sem qualidade. Diversas técnicas podem ser aplicadas nesta fase para ajudar os desenvolvedores a descobrir o que o usuário realmente necessita.

O objetivo desta monografia é propor uma taxonomia para as técnicas utilizadas na fase de elicitação de requisitos, através de um esquema de classificação facetado. As técnicas são classificadas segundo uma lista de parâmetros, ou facetas, que podem auxiliar os desenvolvedores na escolha das técnicas que serão utilizadas na elicitação. As técnicas são descritas e seus processos relacionados, bem como as situações em que seu uso é indicado. A escolha correta da técnica de elicitação cria condições para que os requisitos sejam melhor especificados e para que as possíveis falhas no desenvolvimento de sistemas fiquem num patamar aceitável e tolerável.

Abstract

The requirements engineering is fundamental to the success or failure of a computational system. The requirements engineering is composed of several phases, being elicitation, analysis, specification and validation main ones. The requirements elicitation phase is concerned with discovering the requirements of the system. It is the most subjective of the phases and, because it depends on human beings, it is possible that different and ambiguous interpretations are produced. The definition of requirements is not a mathematical process; there are organizational, technical and social factors involved.

Since the requirements engineering appeared at the beginning of the 90´s, the requirements engineer has faced the problem of finding the best form of identifying the system´s requirement and, since then, several techniques and methods have been proposed.

During the requirements elicitation process, the requirements engineer consults several sources, but people are essentially the most important source. In spite of the technological advance, elicitacion still takes place in an informal way; application of inadequate elicitacion techniques produces errors which propagate to the following development phases, generating a product with failures, not trustworthy and without quality. Several techniques can be applied to help the requirements engineer to discover what the users really need.

The goal of this work is to propose a taxonomy for the techniques used in the requirements elicitation phase, through the use of a faceted classification scheme. The techniques are classified according to a list of parameters, or facets, that can help the requirements engineer to choose the techniques that could be used in the elicitacion phase. The techniques are described and their processes are related, as well as the situations when they should be applied. The use of the appropriate technique leads to better specified requirements, decreasing the problems in the systems´s development as a whole.

Conteúdo

Índice de Figuras

Índice de Tabelas

Capítulo 1

Introdução

Ao longo da história, a humanidade vem sofrendo várias inovações tecnológicas que mudaram definitivamente a maneira do homem viver, trabalhar e se relacionar. Os instrumentos de pedra foram substituídos pelos instrumentos de metal, que contribuíram para o surgimento de uma revolução agrícola. Depois, com as novas descobertas e invenções, que deram base à Revolução Industrial, as mudanças ocorreram cada vez com maior velocidade e estão constantemente transformando a vida do homem. Agora, estamos vivendo uma nova revolução: a da informação. [TROPE2000] citando Alvin Toffler: "...estamos começando a viver a terceira onda, quando inovações tecnológicas originam mudanças nas formas de relacionamento".

[TROPE2000] observa ainda que "presenciamos a passagem de uma sociedade industrial, na qual o recurso estratégico é o capital, para uma sociedade da informação, na qual tal recurso é a informação. Esta passa a ter valor econômico, pois possui um custo para ser produzida e as pessoas estão dispostas a pagar por ela".

Com estas mudanças muitas organizações estão rediscutindo o seu papel, buscando mais flexibilidade, competitividade e produtividade, já que os pressupostos da concorrência também se modificam. A Tecnologia da Informação (TI) surge como a nova situação tecnológica capaz de propiciar a estas organizações condições para sobreviver em um mercado globalizado. Se em períodos anteriores as inovações tecnológicas demoravam mais para alcançar a vida das pessoas, agora as mudanças são praticamente instantâneas e impõem profundas transformações nos campos econômico, social e tecnológico. A velocidade e a grande quantidade das informações globalizadas estão derrubando barreiras econômicas e políticas, criando novos valores e novas formas de se fazer transações econômicas, de trabalhar e de se relacionar.

Para [TROPE2000] "estamos vivendo uma era de profundas mudanças nos campos político, econômico, tecnológico, social e dos valores pessoais. A alta complexidade e velocidade das informações, a interdependência dos fenômenos, o desenvolvimento de uma economia sem fronteiras e regionalismos, um elevado desenvolvimento tecnológico, a alta competitividade entre as empresas e uma crescente exigência dos consumidores podem ser citados como aspectos da época contemporânea. A velocidade de transmissão das informações derruba barreiras antes existentes."

[PASTORE1999] ressalta que "a revolução tecnológica produz mudanças a fundo. Na década de 70, uma inovação industrial durava, como novidade, 2 anos; na década de 80, 1 ano; hoje, 6 meses. Em certos campos, menos que isso. Na informática e nas telecomunicações, as novidades duram três ou quatros semanas. Muitas se tornam obsoletas quando entram no mercado."

Com a globalização, cada vez mais as pessoas e organizações dependem de sistemas automatizados. As novas tecnologias de telecomunicação e computação criaram condições necessárias para a explosão da Internet e, com isto, as informações são facilmente acessadas, ou disponibilizadas a qualquer pessoa, independentemente do local em que ela se encontra. A execução de tarefas que dependem da informação ganharam formas diferentes para serem desempenhadas, pois não há mais limite de distâncias impedindo o acesso a elas. Este acesso cada vez mais facilitado à Internet resulta na possibilidade de coleta e envio de uma grande quantidade de informação, anteriormente dificultada por distâncias físicas. Tudo isto está causando mudanças definitivas na vida das pessoas. A atual situação tecnológica disponibiliza vários recursos que podem ajudar as empresas ou organizações a serem competitivas em um mundo de competição acirrada. Quem melhor utilizar estes recursos pode ter vantagens sobre seus concorrentes.

Para [PRESSMAN1995] o "software é um fator que diferencia. A inteireza e a oportunidade das informações oferecidas pelo software diferenciam uma empresa de suas concorrentes. O projeto e a capacidade de ser 'amigável ao ser humano' de um produto de software diferenciam-no dos produtos concorrentes que tenham função idêntica em outros aspectos. A inteligência e a função oferecidas pelo software muitas vezes diferenciam dois produtos de consumo ou indústrias idênticas. É o software que pode fazer a diferença."

Hoje, qualquer oscilação nas bolsas em Nova Yorque, por exemplo, é sentida instantânea e simultaneamente no mundo todo, interferindo na economia dos países. Era difícil imaginar isto há poucos anos: quanto tempo as notícias sobre a 1a. e 2a. guerras mundiais levavam para chegar até as pessoas (no Brasil)? Como era feita uma compra externa (internacional) há 20 anos? Como eram os serviços bancários na década de 70? Apesar de ainda existir uma grande parcela da população sem acesso a estes avanços, principalmente nos países em desenvolvimento, é notório o aumento de pessoas com acesso às novas tecnologias. Em praticamente todas as áreas de atuação do homem existem sistemas computacionais, ou softwares presentes. Indústria, comércio, medicina, transportes, educação, esportes, lazer, governo, serviços, são algumas das áreas em que existem softwares cada vez mais sofisticados, concebidos para melhorar, pelo menos em tese, a nossa vida.

Nos últimos anos houve uma redução dos preços dos computadores. [CARVALHO+2001] ressalta que "durante a década de 80, avanços na microeletrônica resultaram em aumento do poder computacional a um custo menor. Entretanto, tanto o processo de desenvolvimento como o software produzido ainda deixavam muito a desejar: cronogramas não eram cumpridos, custos excediam o valor orçado, o software não cumpria o estipulado". O barateamento do hardware e seu poder de processamento cada vez maior, fizeram com que a produção de software se desse em escala industrial, pois houve um aumento do mercado de computadores. Mas para [LOPES2002] "o software transformou-se na parte mais cara dos sistemas computadorizados complexos, sendo desenvolvido com baixa qualidade, com pouca previsibilidade de custo e recursos, colocando em risco o avanço da tecnologia de hardware. A necessidade de se encontrar métodos que pudessem ajudar na evolução do processo de construção de software, culminaram com o desenvolvimento da Engenharia de Software."

A Engenharia de Software (ES) introduziu uma série de conceitos e propostas para melhorar o processo de desenvolvimento de software. Muitos se tornaram padrão e se popularizaram como, por exemplo, a Análise e Projeto Estruturados de Sistemas [GANE1988];[MARTIN+1991], Análise Essencial e a Análise e Projeto Orientados a Objetos [JACOBSON+1992]. Contudo, para [PORTELLA1994] "apesar de muitas dessas propostas de soluções terem se transformado em realidade, a maior parte delas ainda persiste apenas como promessa. Como se não bastasse esse fato, parte significativa das promessas que se transforma em software, não traz a esperada satisfação de seus usuários, fato constatado mundialmente, e que tem sido cunhado pela literatura técnica como sendo a 'crise do software' ".

[PORTELLA1994] ressalta ainda que "a quase totalidade do software produzido é criada como objeto de arte, para o qual o construtor é um artista ou artesão e a criatividade é a grande ferramenta. O ideal seria que o software estivesse sendo desenvolvido como artefato de manufatura, onde o construtor é um técnico e o rigor científico das bases de seu desenvolvimento, sua principal ferramenta." Para tentar resolver este problema foram desenvolvidas ferramentas e metodologias que ajudam a definir, apoiar e aplicar o processo de desenvolvimento de software de uma maneira produtiva, eficaz e com qualidade, dando a ele um caráter científico. Entretanto, softwares continuam a ser produzidos com erros, causando prejuízos às organizações e pessoas, principalmente, devido às falhas nas definições dos seus requisitos.

Até há pouco tempo era aceitável e comum que sistemas fossem desenvolvidos em dois ou até três anos, considerando desde o início do levantamento dos requisitos até a sua entrada em produção. Hoje, isto é praticamente inaceitável, existindo, naturalmente,

algumas poucas exceções. Em função da acirrada concorrência entre empresas, deve-se desenvolver o software cada vez mais rapidamente.

"A ES herda da engenharia o conceito de disciplina na produção de software, através de metodologias que, por sua vez, seguem métodos que utilizam-se de ferramentas automatizadas para englobar as principais atividades do processo de produção de software." [CARVALHO+2001].

Segundo [CARVALHO+2001] "o objetivo da ES é auxiliar no processo de produção de software, de forma que o processo dê origem a produtos de alta qualidade, produzidos mais rapidamente e a um custo cada vez menor"; para [PORTELLA1994], também devem ser aplicados conhecimentos científicos neste processo.

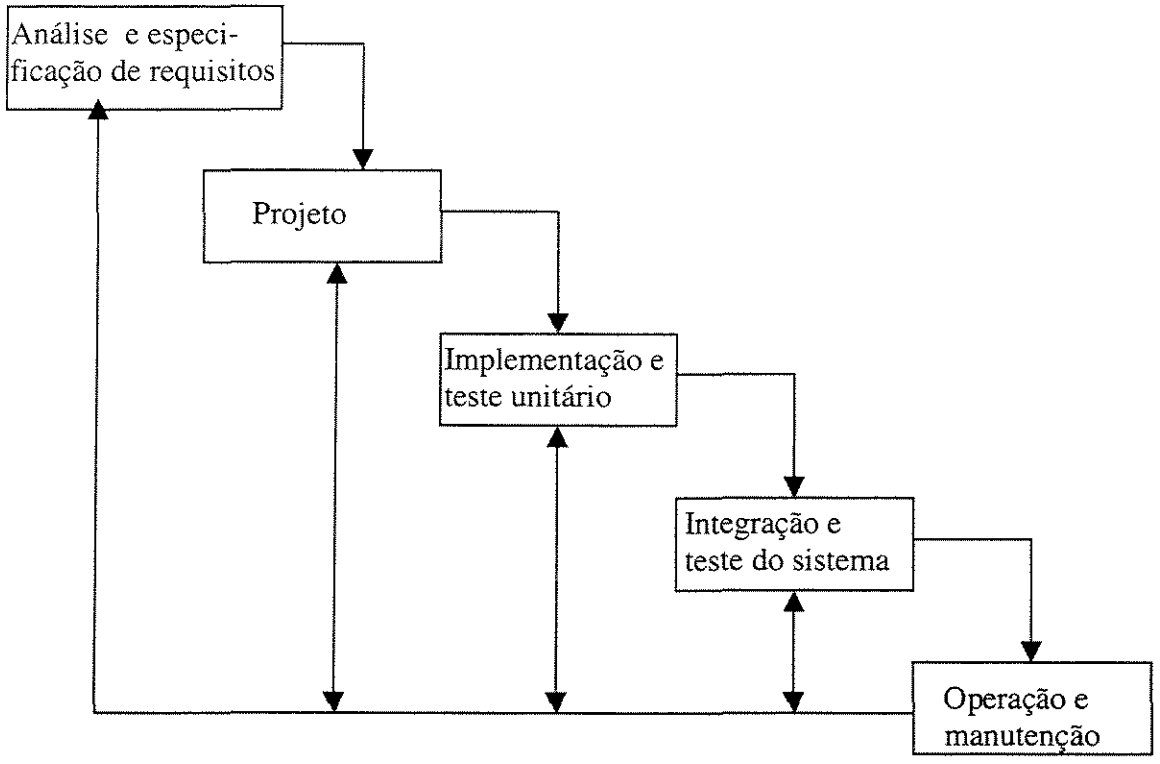

Figura 1 - Modelo de ciclo de vida clássico ou cascata

A construção de um software é um processo contínuo, que se inicia a partir da necessidade da solução de um problema. Pode-se dizer contínuo pelo fato de que os requisitos e tecnologias aplicados para a solução estão permanentemente sofrendo alterações. Daí vem a importância de que um desenvolvimento de software seja o mais

adequado possível às constantes mudanças exigidas ao longo de sua vida, também conhecida como ciclo de vida do software. Dentre os diversos modelos de ciclo de vida do software existentes, podemos citar como os mais comuns, o modelo clássico ou em cascata mostrado na figura 1, em espiral e a prototipação.

[LOPES2002] diz que "a abordagem adotada pelo modelo em cascata é bastante simplista e, de forma geral, não pode ser utilizada na construção de sistemas moderadamente complexos". Entretanto, o modelo em cascata pode ser utilizado nestes sistemas quando se tem uma boa idéia do que fazer ou experiência com algum desenvolvimento semelhante. Na prática, ao lidar com sistemas complexos, é discutível se devemos ou não esperar até que tenhamos todos os requisitos do sistema identificados para podermos prosseguir com sua construção. [LOPES2002] continua argumentando que "tampouco os requisitos são estáveis a ponto de podermos supor que eles permaneçam estáticos até o final da construção do software".

[ZANLORENCI+1998] mostra que o interesse por uma abordagem sistemática do conhecimento do problema levou à busca de apoio de uma nova área de pesquisa, a Engenharia de Requisitos (ER).

A ER surge para organizar o processo de definição de requisitos. Ela é uma atividade essencial da ES, pois visa resolver estes problemas, mostrando claramente o que deve ser feito, procurando estabelecer definições claras, precisas e não ambíguas. Desta maneira ela se propõe a reduzir os problemas no desenvolvimento de software.

ER pode ser descrita como um processo de engenharia que consta de uma série de atividades que vão engendrar, fundamentalmente, uma especificação do produto de software que se decidiu construir, após extrair e analisar os requisitos do sistema. A ER é peça chave, um fator fundamental, para o sucesso ou fracasso de um sistema computacional e sua má gerência pode aumentar o custo de desenvolvimento, extrapolando o valor orçado e também ao descumprir o cronograma estabelecido. Num mundo altamente competitivo, nenhuma organização gostaria de arcar com estes desvios, erros e falhas. A organização não pode desempenhar suas atividades satisfatoriamente se o software que ela se propôs a desenvolver ou utilizar não a atende em suas atividades básicas, essenciais e diárias. "O sucesso de um sistema de informação depende fortemente da qualidade da definição dos requisitos" [PORTELLA1994].

Desde que surgiu, no início dos anos 1990 até agora, a ER tem se transformado, sua importância foi admitida e foram reconhecidos os riscos a que um projeto de software pode estar sujeito se este processo se realiza de forma incompleta ou incorreta. Entretanto, seu pleno uso ainda não foi alcançado. [LOPES2002] relata que numa recente conferência de

ER, a ICRE2000[1], foi conduzida uma seção de discussão com o tema "*Porque é tão difícil introduzir os resultados da pesquisa na área de Engenharia de Requisitos na prática da Engenharia de Requisitos?*". Após duas horas de intenso debate, o ponto comum entre todos os debatedores foi que a pesquisa ainda não está suficientemente madura para oferecer uma base sólida para a utilização das técnicas prescritas na prática. [LOPES2002] observa ainda que "historicamente, novas tecnologias saem do estado da arte para o estado da prática em aproximadamente 20 anos." Nestes termos, teríamos mais ou menos 10 anos de pesquisas em ER para que ela produza resultados práticos e satisfatórios.

A ER é a mais subjetiva das fases e, por ser dependente dos seres humanos, é possível ter interpretações diferentes e ambíguas. Paradoxalmente, com todo o avanço tecnológico ainda existe desenvolvimento de software de maneira informal, no qual não é aplicada técnica alguma de coleta de requisitos ou são coletados por pessoas não capacitadas, resultando em um produto não confiável e sem qualidade. Entretanto, a utilização da ER no processo de desenvolvimento de software está propiciando o surgimento de normas, roteiros, padrões e interpretações que permitem o estudo da conformidade de processos e produtos da ER com padrões de qualidade que visam garantir a satisfação do usuário. Para [CARVALHO+2001a] construir sistemas em tempo hábil para serem úteis aos negócios, com a qualidade e custos adequados à sua importância para a organização, é o desafio que todos os desenvolvedores estão enfrentando.

Uma das maiores preocupações que atualmente envolve a ES são os erros de requisitos. [DAVIS+2000] mostra que 71% de todos os projetos de desenvolvimento de software resultam em falhas, com conseqüente cancelamento prematuro de seu uso por parte do usuário final ou são engavetados antes de entrar em operação. Mau gerenciamento dos requisitos é, freqüentemente, considerado uma das maiores causas de falhas dos produtos de software. Afinal de contas, se são feitas uma má extração e uma má compreensão das necessidades dos usuários, se é feita uma pobre e incorreta definição das características do produto a ser construído, se é feita uma má transcrição/descrição do que se pensa e do que se quer de um sistema, não há como se esperar pelo sucesso de um projeto de software [DAVIS+2000]. Todas as técnicas e ferramentas de desenvolvimento de software não terão a menor utilidade se não for construído o produto correto pretendido pelo usuário.

Devido ao grande interesse que a ER tem despertado na comunidade dos desenvolvedores de software, ela vem sendo foco de estudos e pesquisas e diversas ferramentas, métodos, técnicas e padrões têm sido propostos. Livros, artigos, revistas vêm sendo publicados e conferências e seminários realizados. Mesmo com tantas pessoas

[1] *Fourth IEEE International Conference on Requirements Engineering* - Junho/2000

engajadas e pesquisas em andamento, ainda não se têm grandes resultados e, portanto, a ER ainda não está suficientemente madura. Com tantas pessoas pesquisando e escrevendo sobre o assunto, propondo diferentes maneiras para lidar com as fases da ER, várias nomenclaturas são geradas, quase sempre com o mesmo objetivo, porém causando, muitas vezes, diversas interpretações. Entendimento do domínio, elicitação[2], análise, validação, especificação e documentação, gerenciamento dos requisitos podem ser consideradas fases do processo de ER.

A ER tem que garantir que os requisitos sejam bem definidos para que o projeto do sistema não comece a enfrentar problemas. Ela tem que assegurar que os requisitos do negócio da organização e dos usuários sejam todos contemplados e especificados. Se a elicitação de requisitos não for bem feita, a análise, a especificação e a documentação dos requisitos ficarão comprometidas e a qualidade final dos requisitos ficará sob suspeita, inviabilizando todo o processo de ER e, conseqüentemente, a fase de projeto.

Descobrir o que o usuário realmente quer ou necessita é uma das tarefas mais difíceis do processo de desenvolvimento de software. Para [GOGUEN+1993] um projeto pode estar condenado por alguns fatores: nenhum sistema pode ser construído caso os seus requisitos não sejam satisfeitos ou se os requisitos acordados não refletem as necessidades reais. [DEBORTOLI1998] relata que as conseqüências de um desenvolvimento com uma inadequada definição de requisitos são: sistemas que não atendem às necessidades dos usuários, aumento dos custos, realização de atividades desnecessárias, usuários insatisfeitos, desentendimentos com desenvolvedores e aumento da tarefa de manutenção. O objetivo principal da ER é evitar tais problemas e isto envolve **um significativo esforço na fase de elicitação para descobrir os requisitos. Portanto, é necessário que esta fase seja desempenhada com critérios e técnicas.**

Diversas técnicas podem ser aplicadas à fase de elicitação dos requisitos para ajudar a equipe de desenvolvimento a descobrir o que o usuário realmente necessita. Técnicas tradicionais estão convivendo cada vez mais com novas técnicas, para juntas aprimorar a identificação dos requisitos e diminuir os problemas de manutenção de sistemas causados por má definição de requisitos.

As técnicas estruturada e orientada a objetos são as duas abordagens mais utilizadas na análise e especificação de sistemas atualmente; contudo para [DEBORTOLI1998] estas abordagens não auxiliam eficientemente os analistas na coleta de informações entre os

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

usuários, a fim de ajudá-los no entendimento do problema. Estas abordagens assumem que a atividade de elicitação dos requisitos foi previamente efetuada.

A elicitação tem uma relação muito forte, e com bastante dependência, com as pessoas que solicitaram o desenvolvimento do software e com as pessoas escolhidas como fontes de obtenção das informações e, nesta relação, sempre está presente o problema da comunicação entre estas pessoas e os desenvolvedores. Cada uma destas pessoas tem formação, conhecimentos, valores e pontos de vista diferentes, o que acaba gerando uma enorme barreira na comunicação entre os envolvidos. A maioria das técnicas de elicitação atua junto às pessoas para obter as informações e tentam resolver os problemas de comunicação e, principalmente as mais recentes, também se preocupam com os aspectos organizacionais, sociais e psicológicos. Os problemas da elicitação de requisitos não podem ser resolvidos de uma maneira puramente tecnológica, pois o contexto social nesta fase é muito mais crucial do que em outras fases como a especificação, projeto e programação [GOGUEN+1993].

A escolha das técnicas que serão utilizadas na fase de elicitação não é uma tarefa trivial. Uma má escolha pode causar os problemas descritos anteriormente, entre eles a má especificação de requisitos. Cada tipo de sistema pode requerer técnicas diferentes para a elicitação de requisitos e, para ter maiores possibilidades de escolha e aplicação, a equipe de desenvolvimento deve conhecer o maior número possível destas técnicas.

O objetivo deste trabalho é propor parâmetros que possam ajudar a equipe de desenvolvimento a ter melhores condições de escolher as técnicas que serão utilizadas na fase de elicitação de requisitos. Para esta escolha ser boa e coerente é necessário conhecer alguns parâmetros de comparação entre as diversas técnicas. Para isto, é sugerida uma taxonomia das técnicas de elicitação, classificando-as através da utilização de um esquema de classificação facetado. Esta taxonomia propõe a classificação das técnicas em diversas facetas que, por sua vez, são subdivididas em termos. As facetas foram definidas a partir de parâmetros julgados úteis para dar suporte à escolha como, por exemplo, o papel exercido pelo usuário, formalidade, o nível de participação do usuário, os custos em termos de tempo e esforços, as habilidades do desenvolvedor dentre outros. Os termos foram definidos a partir do valor que o parâmetro pode assumir. Exemplificando, termo no qual o papel do usuário pode ser consultivo ou decisório; outro no qual a técnica pode requer muita ou pouca participação do usuário. As facetas e os seus termos são identificados através de elementos-chave encontrados na definição, descrição e nas características de cada técnica.

Para dar uma boa visão das técnicas, além das descrições são listados suas limitações, benefícios, desvantagens, recursos exigidos, situações em que são melhor aplicadas e

diretrizes de utilização. Este conjunto de tópicos fornece as facetas e termos, ou parâmetros, para escolha da técnica eventualmente indicada para ser aplicada.

Este processo é realizado independentemente da metodologia de desenvolvimento de sistemas utilizada e o objetivo é criar condições para que as possíveis falhas da elicitação sejam resolvidas, não se propagando para outras fases da ER e nem para a fase de projeto de desenvolvimento do sistema.

Portanto, este trabalho mostra alguns critérios que devem ser levados em consideração para escolher uma técnica de elicitação. Eliminar as falhas de requisitos é praticamente impossível; porém, o uso correto de técnicas de elicitação pode diminuí-las e evitar que causem danos graves ao sistema, restringindo possíveis falhas a um patamar aceitável e tolerável.

1.1 Organização Deste Documento

Este documento foi dividido em 7 capítulos, organizados e divididos da seguinte forma:

- **Capítulo 2 – A Engenharia de Requisitos:** São definidos os conceitos de requisitos e de ER. É traçado um panorama da evolução da ER no processo de desenvolvimento de sistemas. São mostradas também as diversas formas de dividir a ER encontradas na literatura. Em seguida, são explicadas as fases da ER propostas para este trabalho e são brevemente discutidas as seguintes: entendimento do domínio, elicitação e análise, especificação e documentação e validação. Também é discutido o gerenciamento dos requisitos, que deve estar presente em todas as fases e são abordados aspectos de qualidade que devem estar presentes nos requisitos e nas especificações.
- **Capítulo 3 - Taxonomia e Esquemas de Classificação**: São apresentados os conceitos de taxonomia e esquemas de classificação. A classificação facetada, que é o embasamento teórico desta monografia, é introduzida e exemplificada.
- **Capítulo 4 - Trabalhos Correlatos Abordando Classificações das Técnicas de ER:** São mostradas algumas formas de classificação das técnicas de ER encontradas na literatura.
- **Capítulo 5 - Parâmetros para a Escolha das Técnicas de Elicitação de Requisitos:** São apresentados alguns parâmetros que podem ser considerados pelos desenvolvedores no momento de escolha da técnica.
- **Capítulo 6 - A Taxonomia Proposta:** É apresentada a taxonomia das técnicas de elicitação de requisitos proposta. Diversas técnicas são classificadas na

taxonomia, utilizando o esquema de classificação facetado; o uso destas técnicas é discutido, abordando os seguintes aspectos: o que é a técnica e quando ela pode ser usada, áreas típicas de aplicação, benefícios, vantagens, limitações, processos, necessidades e diretrizes para bom uso.

- **Capítulo 7 – Conclusão:** Este capítulo sintetiza as contribuições deste trabalho, abre possibilidades para trabalhos futuros e apresenta as conclusões.

Capítulo 2

A Engenharia de Requisitos

Neste capítulo são definidos os conceitos de requisitos e de ER. É traçado um panorama da ER no processo de desenvolvimento de sistemas, bem como a sua evolução ao longo dos últimos anos.

Para [ROCHA2000], ER é entendida como a "atividade inicial do processo de desenvolvimento de sistemas em que se determinam e especificam os requisitos para os Sistemas de Informação (SI), ou seja, o que um sistema deve fazer, bem como as circunstâncias sob as quais deve operar. Envolve geralmente um esforço colaborativo entre engenheiros de requisitos e usuários, em que os primeiros procuram obter a partir dos segundos, num processo gradual e cumulativo, o maior conhecimento possível acerca do domínio do discurso do sistema e respectivo ambiente".

2.1 Um Panorama da Engenharia de Requisitos

As maiores deficiências num processo de produção de software se encontram nas primeiras fases do desenvolvimento. O ponto inicial do desenvolvimento de um sistema é a definição do que se quer fazer. A construção do software (programação) só pode ser iniciada quando se tem bem claro o que se deseja produzir. Fazer uma boa definição que abranja todas as necessidades de um sistema complexo é uma tarefa muito difícil. Executar esta tarefa sem métodos, técnicas e ferramentas adequadas torna-a ainda mais difícil. Segundo [LEITE+2002] existem diversos argumentos para se fazer uma boa definição dos requisitos e um destes argumentos está relacionado à correção de erros. É muito mais barato corrigir um erro enquanto se define o projeto do que consertá-lo quando o sistema já estiver pronto.

Tomando como base o ciclo de vida de um software, a primeira fase do desenvolvimento é a análise e especificação dos requisitos, que tem como objetivo o estudo do problema, visando à compreensão de sua natureza; dos elementos envolvidos e de suas relações internas e externas. Como resultado, espera-se obter uma especificação completa, clara e consistente, definindo muito bem o que o sistema será, de acordo com o objetivo inicial estabelecido, sem se preocupar como o sistema irá funcionar.

O entendimento do problema é de fundamental importância para o desenvolvimento de um sistema. Um problema mal entendido pode resultar em um software tecnologicamente correto, mas inadequado à solução e às necessidades do usuário. Entretanto, esta fase não é fácil de ser desenvolvida, pois estamos lidando, essencialmente, com pessoas que além de, obviamente, utilizar a língua natural, também demonstram suas dúvidas; interpretações dúbias; ansiedades; mudanças constantes de opiniões e posições e envolvem-se algumas vezes em questões políticas da organização. Muitas vezes parecem receosas, principalmente devido ao fato de o novo sistema poder tirar o seu posto de trabalho. Temos aí caracterizados os aspectos sociais da ER. [NUSEIBEH+2000] relata esta situação. O contexto em que a ER se dá é geralmente um sistema da atividade humana e os "donos do problema" são pessoas. O engajamento destas pessoas num problema de ER pressupõe que o novo sistema baseado em computador pode ser realmente útil, mas tal sistema mudará as atividades que estas pessoas desempenham ou, pelo menos, a forma como as desempenham. Conseqüentemente, a ER precisa ser capaz de captar o modo como as pessoas percebem e compreendem o mundo em torno delas, como elas interagem e como a sociologia do ambiente de trabalho afeta suas ações [NUSEIBEH+2000].

[ROCHA+2002] também argumenta neste sentido. A ER "define o suporte que os SI deverão dar ao trabalho realizado nas organizações. A definição incorreta dos requisitos levará à obtenção e disponibilização de sistemas inadequados ao sistema organizacional. Como os SI são sistemas que apóiam o trabalho e este tem dimensões técnicas, sociais e estruturais/organizacionais, em situações normais um SI bem sucedido não só estará bem concebido tecnicamente, como apoiará adequadamente a natureza social e organizacional do trabalho".

Apesar da importância que a ER vem conquistando, ainda há muitas organizações que não a utilizam, ou a usam precariamente, no seu processo de desenvolvimento de SI. Um dos principais motivos alegados para "pular" esta fase é a falta de tempo e, freqüentemente, há uma grande pressão para atender o usuário o mais cedo possível e, assim, mantê-lo feliz [LEITE+2002]. Muitas vezes, esta felicidade é momentânea - mais tarde este mesmo usuário poderá bater às portas do departamento de TI solicitando correções no sistema.

Existem diversos fatores que impedem a plena utilização e disseminação da ER e [LEITE+2002] mostra os quatro principais fatores:

- falta de uma educação formal, em nível universitário, em ER;
- tempo cada vez mais curto para responder às constantes demandas dos usuários;
- guardar as informações na "cabeça". As informações não são transcritas num documento de requisitos; e

- o problema de transferência de tecnologia, principalmente em relação aos pacotes.

2.2 Evolução da Engenharia de Requisitos

O processo de ER é muitas vezes entendido como uma parte "chata" do desenvolvimento de um sistema. A partir de fins da década de 70 passou a ser visto de um modo diferente, ganhando mais importância. Ao lermos uma publicação mais antiga, por exemplo, da década de 1980, é interessante notar como era visto, por parte dos autores, o envolvimento do usuário no processo de obtenção dos requisitos nos projetos de sistemas. Poucas páginas foram dedicadas ao tema. A importância era quase sempre na construção e implementação do software. Ou seja, dava-se muita ênfase à parte tecnológica do sistema: questões de hardware, sistemas operacionais, linguagens de programação, etc.

[MARTIN+1991] dedica 5 páginas ao assunto na parte introdutória e mais algumas nos capítulos sobre as técnicas estruturadas, sendo que a versão traduzida tem mais de 800 páginas. Um destes trechos é o seguinte: "É essencial, em um projeto de sistema de computador, o envolvimento dos usuários finais. Alguns projetistas e programadores têm a tendência de trabalhar isolados, criando projetos elegantes, sem interferências ou discussões com o usuário final. Isto é arriscado, e já resultou em muitos desastres que envolveram milhões de dólares" [MARTIN+1991].

[GANE1988] apresenta sua proposta de desenvolvimento rápido de sistemas, na qual dedica um capítulo sobre técnicas de obtenção de requisitos. Aborda as reuniões em grupo, ou *workshop* e, principalmente JAD (*Joint Application Design*), como maneira rápida de obter os requisitos. Embora não use muitas vezes o termo requisito, e sim necessidades ou levantamento de dados, ele já mostra preocupação com a especificação dos requisitos de uma maneira mais clara.

[ROCHA1987] dedica um capítulo à especificação de requisitos. A preocupação é com a qualidade dos requisitos e os impactos que uma especificação mal feita pode gerar no processo de desenvolvimento. A obtenção dos requisitos não é o principal foco de abordagem da autora.

Certamente estas publicações e pesquisas foram importantes no processo de evolução da extração de requisitos. Para interpretarmos esta situação, temos que deixar de lado o atual momento e tentarmos nos imaginar naquela época, quando os sistemas eram definidos nos antigos CPDs[3] e os usuários tinham que se adaptar a eles. É claro que tínhamos

[3] Centro de Processamento de Dados – antiga definição do setor de TI .

também a questão do hardware, que limitava o uso do software. Hoje a situação é bem diferente, há uma segmentação. Os sistemas são encomendados a partir de uma necessidade dos usuários (grupos, departamentos) e são feitos sob medida.

O termo ER foi utilizado pela primeira vez no Simpósio Internacional de Engenharia de Requisitos, em 1993. Antes disto, as pesquisas em ER estavam concentradas em áreas como gerenciamento de requisitos ou análise de requisitos. Desde então, o termo ER vem se difundindo e sendo utilizado para caracterizar todas as atividades envolvendo requisitos de software, do usuário ou do negócio. O termo ER se refere, também, à necessidade de se empregar um processo sistemático na obtenção e especificação dos requisitos. Para [DEBORTOLI1998] "esta sistematização é necessária porque a complexidade dos sistemas exige que se preste mais atenção ao correto entendimento do problema antes do comprometimento de uma solução".

2.3 Requisitos

Um problema encontrado na indústria do software é a falta de definições comuns para termos que são usados para descrever aspectos do desenvolvimento de software e na ER não é diferente. Existem diversas interpretações de requisitos. Para definirmos requisitos temos que levar em conta a visão do usuário e a do desenvolvedor. Em [WIEGERS1999a] é relatado que um mesmo requisito pode ter significados diferentes dependendo de quem está tratando com ele. A definição de um usuário para requisitos pode soar para o desenvolvedor como um conceito de alto nível do sistema. Já a noção de requisitos para o desenvolvedor pode parecer ao usuário uma definição detalhada do projeto. Existem múltiplos níveis de requisitos de software, todo eles legítimos, representando diferentes perspectivas e diferentes graus de detalhe e de precisão.

Dentre as muitas definições existentes para requisitos, uma constantemente apresentada é a que aparece no glossário da IEEE[4] [IEEE90]:

> (1) Uma condição ou necessidade de um usuário para resolver um problema ou alcançar um objetivo.
>
> (2) Uma condição ou capacidade que deve possuir um sistema ou componentes de sistema para satisfazer um contrato, norma, especificação ou outro documento formal.
>
> (3) Uma representação documentada de uma condição ou capacidade satisfazendo (1) ou (2).

[WIEGERS1999a] mostra que a definição publicada pela IEEE cobre tanto a visão do usuário para requisitos (o comportamento externo do sistema) quanto a visão do

[4] *Institute of Electrical and Electronics Engineers*

desenvolvedor (algumas características de implementação e técnicas). Já para [DAVIS+2000] requisitos são todas aquelas características externas observáveis de um sistema que um usuário, um comprador, um cliente ou outro *stakeholder*[5] desejam que estejam presentes no sistema.

Em outras palavras, requisitos são as características que um sistema deve ter para satisfazer os usuários e as propriedades do negócio.

Os requisitos se dividem em funcionais e não funcionais. Os funcionais especificam os serviços que o sistema deverá proporcionar. Descrevem as transformações que o sistema realiza sobre as entradas para produzir as saídas. Os não-funcionais especificam as restrições que interferem no funcionamento do sistema como, por exemplo, tempo de resposta, uso de recursos, segurança, interfaces de usuários e outras. De acordo com [PRESSMAN1995], a funcionalidade deve cobrir a representação e a compreensão do ambiente específico e deve conter todas as funções essenciais que o software deverá realizar. De concreto, os requisitos funcionais dizem respeito às funções específicas do software e definem o que se espera que o produto de software a ser desenvolvido realize.

[CARVALHO+2001] acrescenta um terceiro tipo de requisitos: os de desenvolvimento e manutenção, que incluem procedimentos de controle de qualidade, particularmente procedimentos de teste de sistema, prioridades das funções desejadas, mudanças prováveis nos procedimentos de manutenção do sistema e outros.

2.4 Problemas Envolvendo Requisitos

[OBERG+2000] apresenta uma pesquisa feita com gerentes e desenvolvedores de sistemas na qual são mostrados os problemas reais enfrentados na definição de requisitos. A tabela 1 exibe a porcentagem das respostas dos entrevistados e os problemas mais freqüentes que eles mencionaram em relação ao processo de definição de requisitos.

Além destes problemas, outros importantes foram relacionados [OBERG+2000]:

- requisitos nem sempre são óbvios e possuem várias fontes;
- nem sempre é fácil expressar os requisitos em palavras;
- existem diferentes tipos de requisitos e diferentes níveis de detalhes;
- o número de requisitos pode ficar ingerenciável se não for controlado;

[5] Os possíveis requisitos candidatos podem vir de clientes potenciais, clientes atuais, clientes dos clientes, fornecedores, *marketing*, relatórios de problemas, solicitação de mudanças, características rejeitadas de versões anteriores, pessoal interno de desenvolvimento, pesquisadores da empresa, pessoal de suporte a clientes, manufatura, usuários internos, gerentes e assim por diante. Todas estas fontes candidatas de características são denominadas *de stakeholders* [DAVIS+2000].

- requisitos são relatados de maneiras diferentes;
- existem muitas partes interessadas e responsáveis, o que significa que requisitos têm que ser gerenciados numa referência cruzada de pessoas envolvidas; e
- requisitos mudam.

Tabela 1 - Problemas comuns envolvendo os requisitos [OBERG+2000]

Problema	%
Não poder rastrear mudanças	71
Dificuldade de escrever	70
Não são bem organizados	43

Segundo a pesquisa do *Standish Group* [STANDISH1995] as razões principais que levam a problemas de projeto são identificadas na tabela 2.

Tabela 2 - Fatores de falha de um projeto [STANDISH1995]

Fatores de Falhas de um Projeto	**% de Respostas**
1.Falta de entradas do usuário	12.8%
2.Requisitos e especificações incompletos	12.3%
3. Mudança de requisitos	11.8%
4. Falta de apoio executivo	7.5%
5. Tecnologia imatura	7.0%
6. Falta de recursos	6.4%
7. Expectativas irreais	5.9%
8. Objetivos obscuros	5.3%
9. Tempo/cronogramas irreais	4.3%
10. Novas tecnologias	3.7%
11. Outros	23.0%

Pode-se perceber que problemas envolvendo requisitos incompletos e as mudanças de requisitos são dois importantes fatores que podem tornar um projeto crítico e conduzir a falhas na construção de software.

2.5 Fases e Processos da Engenharia de Requisitos

Na literatura existem diferentes definições e interpretações para as fases, atividades ou processos da ER. Os processos utilizados na ER variam muito. Dependem da cultura organizacional, das pessoas envolvidas e do domínio da aplicação. Nesta seção são mostradas as diversas formas de dividir a ER encontradas na literatura. Em seguida, são explicadas as fases da ER propostas para este trabalho. São brevemente discutidas as seguintes fases: entendimento do domínio, elicitação e análise, especificação e documentação e validação. Também é discutido o gerenciamento dos requisitos, que deve estar presente em todas as fases. Finalmente são abordados aspectos de qualidade que devem estar presentes nas especificações de requisitos.

2.6 Fases e Processos da Engenharia de Requisitos Encontrados na Literatura

Foi realizada uma pesquisa envolvendo publicações que abordam a ER para apurar como os diversos autores/pesquisadores abordam as fases ou processos da ER. O que pode ser verificado é que não há consenso ou padronização na nomenclatura empregada na ER.

[DAVIS+2000] denomina todo o processo de definição de requisitos de gerenciamento de requisitos, que é o conjunto de atividades que compreende a coleta, controle, análise, filtragem e documentação dos requisitos do sistema. Consiste em três fases: elicitação, triagem e especificação.

Para [CARVALHO+2001] o processo de extração de requisitos consiste nos seguintes passos: entendimento de domínio, extração e análise de requisitos, especificação dos requisitos e validação dos requisitos.

[BUREN+1998] divide a ER em categorias de validação e verificação, elicitação, análise, gerenciamento e documentação.

[FIORINI+1998] apresenta conceitos diferentes. O processo de determinar requisitos é chamado de processo de definição de requisitos e compreende basicamente três estágios: elicitação, modelagem e análise. Além do processo de definição de requisitos, salienta a gerência de requisitos, como outro processo que ocorre após a definição.

Para [WIEGERS2000b] a falta de acordo sobre como chamar todo o campo de requisitos de software, pode ensejar interpretações confusas e, para tentar melhorar a compreensão, [WIERGES91999a] considera útil dividir todo o domínio da ER em desenvolvimento dos requisitos e gerenciamento dos requisitos. Subdivide o desenvolvimento em: elicitação, análise, especificação e verificação.

Para [RAGHAVAN+1994] há muitos termos que são usados para descrever o processo de compreender os requisitos para um sistema de software e usa ER como um termo geral que abrange todas as atividades relacionadas aos requisitos. Em particular, a ER compreende quatro seguimentos específicos: elicitação, análise, especificação e validação.

Para [BRACKETT1990] o processo de definição de requisitos compreende os seguintes passos: identificação dos requisitos, identificação das restrições do desenvolvimento do software, análise dos requisitos, especificação dos requisitos, comunicação e validação dos requisitos do software.

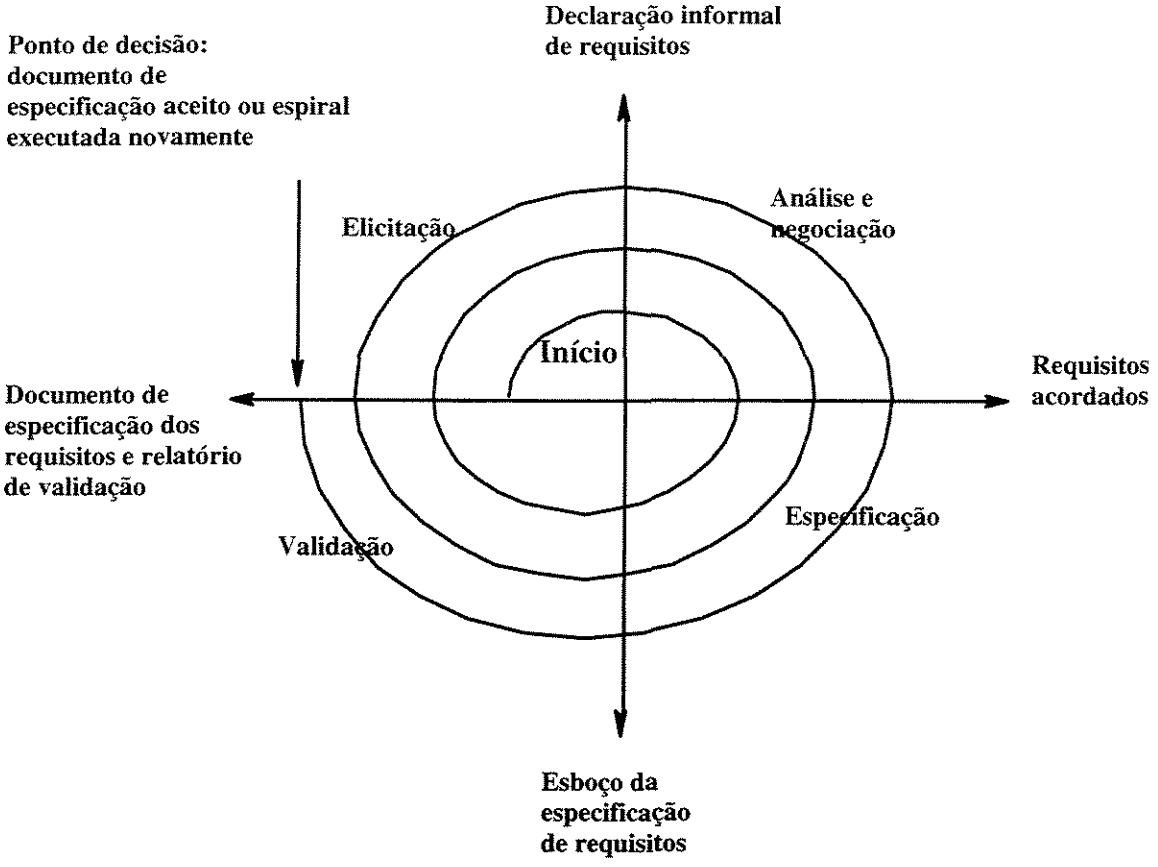

Figura 2 - Modelo espiral do processo de engenharia de requisitos [KOTONYA+1998]

[KOTONYA+1998] define o processo de ER como um conjunto estruturado de atividades envolvidas para derivar, validar e manter um documento de requisitos de sistema. As operações do processo são: elicitação, análise e negociação, documentação e validação. Um processo de ER varia radicalmente de uma organização para outra e os fatores que contribuem para esta variação incluem maturidade técnica, disciplina, cultura

organizacional e domínio da aplicação, não existindo, portanto, um processo de ER ideal para ser utilizado por uma organização. Poucas organizações possuem um processo de ER bem definido e padronizado.

As atividades do processo de ER apresentadas por [KOTONYA+1998] são:

- Elicitação de requisitos: os requisitos são descobertos ou coletados através de consultas aos usuários e da documentação dos sistemas, por exemplo;
- Análise e negociação dos requisitos: os requisitos são analisados e os conflitos são resolvidos através de negociação;
- Documentação de requisitos: um documento de requisitos é produzido, contendo os requisitos acordados num apropriado nível de detalhe;
- Validação de requisitos: o documento de requisitos é cuidadosamente checado, para garantir consistências e completude. Isto é feito para detectar problemas no documento de requisitos antes dele ser usado como base para o desenvolvimento do sistema.

Um processo de ER é iterativo, isto é, a seqüência de execução das atividades pode ocorrer várias vezes, uma após a outra, caracterizando um modelo espiral. A figura 2, extraída de [KOTONYA+1998], representando um modelo espiral dos processos de ER mostra que as diferentes atividades na ER são repetidas até que o documento de requisitos seja aceito, finalizando, então, o processo de ER.

As terminologias empregadas para explicar a ER muitas vezes acabam provocando mais dúvidas do que facilitando o seu pleno entendimento e, com isto, podem admitir interpretações diferentes. Há diversas atividades comuns apresentadas nas definições e outras distintas. [WIEGERS1999a] afirma que as definições que mais aparecem para explicar todo o processo de requisitos são os termos "engenharia de requisitos" e "gerenciamento de requisitos".

[DAVIS+2000] apresenta uma preocupação com relação à ER. Para ele todo o processo de definição tem se tornado cada vez mais complexo. Perdeu-se de vista o fato que a ER foi criada para simplificar o desenvolvimento de software, reduzir custos e os riscos associados à construção do software. Ao invés disso, a ER tem se tornado uma atividade propensa ao erro e muitas vezes ela é um desagradável e desinteressante trabalho.

2.7 Fases da Engenharia de Requisitos Utilizadas neste Trabalho

Para melhor entendimento e clareza é estabelecido aqui que a ER compreende todo o processo de definição de requisitos que vai desde o entendimento do domínio até o momento anterior ao início do projeto, contando com as seguintes fases:

- Entendimento do domínio;
- Elicitação e análise;
- Especificação e documentação; e
- Validação

Além destas quatro fases, o Gerenciamento e o Controle de Qualidade também podem ser considerados como fases presentes na ER. Em cada uma delas é necessário ir mantendo um controle de qualidade dos requisitos e o gerenciamento das eventuais mudanças de requisitos, como mostrado na figura 2.

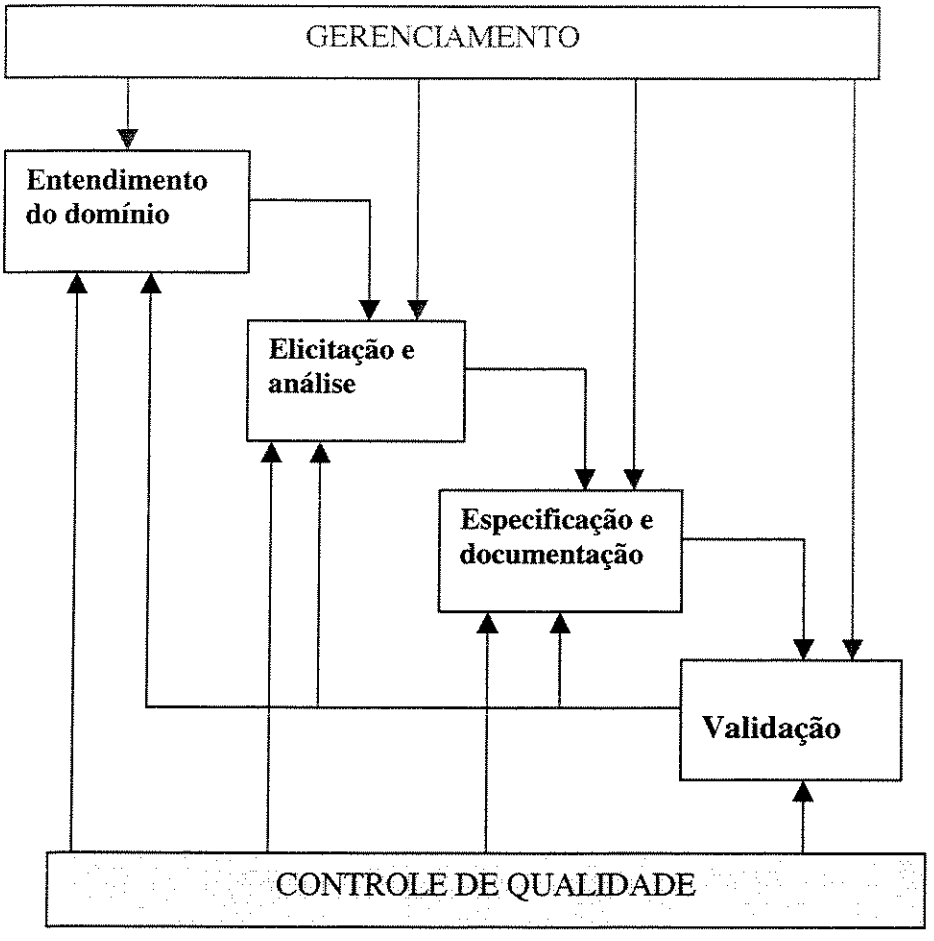

Figura 3 - Fases da engenharia de requisitos para este trabalho

Numa situação real, estas fases não podem ser estritamente separadas e executadas seqüencialmente. Todas são intercaladas e executadas repetidamente. Por exemplo, a especificação de requisitos numa representação formal ou gráfica é freqüentemente útil para identificar conflitos ou requisitos esquecidos e a validação de alguns requisitos muitas vezes revela requisitos ou detalhes que os usuários não reconheceram previamente ou esqueceram [RAGHAVAN+1994].

2.7.1 Entendimento do Domínio

É a fase de entender o que se deseja do novo sistema. Num primeiro momento, todos os usuários relevantes e pessoas envolvidas devem ser identificados. [LEITE+2002] defende que a identificação do usuário necessita ir além da seleção de um cliente vocal. É importante identificar todos os indivíduos ou organizações que têm algo a dizer no desenvolvimento do sistema ou são afetados pelo seu sucesso ou falha. Estabelece-se, então, quais pessoas serão as fontes de informações. Os usuários, ou grupos de usuários, não são homogêneos e é importante determinar os requisitos destes diferentes atores. Após a escolha de quem serão os usuários que fornecerão as informações para o sistema, o primeiro passo é o entendimento do domínio. É o momento de captar as expectativas dos usuários.

Já [TORO+2000] argumenta que, antes de manter as reuniões com os clientes e usuários e identificar os requisitos, é fundamental conhecer o domínio do problema e os contextos organizacional e operacional, isto é, a situação atual. Começar um desenvolvimento sem conhecer as características principais nem o vocabulário próprio de seu domínio pode resultar num produto final que não é o esperado pelos clientes nem pelos usuários. Por outro lado, ao manter reuniões com clientes e usuários sem conhecer as características de sua atividade, provavelmente tornará o entendimento das suas necessidades mais difícil e sua confiança inicial no desenvolvimento pode se deteriorar bastante. Esta tarefa é opcional, já que pode ser desnecessária realizá-la se a equipe de desenvolvimento tiver experiência no domínio do problema e o sistema atual já for conhecido.

"Nesta fase, os desenvolvedores devem entender o domínio da aplicação o mais completamente possível" [CARVALHO+2001]. Devem ser estabelecidos um contexto, escopo ou limites dos requisitos gerais do negócio. Além de mostrar este limite que será contemplado quando o sistema for construído, também é importante identificar as funções ou facilidades que não existirão (não serão implementadas) no novo sistema [WIEGERS2000b].

2.7.2 Elicitação

Uma questão básica na ER é como ficar sabendo o que os usuários realmente necessitam. [GOGUEN+1993] relata que "pesquisas mostraram que muitos projetos falham por causa de requisitos inadequados e, além disso, esta inadequação é freqüentemente relacionada aos fatores sociais, políticos e cultural". Para [WIEGERS2000b] elicitação é primariamente uma atividade de comunicação, não uma atividade técnica.

Alguns cientistas da computação dizem que "elicitação de requisitos é onde a ciência pára e o caos começa" [GOGUEN+1993]. É a mais difícil das fases da ER, afirma [DAVIS+2000], pois está sendo criada alguma coisa a partir do nada.

Elicitação é a arte de determinar a lista de todas as possíveis características que poderiam ser incluídas no sistema [DAVIS+2000]. É o processo de descobrir ou encontrar os requisitos do sistema e determinar o que o sistema deve ser.

[RAGHAVAN+1994] define elicitação como o processo através do qual os clientes, os compradores ou os usuários de um sistema de software descobrem, revelam, articulam e compreendem seus requisitos e que o termo elicitação não é universalmente aceito para descrever tal processo. Alguns engenheiros de software usam termos tais como identificar, recolher, determinar, formular, extrair ou expor requisitos. Cada um destes termos tem conotações diferentes. Por exemplo, recolher sugere que os requisitos já estão presentes em algum lugar e é necessário somente recuperá-los; formular sugere que já os entendemos; extrair e expor sugerem que os requisitos estão sendo escondidos pelos usuários.

De acordo com [WIEGERS1999a], em nenhum outro momento do projeto do sistema os interesses dos usuários e clientes cruzam-se mais do que no processo de elicitação de requisitos. Quando bem feita, esta interseção pode produzir softwares bem elaborados, a clientes felizes e desenvolvedores realizados (com a sensação do dever cumprido). Mal feita e mal conduzida, esta interseção é fonte de equívocos, erros, frustração e de atritos que podem minar e comprometer a qualidade do produto final, pois os requisitos fornecem os fundamentos (alicerces) para a ES e para as atividades de gerência do projeto. Todos os usuários e pessoas envolvidas devem ser convidados a aderir e estar comprometidos e engajados e, assim, gerar um efetivo processo de requisitos.

Segundo [KUCHMISTAYA2001] a fase de identificação dos requisitos "é uma das partes mais importantes do processo, que pode levar ao sucesso ou ao fracasso do projeto". A fim de ajustar os requisitos do sistema que são úteis à organização, os usuários primários do sistema têm que ser envolvidos. Conseqüentemente, o bom processo de identificação dos requisitos abrange a elicitação dos requisitos do sistema, a partir das pessoas que interagirão com o novo sistema regularmente.

[HUSSEIN1997] apresenta definições semelhantes ao afirmar que a elicitação de requisitos é a atividade durante a qual os requisitos de um sistema são descobertos, articulados e revelados, a partir das pessoas envolvidas ou derivados dos requisitos de um outro sistema. As fontes podem ser documentação dos requisitos de um sistema mais amplo, clientes, usuários finais ou especialistas.

Esta atividade é especialmente crítica e deve ser realizada com redobrado cuidado, já que geralmente a equipe de desenvolvimento não conhece os detalhes específicos da organização para a qual irá desenvolver o sistema e, por outro lado, os clientes e possíveis usuários não sabem o que a equipe de desenvolvimento necessita saber para desempenhar o seu trabalho [TORO+2000].

A tentação de começar a escrever o código após os primeiros contatos com os usuários e as primeiras definições é grande. Ainda existem gerentes, desenvolvedores e até mesmo usuários que acham que é perda de tempo despender muito esforço na elicitação dos requisitos. Entretanto, as pessoas e organizações envolvidas com o desenvolvimento de sistemas têm demonstrado um interesse cada vez maior com a ER e, em particular, com a elicitação de requisitos. Elas perceberam, segundo [WIEGERS2000b], que o tempo gasto com a compreensão dos problemas do negócio é um excelente investimento porque os prejuízos causados na construção de um sistema mal implementado têm se tornado muito grandes.

O resultado da elicitação é uma lista de pedidos ou das necessidades dos usuários contendo as características candidatas, cada uma anotada com o seu risco técnico relativo, uma estimativa do esforço requerido para atender à característica, e de uma prioridade relativa ou de uma medida da importância sob a perspectiva do usuário. Esta lista pode ser representada textual ou graficamente [DAVIS+2000].

Um Procedimento Geral de Elicitação

Existem diversas técnicas de elicitar requisitos, tanto a partir de pessoas como a partir de documentos ou observações. [RAGHAVAN+1994] afirma que de longe, o tipo mais comum de esforço da elicitação de requisitos é o que obtém a informação diretamente das pessoas que usarão o sistema. Nesses casos, o procedimento de elicitação pode ser descrito em termos gerais em cinco etapas:

1. Identificar as fontes relevantes de requisitos (usuários);
2. Fazer perguntas apropriadas para obter uma compreensão de suas necessidades;
3. Analisar as informações recolhidas, procurando implicações, inconsistências ou tópicos não resolvidos;
4. Confirmar o entendimento dos requisitos dos usuários; e
5. Resumir os requisitos estabelecidos.

As técnicas específicas de elicitação evoluíram deste procedimento geral definindo processos detalhados, perguntas específicas ou categorias de perguntas, formatos de reuniões estruturadas, comportamentos específicos do indivíduo ou do grupo ou uso de *templates* para organizar e registrar as informações. Várias destas técnicas são descritas no capítulo 6.

Participantes da Elicitação de Requisitos

A elicitação envolve o pessoal técnico e os usuários diretos ou indiretos para conhecer o domínio da aplicação e definir os serviços ou funções que o novo sistema irá disponibilizar, assim como suas restrições.

[ROCHA+2002] divide em três níveis o papel dos usuários quanto ao envolvimento no processo de definição de requisitos:

- Passivo/Consultivo: Usuários são consultados sobre o que querem, normalmente por meio de uma consulta individual, na qual desempenham um papel passivo;
- Representativo: Usuários que representam grupos de usuários são consultados, desempenhando um papel relativamente passivo; e
- Participativo/decisório: Grupo de usuários que participa ativamente, tomando decisões e assumindo responsabilidades pelos modelos e especificações.

O pessoal técnico envolve desenvolvedores em geral, tais como: analista de sistemas, engenheiro de software, analista de negócios, etc. Todos eles estão desempenhando neste momento o papel de engenheiro de requisitos. Para [CARVALHO2001] um dos conhecimentos imprescindíveis a um desenvolvedor é a habilidade de empregar um processo sistemático na elicitação de requisitos.

Mais recentemente, houve um interesse na elicitação dos requisitos porque trabalhar com pessoas não técnicas pode ser um dos maiores desafios entre as atividades de desenvolvimento de software. Em um esforço para comunicar-se com os clientes não

técnicos e compreender suas necessidades, os desenvolvedores de software têm trabalhado com os especialistas das ciências sociais para ganhar novas perspectivas em resolver os problemas de comunicação que podem fazer com que o esforço da aquisição do conhecimento seja nulo e, praticamente, sem resultados. Por isto, a importância do envolvimento dos usuários tem sido reconhecida cada vez mais. Retornos dos usuários, análises e revisões mostram que a direta e regular interação com os usuários é fator-chave de sucesso de um projeto. Projetos que falham tendem a apresentar uma grande falta de participação e retorno dos usuários aos passos iniciais da extração de requisitos [LEITE+2002].

Tabela 3 - Fatores de sucesso de um projeto [STANDISH1995]

Fatores de Sucesso de um Projeto	% de Respostas
1. Envolvimento do usuário	15.9%
2. Apoio da administração	13.9%
3. Declaração/especificação clara de requisitos	13.0%
4. Planejamento próprio	9.6%
5. Expectativas realísticas	8.2%
6. Medidas de menores projetos	7.7%
7. Pessoal competente	7.2%
8. Propriedade	5.3%
9. Visão clara & objetivos	2.9%
10. Trabalho duro e focalizado do pessoal	2.4%
11. Outros	13.9%

Uma vez que o escopo do projeto esteja definido é importante garantir que os usuários terão um envolvimento intensivo. Vários estudos indicam um insuficiente envolvimento dos usuários como um motivo comum do porquê de muitos projetos de software caminharem lentamente ou falharem [WIEGERS2000b]. A tabela 3, obtida de [STANDISH1995], mostra que o principal fator de sucesso de um projeto é o envolvimento do usuário.

Trabalhos nesta área foram inspirados na psicologia cognitiva, antropologia, sociologia e lingüística. A psicologia cognitiva ajuda a entender as dificuldades que as pessoas têm em expressar o que elas querem obter de um sistema. A antropologia tem mostrado a importância de observar pessoas a fim de se descobrir como um sistema pode ajudá-las em seus trabalhos. A sociologia pode ajudar na interação entre os *stakeholders* e a lingüística está relacionada ao problema da comunicação e a clara representação dos requisitos [LEITE+2002].

Dificuldades na Elicitação de Requisitos

Como mostrado na tabela 3, o envolvimento dos usuários é de fundamental importância, pois eles sabem o que o sistema deveria fazer e eles possuem o conhecimento do domínio. Entretanto, para [HUSSEIN1997] e [CHRISTEL1992], os usuários têm problemas para articular suas necessidades por várias razões:

- Usuários podem saber o que eles querem, mas são incapazes de articular os requisitos;
- Usuários podem não saber do que a tecnologia é capaz e podem não considerar o que é possível;
- Usuários podem ter motivos para não querer revelar os requisitos;
- Às vezes usuários e desenvolvedores não falam a mesma língua;
- Um único usuário nunca terá todas as respostas, pois os requisitos podem vir de muitas fontes; e
- Usuários expressam os requisitos nos seus próprios termos.

Alguns problemas tocantes ao desenvolvedor são:

- Desenvolvedores podem não ter as habilidades necessárias para obter os requisitos dos usuários;
- Desenvolvedores podem não ser capazes de manter a harmonia no processo de obtenção de requisitos ou não entender as necessidades dos usuários;
- Muitas vezes os desenvolvedores não são capazes ou não se identificam com as necessidades e preocupações dos usuários; e
- Muitas vezes os desenvolvedores tendem a intimidar os usuários, fazendo com que eles concordem com os requisitos estabelecidos pelos próprios desenvolvedores.

Uma dificuldade extra para os desenvolvedores é obter a informação de várias pessoas envolvidas e sistematizar mais tarde todos seus desejos em requisitos. Há diversos motivos para esta dificuldade segundo [KUCHMISTAYA2001]:

- Os usuários vêm de departamentos diferentes da organização e são de níveis profissionais diferentes;
- Trazem propostas diferentes para as funções e tarefas que o sistema deveria estar apto a fazer;
- É difícil encontrar tempo para elicitar requisitos de muitos usuários;

- Os requisitos coletados dos usuários não são completos ou não são compreendidos pelos analistas; e
- Usuários diferentes podem ter requisitos que entram em conflito.

2.7.3 Análise dos Requisitos

É o processo de raciocinar sobre os requisitos que foram elicitados e envolve atividades como examinar os requisitos para encontrar conflitos, inconsistências ou ambigüidades; combinar requisitos relacionados e identificar requisitos faltantes ou extras [RAGHAVAN+1994]; [HUSSEIN1997]. A análise de requisitos envolve a quebra de requisitos de alto nível em requisitos funcionais detalhados. Durante esta atividade, ocorre uma negociação entre todos as pessoas envolvidas para obter um conjunto de requisitos comumente acordados e aceitos. A figura 4, adaptada de [KOTONYA+1998], mostra esta operação.

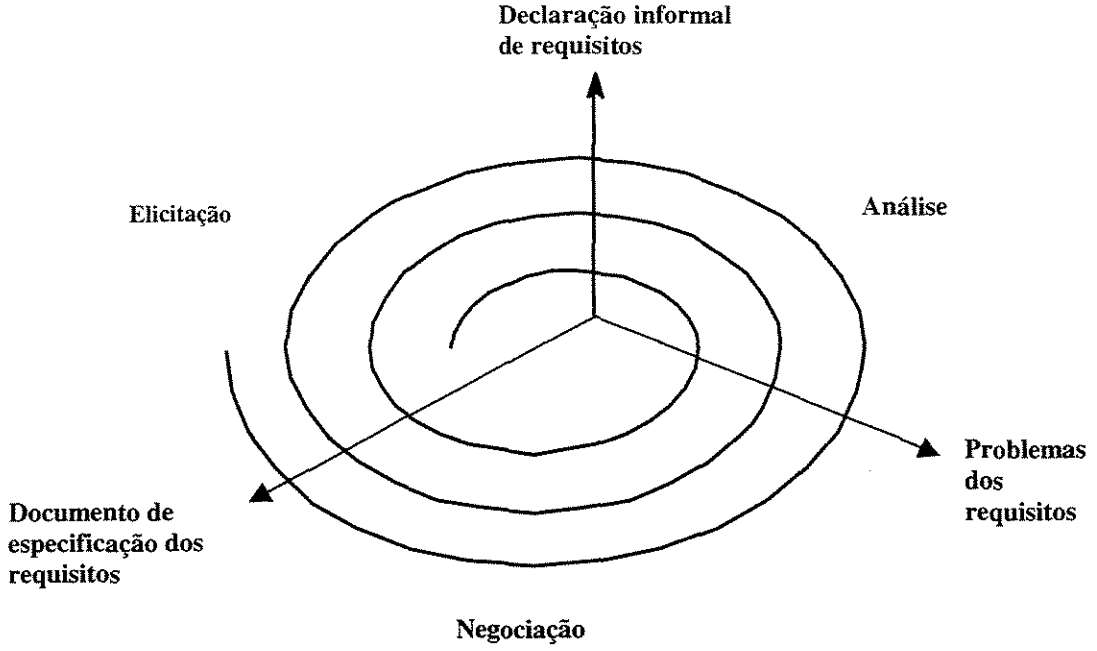

Figura 4 - Interação entre a elicitação e a análise de requisitos [KOTONYA+1998]

A análise de requisitos é uma atividade que deve ser desempenhada simultaneamente com a elicitação, conforme mostrado na figura 4. Como os requisitos são descobertos durante a elicitação, a análise deve ocorrer, inevitavelmente, nesta fase, para discutir e resolver os problemas dos requisitos que eventualmente aparecerem [KOTONYA+1998]. Deste modo, melhores resultados podem ser obtidos.

Os requisitos mudam durante o processo de análise. Novos usuários podem surgir, o ambiente do negócio pode mudar, novas necessidades podem aparecer. A análise de requisitos também pode sofrer a influência de fatores políticos e organizacionais.

2.7.4 Especificação e Documentação dos Requisitos

É a tarefa de registrar os requisitos em um ou mais formulários, usando uma linguagem formal, natural, representações simbólicas ou gráficas. A especificação é feita geralmente num documento chamado Especificação de Requisitos de Software (ERS). A especificação é utilizada para que os requisitos possam ser comumente interpretados pelos clientes, usuários finais, gerentes, projetistas e desenvolvedores [RAGHAVAN+1994]; [HUSSEIN1997].

[DAVIS+2000] mostra que a especificação de requisitos é a tarefa de documentar com precisão o comportamento externo do sistema que está sendo construído. Faz-se o exame das características selecionadas durante a análise dos requisitos, expandindo-as num nível de detalhe considerável. As finalidades primárias de fazer isto são:

- assegurar que os usuários e desenvolvedores tenham a mesma compreensão do que deve ser construído;
- assegurar que todos os desenvolvedores tenham compreensão idêntica do que deve ser construído;
- assegurar que os verificadores estejam testando as mesmas qualidades que os desenvolvedores estão construindo; e
- ter garantias que a gerência está aplicando recursos nas mesmas tarefas que os desenvolvedores estão executando.

A especificação de requisitos é o fundamento teórico no qual se apoiará todo o trabalho de desenvolvimento, portanto, deve ser feita com muito cuidado [PORTELLA1994] para evitar erros nas fases seguintes.

O documento de especificação define os atributos desejáveis do software a desenvolver. Dentre estes requisitos se encontra a funcionalidade, ou requisitos funcionais, que especifica os detalhes mais relevantes das funções que o produto realizará, com independência dos requisitos não funcionais.

Documentar os requisitos não é fazer anotações em "papel de pão", rascunhos, textos que foram escritos e não validados. Sem contar que muitos guardam na "cabeça" os requisitos necessários. Se um membro da equipe for trocado e chegar um outro, como é que este novo analista vai conhecer os requisitos do sistema? Fará novas reuniões com os usuários? Os usuários terão tempo e paciência?

O documento de requisitos estabelece o que se espera do sistema. Deve incluir tanto a definição quanto a especificação. Não é um documento de projeto do sistema e deve estabelecer o que faz o sistema e não como faz. Os requisitos devem estar documentados de tal modo, que seja possível rastreá-lo ou encontrá-lo no sistema final. Deve especificar somente o comportamento externo do sistema e suas restrições de implementação. Deve ser de fácil acesso e serve de base para futuras manutenções.

Os requisitos estão sempre em mudança, não são estáticos. Com o passar do tempo entende-se melhor a necessidade dos usuários e também porque as necessidades dos usuários mudam. Documentar todas as mudanças deve ser obrigatório, o que vai produzir novas versões da especificação. É de fundamental importância o controle das versões implementadas.

Segundo [HUSSEIN1997] uma mesma especificação pode ter significados diferentes para muitas pessoas:

- Descreve o que o sistema será para o usuário;
- Descreve o tipo de sistema que os projetistas necessitam construir;
- Descreve que tipos de coisas necessitam ser testadas pelos verificadores; e
- Estabelece um acordo entre o usuário e o desenvolvedor do software.

Para [PORTELLA1994], o sistema desenvolvido pode falhar em seus objetivos em atender as especificações definidas a partir da extração, junto aos usuários, por três razões:

1. O sistema é desenvolvido corretamente, porém a partir de uma especificação errada ou incompleta;
2. O sistema é desenvolvido incorretamente a partir de uma especificação correta. Essa incorreção pode ser fruto de erros naturais do processo de desenvolvimento, dada a sua complexidade ou incapacidade técnica no uso de metodologias ou ferramentas; e
3. Corte voluntário do escopo do projeto, por redução de verba e/ou tempo, diminuindo sua funcionalidade, facilidade, segurança, precisão e até, muitas vezes, os requisitos de hardware necessários para sua implantação.

Das três causas acima, a primeira (desenvolvimento correto a partir de especificações erradas) é a de maior incidência. [PORTELLA1994] continua relatando que "o calcanhar de Aquiles da ES está em produzir sistemas que atendam às especificações dos usuários, quando é consenso que normalmente os requisitos são incompletos ou mal formulados".

Qualidade do Documento de Requisitos

A seguir são apresentadas as características desejáveis que um documento de ERS deve apresentar. Uma ERS completa é mais do que uma longa lista de requisitos funcionais, ela é um documento que inclui também as descrições das interfaces externas e requisitos não-funcionais, tais como atributos da qualidade e expectativas de desempenho. As seguintes características devem estar presentes numa ERS de qualidade [WIEGERS1999b]; [PORTELLA1994];[DAVIS+2000]:

- **Completitude.** Nenhum requisito ou informação necessária deve faltar na ERS. Ela deve descrever algo que os usuários querem e abranger toda a necessidade do usuário em termos de informações. Organizar os requisitos hierarquicamente na ERS ajuda os revisores a compreender a estrutura da funcionalidade descrita, assim será mais fácil para eles relatarem se está faltando algo. Modelos de análise gráficas que representam diferentes visões dos requisitos podem também revelar a não completitude. Se o desenvolvedor souber que está faltando determinada informação, ele pode usar o rótulo <<a determinar>> como uma marca para destacar as falhas nos requisitos. Todos os requisitos <<a determinar>> devem ser resolvidos antes do prosseguimento da construção do software.
- **Consistência**. Uma ERS não pode conter definições erradas ou mal formuladas das necessidades de informações. Os requisitos devem ser consistentes e as discrepâncias entre eles devem ser resolvidas antes que o desenvolvimento prossiga. Deve-se tomar cuidado quando modificar os requisitos, pois inconsistências podem não ser detectadas se for revista somente a mudança específica e não todos os requisitos relacionados.
- **Não redundância**: Cada necessidade de informação deve ser definida somente uma vez nas especificações. Ser repetitivo, tentando reforçar aspectos importantes, apenas cria dificuldade na análise de consistência, não garantindo exatidão;
- **Modificabilidade:** Uma ERS pode ser revista a qualquer momento e um histórico das mudanças feitas para cada requisito deve ser mantido. Para fazer isto, cada requisito deve ser univocamente identificado e mapeado separadamente de outros requisitos; assim é possível consultar o requisito de forma não ambígua. Uma ERS pode ser mais facilmente modificável se ela for organizada

de modo que os requisitos relacionados sejam agrupados, com a criação e tabelas, índices e uma lista de referências cruzadas; e

- **Não ambigüidade:** O leitor de uma especificação de requisitos deve poder extrair somente uma interpretação dela. Vários leitores de uma especificação de requisitos também devem chegar a uma mesma interpretação. A língua natural é altamente propensa à ambigüidade. Devem-se evitar palavras subjetivas e genéricas como amigável, fácil, simples, rápido, eficiente, diversos, último modelo, melhorado, maximizar e minimizar. Palavras que estão claras ao autor da ERS podem não ser suficientemente claras para os leitores. Cada requisito deve ser escrito numa linguagem sucinta, simples e direta, de domínio do usuário, não usando termos técnicos da computação; evitando, desta forma, confusões para o leitor.

2.7.5 Validação dos Requisitos

É o processo de confirmação com o usuário do software se os requisitos especificados e documentados são válidos, corretos e estão completos [RAGHAVAN+1994]. É a última etapa da ER e tem como objetivo certificar se os requisitos especificados representam uma descrição aceitável do sistema a ser implementado [KOTONYA+1998]

[HUSSEIN1997] define validação como a operação na qual os requisitos são checados para prevenir requisitos omitidos, ambíguos, errados, inconsistentes ou extras. Esta operação serve também para assegurar que todos os requisitos estão seguindo padrões de qualidade indicados. Através da validação pode-se demonstrar se os requisitos definem o sistema que o usuário realmente quer: está sendo construindo o sistema que o usuário espera? A validação dos requisitos tem como objetivo demonstrar que a definição dos requisitos está compatível com o sistema que o usuário deseja. Se uma validação adequada não for realizada, os erros se propagam para as fases seguintes. Quanto mais tarde um erro é detectado, maior o custo de corrigi-lo [LEITE+2002] e o custo de corrigir um erro de requisito mal definido é maior que o de corrigir um erro depois de ter o programa implementado. Para se fazer uma boa validação alguns aspectos podem ser considerados: os requisitos não devem ser contraditórios e sim consistentes, as definições devem conter todas as funções e restrições, isto é, devem estar completas. Também deve ser verificado se existem requisitos impossíveis cuja implementação é impossível.

Na operação de validação devem-se fazer constantes revisões dos requisitos com a participação dos usuários e da equipe de desenvolvimento. Nesta etapa é fundamental a boa comunicação entre usuários e desenvolvedores. Todas as alterações devem ser

documentadas, em caráter formal. Devem-se também realizar revisões informais e um método padronizado deve ser seguido. A prototipação é uma técnica que pode ser usada para validar requisitos.

Alguns problemas descobertos durante a validação segundo [KOTONYA+1998] são:

- Falta de conformidade com os padrões de qualidade;
- Pobre formulação (descrição) de requisitos, que acaba gerando ambigüidade;
- Erros nos modelos do sistema ou do problema a ser resolvido; e
- Requisitos com conflitos que não são detectados na fase de análise.

Estes problemas devem ser resolvidos antes que o documento de requisitos seja aprovado e usado como base para o desenvolvimento do sistema. Alguns problemas podem ser simplesmente problemas de documentação e podem ser resolvidos corrigindo a especificação, enquanto outros podem advir de falhas resultantes da elicitação, análise ou especificação o que pode levar a uma nova execução destas fases conforme a espiral da ER mostrada na figura 2 [KOTONYA+1998].

Segundo [KOTONYA+1998], o principal problema da validação de requisitos é que não existe um documento para ser tomado como base na validação, pois não existe maneira de comparar uma ERS com outra. Não há como saber se ela está correta, pois ela é o primeiro documento do sistema. Entretanto uma ERS tem que assegurar que os requisitos representam uma clara descrição do sistema a ser implementado e é uma verificação final para garantir que os requisitos vão ao encontro das necessidades do usuário. As entradas e saídas de uma validação de requisitos estão representadas na figura 5, obtida de [KOTONYA+1998], e estão comentadas a seguir:

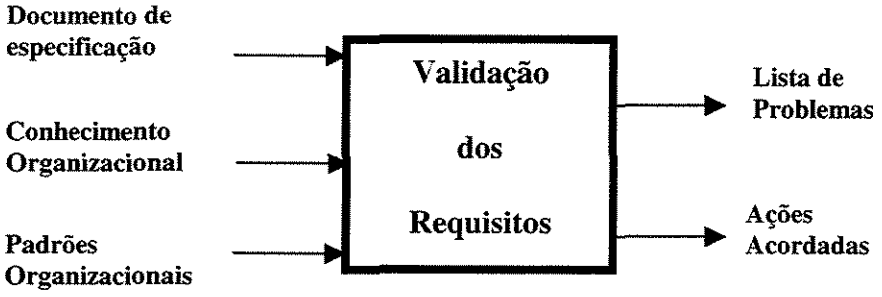

Figura 5 - Entradas e saídas do processo de validação dos requisitos [KOTONYA+1998]

Processos de entradas:

- Documento de requisitos: deve estar completo com todos os requisitos acordados com os usuários, formatados e organizados de acordo com os padrões organizacionais;
- Padrões organizacionais: o processo de validação de requisitos deve ser revisado para estar de acordo com os padrões da organização.
- Conhecimento organizacional: Não é uma entrada tangível na prática, mas é muito importante. As pessoas envolvidas na validação conhecem a organização, terminologias e perfis dos usuários envolvidos com o sistema. O conhecimento organizacional está, portanto, estreitamente relacionado com a cultura, estrutura organizacional e padrões.

Processos de saídas:

- Lista de problemas: deve conter os problemas encontrados na especificação de requisitos. Apesar de na prática ser difícil categorizar os tipos de problemas, esta lista pode ser classificada por tipos de problema, como: ambigüidade, incompletitudes, etc.
- Ações acordadas: É uma lista de ações a ser executadas para resolver os problemas presentes na lista de problemas. Alguns problemas podem ter várias ações corretivas, enquanto outros podem ter somente uma ação ou mesmo nenhuma.

[LEITE+2002] citando [SOMMERVILLE+1997] ressalta que sempre devemos ficar atentos à clássica distinção entre verificação e validação. Na verificação devemos nos perguntar: "*estamos construindo o produto de forma correta?*" e na validação: "*estamos construindo o produto certo?*". Então, teste (validação) é diferente de inspeção (verificação).

2.7.6 Gerenciamento dos Requisitos

Os requisitos não são estáticos; durante o ciclo de vida do software podem surgir novos requisitos ou mudanças nos já existentes. Estas mudanças ocorrem porque os objetivos organizacionais podem mudar, o foco do negócio pode variar, as pessoas têm novas necessidades, compreendem melhor o sistema, podem ocorrer mudanças legais, etc. Mesmo durante a fase da ER, quando estão sendo especificados, os requisitos podem mudar e para controlar estas mudanças é necessário um ótimo processo de gerenciamento de requisitos; esta gerência é uma atividade que deve ser contínua e ocorrer

concomitantemente com todos os outros processos da ER. Já que não é possível evitar as mudanças, deve ser possível controlá-las.

Gerenciamento de requisitos é o processo de controlar mudanças nos requisitos do sistema [KOTONYA+1998]. [ROCHA2000] define gerenciamento de requisitos como "o processo de gestão das mudanças dos requisitos de um sistema. Os requisitos de um sistema mudam sempre como reflexo das mudanças das necessidades dos usuários, do ambiente onde o sistema vai funcionar, do negócio, etc."

Ainda segundo [KOTONYA+1998] as principais preocupações do gerenciamento de requisitos são:

- Gerenciar mudanças dos requisitos acordados;
- Gerenciar os relacionamentos entre os requisitos; e
- Gerenciar as dependências entre o documento de requisitos e outros produzidos durante o processo de ES.

A gerência bem sucedida dos requisitos requer um processo do controle da mudança que gerencie como as mudanças são submetidas, avaliadas, escolhidas e implementadas. Uma boa prática nesta atividade é o uso de uma ferramenta de gerência de mudanças que preserve automaticamente trilhas ou pistas das mudanças individuais dos requisitos, disponibilize relatórios das mudanças e mantenha históricos das revisões [WIEGERS2000a]. Sistemas de controle de versões também podem ser usados para ajudar neste esforço.

[KOTONYA+1998] argumenta que as mudanças podem ocorrer mesmo quando os requisitos estão sendo elicitados, analisados, validados ou mesmo quando o sistema já estiver em produção. As mudanças nos requisitos são inevitáveis e isto não significa que o processo de ER tenha sido falho. Ainda segundo [KOTONYA+1998], essas mudanças resultam de uma combinação de fatores, tais como:

- Erros, conflitos e inconsistências nos requisitos: Assim que os requisitos são analisados e implementados, erros ou inconsistências surgem e precisam ser corrigidos. Estes erros são descobertos durante a validação ou mais tarde, no desenvolvimento do sistema;
- Evolução do conhecimento do usuário sobre o sistema: Assim que os requisitos são definidos, os usuários têm uma melhor compreensão do que realmente eles querem do sistema;
- Problemas técnicos, de cronograma ou de custos: Problemas podem ser encontrados durante a implementação dos requisitos. Alguns requisitos podem ter um alto custo ou levar muito tempo para ser implementados;

- Mudanças na prioridade do cliente: As prioridades do usuário podem se alterar durante o processo de ER devido às mudanças no negócio, surgimento de novos concorrentes, mudanças gerencias, etc.
- Mudanças ambientais: o ambiente no qual o sistema será instalado pode mudar e isto leva às mudanças de requisitos para manter a compatibilidade; e
- Mudanças organizacionais: A organização que pretende usar o sistema pode sofrer mudanças estruturais, ocasionando, por exemplo, o surgimento de novos requisitos.

2.7.7 Qualidade dos Requisitos

[PRESSMAN1995] define qualidade de software como "conformidade a requisitos funcionais e de desempenho explicitamente declarados, a padrões de desenvolvimento claramente documentados e a características implícitas que são esperadas de todo software profissionalmente desenvolvido." Qualidade de software depende de uma série de fatores. Um fator que certamente influencia a qualidade do produto final de software é a qualidade dos requisitos. Segundo [PRESSMAN1995] os requisitos do software são a base a partir da qual a qualidade é medida. A falta de conformidade aos requisitos significa falta de qualidade. O processo de controle de qualidade dos requisitos deve ser efetuado durante toda a fase da ER.

A satisfação do usuário é a melhor forma de se avaliar a qualidade de um sistema. Segundo [WIEGERS1999b], muitas ERS são elaboradas com requisitos mal escritos. Devido ao fato de que a qualidade de qualquer produto depende da qualidade das informações coletadas e colocadas na especificação, requisitos mal definidos não podem levar a um excelente software. Infelizmente, poucos desenvolvedores de software foram educados sobre como elicitar, analisar, documentar e verificar a qualidade dos requisitos. Não há muitos exemplos de bons requisitos disponíveis para aprender com eles, em parte porque poucos projetos têm bons exemplos a compartilhar, e em parte porque poucas companhias estão dispostas a disponibilizar suas especificações de produtos ao domínio público.

[WIEGERS1999b] mostra que existem várias características que dão indicações de uma alta qualidade nos requisitos de software. Não é de se esperar que seja criada uma ERS em que cada requisito exiba todas as características desejadas. Não importa muito como se depure, analise, reveja e refine os requisitos, eles nunca serão perfeitos. Entretanto, se algumas das características de qualidade forem mantidas em mente, melhores requisitos serão produzidos e melhores sistemas construídos.

Atributos que Indicam Qualidade dos Requisitos

Os requisitos têm que ser elaborados contemplando alguns atributos e possuir algumas propriedades ou características individuais, demonstradas a seguir, que podem ser indicativos de boa qualidade [WIEGERS1999b]; [PORTELLA1994]:

- **Exatidão**. Cada requisito deve descrever exatamente a funcionalidade a que ele corresponde. Um requisito está completo se ele proporciona informações suficientes para a sua compreensão, não necessitando mais detalhes. A referência para a exatidão é a fonte do requisito, tal como um cliente/usuário real ou uma especificação de requisitos de alto nível de um sistema.
- **Viabilidade.** Deve ser possível implementar cada requisito dentro das conhecidas capacidades e limitações do sistema e de seu ambiente. Para evitar requisitos impraticáveis os analistas envolvidos no processo de elicitação devem fazer constantemente uma real verificação do que pode e do que não pode ser feito tecnicamente ou que pode ser feito somente com um alto custo.
- **Rastreabilidade:** Deve ser possível associar cada requisito com a sua fonte de origem no sistema. Os requisitos devem ser univocamente identificados e escritos em uma maneira estruturada e refinada, e não em grandes parágrafos narrativos ou grandes listas endentadas, para facilitar a sua rastreabilidade. Requisitos podem ser dependentes de outros e para que as equipes possam determinar o impacto das mudanças e confiem que o sistema está de acordo com as expectativas, o relacionamento da rastreabilidade deve ser entendido, documentado e mantido. Enquanto a rastreabilidade é um dos conceitos mais difíceis de ser implementados em todo o processo da ER, é essencial acomodar as mudanças. Estabelecendo-se claramente os tipos de requisitos e elaborando a referência cruzada, pode-se fazer mais facilmente a implantação e a manutenção da rastreabilidade [OBERG+2000].
- **Necessidade.** Cada requisito deve documentar algo que os clientes realmente necessitam ou algo que é exigido para manter a conformidade de um requisito a uma interface externa ou a um padrão. Uma outra maneira de se pensar sobre "necessidade" é que cada requisito deve ser originário de uma fonte que seja reconhecida como autoridade para especificar requisitos. Rastrear cada requisito até a sua origem é uma maneira de verificar sua real necessidade. Se não for possível identificar a origem, talvez o requisito não seja realmente necessário. Um requisito é necessário se sua omissão provoca uma deficiência no sistema a ser construído e, além disso, se sua capacidade, características físicas ou fator de

qualidade não podem ser substituídos por outras capacidades do produto ou do processo.

- **Prioridade**. Atribuir uma prioridade de implementação para cada requisito ajuda a indicar o quão essencial ele é para ser incluído em um sistema. Os usuários são as pessoas que têm condições e responsabilidades para estabelecer as prioridades. Se todos os requisitos forem considerados igualmente importantes, o gerente de projeto tem menos condições para agir e tomar as decisões em determinadas situações como, por exemplo, de que maneira tratar os requisitos novos adicionados durante o desenvolvimento, como lidar com os cortes do orçamento; redefinir prazos em função de atrasos ou em quais atividades concentrar esforços se um membro da equipe sair. Com os requisitos priorizados fica mais fácil tomar uma decisão. Requisitos podem ser priorizados baseando-se no custo, dependência e necessidade dos usuários. A prioridade pode ser dividida em três níveis:
  - A prioridade alta significa que o requisito deve ser incorporado na próxima liberação/versão do produto;
  - A prioridade média significa que o requisito é necessário, mas pode ser postergado para uma liberação mais adiante, se necessário; e
  - A prioridade baixa significa que seria agradável ter o requisito, mas percebe-se que ele pode ser eliminado se não houver tempo ou recursos suficientes.
- **Verificabilidade.** Planejar testes ou usar outra forma de verificação, como a inspeção, a demonstração ou provas para determinar se cada requisito está implementado corretamente no produto. Em outras palavras, deve-se verificar se é possível encontrar o requisito no sistema. Requisitos que são inconsistentes, impraticáveis ou ambíguos também são inverificáveis.

2.8 Definição de Requisitos com Qualidade

Não há nenhuma regra formulada para escrever excelentes requisitos. Isto é alcançado muito mais pela experiência, aprendizagem e superação dos problemas enfrentados em definições anteriores de requisitos. Entretanto, algumas regras devem ser lembradas no momento em que se descrevem requisitos [WIEGERS2000a]:

- Elaborar sentenças e parágrafos curtos;
- Usar a voz ativa;
- Usar a gramática, a ortografia e a pontuação apropriadas; e
- Usar termos consistentes e definí-los em um glossário ou dicionário de dados.

Para verificar se um requisito está suficientemente definido, ele deve ser lido sob a perspectiva do usuário. O analista pode verificar se ele consegue entender completamente o que está escrito. Ou seja, é necessário um esclarecimento adicional do autor da ERS para compreender se o requisito está bom o bastante para projetá-lo e implementá-lo ou se este requisito deve ser melhor elaborado antes de se prosseguir com a construção do software.

Os desenvolvedores devem esforçar-se freqüentemente para encontrar o nível correto de refinamento, evitando os longos parágrafos narrativos que contém múltiplos requisitos. Um guia útil de refinamento é escrever individualmente os requisitos. Se se pode pensar em um número pequeno de testes relacionados para verificar a correta implementação de um requisito, ele provavelmente está escrito no nível certo de detalhe. A visualização de muitos tipos diferentes de testes pode significar que talvez vários requisitos foram reunidos e devem ser separados.

Prestar atenção aos múltiplos requisitos que foram agregados em uma única definição. Conjunções como "e" e "ou" em um requisito sugerem que diversos requisitos foram combinados. Nunca usar "e/ou" num documento de especificação de requisitos. Para [KOTONYA+1998], independentemente do nível de detalhe da descrição de requisitos, existem três aspectos essenciais que se deve ter em mente no momento de descrever os requisitos:

- Normalmente os requisitos são mais lidos do que escritos. Investir esforços em escrever requisitos fáceis de ler e de compreender é uma forma efetiva de economizar tempo e esforço;
- Os leitores de requisitos vêm de diferentes áreas e possuem formações diferentes. O descritor/escritor dos requisitos não pode assumir que tenham o mesmo conhecimento que ele;
- Escrever requisitos claros e concisos não é fácil. Se a pessoa que estiver escrevendo os requisitos não tiver tempo suficiente para fazer um esboço, uma boa revisão e melhorias, a especificação será inevitavelmente escrita de forma imprecisa e com falhas.

Capítulo 3

Taxonomia e Esquemas de Classificação

Neste capítulo é apresentado o conceito de taxonomia, assim como os esquemas de classificação que fornecem o embasamento teórico que sustentará a taxonomia proposta no Capítulo 6.

3.1 Taxonomia

No dicionário Aurélio [FERREIRA1975] taxonomia, ou taxinomia é assim definida: *S. f.* 1. Ciência da classificação. 2. *Biol.* Sistemática. 3. *Gram.* Classificação das palavras em grupos ou categorias. A taxonomia tem como objetivo a divisão sistemática dos seres, de forma que possa facilitar seu estudo.

Segundo [PRIETO-DÍAZ2002] uma taxonomia é uma estrutura de categorias e a classificação é o ato de atribuir entidades às categorias dentro de uma taxonomia. [REIMER2001] define taxonomia como um "vocabulário controlado que é arranjado em uma hierarquia de conceitos".

Um esquema de classificação na biblioteconomia é uma ferramenta para a produção de uma ordem sistemática baseada em um índice de vocabulário controlado e estruturado. Este índice de vocabulário é chamado de glossário/catálogo de classificação e consiste de um conjunto de nomes ou termos representando conceitos ou classes, listados numa ordem sistemática, para indicar os relacionamentos entre classes.

Um esquema de classificação e seu respectivo glossário podem, então, ser considerados uma taxonomia ampliada que vai além de um mero arranjo de categorias, pois incluem relacionamentos entre categorias e algumas breves definições [PRIETO-DÍAZ2002].

3.2 Classificação

Classificar é agrupar coisas semelhantes de acordo com algum critério pré-estabelecido. Os grupos ou classes obtidos com a classificação compartilham ao menos uma característica que os membros de outra classe não compartilham. A classificação mostra o relacionamento entre coisas e entre classes de coisas. O resultado de uma classificação é uma rede ou estrutura de relacionamentos.

De acordo com [PRIETO-DÍAZ 1987], existem dois tipos de relacionamentos que uma classificação deve expressar: hierárquico e sintático. Relacionamentos hierárquicos baseiam-se no princípio de subordinação ou inclusão, enquanto que relações sintáticas são criadas para relacionar dois ou mais elementos de diferentes hierarquias. Ainda segundo ele, esquemas de classificação típicos são estritamente hierárquicos. Relações sintáticas são posteriormente apresentadas através de classes compostas. Por exemplo, a "reprodução das serpentes" relaciona o termo "reprodução", da classe processos, com o termo "serpentes" da classe "répteis".

Classificações podem ser organizadas de duas formas: através de enumeração ou através de facetas.

3.2.1 Classificação Enumerativa

Na classificação enumerativa os componentes são descritos por uma estrutura hierárquica definida em que as classes e os níveis das subclasses são definidos. Os componentes reais estão no último nível. A estrutura hierárquica é fácil de compreender e usar, porém não é flexível.

Os esquemas de classificação enumerativos mais conhecidos são o *Dewey Decimal Classification* (DDC) e o *Library of Congress Classification* (LCC). O DDC, publicado primeiramente em 1876, foi projetado, originalmente, para organizar os livros em prateleiras das bibliotecas. O LCC, publicado e mantido pela Biblioteca Nacional dos Estados Unidos, é usado para classificar os milhões de documentos, livros e outros materiais da vasta coleção desta biblioteca. O DDC e o LCC, juntamente com outros, como o *Universal Decimal Classification*, ou UDC, são exemplos de classificações enumerativas, que têm como objetivo enumerar ou listar todos os assuntos presentes na literatura que se pretende classificar em um esquema.

A enumeração é normalmente realizada identificando primeiramente as disciplinas principais a ser cobertas pelo esquema, em uma base filosófica, e alocando cada uma como uma classe principal. Cada disciplina é então dividida em subclasses. Este processo de subdivisão continua até que um nível apropriado de especificidade seja obtido. O objetivo é fornecer um lugar, e somente um, para cada assunto. Por exemplo, o DDC classifica a "filosofia e a psicologia" como a classe 100, que por sua vez é subdividida em 100: Filosofia, 110: Metafísica, 120: Epistemologia, etc.. Assim, esquemas enumerativos são também hierárquicos. As classificações enumerativas são essencialmente métodos de análise descendente: começam no alto da hierarquia e atravessam-na até que seja encontrado um título ou um termo apropriado que descreva ou classifique o texto

[LEISE2001].

O método de classificação enumerativo postula um universo de conhecimento subdividido em classes sucessivamente menores que incluem todas as possíveis classes compostas (relações sintáticas). Essas classes são, então, organizadas de forma a apresentar suas relações hierárquicas [PRIETO-DIAZ1987].

3.2.2 Classificação Facetada

Classificações facetadas, ao contrário da classificação enumerativa, não dependem da segmentação de um universo, mas em sintetizar a declaração de conteúdo dos elementos a serem classificados. Por este método, esses elementos são analisados segundo suas classes elementares e essas classes são, então, relacionadas no esquema. Seus relacionamentos genéricos são os únicos apresentados [PRIETO-DÍAZ+1987].

Tabela 4 - Um exemplo de classificação facetada [PRIETO-DÍAZ+1987]

{Faceta de processo}
- Fisiologia
  - Respiração
  - Reprodução

{Faceta do habitat animal}
- Animais aquáticos
- Animais terrestres

{Faceta da taxonomia zoológa}
- Invertebrados
  - Insetos
- Vertebrados
  - Répteis

Quando é necessário, através deste esquema, expressar uma classe composta utiliza-se uma combinação de suas classes elementares. Esse processo é denominado síntese. Os grupos de classes elementares que compõem o esquema de classificação são as facetas. Os elementos ou classes que constituem uma faceta são denominados termos. Na tabela 4 é mostrado um exemplo de uma classificação facetada, adaptado de [PRIETO-DÍAZ+1987].

A tabela 5, também adaptada de [PRIETO-DÍAZ+1987], mostra a mesma classificação usando o método enumerativo ou hierárquico.

Ambos os esquemas podem representar o mesmo número de classes. Contudo, no esquema enumerativo, classes com mais de um componente elementar são imediatamente incluídas na classificação, enquanto que para o esquema facetado tem-se que sintetizar elementos de múltiplas classes. Ordenação sistemática em uma classificação facetada consiste em dispor as facetas em ordem de citação, de acordo com sua relevância para os

usuários da classificação. Termos nas facetas são ordenados por seu inter-relacionamento (proximidade conceitual) [PRIETO-DÍAZ+1987].

Tabela 5 - Um exemplo de classificação enumerativa [PRIETO-DÍAZ+1987]

Fisiologia
Respiração
Reprodução
Animais aquáticos
Fisiologia de animais aquáticos
Respiração de animais aquáticos
Reprodução de animais aquáticos
Animais terrestres
Fisiologia de animais terrestres
Respiração de animais terrestres
Reprodução de animais terrestres
Invertebrados
Fisiologia de invertebrados
etc.

A classificação facetada, segundo [LEISE2001], pode ajudar a analisar textos, capturar todos os itens relevantes que devem ser classificados e assegurar a consistência ao classificar os mesmos tipos das coisas. [PRIETO-DÍAZ 1987] argumenta ainda que esquemas facetados são mais flexíveis, mais precisos e mais adequados à classificação de coleções de muitos elementos e com crescimento permanente. Estes esquemas facetados são construídos utilizando um processo denominado de *Literary Warrant*, que consiste em selecionar uma amostra aleatória dos títulos da coleção a ser classificada, listar os termos individuais dos títulos, agrupar termos relacionados em classes comuns e organizar estas classes num esquema de classificação definido a partir dos grupos. As facetas são então classificadas em ordem de citação e os termos em cada faceta ordenados arbitrariamente, de acordo com as necessidades dos usuários. Um esquema construído a partir de *Literary Warrant* gera esquemas específicos e personalizados. Contudo, novos termos podem ser facilmente adicionados ao esquema de classificação.

O processo *Literary Warrant* pode ser melhor ilustrado através de um exemplo encontrado em [PRIETO-DÍAZ2002]. Suponhamos que tenha sido solicitada a construção de uma classificação para uma lista de títulos de livros relacionados com zoologia. A primeira etapa é selecionar uma amostra representativa da coleção. Vamos supor que os seguintes títulos tenham sido selecionados:

- Ensaios sobre fisiologia da fauna marinha
- Animais das montanhas
- Animais anfíbios
- Répteis do deserto
- Pássaros migratórios
- Peixes de água salgada
- Reprodução dos mamíferos
- Serpentes do Rio Amazonas
- Relatórios experimentais sobre a respiração dos vertebrados
- Traça/mariposa de folhas tropicais

A etapa seguinte é agrupar termos comuns, isto é, aglomerar conceitos:

- fisiologia, reprodução, respiração
- trópico, deserto, montanhas, água salgada
- marinho, anfíbio
- fauna, animais, vertebrados, répteis, serpentes, pássaros, peixes, traças, mamíferos
- ensaios, relatórios experimentais

Estes cinco grupos são as facetas iniciais da coleção de livros de zoologia. Cada grupo é nomeado pelo conceito geral que representa:

- pelo processo
- pelo habitat
- pelo elemento
- pela taxonomia[6]
- pelo formulário literário

Estas cinco facetas são ordenadas por sua relevância para os usuários da coleção e os termos dentro de cada grupo são listados em uma ordem lógica. Supõe-se neste exemplo que os usuários da coleção são principalmente ecólogos assumindo o habitat como a faceta mais relevante. A área de domínio ou assunto é animais/fauna e o esquema de classificação facetado resultante é mostrado na tabela 6, extraído de [PRIETO-DÍAZ2002].

Como títulos novos podem incorporar a coleção, novas facetas podem ser definidas e

[6] Taxonomia aqui é uma classe de classificação na Zoologia

termos novos adicionados ao esquema, assim ampliando e enriquecendo o esquema facetado. Este é o núcleo da *Literary Warrant*.

Este esquema é um vocabulário estruturado e controlado que pode ser usado sistematicamente para definir cada título da coleção. Cada título pode agora ser reduzido a uma forma normal de termos de cada faceta. Para descrever um título usando este esquema combina-se, pela ordem de relevância, cada termo no título ao termo no esquema.

Por exemplo, o título "Ensaios sobre fisiologia da fauna marinha" pode ser representado (isto é, classificado) pelo seguinte conjunto de termos selecionados do esquema facetado:

/null/marinho/animais/fisiologia/ensaios/

A primeira entrada é nula porque não há nenhum termo (isto é, conceito) no título que corresponda a qualquer conceito na faceta do habitat. Os termos restantes são selecionados combinando, conceitualmente, palavras-chaves do título aos termos da faceta no esquema. Similarmente, os títulos abaixo são normalizados (isto é, classificados) seguindo o mesmo processo:

"Animais das montanhas" ⇒ /montanhas/null/animais/null/null/

"Répteis do deserto" ⇒ /deserto/null/répteis/null/null/

Tabela 6 - Esquema de classificação facetado resultante do domínio animais/fauna [PRIETO-DÍAZ2002]

{ habitat}	*{elemento}*	*{ taxonomia}*	*{ processo}*	*{formulário literário}*
Terra	Marinho	Animais/fauna	Fisiologia	Ensaios
Trópico	:	Invertebrados	Respiração	:
Deserto	:	Insetos	Reprodução	:
Montanha	Anfíbio	Traças	:	Relatórios
:	:	:	:	Experimental
:	:	Vertebrados	:	:
Água		Mamíferos		
Mar		:		
Rio		Pássaros		
Amazonas		:		
:		Répteis		
:		Serpentes		
		:		
		Peixes		

Este é o processo de classificação sintética usando um esquema de classificação facetada. O esquema fornece um vocabulário e algumas regras básicas para converter títulos em um conjunto normalizado de conceitos. O conjunto das facetas que descreve um

componente é chamado de descritor da faceta.

Em resumo, foi produzida, utilizando uma abordagem ascendente, uma estrutura do conhecimento que pode ser usada para gerar descritores normalizados das declarações de conteúdo que facilitam sua categorização; títulos que compartilham os mesmos termos da faceta pertencem à mesma categoria.

A vantagem da classificação facetada é a flexibilidade. As facetas e seus respectivos valores podem ser facilmente adicionados, suprimidos ou modificados.

Tabela 7 - Um esquema simplificado facetado para componente Unix [PRIETO-DÍAZ1991]

{pela ação}	{pelo objeto}	{pela estrutura de dados}	{pelo sistema}
get	file-names	buffer	line-editor
put	indentifiers	tree	text-formatter
update	line-number	table	
append	character	file	
check	number	archive	
detect	expression		
locate	entry		
search	declaration		
evaluate	line		
compare	pattern		
make			
build			
start			

Outro exemplo mostrado na tabela 7 é um esquema facetado desenvolvido para componentes do sistema operacional Unix e encontrado em [PRIETO-DÍAZ1991], que consiste de quatro facetas: a função executada pelo componente, os objetos manipulados pela função, a estrutura de dados onde a função ocorre e o sistema a que a função pertence. O exemplo mostra a classe dos componentes do Unix que encontram números da linha em um arquivo e são partes de um editor de linha.

3.3 Uso da Classificação Facetada

[MURRAY2002] explica que um dos benefícios primários da classificação facetada é que, mesmo não sabendo o nome de um objeto, pode-se conseguir uma compreensão com certa exatidão do que ele é através de uma descrição de diversas categorias de informações mutuamente exclusivas acerca deste objeto. Para descrever um refrigerador, por exemplo, pode-se definir o que ele é com bastante precisão especificando seu tamanho, o material com que ele é geralmente feito, sua cor, sua posição típica em uma casa e sua função primária.

Em ambientes de recuperação de informações baseados em computador, o conjunto das facetas não tem que ser exaustivo durante a classificação ou a recuperação da informação. Os projetistas de uma base de conhecimento compartilhada podem adicionar uma nova faceta a qualquer tempo e os usuários podem selecionar elementos tanto de poucas como de muitas facetas, como convier a eles. Segundo [MURRAY2002] em um ambiente de rápidas mudanças, facetas mutuamente exclusivas podem incluir:

- Produtos: uma descrição hierárquica de cada um dos produtos (sistemas) da organização. Pode ser uma descrição completa do sistema ou apenas uma descrição de cada comando (opção de menu) apresentada em um campo ou em uma caixa de diálogo;
- Aplicações: os usos práticos a que se destinam os sistemas ou onde suas características se aplicam;
- Organizações: negócios e outros grupos, que incluem clientes e prospectos de uma companhia;
- Pessoas: pessoas de dentro e fora da organização, organizadas de diversas maneiras e associadas aos elementos em outras facetas, tais como organizações, eventos, publicações, etc.;
- Objetos do domínio: por exemplo, as tecnologias aplicadas ao *marketplace* em que a organização atua;
- Eventos: não apenas conferências, mas qualquer coisa que seja, primariamente, baseada no tempo; e
- Publicações: documentos, páginas *Web*.

As unidades da informação podem ser associadas aos elementos de diferentes hierarquias de facetas. Inversamente, os usuários querem, freqüentemente, pesquisar as hierarquias; assim, eles podem ver quais unidades de informação estão associadas a um elemento em particular como, por exemplo, um comando ou uma interface usados em um

produto específico. As novas hierarquias de facetas podem ser adicionadas a qualquer momento, quando necessário. Elas simplesmente representam uma perspectiva de interesse de pelo menos uma audiência [MURRAY2002].

O vinho é um assunto que cativa muitas pessoas e durante muitos anos foi desenvolvida uma linguagem e uma organização próprias. Esta organização pode ser considerada facetada, pois os vinhos são organizados tipicamente pela cor, variedade, pela região e ocasionalmente pelo preço. Os *sites* da Internet que vendem vinho, como o Wine.com, exploraram este conceito de classificação facetada, permitindo que seus visitantes comprem o vinho selecionando características desejadas a partir das facetas.

Capítulo 4

Trabalhos Correlatos Abordando Classificações

Há na literatura diversas propostas de classificação e organização das técnicas empregadas na elicitação ou no processo completo da ER. Muitas destas técnicas são bastante conhecidas e tradicionais enquanto outras são novas. Existem ainda algumas outras que são aplicadas conjuntamente com uma ferramenta, que algumas vezes acaba sendo empregada também como técnica. Como já discutido anteriormente, não há consenso em relação à nomenclatura empregada na ER. Isto também se aplica à elicitação em particular. A seguir são apresentadas algumas destas classificações.

4.1 Escolha de Técnicas que Ajudam a Encurtar o Tempo de Desenvolvimento

Segundo [McPHEE2001] existem três métodos fundamentais para reduzir o tempo exigido para completar uma tarefa ou um conjunto de tarefas:

1. Desenvolver as tarefas mais rapidamente.
2. Desenvolver tarefas concorrentemente.
3. Desenvolver menos tarefas.

Os itens 1 e 3 podem diminuir o esforço de desenvolvimento se este for definido em termos de pessoa-hora ou pessoa-mês, mas o item 2 não produz um decréscimo de esforço de desenvolvimento. Além disso, o item 2 torna a coordenação mais complexa e eleva o risco, pois desenvolvedores devem executar tarefas concorrentemente.

[McPHEE2001] pesquisou técnicas e ferramentas que poderiam ajudar a encurtar o tempo de desenvolvimento de um projeto. Cada técnica foi analisada com relação aos três métodos de redução de tempo anteriormente citados. O objetivo desta análise é determinar se uma técnica permite a redução de tempo exigido ou pelo processo de ER, ou pelo projeto como um todo. É possível fazer uma fase de ER bastante curta, reduzindo, assim, o tempo gasto na ER. Contudo, isto aumenta o risco do projeto, pois um tempo muito grande pode ser desperdiçado para refazer, invariavelmente, a definição de requisitos. Não é gastando menos tempo na ER que se encurta o tempo do projeto. A análise destas técnicas de ER

assume que cada técnica, sozinha ou em combinação com outras, será usada para construir uma especificação de requisitos razoavelmente completa.

Para prover uma análise quantitativa [McPHEE2001] atribui a cada técnica uma nota de acordo com os três métodos de redução de tempo acima mencionados. A nota é baseada em uma das seguintes escalas:

Nota 1 => A técnica não leva em conta o critério

Nota 2 => A técnica pode ser adaptada para levar em conta o critério

Nota 3 => A técnica explicitamente leva em conta o critério

Tabela 8 - Análise de técnicas aplicáveis na elicitação visando redução de tempo [McPHEE2001]

Técnica	Tarefas mais rápidas	Tarefas concorrentes	Menos tarefas	Total
Entrevistas	2	2	2	6
Análise de Documentos	1	1	1	3
Observação	1	2	2	5
Metodologia *Sof Systems*	1	1	2	4
JAD	3	3	2	8
Cenários / Casos de Uso	2	3	2	7
Prototipação	2	2	2	6
Reuso de Requisitos	3	2	2	7

Assim, para a técnica de entrevistas temos a seguinte situação:

a)Permite desempenhar as tarefas mais rapidamente

A única maneira de fazer com que entrevistas permitam que tarefas sejam executadas mais rapidamente é assegurar que o entrevistador seja experiente e esteja preparado para a entrevista. Dependendo do número de usuários, entrevistas podem consumir uma grande quantidade de tempo. Os entrevistadores devem manter a entrevista sob controle e obter eficientemente conhecimentos dos entrevistados. Portanto, neste critério a nota recebida é 2.

b)Permite desempenhar as tarefas concorrentemente

Pequenos projetos ou implementações podem ser executados antes que as entrevistas iniciais estejam completas. Se algum participante da entrevista tiver conhecimento suficiente tanto do domínio do problema quanto da solução, é possível que a validação dos requisitos possa ser parcialmente feita durante a entrevista. Com um competente escriba tomando notas, é possível que a especificação possa ser executada durante a entrevista de forma parcial também. Para este critério a nota atribuída é 2.

c)Permite desempenhar menos tarefas

Se os devidos cuidados forem tomados durante a entrevista no sentido de capturar precisamente os requisitos e modelos discutidos, é possível que estes artefatos possam ser usados no projeto, ao invés de recriar as informações a partir da memória. A nota atribuída para este critério é 2.

[McPHEE2001] analisou diversas técnicas e a tabela 8 mostra algumas destas técnicas analisadas sob o ponto de vista de redução de tempo. Quanto mais alto o valor total atingindo, melhor a técnica se aplica ao propósito da redução do tempo da fase de elicitação de requisitos. Nota-se, por exemplo, que JAD pode produzir rapidamente os requisitos do sistema, enquanto que a análise de documentos é um processo mais lento.

4.2 Estudo Comparativo Visando Ajudar na Escolha da Técnica de Elicitação

[BELGAMO+2000] descreve algumas técnicas de elicitação de requisitos apontando as diferenças e semelhanças entre elas. Neste trabalho foram propostos vários tópicos de avaliação que serviram de base para comparação. Os tópicos propostos são explicados abaixo:

- Grupo/Indivíduo: indica se a técnica é aplicada em grupo ou individualmente;
- Contexto: indica se a técnica leva em conta o ambiente onde está se realizando a elicitação;
- Caráter de interação: indica se o usuário e o desenvolvedor sentem-se à vontade, num clima de estímulo e de aceitação mútua;
- Usa lado introspectivo: indica se a técnica faz com que o desenvolvedor volte-se para si e pense como será o serviço;
- Confiabilidade: indica se as informações colhidas são confiáveis para o desenvolvimento do projeto;

- Custo: indica o esforço gasto no uso da técnica;
- Qualidade: indica se na técnica há democracia, aprendizado mútuo, educação e resolução de conflito;

Tabela 9 - Cruzamento entre as técnicas de elicitação com os parâmetros de escolha [BELGAMO+2000]

Técnicas de Elicitação/ Parâmetros	Observação	Entrevista	Análise de Protocolo	JAD	Prototipação	Cenários
Grupo/ Indivíduo	Grupo/ Indivíduo	Indivíduo	Indivíduo	Grupo	Grupo/ Indivíduo	Grupo/ Indivíduo
Considera o contexto	Sim	Sim	Sim	Sim	Sim	Sim
Caráter de interação	Não	Sim	Não	Sim	Sim	Sim
Liberdade de percurso	Sim	Sim	Não	Não	Não	Sim
Usa lado introspectivo	Sim	Sim	Sim	Não	Não	Não
Confiabilidade	Média	Alta	Baixa	Alta	Alta	Alta
Custo	Baixo	Médio	Médio	Alto	Alto	Alto
Qualidade	Média	Média	Baixa	Alta	Alta	Alta
Padronização	Não	Não	Sim	Sim	Sim	Sim
Produtividade	Baixa	Média	Baixa	Alta	Alta	Alta
Quantidade	Baixa	Alta	Baixa	Alta	Média	Média
Compartilha-mento de informações	Não	Não	Não	Sim	Sim	Sim
Tempo	Longo	Médio	Médio	Longo	Médio	Longo
Promove cooperação	Não	Não	Não	Sim	Sim	Sim
Facilitador	Não	Não	Não	Sim	Não	Não
Valida requisitos com os usuários	Não	Sim	Não	Sim	Sim	Sim
Conflitos entre os usuários do grupo	Não	Não	Não	Sim	Sim	Sim
Evita atividade de projeto prematura	Não	Não	Não	Sim	Sim	Sim

- Padronização: indica se a técnica possui uma regra para seu uso;
- Produtividade: indica se é uma técnica que faz com que se aumente a produtividade;
- Quantidade: indica se na técnica há índices de desempenho e economia de tempo;

- Compartilhamento de informações: indica se todos os indivíduos do grupo compartilham as informações;
- Tempo: indica tempo despendido para a elicitação de requisitos;
- Promove cooperação: se a técnica promove a cooperação entre os indivíduos do grupo;
- Facilitador: se a técnica pressupõe a existência de uma pessoa com a função de guiar, levantar questões e conduzir discussões num grupo;
- Validar requisitos com os usuários: se a técnica valida os requisitos com os usuários;
- Conflitos entre usuários do grupo: se a técnica provê um meio para lidar com conflitos em grupo;
- Atividade prematura de projeto: se a técnica evita que os analistas pensem prematuramente que todos os requisitos já foram elicitados e que o projetista já comece a elaboração do sistema.

O objetivo deste estudo é ajudar na escolha das técnicas com as quais os desenvolvedores podem analisar os parâmetros de avaliação, conforme exibido na tabela 9, extraída e adaptada de [BELGAMO+2000]

4.3 ACRE

ACRE (*ACquisition Requirements)* é um *framework* proposto por [MAIDEN+1996] para ajudar os desenvolvedores selecionar técnicas a ser utilizadas num processo de ER. É um trabalho que considera um número importante de técnicas de elicitação e provê um guia para selecionar, a partir de um amplo domínio, diferentes técnicas com diferentes características e diferentes origens. ACRE foi projetado para ajudar a selecionar uma ou mais técnicas para uma sessão de elicitação e doze técnicas de elicitação são contempladas, para as quais são definidas seis facetas representando diferentes conhecimentos ou características sobre o problema. Estas facetas são os critérios de avaliação de cada técnica. A ferramenta proporciona uma interação para responder as perguntas sobre as facetas e, assim, ajudar o desenvolvedor a decidir sobre as técnicas a eleger.

Os critérios de avaliação propostos são os seguintes:

1. **Finalidade ou propósito dos requisitos**

Requisitos podem ser elicitados para diferentes propósitos como: especificação de sistemas, seleção de pacotes de software e uma base para prover um acordo sobre

requisitos. Diferentes técnicas ajudam cada uma destas seleções.

2. Tipos de conhecimentos

Linguagens de modelagem de requisitos incluem primitivas semânticas tais como eventos, estados e agentes. Diferentes métodos capturam diferentes tipos de conhecimentos.

3. Filtragem interna dos requisitos

Freqüentemente os usuários não têm consciência do seu próprio conhecimento e limites. Os problemas podem incluir má lembrança e a comunicação incompleta ou incorreta do conhecimento. Existem métodos para superar estas limitações.

4. Fenômenos observáveis

Alguns conhecimentos não podem ser comunicados pelos usuários e somente podem ser apreendidos pela observação de um sistema e seu ambiente.

5. Contexto de aquisição

A escolha do método também depende do contexto e do seu uso. A aquisição não ocorre num vácuo. Complexidade das organizações, pressões políticas, financeiras e temporais influenciam a aquisição.

6. Interdependência do método

Numa sessão de elicitação, várias técnicas podem ser empregadas. ACRE usa um conjunto de planos de aquisição que representa seqüências típicas dos métodos usados na prática. Os métodos prévios na seqüência cumprem precondições para os métodos subseqüentes. Um exemplo de um plano simples é usar entrevistas não estruturadas seguidas pelas entrevistas estruturadas. A entrevista não estruturada identifica entidades tais como problemas, requisitos e características amplas dos domínios que podem ser exploradas de uma maneira mais sistemática pelas entrevistas estruturadas.

Algumas das 12 técnicas que o *framework* contempla são as seguintes:

- Observação
- Entrevistas Não Estruturadas
- Entrevistas Estruturadas
- *Brainstorming*
- Prototipação Rápida
- Análise de Cenários

Para melhor entendimento dos critérios de avaliação, o item **2, Tipos de Conhecimentos,** é tomado como exemplo para uma análise mais detalhada conforme as definições de [MAIDEN+1996]. O tipo de conhecimento a ser adquirido é um fator que influencia a escolha da técnica. ACRE segue as práticas padrões da ES e divide o conhecimento em três amplos tipos: comportamento, processos e dados. A maioria das técnicas é eficaz para capturar o conhecimento sobre comportamentos e processos, desde que este conhecimento possa ser freqüentemente observado, verbalizado e comunicado durante as entrevistas, por exemplo. Entretanto, obter dados é mais problemático, pois a maioria dos usuários é mais consciente das suas próprias ações do que das informações existentes em seu ambiente.

A tabela 10, adaptada de [MAIDEN+1996], mostra um exemplo desta avaliação. As convenções utilizadas são as seguintes:

√√ Indica ótima adequação da técnica com o tipo de conhecimento;

√ Indica boa adequação;

= Indica fraca adequação.

Tabela 10 - Eficácia das técnicas para obter tipos diferentes de conhecimento [MAIDEN+1996]

Tipo do Conhecimento	Observação	Entrevistas Não Estruturadas	Entrevistas Estruturadas	Protocolos	*Brainstorming*	Prototipação Rápida	Análise de Cenários
Comportamento	√√	√	√	√	√	√	√√
Processo	√	√	√	√	√	√	√√
Dados	=	=	=	=	=	√	√

Para cada um dos outros 5 critérios mostrados os autores fazem uma apreciação semelhante. Por esta avaliação percebe-se que a análise de cenários é uma boa técnica para elicitar o tipo de conhecimento como um todo, enquanto que as técnicas tradicionais, como entrevistas e *brainstorming,* não são indicadas para coletar em detalhe os tipos de conhecimento, principalmente em relação aos dados.

4.4 Uma Proposta para Classificar as Publicações que Abordam a ER

[LOPES2002] propôs uma taxonomia da pesquisa em ER, organizada ontologicamente, isto é, estruturada de acordo com as atividades conduzidas durante o processo de ER. São discutidas as dimensões que afetam uma possível classificação e é apresentado um esquema facetado para classificar a pesquisa na área de ER.

[LOPES2002] partiu da premissa de que há muita pesquisa ainda a ser feita em ER e que qualquer esforço para encurtar o tempo de familiarização de novos estudantes ou pesquisadores com os conceitos da área seria bem vindo. Uma idéia nessa direção seria criar um mecanismo que facilite a seleção dos textos a serem lidos, classificando-os de acordo com critérios específicos. Isso poderia não só auxiliar estudantes e pesquisadores, mas também profissionais da área em busca de soluções ou idéias para resolver problemas específicos, a atingir seus objetivos mais rapidamente.

Tabela 11 - Esquema de classificação facetado proposto por [LOPES2002] para classificar publicações em ER

(Tipo de publicação)
Conceituação
Método-procedimento
Estudo de caso
Investigação-solução
Compêndio
Posicionamento
(Problema original)
Especificação
Representação
Consenso
(Atividade do ciclo de vida)
Elicitação
Análise e negociação
Documentação
Validação
Gerenciamento
(Domínio de Aplicação)
Sistemas de Informação
Outros Sistemas

Segundo [LOPES2002] para "classificar a pesquisa na área de ER é necessário definir claramente o que está sendo considerado como pesquisa. A princípio, as idéias obtidas com a investigação científica em algum campo do conhecimento são a verdadeira produção da pesquisa. Essas idéias são então desenvolvidas e, algumas delas acabam por encontrar um caminho que possibilite sua aplicação – prática ou teórica – que justifique sua ampla divulgação e discussão no meio acadêmico e, posteriormente, no meio industrial, se for o caso. Em geral, idéias obtidas com uma linha de investigação científica são associadas a modelos e/ou idéias anteriores na produção de um artigo que, submetido a algum veículo de divulgação técnica, junta-se a outros artigos e transforma-se em uma publicação. Essas publicações revestem as idéias oriundas da pesquisa de um caráter mais sólido, uma vez que o crivo editorial aumenta-lhe a confiabilidade".

Assim, para classificar a pesquisa na área de ER, [LOPES2002] considerou que ela está identificada na produção literária resultante da publicação de artigos nos diversos veículos reconhecidos, nacional e internacionalmente, pela comunidade acadêmica e industrial.

A taxonomia proposta por [LOPES2002] é apresentada na forma de um esquema de classificação facetada, no qual as facetas são definidas em função de dimensões específicas escolhidas com base na argumentação apresentada e na bibliografia coletada durante o trabalho. Por esta proposta, cada publicação é analisada e classificada nas facetas adequadas. Na tabela 11, extraída de [LOPES2002], descreve o esquema de classificação facetado utilizado na taxonomia proposta.

De acordo com a proposta de [LOPES2002], a publicação de [GOGUEN+1993] é representada na taxonomia conforme pode ser observado na figura 6.

Referência	Techniques for Requirements Elicitation Joseph A.Goguen e Charlotte Linde Proceedings of the International Symposium on Requirements Engineering, 1993	**[GOGUEN1993]**
Descrição	Análise das técnicas aplicáveis à elicitação de requisitos, enfatizando, particularmente, a forma como cada uma delas lida com as questões sociais da atividade.	
Classificação	**Compêndio / consenso / elicitação**	

Figura 6 - Exemplo de classificação facetada proposto por [LOPES2002]

Capítulo 5

Parâmetros Para a Escolha das Técnicas de Elicitação

Neste capítulo são apresentados alguns parâmetros que podem ser levados em consideração para a escolha das técnicas a ser utilizadas no processo de elicitação dos requisitos.

5.1 Processo de Escolha das Técnicas

Desde que a ER surgiu os desenvolvedores de sistemas têm se defrontado com a questão de encontrar a melhor forma para a identificação dos requisitos do sistema. Durante o processo de elicitação pode ser utilizada uma ou diversas técnicas e muitas destas técnicas partem da premissa de que a organização para a qual a aplicação irá ser desenvolvida é estável (no sentido de que todos os seus processos já estão definidos, todos os produtos desenvolvidos possuem a sua fatia de mercado, os concorrentes não incomodam, etc.) e que os requisitos existem, bastando apenas coletá-los e documentá-los. Entretanto, a atual situação mostra que isto nem sempre pode ser tomado como verdade. A tecnologia da informação está mudando o modo de vida das pessoas e a forma de relacionamento entre pessoas, entre pessoas e organizações e entre organizações. Isto acaba gerando empresas em constantes mudanças, não existindo requisitos fixos para um determinado sistema de informações. Muitas vezes, por uma situação de mercado, por exemplo, é necessário o desenvolvimento de um novo produto rapidamente e este novo produto tem que ser confiável, seguro e desempenhar suas funções de acordo com o especificado. Neste caso, é indispensável o correto processo de definição dos requisitos, exigindo por parte dos desenvolvedores o uso acertado das técnicas de elicitação [McPHEE2001].

Para [WESSELS2002] em qualquer projeto de análise, algumas das dificuldades-chave estão na decisão de como coletar a informação, quantas informações obter e quanto tempo gastar com a coleta da informação.

A seguir é apresentada e discutida uma lista de parâmetros que podem fornecer os subsídios necessários ao processo de seleção das técnicas.

5.2 Parâmetros para Agrupamento das Técnicas

Existem diversas maneiras de agrupar as técnicas e aqui está sendo proposta a utilização de alguns parâmetros básicos para classificá-las.

5.2.1 Papel Exercido pelo Usuário no Uso das Técnicas

Várias técnicas foram desenvolvidas para envolver usuários e cada uma varia no nível de participação destes usuários, ou seja, cada uma delas exige um tipo de comportamento dos usuários durante o processo de elicitação. A escala de variação da participação pode ser classificada em três estilos principais [HUSSEIN1997];[ROCHA2002]:

- **Consultivo:** O poder de tomar decisão está nas mãos dos desenvolvedores e os usuários são fontes de informação com quase nenhuma influência, desempenhando um papel passivo. Exemplos de técnicas neste estilo são: entrevistas, reuniões estruturadas e *brainstorming*.

- **Representativo:** Os usuários representativos estão envolvidos na formulação do projeto e em algumas tomadas de decisões não muito importantes. Um exemplo de técnica neste estilo é JAD.

- **Decisório:** Os usuários são envolvidos durante todo o processo do projeto, inclusive tomando decisões. O desenvolvimento do sistema é de responsabilidade deles também. Há um consenso entre usuários e desenvolvedores. Algumas das técnicas neste estilo são: *Participatory Design (PD)*[7] e aprendizagem com o usuário

Um quarto tipo de participação, **Apoio Geral**, pode ser agregado para demonstrar situações em que o usuário pode exercer diversos papéis, até mesmo englobando algum dos três listados acima. Algumas técnicas neste estilo são casos de uso e prototipação que ajudam os usuários a ir conhecendo os resultados da elicitação.

[7] Projeto Participatório

5.2.2 Formalidade

Existem diversas maneiras para modelar o domínio das informações. Para descrever um modelo utiliza-se uma linguagem ou uma notação. De acordo com os objetivos a ser alcançados, as características do ambiente e o objeto a ser modelado aplicam-se técnicas de elicitação que melhor se ajustam a esta situação. Estas técnicas de elicitação de requisitos podem ser divididas em formais, semi-formais e informais.

As técnicas informais são baseadas em comunicação estruturada e interação com o usuário, questionários, estudo de documentos, etc. Os modelos do problema e do produto são construídos na mente dos desenvolvedores, que podem fazer uso de notação informais, que são traduzidas diretamente para o documento de especificação de requisitos. São exemplos de técnicas informais o JAD, *brainstorming* e as entrevistas [CARVALHO+2001]. Nesta abordagem, emprega-se a língua natural para descrever os modelos.

Os modelos semi-formais utilizam-se de uma notação gráfica que permite a descrição dos elementos do domínio de maneira clara, abstrata e com pouca ambigüidade. A maioria das metodologias de análise utilizam notações semi-formais. Estes modelos possibilitam uma visão ampla do sistema e podem ser utilizadas para a descrição de informações. Exemplos de técnicas utilizadas nesta abordagem são o Diagrama Entidade-Relacionamento, Diagrama de Funções e Diagrama de Fluxo de Dados [LEITE+1999].

Já as técnicas formais pressupõem a construção de um modelo conceitual do problema que está sendo analisado[CARVALHO+2001]. "Modelos formais requerem uma linguagem formalmente definida, isto é, com a semântica e a sintaxe definidas matematicamente. Modelos formais são bastante utilizados para a especificação formal da funcionalidade do sistema" [LEITE+1999]. Exemplos de técnicas formais: Z [MOURA2001], B e VDM.

Linguagens e métodos de especificação formal baseadas em modelos/fundamentos matemáticos têm vantagens sobre as abordagens informais de captura de requisitos de software, defende [CYBULSKI1996]. A descrição formal do domínio do problema também permite a criação de estruturas formais de desenvolvimento, que podem levar a custos adicionais, porém levam também a ganhos de produtividade.

Infelizmente, a extração de requisitos não se resume às conveniências de notações, à rigidez do processo de elicitação ou à completitude e à coesão dos requisitos coletados. Como observado em algumas práticas de engenheiros de requisitos e relatos de pesquisadores, as notações informais são, freqüentemente, a única forma de expressão de requisitos aceita e compreendida pelos usuários. Enquanto é desejável especificar

formalmente requisitos em algum estágio da ER, é também crítico que os usuários definam estes requisitos, uma vez que não é esperado que eles aprendam e usem eficazmente formalismos complexos. Como os primeiros estágios da aquisição dos requisitos são motivo de muita negociação entre os usuários e desenvolvedores, os requisitos iniciais de usuários são, necessariamente, informais, isto é, vagos, ambíguos e incompletos. Os métodos formais podem ser bem sucedidos quando usados depois que o consenso sobre os requisitos foi alcançado [CYBULSKI1996].

A língua natural é a forma mais familiar de comunicação e ela também faz com que a atenção dos potenciais leitores seja direcionada aos componentes importantes da especificação explicitamente expressados. Por outro lado, ela é também a forma pela qual a informação implícita pode ser omitida mais facilmente. As especificações de requisitos de software elaboradas em língua natural são mais concisas, menos complexas e, portanto, mais baratas na manutenção do que aquelas escritas em uma notação formal [CYBULSKI1996].

5.2.3 Categorias de Aplicação

Uma outra maneira de classificação é agrupar as técnicas por alguma categoria de aplicação. [ROCHA2000] mostra um conjunto de técnicas que são comumente usadas na obtenção e modelagem de requisitos, sabendo-se de antemão que, individualmente, nenhuma consegue satisfazer as diferentes categorias de requisitos de informação. As técnicas foram divididas em 4 categorias relacionadas à sua aplicabilidade: técnicas de observação, técnicas de levantamento não-estruturado, técnicas de mapeamento e técnicas de levantamento estruturado.

A seguir são mostrados exemplos de técnicas classificadas por estas categorias:

- **observação:** observação do comportamento, aprendizagem com o usuário, prototipação;
- **levantamento não-estruturado:** entrevistas abertas, *brainstorming*;
- **mapeamento:** Metodologia *Soft Systems* (MSS) ou *rich pictures*, diagramas de fluxos de dados (DFDs), análise de dados, análise de decisões, análise de objetos; e
- **levantamento estruturado:** cenários, entrevistas estruturadas, reuso de requisitos, JAD.

5.2.4 Abordagens Organizacionais

[ROCHA2000] relembra que durante um processo de ER devem ser considerados vários aspectos do domínio de um problema de modo que o resultado desta atividade seja o mais proveitoso possível na satisfação da sua solução. Olhando para esses aspectos, encontram-se essencialmente dois grupos de preocupações principais: **técnico/tecnológicas e sociais e/ou organizacionais.**

Tabela 12 - Principais características das abordagens tecnológicas e sócio-organizacionais [ROCHA2000]

	Tecnológicas	**Sócio-organizacionais**
Princípios	Objetivas	Subjetivas
Enfoque	Sistema formal Dados	Sistema Informal Informações
Modelos	Primeiro modelo conceitual do sistema existente	Primeiro modelo conceitual das perspectivas ou ideais das pessoas que fazem parte do sistema
Tipos de técnicas	Estruturadas e rígidas	Etnográficas, interativas e flexíveis
Estratégia	Divide um problema em partes para análise individual	Vê um problema como um todo que pode ser melhorado
Papel dos analistas	Decisores/condutores	Facilitadores
Papel dos usuários	Consultivo/passivo	Participativo e decisório
Ponto forte	Resultados previsíveis	Proporcionam ou potencializam a reengenharia e inovação de sistemas
Ponto fraco	Proporcionam ou potencializam a reparação e polimento de sistemas velhos e obsoletos	Resultados não previsíveis

Considerando esta constatação, [ROCHA2000] apresenta três tipos de abordagens, que caracterizam os diferentes tipos de processos de ER que se podem encontrar nas organizações:

- **Abordagens Tecnológicas:** realçam a visão objetiva dos requisitos e os aspectos técnicos/tecnológicos dos SI e que englobam os conhecidos métodos *hard*, tradicionais ou estruturados.
- **Abordagens Sócio-organizacionais**: enfatizam a visão subjetiva dos requisitos e os aspectos sociais e/ou organizacionais dos SI, e que englobam os métodos que são conhecidos por métodos *soft*, interpretativistas ou construtivistas.
- **Abordagens Mistas**: apresentam situação intermediária, isto é, em que existe algo de cada uma das duas abordagens anteriores. Geralmente consideram-se primeiro as questões sócio-organizacionais, deixando as questões técnicas/tecnológicas para uma segunda fase.

De acordo com [ROCHA+2002], algumas das técnicas representativas de cada abordagem são:

- **Tecnológica:** análise de dados, análise de decisões, análise de objetos, análise de textos, entrevistas estruturadas, reuso de requisitos;
- **Sócio-organizacional:** aprendizagem com o usuário, *brainstorming*, MSS (*rich pictures)*;
- **Mista:** observação do comportamento, prototipação, entrevistas abertas, mapeamento cognitivo, cenários, JAD.

Na tabela 12 são mostradas as principais características das abordagens tecnológicas e sócio-organizacionais segundo [ROCHA+2002]. A abordagem mista não aparece na tabela, uma vez que ela é uma conjugação das características das outras duas.

5.2.5 Fontes de Obtenção dos Requisitos

As pesquisas em ER são feitas com o objetivo de desenvolver técnicas que permitem que o processo de elicitação seja executado de forma mais eficiente, produtiva e segura, criando formas de facilitá-lo. Segundo [DAVIS2002], estas técnicas variam consideravelmente dependendo da situação. Algumas se aplicam exclusivamente ao processo de extração de requisitos a partir de pessoas. Outras servem para obter requisitos de outras fontes, como, por exemplo, documentação dos sistemas existentes.

Na grande maioria das técnicas há o envolvimento de desenvolvedores e usuários que se encontram em sessões de trabalho, num determinado período, com o objetivo de acelerar as negociações entre pessoas com opiniões diferentes, prover os analistas de uma profunda compreensão das necessidades do sistema e dos usuários e elaborar um rascunho/anotações

com os requisitos descobertos [BRACKETT1990]; [KUCHMISTAYA2001].

[CARVALHO+2001] mostra que as técnicas de elicitação servem para superar as várias dificuldades inerentes ao processo. Algumas tratam das dificuldades de comunicação, enquanto outras tratam de dificuldades técnicas ou do comportamento humano. Nenhuma técnica por si só é suficiente para projetos reais. O desenvolvedor deve ser capaz de escolher um conjunto de técnicas que melhor se adaptem ao sistema a ser desenvolvido e às condições de obtenção. Portanto, para capturar os requisitos existem técnicas que se aplicam na obtenção de requisitos a partir de um indivíduo, grupo de pessoas, documentos, observação. Outras podem ser mistas, ou seja, usam fontes combinadas. Alguns exemplos dessas técnicas são:

- **Indivíduo:** Entrevistas, Análise de protocolo;
- **Grupos:** JAD, QFD[8];
- **Mista:** Prototipação, Cenários;
- **Documentos:** Análise de documentos, Reuso de Requisitos;
- **Observação:** Observação e Análise Social.

5.2.6 Técnicas Aplicáveis às Diferentes Fases da ER

Uma mesma técnica pode ser empregada em fases diferentes da ER e com objetivos diferentes com o propósito de facilitar o trabalho dos desenvolvedores e a interação com os usuários. [McPHEE2001] pesquisou diversas destas técnicas aplicáveis em qualquer uma das fases da ER e as categorizou baseando-se na fase da ER, na qual elas seriam mais comumente usadas, ou seja, na sua finalidade primária. Por exemplo, embora entrevistas possam ser usadas para ajudar a validação, ou até mesmo analisar os requisitos, elas são mais comumente usadas como uma técnica de elicitação. A tabela 13 mostra um resumo dessa categorização.

Pode-se observar, por exemplo, que os casos de uso podem ser utilizados em todas as fases da ER, enquanto que a análise de documentos serve exclusivamente para a elicitação.

[ALENCAR1999] destaca também técnicas que são usadas, primariamente, para realizar a elicitação dos requisitos, dentre as quais distinguem-se:

- Entrevista;
- Leitura de Documentos;
- Questionários;
- Análise de Protocolos;

[8] *Quality Function Deployment*

- Cenários;
- MSS;
- Observações;
- Reuso de Requisitos; e
- Prototipação.

Tabela 13 - Aplicabilidade das técnicas e ferramentas às fases da ER.

Ferramenta/Técnica	Elicitação	Análise	Especificacão	Validação
Entrevistas	Primária			Secundária
Análise de documentos	Primária			
Observação	Primária			Secundária
MSS	Primária	Secundária		Secundária
JAD	Primária			Secundária
Cenários / Casos de Uso	Secundária	Primária	Secundária	Secundária
Prototipação	Secundária		Secundária	Primária
Reuso de Requisitos	Secundária		Primária	

O domínio do problema e a equipe participante são fatores fundamentais que o desenvolvedor terá de considerar para selecionar as técnicas a serem utilizadas e estabelecer de que maneira elas serão integradas.

Para [ALENCAR1999], dentre as técnicas apontadas, a prototipação apresenta-se como uma das mais efetivas para validação de requisitos. Quando se dispõe de uma especificação de requisitos executável, a validação fica facilitada, pois aquela já pode ser diretamente utilizada como teste pelos próprios usuários. Com isto pode-se ter redução dos custos e dos prazos de desenvolvimento dos sistemas. Daí também a importância do desenvolvimento de protótipos nesta fase.

5.2.7 Nível de Treinamento/Conhecimento do Desenvolvedor na Técnica

O desenvolvedor deve conhecer as técnicas que serão empregadas no processo de elicitação. Deve passar por algum processo de treinamento, pois cada técnica exige níveis diferentes de domínio. O uso correto das técnicas pode evitar muitos erros de requisitos. O investimento em treinamento pode ser alto, porém, desenvolvedores mal treinados podem acarretar perdas ainda maiores. O retorno do investimento no treinamento pode ser recuperado já no primeiro projeto. Nos próximos projetos, a habilidade da equipe na técnica pode trazer mais economias ainda, sem contar os ganhos intangíveis.

Por exemplo, é desejável que o desenvolvedor seja treinado em técnicas de entrevista.

Na prototipação, o treinamento não é exigido, embora o método possa requerer algum grau de treinamento a fim de se adquirir as habilidades básicas. Por outro lado, a experiência do desenvolvedor na aplicação da técnica, adquirida ao longo de sua carreira, também conta muito. O coordenador/gerente deve avaliar o conhecimento de cada membro da equipe em cada técnica para verificar a viabilidade do seu uso. Os níveis de conhecimentos podem ser baixo, médio, alto, domínio completo [KIRAKOWSKI1997].

Alguns conhecimentos fundamentais e práticos desejados/exigidos do desenvolvedor são:

- Experiência/domínio no uso de técnicas de elicitação;
- Conhecimento e experiência no uso de alguma ferramenta CASE;
- Conhecimento e experiência em técnicas de modelagem;
- Conhecimento de conceitos, técnicas e ferramentas de interface homem-computador; e
- Conhecimento de técnicas de planejamento e marketing.

5.2.8 Habilidades Exigidas do Desenvolvedor

Como já mostrado anteriormente a ER não é uma atividade isolada no ciclo de vida do software. Ela é parte integrante de um processo mais amplo. Os ciclos de vida do software devem ser projetados prevendo que mudanças ocorrerão, pois isto é inerente ao processo. Haverá também uma necessidade contínua para justificar os requisitos e existirão conflitos/divergências nos requisitos, que precisam ser identificados e resolvidos. Além disso, o desenvolvedor estará sempre em contato com algum usuário, o que faz com que ele deva possuir um perfil social adequado para relacionar-se com as pessoas de diferentes personalidades.

Como conseqüência, o desenvolvedor, aqui exercendo o papel de engenheiro de requisitos, necessita ter algumas qualidades para conduzir este processo. É necessário que o engenheiro esteja apto para entrevistar os usuários; tenha recursos para identificar problemas; analisar situações; seja hábil para facilitar, negociar e resolver problemas ou propor soluções alternativas e possua habilidades de apresentação e liderança e saiba trabalhar em grupo [SPRINGL1998].

O desenvolvedor deve também ser hábil para tornar evidentes questões sociais ou organizacionais no momento da elicitação. Segundo [ROCHA+2002] "O que normalmente se vê na definição de requisitos são engenheiros de requisitos assumirem uma postura de condutores e decididores, perguntando aos usuários que sistema querem, mas sem deixá-los pensar e envolverem-se, ou seja, sem a preocupação de descobrirem necessidades sociais

ou organizacionais do domínio do problema". Com efeito, a introdução de um SI numa situação real de trabalho sempre afeta o ambiente social e organizacional onde o sistema atua.

O engenheiro de requisitos deve ter habilidades para conduzir a atividade de coleta de requisitos, para obter o máximo possível de informações sobre o domínio do problema e deve ser capaz de gerar idéias, motivar e evitar inibições. Por exemplo, numa sessão de *brainstorming* o desenvolvedor deve ser capaz de iniciar uma discussão e motivar o grupo. Se estiver empregando a técnica de observação ele deve ser capaz de registrar os eventos observados.

A seguir são listadas algumas habilidades que o desenvolvedor deveria possuir:

- Motivador do grupo;
- Saber conduzir reuniões;
- Saber planejar;
- Ser neutro;
- Capacidade de registrar/anotar eventos;
- Capacidade de apresentar e capturar idéias;
- Gerenciar o grupo;
- Ser receptivo;
- Capacidade de projetar;
- Saber montar projetos e analisar pesquisas;
- Saber entrevistar;
- Capacidade de edição, numa vídeo-prototipação por exemplo;
- Elaborar testes/testar;
- Trabalhar em grupo;
- Negociador (visando conseguir consenso);
- Capacidade de apresentação em público;
- Saber encaminhar a resolução de problemas propondo soluções alternativas;
- Saber especificar/modelar;
- Capacidade de desenvolvimento de software ou construção de modelos;
- Saber aproveitar análises anteriores;
- Capacidade de redigir ou escrever; e
- Possuir habilidades analíticas para facilitar, por exemplo, a análise da situação organizacional.

5.2.9 Custo da Técnica

O custo aqui estudado contempla o tempo e esforços empregados no uso da técnica e não o custo financeiro na aplicação da técnica. Desenvolver rapidamente um sistema é desejo de todos e a escolha da técnica influencia nisto. Para [WESSEL2002], um dos aspectos-chave que o analista deve considerar é justamente quanto irá custar o esforço e o tempo gastos no uso de cada técnica. JAD e entrevistas, por exemplo, são significativamente mais caras do que outras técnicas, pois as entrevistas exigem uma considerável preparação do entrevistador. Em termos de tempo *brainstorming* pode ter um custo baixo, pois uma sessão não precisa levar mais que uma hora, principalmente se as pessoas forem criativas e já possuírem alguma experiência no uso da técnica. Já em termos de esforços, *brainstorming* pode ter um custo relativamente alto, pois requer um número significativo de pessoas. Questionários dependem muito da complexidade da pesquisa e do número de pessoas que os responderão. Dependendo do meio a ser utilizado pode haver um custo financeiro relativamente alto como, por exemplo, postagem, se for usado algum serviço de envio, ou custos com telefonemas, se a pesquisa for feita por este meio.

Entretanto, tudo isto deve ser ponderado levando-se em conta a quantidade de informação útil coletada versus o custo para obtê-la. O uso de uma técnica pode ter custo baixo, médio ou alto.

5.2.10 A Finalidade da Informação Coletada pela Técnica

Uma técnica normalmente pode ser útil para capturar requisitos tanto para um sistema novo como para outro já existente. Entretanto, algumas oferecem melhores condições para um ou outro caso; a maioria das técnicas é útil para compreender o sistema atual, mas a análise de documentos e a observação são geralmente menos úteis para projetar um novo sistema. De uma maneira geral, os questionários podem ser úteis para sistemas novos, mas não no mesmo grau de entrevistas ou sessões JAD [WESSEL2002].

O tipo de informação coletado pela técnica é indicado para:

- Sistema novo
- Sistema atual
- Ambos

5.2.11 A Quantidade de Informação Coletada pela Técnica

As entrevistas e as sessões JAD produzem muita informação em profundidade, enquanto que análise de documentos e a observação são úteis em extrair fatos, mas não em muita profundidade. Por outro lado questionários e análise de documentos conseguem capturar efetivamente uma grande quantidade de informações de diversas áreas. JAD e entrevistas requerem que o analista visite cada uma das fontes de informação e isto pode implicar numa grande disponibilidade de recursos [WESSEL2002].

Uma técnica pode produzir informações em:

- profundidade;
- largura (amplitude).

5.2.12 Nível de Participação do Usuário

Sessões JAD requerem uma grande participação e preparação do usuário, entrevistas um pouco menos e as outras técnicas bem menos. Em geral, uma grande participação do usuário dá uma exatidão maior, mas a um custo significativo [WESSEL2002]. O tipo de participação abordado neste trabalho é a presencial nas reuniões ou sessões. Na observação, por exemplo, o usuário participa muito pouco, contudo, ele é intensamente observado.

A participação exigida dos usuários pode ser:

- Baixa
- Média
- Alta

Capítulo 6

A Taxonomia Proposta

Neste capítulo é apresentada a taxonomia proposta para as técnicas de elicitação de requisitos. A proposta é utilizar o esquema de classificação facetado, explicado no Capítulo 3 para classificar as técnicas criando, assim, a taxonomia. Os parâmetros de classificação propostos e descritos no Capítulo 5 comporão as facetas e termos do esquema.

Como no exemplo do refrigerador, citado anteriormente, **aqui entende-se que a descrição, os processos, a aplicabilidade, as diretrizes, as vantagens, as desvantagens e o processo da técnica fornecem os elementos que se encaixam em algum termo das facetas propostas.** Assume-se que o papel exercido pelo usuário é a faceta mais importante a partir do ponto de vista dos desenvolvedores. Então, o esquema de classificação facetado proposto para as técnicas de elicitação de requisitos, contendo as facetas e os seus respectivos termos, é o seguinte:

{Faceta do papel exercido pelo usuário}

- Consultivo;
- Representativo;
- Decisório; e
- Apoio geral.

{Faceta da formalidade da técnica}

- Formal;
- Semi-formal; e
- Informal.

{Faceta das categorias de aplicação}

- Observação;
- Levantamento não-estruturado;
- Mapeamento; e
- Levantamento estruturado.

{Faceta das abordagens organizacionais}

- Tecnológica;
- Sócio-organizacional; e
- Mista.

{Faceta das fontes de obtenção dos requisitos}

- Indivíduo;
- Grupos;
- Mista;
- Documentos; e
- Observação.

{Faceta das técnicas aplicáveis às diferentes fases da ER}

- Entendimento do domínio;
- Elicitação e análise;
- Especificação e documentação;
- Validação;
- Gerenciamento; e
- Controle de qualidade.

{Faceta do treinamento do desenvolvedor na técnica}

- Baixo;
- Médio;
- Alto; e
- Domínio completo.

{Faceta das habilidades exigidas do desenvolvedor}

- Motivador do grupo;
- Saber conduzir reuniões;
- Saber planejar;
- Ser neutro;
- Capacidade de registrar/anotar eventos;
- Capacidade de apresentar e capturar idéias;
- Gerenciar o grupo;
- Ser receptivo;
- Capacidade de projetar;
- Saber montar projetos e analisar pesquisas;
- Saber entrevistar;
- Capacidade de edição, numa vídeo-prototipação por exemplo;
- Elaborar testes/testar;
- Trabalhar em grupo;
- Negociador (visando conseguir consenso);
- Capacidade de apresentação em público;
- Saber encaminhar a resolução de problemas propondo soluções alternativas;
- Saber especificar e modelar;
- Capacidade de desenvolvimento de software ou construção de modelos;

- Saber aproveitar análises anteriores;
- Capacidade de redigir ou escrever; e
- Possuir habilidades analíticas para facilitar, por exemplo, a análise da situação organizacional.

{Faceta do custo da técnica}

- Baixo;
- Médio; e
- Alto.

{Faceta da finalidade da informação coletada}

- Sistema novo;
- Sistema atual; e
- Ambos.

{Faceta da quantidade informação coletada}

- Profundidade; e
- Largura (amplitude).

{Faceta do nível de participação do usuário}

- Pouca;
- Média; e
- Muita.

Como escolher as técnicas corretas para a elicitação? Como aplicar as técnicas certas para o usuário certo? O uso de alguma técnica é realmente necessário ou a elicitação pode ser feita empiricamente?

Nesta seção é apresentada uma proposta de classificação das técnicas que fornece subsídios para ajudar os desenvolvedores a escolher as técnicas adequadas. O objetivo não é direcionar a escolha para alguma técnica que sempre pode apresentar bons resultados, mas fornecer parâmetros que os ajudem nesta escolha. Cada projeto tem características diferentes e o que é fundamental para alguns pode não ser para outros.

Diferentes técnicas são apropriadas para uso em contextos diferentes, em diferentes tipos de projetos, em estágios diferentes de um projeto, para desenvolver diferentes tipos de sistemas ou para resolver diferentes tipos de problemas na elicitação.

A seguir algumas técnicas são **classificadas de acordo com a taxonomia proposta** e o uso destas técnicas é discutido, abordando os seguintes aspectos: **descrição da técnica, áreas típicas de aplicação e quando ela pode ser usada, processo, benefícios, limitações, vantagens, desvantagens, pontos fortes e fracos, necessidades e diretrizes**

para bom uso. A idéia básica de como abordar cada técnica foi obtida de [MAGUIRE1998] e [KIRAKOWSKI1997].

Durante a pesquisa para a elaboração deste trabalho foram encontradas mais de cinqüenta técnicas envolvendo todo o processo da ER. Nem todas as técnicas mencionadas até agora serão descritas, somente as comumente utilizadas na elicitação.

6.1 Entrevistas

6.1.1 A classificação na taxonomia

/ Consultivo / Informal / Levantamento não estruturado / Mista / Mista / Validação / Médio / Ser neutro / Médio / Ambos / Profundidade/ Média /

6.1.2 Descrição da técnica

Entrevista é a técnica de elicitação mais utilizada. Entrevistas são praticamente inevitáveis em qualquer desenvolvimento já que são algumas das formas de comunicação mais natural entre pessoas [TORO+2000]. Entrevistas não são tão simples como inicialmente pode parecer e entrevistar não é somente fazer perguntas; é uma técnica estruturada que pode ser aprendida e na qual os desenvolvedores podem conquistar a proficiência com treinamento e prática [CARVALHO+2001]. Entrevistas são usadas para obter conhecimentos sobre um domínio através de perguntas feitas aos usuários especialistas deste domínio por um ou mais entrevistadores. Entrevistas permitem também que os desenvolvedores entendam os processos atuais da organização, percebam o que está faltando no sistema existente e as expectativas dos usuários do novo sistema [KUCHMISTAYA2001].

Entrevistas podem ser divididas em três tipos: entrevistas não estruturadas, entrevistas semi-estruturadas e entrevistas estruturadas.

Entrevistas não estruturadas

As 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. Sob estas circunstâncias, não é possível ter um conjunto pré-determinado de perguntas.

A aparente simplicidade de uma entrevista não estruturada está no fato de que

entrevistar parece ser uma habilidade que a maioria das pessoas sentem que possuem devido à experiência da conversação social, ou seja, a atividade básica do ser humano de falar e se comunicar. É caracterizada por uma pauta ou agenda irrestrita e é uma técnica que é conduzida em praticamente todas as áreas de atuação do homem [MAGUIRE1998].

Entrevistas semi-estruturadas

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. Ainda há muita flexibilidade e a ordem e o mecanismo das perguntas não são determinados antecipadamente. Além disso, dentro da lista de tópicos ou de assuntos, o entrevistador está livre para explorar determinadas perguntas com maior profundidade.

Entrevistas estruturadas

As entrevistas estruturadas consistem em um conjunto de perguntas fechadas, antecipada e cuidadosamente elaboradas. O entrevistador faz as mesmas perguntas a cada entrevistado mantendo, essencialmente, as mesmas palavras e na mesma seqüência. Este tipo de entrevista pode ser particularmente apropriado quando há diversos entrevistadores e se deseja minimizar a variação nas perguntas apresentadas. É também 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.

6.1.3 Áreas típicas de aplicação da técnica e quando pode ser usada

Úteis para obter informações em profundidade sobre uma atividade específica ou um conjunto de tarefas. Na fase inicial da elicitação o desenvolvedor pode entrevistar os usuários para obter informações sobre o trabalho que eles executam. Entrevistar é ainda o método mais amplamente utilizado para descobrir o que os usuários querem.

Em um contexto de elicitação, a entrevista semi-estruturada é comumente mais frutífera. Geralmente, as entrevistas iniciais serão menos estruturadas que as seguintes. Nos primeiros encontros as questões tendem a ser mais genéricas e tornam-se mais específicas nos encontros posteriores, conforme o desenvolvedor entende o domínio e fique claro que tipo de informações ele realmente necessita obter dos usuários.

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 [MAGUIRE1998].

6.1.4 Benefícios/Vantagens

São úteis para extrair informações detalhadas dos usuários e também para identificar possíveis áreas para uma análise mais detalhada. Os dados coletados fornecem informações em princípios e regras gerais e são mais rápidas do que as técnicas de observação. Entrevistas são populares, bem conhecidas, amplamente aceitas e também são úteis para descobrir os eventos que ocorrem esporadicamente [MAGUIRE1998].

As perguntas abertas nas entrevistas são mais eficazes, permitindo que o entrevistado fale sobre o que ele está fazendo e expresse sua compreensão e sentimento sobre o problema [KUCHMISTAYA2001].

6.1.5 Pontos fortes

O ponto forte das entrevistas não estruturadas é que a entrevista é flexível e pode ajudar os desenvolvedores a encontrar respostas para as diferenças individuais, para as mudanças situacionais e para informação nova emergente.

A utilização do guia da entrevista nas entrevistas semi-estruturadas possibilita entrevistar diferentes pessoas de uma maneira mais sistemática e completa, através da limitação dos assuntos tratados na entrevista.

As entrevistas estruturadas com questões fechadas permitem que o entrevistador colete sistematicamente dados detalhados, facilitando a comparação das respostas de todos os respondentes.

6.1.6 Limitações/Desvantagens

Segundo [KUCHMISTAYA2001] entrevistar é um processo longo e, conseqüentemente, caro. As seguintes desvantagens são evidentes:

- É difícil ajustar uma hora conveniente para entrevistar todas as pessoas capazes de fornecer uma boa visão do problema.
- Geralmente muitos *follow-ups* são necessários para o esclarecimento, tornando o processo mais caro;
- Como nem todas as pessoas dos vários departamentos da organização são entrevistadas, devido ao custo, o risco de parcialidade ou desvios nas informações coletadas é alto; e

- Informação de qualidade pode ser coletada através de perguntas bem elaboradas e feitas corretamente. Entretanto, não há nenhum procedimento padrão para fazer entrevistas mais eficazes.

6.1.7 Pontos fracos

O ponto fraco das entrevistas não estruturadas é que elas podem gerar dados menos sistemáticos que são mais difíceis e custosos para classificar e analisar.

Um dos pontos fracos das entrevistas semi-estruturadas é que ela não permite ao entrevistador explorar tópicos ou assuntos de interesse que surgem no processo, porque não foram contemplados quando o guia da entrevista foi elaborado. A flexibilidade do entrevistador na formulação e no seqüenciamento das perguntas pode resultar em respostas substancialmente diferentes de pessoas diferentes reduzindo, assim, as maneiras de comparação das respostas.

Já a entrevista estruturada não permite ao entrevistador explorar os tópicos ou itens que não foram contemplados quando o instrumento da entrevista foi elaborado. As entrevistas estruturadas limitam o uso de linhas alternativas de questionamentos com pessoas diferentes relatando suas experiências particulares. Isto reduz a extensão em que as diferenças e as circunstâncias individuais podem ser inteiramente incorporadas na avaliação.

6.1.8 O que é necessário

Requerem uma considerável preparação por parte do entrevistador, que deve formular perguntas introdutórias para conduzir o entrevistado ao foco do que está sendo discutido. É indispensável que o entrevistador se prepare para a interação, elaborando pelo menos as especificações do núcleo do problema que necessitam ser resolvidas.

Os desenvolvedores precisam ter habilidades sociais, de ouvir e de registrar informações da forma mais organizada possível. O local da entrevista deve ser confortável e agradável. Se a entrevista for realizada numa sala de reunião da própria organização, deve ser solicitado aos entrevistados que evitem interrupções como, por exemplo, chamadas telefônicas ou de algum companheiro de departamento. Os telefones celulares devem ser desligados. Nas entrevistas, em geral, há também um espaço para uma considerável parcialidade na forma como as questões são colocadas e em como as respostas são interpretadas. Pode ser necessário que o entrevistador adquira conhecimento sobre o domínio para saber que perguntas fazer. O que as pessoas dizem freqüentemente difere do que realmente elas fazem [MAGUIRE1998].

6.1.9 Processo

Há tipicamente quatro fases na entrevista [RAGHAVAN+1994]; [CARVALHO+2001]; [TORO+2000]: identificação das pessoas que serão entrevistadas, preparação, realização e análise das entrevistas.

Identificando as pessoas que serão entrevistadas. A elicitação de requisitos através da entrevista começa com a identificação das pessoas a ser entrevistadas. Deve-se diminuir o número de entrevistas e, por isso, é fundamental selecionar as pessoas corretas. Normalmente se começa pelos diretores, executivos ou patrocinadores que podem oferecer uma visão global e, em seguida, os usuários podem ser entrevistados, fornecendo informações mais detalhadas sobre a organização e o sistema novo ou atual. Em cada entrevista é possível descobrir se é necessário entrevistar mais pessoas, tanto possíveis novos usuários quanto pessoas que não são usuários diretos do sistema, mas que de alguma forma vão interagir com o novo produto. Também é útil estudar os perfis dos entrevistados, buscando pontos comuns ou divergentes com o entrevistador, que podem ajudar a quebrar algumas barreiras no início da entrevista.

Preparando a entrevista. Existem algumas atividades básicas no preparo de uma entrevista: estudar o domínio do problema, determinar objetivos e lista das questões e planejar e agendar os encontros. Conhecer o domínio, conceitos e termos da comunidade de usuários é fundamental para entender as necessidades desta comunidade, sua forma de expressá-las e para dar aos usuários a certeza de que o desenvolvedor entende seus problemas. Para obter este primeiro conhecimento sobre o domínio pode-se ler a documentação existente do sistema atual ou qualquer material sobre o novo sistema, tais como a bibliografia e a documentação de projetos semelhantes ou conversar com outros desenvolvedores. Tudo isto vai guiar o desenvolvedor a preparar com antecedência uma lista de questões.

Não é possível preparar todas as questões antecipadamente; as informações obtidas durante a entrevista serão a base para a preparação de novas perguntas, que serão formuladas à medida que a entrevista for avançando. Para encurtar o tempo da entrevista deve-se deixar claro os objetivos que se pretende alcançar e a duração da entrevista. Antes da realização pode-se enviar aos entrevistados um documento introdutório sobre o tema e uma lista de perguntas que eles devem ler e devolver. Isto é importante para que o entrevistado conheça os temas que serão tratados e tenha tempo de se preparar.

A data, hora, local e duração das entrevistas devem ser fixados tendo em conta sempre a agenda do entrevistado. Um ou dois dias antes os usuários devem ser relembrados da entrevista. Se se pretende gravar a entrevista, os usuários devem ser consultados com antecedência para obter autorização, evitando constrangimentos. O local da reunião também deve ser checado novamente.

Realização/condução das entrevistas. A realização da entrevista propriamente dita, pode ser dividida em 3 etapas [MAGUIRE1998]; [TORO+2000]: a abertura, o desenvolvimento e o fechamento da entrevista.

- **A fase da abertura**. É o aquecimento inicial da entrevista. As pessoas se apresentam e há as devidas trocas de cumprimentos. Em seguida o entrevistador informa sobre o motivo da entrevista,, o que se espera conseguir, a forma como as perguntas serão colocadas, como a informação será utilizada, etc. Se for utilizar algum tipo de notação gráfica ou matemática, o entrevistador deve se certificar que todos o conhecem e explicar, caso necessário, antes de utilizá-la. É muito importante causar boa impressão e passar segurança para os entrevistados.

- **O desenvolvimento da entrevista**. Esta é a fase principal da atividade, quando o entrevistador estará continuamente explorando, idealmente fazendo perguntas abertas sobre os tópicos ou assuntos, para que todos compreendam o alcance das respostas dos usuários. Perguntas abertas excluem repostas do tipo "sim" ou "não", permitindo maior comunicação e evitando a sensação de interrogatório. É importante neste estágio que o entrevistador permaneça analítico e neutro. Sempre que possível, é importante fazer um resumo do que já foi discutido, facilitando a compreensão. Questões de caráter geral encorajam respostas não reprimidas e podem extrair uma grande quantidade de informações. Elas podem ser muito úteis quando não se conhece o suficiente sobre o produto para fazer perguntas mais detalhadas. Já as questões específicas são úteis quando é preciso informar o usuário sobre um aspecto particular e forçar uma resposta detalhada ou precisa.

- **A fase de fechamento.** A entrevista pode terminar quando as questões tiverem sido feitas e respondidas, quando o tempo alocado tiver se esgotado, ou quando o entrevistador sentir que o entrevistado está exausto. Nesta fase, também conhecida como fase de relaxamento, pode-se fazer uma recapitulação para

confirmar que não há confusão nas informações recolhidas. As ações subseqüentes são anotadas e o planejamento futuro é feito. Deve-se agradecer a colaboração do entrevistado, deixando sempre aberta a possibilidade de voltar a contatá-lo para esclarecer eventuais dúvidas que surjam ao analisar a entrevista ou ao entrevistar outros usuários.

Análise das entrevistas. Uma vez realizada a entrevista, é necessário ler as notas tomadas, passá-las a limpo, reorganizar a informação, confrontá-las com outras entrevistas ou fontes de informação e produzir um resumo escrito. Neste momento podem-se descobrir ambigüidades, informações conflitantes ou ausentes. Assim que o resumo estiver pronto, pode-se enviá-lo aos entrevistados para que eles confirmem o conteúdo. Também é importante avaliar os procedimentos utilizados para preparar e conduzir a entrevista, com o objetivo de melhorar entrevistas futuras [CARVALHO+2001].

6.1.10 Procedimentos Práticos

A seguir são sugeridas algumas orientações práticas para a boa aplicação da técnica [RAGHAVAN+1994]; [CARVALHO+2001]; [TORO+2000]:

- Se a entrevista for conduzida de maneira estruturada, as perguntas devem ser elaboradas e testadas como se fossem para uma pesquisa ou questionário;
- Antes das entrevistas, elaborar uma lista de assuntos que serão apresentados para cada usuário e identificar estratégias e alguns exemplos, caso os usuários encontrem dificuldades para responder alguns tópicos;
- Os objetivos e os limites do contexto da entrevista devem estar definidos para que não haja desvios;
- Estabelecer um tempo para a entrevista;
- A entrevista deve ser conduzida de uma maneira amigável, porém profissional;
- O entrevistador dever ser consistente na forma como apresenta as perguntas;
- O entrevistador deve dar tempo suficiente para o entrevistado elaborar sua resposta antes de avançar para a questão seguinte;
- Deve ouvir o entrevistado;
- Deve evitar conduzir o entrevistado;
- Deve estar preparado para mudar as perguntas prontas (padrão) e investigar mais se uma nova linha interessante da discussão surgir, desde que esteja coerente com os objetivos iniciais da entrevista e com o tipo de entrevista utilizado;

- Se mais de um entrevistado estiver presente, é interessante ter um número maior de entrevistadores que nunca deve exceder o de entrevistados em mais de um;
- Se houver mais de um entrevistador, é interessante que eles tenham perfis variados para que o resultado da entrevista seja mais produtivo. Por exemplo, um poderia registrar todas as respostas enquanto outro poderia conduzir a entrevista, cuidando para que ela não saia da programação previamente estabelecida;
- O entrevistador deve ilustrar a relevância da entrevista destacando algum ponto significativo que ele tenha aprendido;
- Depois das entrevistas, os entrevistadores devem reunir suas anotações e apresentar um resumo das respostas do usuário para cada tópico; e
- Finalmente, o entrevistador deve agradecer aos entrevistados pelo tempo dedicado.

6.2 Brainstorming

6.2.1 A classificação na taxonomia

/ Consultivo / Informal / Levantamento não estruturado / Sócio-organizacional / Grupo / Null / Médio / Motivador do grupo / Médio / Ambos / Largura / Baixa /

6.2.2 Descrição da técnica

O *brainstorming* é uma das diversas técnicas de reuniões de grupo, provavelmente a mais antiga e mais conhecida. Ela é uma técnica básica para geração de idéias. O princípio básico é reunir um conjunto de especialistas (sistemas e negócios) para que cada um possa inspirar ao outro a criação de idéias que contribuam para resolver o problema em uma ou várias reuniões [KIRAKOWSKI1997]. As idéias sugeridas e exploradas nestes encontros não devem ser criticadas ou julgadas.

6.2.3 Áreas típicas de aplicação da técnica e quando pode ser usada

Pode ser aplicada no início da fase do desenvolvimento quando pouco do projeto é conhecido e são necessárias idéias novas [KIRAKOWSKI1997].

Brainstorming é usada para gerar novas idéias deixando a mente livre para aceitar toda a idéia que for sugerida e, assim, permitir a liberdade para a criatividade. O resultado de uma sessão de *brainstorming* bem sucedida é um conjunto de boas idéias e a sensação de

que todos participaram da solução do problema. *Brainstorming* é uma técnica particularmente efetiva para ser aplicada à concepção de um sistema ou na exploração e entendimento do potencial de mercado para este produto [MAGUIRE1998]. É adequado para elicitar informações de alto nível dos domínios e questionar supostas limitações que outras técnicas abordaram [MAIDEN+1996]

6.2.4 Benefícios/Vantagens

- No processo de *brainstorming*, todos no grupo podem levar crédito pelas boas idéias; não se gasta muito tempo para obter dados úteis e a sessão não necessita durar mais de uma hora [MAGUIRE1998];
- O processo de grupo é geralmente recompensador e cria um sentimento de posse ou propriedade do resultado;
- O seu ponto forte é a geração de idéias; e
- Freqüentemente, boas idéias surgem da combinação de idéias aparentemente ruins ou sem relação alguma, sugeridas por pessoas que foram encorajadas a propor idéias, incomuns muitas vezes.

6.2.5 Limitações/Desvantagens

- Os participantes do grupo têm que estar bastante compenetrados e sentir-se à vontade para que a sessão produza bons resultados;
- Sessões de *brainstorming* são relativamente caras e devem ser usadas moderadamente, possivelmente somente uma sessão durante todo o processo deva ser realizada [BRAY2002]; e
- Cuidado para que as idéias geradas não sejam superficiais.

6.2.6 O que é necessário

Os recursos humanos são os mais importantes para o êxito desta técnica, portanto é necessário um grupo adequado de pessoas. As pessoas mais criativas e com alguma experiência nesta área podem obter melhor resultado. Cerca de 5 a 12 pessoas podem participar, ou um número suficiente de pessoas capaz de criar idéias e instigar reações. É preciso somente uma sala, com um quadro para anotar as idéias.

É necessário que todos os participantes possam ter acesso às idéias anteriormente formadas na sessão. Alternativamente ao quadro pode ser usado um *flip chart* para anotar as idéias na forma como elas surgem e as folhas podem ser coladas na parede, por exemplo.

Algum critério de classificação pode ser aplicado para agrupar as idéias semelhantes, mas este não deve ser o objetivo primário [BRAY2002]. A sessão deve ser conduzida por um facilitador bastante experiente.

6.2.7 Processo

- Decidir os objetivos da sessão de *brainstorming* e os participantes;
- Ao contatar os participantes, deve ser explicado claramente quais tópicos deverão ser considerados e o formato da reunião. Deve ser obtida também a concordância prévia para se utilizar alguma técnica de gravação, por exemplo, em vídeo ou áudio;
- Produzir cronograma para a sessão e fazer uma sessão piloto para se certificar de que o cronograma é realista. Se for necessária mais alguma informação individual para validar uma outra, preparar um questionário apropriado para ser aplicado a todos antes ou depois da sessão; e
- Durante a sessão, o líder da discussão deve ser ativo em conduzir a discussão e resumir os resultados no fim de cada tópico. É importante distinguir entre o que é consenso do grupo e o que é a opinião de diferentes participantes;

6.2.8 Procedimentos Práticos

- Não permitir críticas às idéias surgidas durante a sessão;
- Encorajar a participação para criar uma grande quantidade de idéias;
- Não desprezar as idéias que aparentemente são irrelevantes;
- Manter as idéias curtas e precisas;
- Manter uma lista das idéias;
- Evitar associar nome da pessoa à idéia; e
- Tentar combinar e melhorar as idéias.

6.3 JAD

6.3.1 A classificação na taxonomia

/ Representativo / Informal / Levantamento estruturado / Mista / Grupo / Validação / Alto / Apresentar e capturar idéias / Alto / Ambos / Profundidade / Alta /

6.3.2 Descrição da técnica

A técnica JAD desenvolvida na IBM no fim dos anos 70 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 [PORTELLA1994], prevalecendo a cooperação e o entendimento. Os desenvolvedores ajudam os usuários a formular os problemas e explorar possíveis soluções, envolvendo-os e fazendo com que eles se sintam participantes do desenvolvimento [RAGHAVAN+1994].

Esta técnica se baseia em quatro princípios básicos [RAGHAVAN+1994]:

- 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; e
- utilização de documentação-padrão, que é preenchida e assinada por todos os participantes de uma sessão.

A técnica JAD tem duas grandes etapas: planejamento, cujo objetivo é elicitar e especificar requisitos; e projeto, em que se lida com o projeto do software. Nesta monografia somente será tratada a primeira etapa.

Ainda segundo [RAGHAVAN+1994] os participantes de uma sessão de JAD desempenham seis diferentes papéis: líder da sessão, representantes do usuário, especialista, analista, representantes dos sistemas de informação e patrocinador executivo.

Líder da sessão é o responsável por todo o processo e assume o controle durante as reuniões, por isso deve ter bom relacionamento pessoal e capacidade de liderança. Deve estar familiarizado e treinado em todos os aspectos do JAD e ter experiência na aplicação para ser capaz de planejar e entender as várias tarefas da técnica JAD [CARVALHO+2001]. Algumas habilidades importantes que deve ter são: entender e promover a dinâmica de grupo, iniciar e centralizar discussões, reconhecer quando a

reunião está se desviando do tema e reconduzi-la, gerenciar os diferentes tipos de personalidades e maneiras de ser dos participantes, evitar o desinteresse pela reunião ainda que ela seja demorada e difícil [TORO+2000], tudo isto realizado através da prática e da experiência na aplicação da técnica.

Desenvolvedor ou engenheiro de requisitos é o responsável pela produção dos documentos que devem ser gerados durante as sessões JAD. Deve ter a habilidade de organizar bem as idéias e expressá-las claramente por escrito. Quando se utilizam ferramentas de software durante as sessões, deve ser capaz de manejá-las eficientemente [TORO+2000]. Deve ser um desenvolvedor experiente para poder entender questões técnicas e detalhes que são discutidos durante as sessões [CARVALHO+2001].

Executor é o responsável pelo produto a ser construído e deve proporcionar aos demais participantes informações sobre a necessidade do novo sistema e os benefícios que se espera obter dele. Também é responsável pelas decisões executivas como, por exemplo, alocação de recursos e o projeto do novo produto.

Representantes dos usuários são pessoas que atuam na organização e durante a etapa de planejamento do JAD, podem ser gerentes e pessoas-chave, com uma visão global do sistema. Futuros usuários finais do sistema podem ser incorporados durante a etapa de projeto do JAD. Os representantes dos usuários devem ser selecionados de acordo com o conhecimento de suas próprias necessidades dentro da empresa, o entendimento da interação do seu departamento com outros departamentos e algum conhecimento de produtos de software [CARVALHO+2001].

Representantes dos sistemas de informação são pessoas especialistas em sistemas de informação que devem ajudar os usuários a compreender o que é ou não factível com a tecnologia atual e o que é possível o novo sistema fazer. Estes representantes podem, por exemplo, ajudar os usuários a entender as conseqüências da escolha de um ou outro caminho para a resolução de um problema [CARVALHO+2001].

Especialistas são pessoas que podem dar informação detalhada sobre aspectos específicos, tanto do ponto de vista dos usuários, porque conhecem muito bem o funcionamento de uma parte da organização, como do ponto de vista dos desenvolvedores, porque conhecem perfeitamente certos aspectos técnicos da instalação de hardware da organização.

Um especialista da comunidade de usuários pode ser a pessoa que usa um determinado tipo de relatório. Um especialista da comunidade de desenvolvedores pode ser o administrador da rede, cuja participação seria solicitada, por exemplo, durante a definição dos aspectos da rede que o novo sistema requer [CARVALHO+2001].

6.3.3 Áreas típicas de aplicação da técnica e quando pode ser usada

Pode ser usada tanto para elicitar como nas fases iniciais da especificação de requisitos. Ajuda a identificar os assuntos que podem necessitar de rastreamento e fornece uma perspectiva multifacetada dos requisitos.

Sessões JAD permitem aos analistas coletar simultânea e eficientemente uma grande quantidade de requisitos do sistema junto a uma gama de usuários-chave. São úteis por considerar necessidades específicas dos usuários ou opções de projetos.

JAD também pode ser usada em conjunto com outra técnica de elicitação como, por exemplo, a prototipação. À medida que os requisitos sejam obtidos nas sessões, pode-se construir um protótipo que demonstre alguma funcionalidade destes requisitos.

6.3.4 Benefícios/Vantagens

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 [KUCHMISTAYA2001], existem alguns motivos convincentes para incorporar a técnica JAD ao processo da identificação dos requisitos:

- JAD diminui o tempo associado com o processo de elicitação dos requisitos, porém com um custo alto, devido ao número de pessoas envolvidas. Durante o período de 2 a 4 semanas as informações não são somente coletadas, mas os requisitos, acordados por vários usuários do sistema, são identificados. Com uma equipe experiente em JAD as organizações podem tornar seu processo de análise de sistemas mais dinâmico;
- As sessões JAD ajudam a reunir especialistas dando a eles possibilidade de compartilhar suas visões, compreender outras e desenvolver o sentimento de posse do projeto;
- As formas de aplicação de JAD são bem conhecidas porque foi a primeira técnica de projeto acelerada disponível no mercado e, provavelmente, a mais conhecida e pode ser aplicada facilmente por qualquer organização;

- A fácil integração de ferramentas CASE ao processo JAD melhora a produtividade das sessões e fornece aos analistas de sistemas um modelo já discutido e pronto para ser usado nas fases seguintes;
- A metodologia JAD tem o mérito de criar uma abordagem em grupo, para o desenvolvimento do projeto, em que o papel ativo do usuário é altamente valorizado;
- Permite aos analistas obter rapidamente uma variedade de visões de uma quantidade de pessoas com perspectivas diferentes e amplas [MAGUIRE1998]; e
- Muitos conflitos podem ser resolvidos nas sessões.

6.3.5 Limitações/Desvantagens

Os desenvolvedores sempre têm que ter em mente que nenhuma das técnicas são completas para coletar toda informação, inclusive JAD. Esta pode ser ineficaz se não usada corretamente e alguns assuntos pontuais precisam ser resolvidos antes e durante as sessões JAD [KUCHMISTAYA2001]:

- "Fazer a lição de casa". Sem uma preparação multifacetada para a sessão de JAD, o tempo valioso dos profissionais pode facilmente ser desperdiçado, problemas errados podem ser colocados, pessoas erradas podem ser convidadas para participar, recursos inadequados para a resolução dos problemas podem ser usados; todos estes cenários podem se formar caso os organizadores da sessão de JAD não estudem os elementos do sistema que estão sendo analisados.
- A equipe escolhida para participar da sessão de JAD deve incluir os empregados capazes de fornecer o máximo possível de informação sobre o problema ou, pelo menos, das partes fundamentais do problema. É por isso que a seleção dos participantes é importante e deve ser feita com cuidado. O grupo deve ser constituído não somente pelos empregados dos vários departamentos que terão interação com o novo sistema, mas também por outros de diferentes níveis da organização. Esta variedade de pensamentos sobre a compreensão do processo refletirá algumas vezes pontos de vista conflitantes, mas permitirá também aos participantes ver "o outro lado da moeda". Com uma melhor compreensão dos processos JAD, será possível fazer um melhor esboço do modelo.
- O facilitador atuando como força motivadora tem que se certificar de que todos os participantes, não somente os mais comunicativos, têm a possibilidade de expor suas opiniões, idéias e pensamentos. Todos os especialistas do negócio na

equipe do JAD devem ser incentivados a oferecer sua contribuição, tornando as discussões mais frutíferas.

Para [PORTELLA1994], JAD também apresenta a falha mais comum das metodologias de desenvolvimento - crer na possibilidade de se obter dos usuários requisitos essenciais, completos e não ambíguos, através de modelos não-executáveis. Outro problema da metodologia apontado por [PORTELLA1994] é o fato de que, teoricamente, os componentes da etapa de planejamento não são os mesmos da etapa do projeto, o que causa perda de informação obtida sobre o projeto, pois nenhum documento escrito conseguirá transcrever fielmente as percepções obtidas pelos presentes nas reuniões.

Algumas pessoas podem não ter chance de falar sobre suas visões do sistema ou podem se sentir inibidas por outros membros do grupo, particularmente colegas ou alguma pessoa de hierarquia superior. Outras pessoas podem também não pensar criativamente quando estão em grupo, ou preferem ser entrevistadas individualmente, ou ainda responder algum questionário [MAGUIRE1998].

6.3.6 Pontos fortes e fracos

JAD tem dois pontos fortes: (a) para toda a pergunta que surgir é provável que alguém esteja apto a respondê-la; e (b) é provável que divergências e pontos conflitantes sobre o sistema ou processo venham à tona. É neste momento que o líder da sessão deve atuar, conduzindo a discussão para uma solução [WESSELS2002].

Um ponto fraco do JAD está relacionado 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 alguns 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 de maneira *on-line*, usando um *login* anônimo para cada participante [WESSELS2002].

6.3.7 O que é necessário

As sessões JAD geralmente podem ser conduzidas no ambiente de trabalho, mas preferencialmente elas devem ocorrer em local diferente e podem durar diversas horas, ou mesmo, uma semana inteira, envolvendo de dez a vinte pessoas em uma dinâmica de grupo com apoio de ferramentas visuais. A equipe que participa das sessões é a principal parte do processo de JAD e a seleção dos indivíduos é crítica para o sucesso da sessão.

6.3.8 Processo

A técnica JAD é executada em três fases [RAGHAVAN1994], [TORO+2000], [CARVALHO+2001]:

1. **A fase de adaptação**: é de responsabilidade do líder da sessão de JAD, ajudado por um ou dois desenvolvedores, adaptar a técnica JAD para cada produto de software a ser desenvolvido, tornando-a mais efetiva. Compõe-se dos seguintes passos:
   - A adaptação deve começar pela definição em alto nível do projeto, para a qual podem ser necessárias entrevistas prévias com alguns clientes e usuários. Também pode ser necessário obter informação sobre a organização para se familiarizar com o domínio do problema. Por exemplo, utilizando técnicas complementares como a análise da documentação ou observação;
   - Obtida uma primeira idéia dos objetivos do projeto, é necessário selecionar os participantes, avisá-los para participar das reuniões e fornecer uma lista prévia com os temas que serão tratados nas sessões para que eles possam se preparar;
   - O líder da sessão de JAD deve decidir sobre a duração e o número de sessões que serão feitas e ajustar o formato geral dos documentos JAD às necessidades do produto de software a ser construído; e
   - Deve também preparar transparências introdutórias e todo o material audiovisual que considere oportuno.

2. **A fase de sessão**: durante as sessões, os participantes expõem e discutem suas idéias que são analisadas e refinadas até se chegar a um acordo. Os passos recomendados para este processo são:
   - Apresentação: todos os participantes são apresentados e o líder da sessão, juntamente com o patrocinador-executivo, dão boas-vindas a todos participantes. O patrocinador-executivo expõe brevemente as necessidades que levaram ao desenvolvimento e os benefícios esperados. O líder da sessão explica a mecânica das sessões e o planejamento previsto;
   - Definir objetivos e requisitos: o líder da sessão JAD promove a discussão para elicitar os objetivos ou requisitos de alto nível mediante perguntas como: "Por que o sistema será construído?", "Que benefícios são esperados?", "Como a organização pode se beneficiar no futuro?", "Que restrições de recursos disponíveis, normas ou leis afetam o projeto?", "A segurança dos dados é importante?" À medida que se vai elicitando requisitos, o analista os escreve em

transparências ou em algum outro meio que os mantenha visíveis durante a discussão;

- Delimitar o escopo do sistema: uma vez obtidos os requisitos, é necessário organizá-los e chegar a um acordo sobre o âmbito do novo sistema. No caso dos SI, é útil identificar os usuários potenciais e determinar quais tarefas eles realizam;
- Documentar temas abertos: as questões surgidas durante a sessão e não resolvidas devem ser documentadas para ser novamente discutidas nas sessões seguintes, devendo ficar designada uma pessoa como responsável pela sua solução dentro de um prazo pré-estabelecido; e
- Concluir a sessão: o líder da sessão JAD conclui a sessão revisando com os demais participantes a informação elicitada e as decisões tomadas. É dada oportunidade a todos participantes de expressar qualquer consideração adicional, fomentando por parte do líder da sessão JAD o sentimento de propriedade e compromisso de todos os participantes sobre os requisitos elicitados.

3. **A fase de finalização:** terminadas as sessões, é necessário transformar as transparências, notas e demais documentos gerados em documentos formais. Três passos devem ser seguidos:
   - Completar a documentação: os analistas compilam a documentação gerada durante as sessões em documentos conforme as normas ou padrões da organização;
   - Revisar a documentação: a documentação gerada é enviada a todos os participantes para que a comentem. Se os comentários forem suficientemente importantes, outra reunião é convocada para discuti-los; e
   - Validar a documentação: revisados todos os comentários, o líder da sessão JAD envia o documento ao patrocinador executivo para sua aprovação. Aprovado o documento, cópias definitivas são enviadas a cada um dos participantes.

A sessão termina quando se acredita que toda a informação relevante foi obtida. Obviamente, quanto mais longas as sessões, mais custosas elas serão.

6.3.9 Procedimentos Práticos

Um dos elementos chave do processo JAD, e muitas vezes negligenciado, é a preparação das sessões. De uma a três semanas antes do início das sessões a equipe de executivos e a equipe de desenvolvedores deve identificar os seguintes pontos, segundo [KUCHMISTAYA2001]:

- Projeto e seus objetivos: qual a essência do problema que a equipe de desenvolvimento e os participantes das sessões estão enfrentando;
- Objetivo e resultado esperado das sessões: quais os pontos que devem ser elucidados nas sessões e quais os resultados que a equipe de desenvolvimento espera obter das sessões JAD;
- Quais participantes serão mais benéficos ao projeto: pessoas de que área funcional e de que hierarquias profissionais deveriam estar presentes para assegurar a versatilidade da equipe de JAD; e
- Tentar agendar as sessões com antecedência escolhendo local, hora e ferramental a ser utilizado. Sem uma boa preparação e sem um plano claro de ações as sessões JAD podem tornar-se desorganizadas e desperdiçar recursos.

O líder da sessão JAD deve:

- Criar uma boa atmosfera;
- Fornecer aos participantes um formulário simples para completar com detalhes pessoais antes que a reunião comece. Isto pode ajudar a indicar uma atividade quando for necessário um último ajuste no ambiente, ou quando de eventual espera pelos participantes atrasados;
- Ajudar os participantes na formulação do problema e guiá-los quando necessário;
- Impedir comportamentos destrutivos ou negativos por parte de algum participante;
- Procurar evitar polêmicas entre os indivíduos com idéias e opiniões diferentes;
- Não sugerir soluções para o problema e evitar avaliar as soluções propostas;
- Assegurar-se de que todos os participantes estão tendo oportunidade de contribuir;
- Se possível, explicar os conceitos a ser explorados usando slides, quadros brancos, transparências ou outros meios para ressaltar aspectos do sistema ou do produto a ser construído; e
- Oferecer diversas alternativas para enfatizar pontos para os quais há mais de uma solução possível e para estimular a discussão sobre temas comuns e problemas.

6.4 Questionários

6.4.1 A classificação na taxonomia

/ Consultivo / Informal / Levantamento estruturado / Tecnológica / Indivíduo / Null / Alto / Saber montar projetos e analisar pesquisas / Médio / Ambos / Largura / Baixa /

6.4.2 Descrição da técnica

Um questionário abrange a aplicação de várias perguntas escritas para uma ampla amostragem de usuários. Questionários podem ajudar a determinar informações ou clientes, práticas de trabalho e atitudes. Existem dois tipos: **fechados**, quando os pesquisados são solicitados a selecionar respostas a partir de uma lista de opções do tipo múltipla-escolha e **abertos**, quando os pesquisados ficam à vontade para responder as perguntas [KIRAKOWSKI1997].

Se bem elaborado, os questionários podem oferecer uma visão bastante definida da percepção por parte dos usuários do universo atual de informações e sistemas.

6.4.3 Áreas típicas de aplicação da técnica e quando pode ser usada

Úteis para obtenção de dados quantitativos dos usuários sobre as tarefas existentes no sistema atual [MAGUIRE1998]. Os questionários são uma alternativa para conduzir uma entrevista, principalmente quando é grande o número de usuários envolvidos [WESSELS2002].

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 ambigüidade.

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.

6.4.4 Benefícios/Vantagens

Geralmente sua aplicação é rápida e relativamente barata, mas a sua elaboração não é. Quando se está aplicando um questionário fechado, há a possibilidade de análises estatísticas das respostas.

Quando há tempo suficiente para preparar as questões e analisar as respostas, os questionários são uma maneira eficaz de eliciar informações, particularmente de uma quantidade grande de usuários, porém a um alto custo.

Questionários podem ser uma boa maneira de confirmar informações levantadas de um grande número de pessoas por outras técnicas.

6.4.5 Limitações/Desvantagens

Questionários não são flexíveis e a elaboração do questionário não é simples; muitos podem pensar que podem fazê-lo, mas poucos o fazem bem e corretamente, pois são necessárias formação específica e experiência Pode ser difícil analisar comentários interessantes, já que freqüentemente não é desejável ou possível manter registros dos pesquisados [MAGUIRE1998]. Além disso, há uma limitação do universo de respostas.

Na comparação com entrevistas, os questionários são menos ricos em informações e mais baratos. Os questionários não fornecem uma oportunidade de julgar a exatidão das respostas, além de permitir pouca ou nenhuma interação. Também não é possível detectar respostas tendenciosas.

Nem todos que recebem o questionário irão preenchê-lo e devolvê-lo, correio eletrônico e formulários são mais fáceis de ignorar do que entrevistas.

Pode-se gastar um tempo considerável para elaborar bons questionários. Os desenvolvedores precisam ser também bons planejadores.

6.4.6 O que é necessário

Depende muito da complexidade da pesquisa e do número de pessoas a serem pesquisadas. Elaborar perguntas boas é mais importante aqui do que nas entrevistas, porque não há oportunidade de se explorar as respostas que são dadas [WESSELS2002]. Isso também é crítico para se ter uma boa compreensão de como será usada a informação antes de enviar o questionário pois, caso contrário, poderão ser obtidos resultados inúteis.

6.4.7 Processo

Os questionários incluem, geralmente, perguntas fechadas, embora perguntas abertas também possam ser usadas. Em termos práticos, é comum agrupar perguntas por área de assunto e iniciar com questões importantes ou interessantes para motivar as pessoas que estão completando o questionário.

Os passos iniciais são os mesmos usados para se preparar uma entrevista, tendo em mente que entrevistas semi-estruturadas são similares aos questionários abertos (por exemplo, os problemas são conhecidos, mas o alcance das respostas do usuário não são); e entrevistas estruturadas são similares aos questionários fechados [MAGUIRE1998].

Como nas entrevistas, pessoas necessitam ser escolhidas para responder ao questionário. Geralmente, escolhe-se uma amostragem representativa do grupo de usuários, isto é, escolhendo uma amostra grande o bastante para dar alguma confiabilidade aos resultados [WESSELS2002].

Os questionários devem empregar um método estatístico rigoroso nestas amostragens para assegurar que os resultados não sejam tendenciosos. Entretanto, essa recomendação é raramente observada. Valendo-se deste método rigoroso de análise, é possível devolver os resultados tabulados aos pesquisados, oferecendo uma forma de recompensa pelo retorno da pesquisa preenchida. Ao empregar esta tática o índice de retorno dos questionários pode aumentar bastante. Um baixo índice de resposta pode ser corrigido com a re-postagem da pesquisa ou ainda com um contato telefônico. Entretanto, esse método requer que os usuários sejam identificados pelo nome e algumas pesquisas podem requerer total anonimato. É comum anexar uma carta explicativa ao enviar a pesquisa, solicitando ao respondente para responder e devolver no envelope já selado e endereçado para facilitar a devolução [MAGUIRE1998].

Antes de enviar um questionário oficialmente é importante fazer um teste com uma pequena amostragem da audiência escolhida para se obter um retorno dela sobre o questionário. Isso ajudará a eliminar perguntas confusas ou redundantes, e pode levar a outras perguntas que poderão ser incluídas na versão final [WESSELS2002].

Sob pena de aumentar o custo do trabalho de análise, as perguntas abertas também podem ser empregadas nos questionários, mas deve ser considerada a habilidade do público alvo em compreendê-las e respondê-las sem ambigüidades.

Assim como outras técnicas, os questionários podem ser vistos como um complemento das entrevistas e ser usados somente sob certas circunstâncias.

6.4.8 Procedimentos Práticos

Alguns critérios devem ser seguidos para obter boas informações num questionário e [MAGUIRE1998] sugere alguns:

- Explicar o objetivo da pesquisa no início do formulário. Se for uma pesquisa via correio, incluir uma carta mais formal explicando o propósito. Se os pesquisados conhecerem a razão para responder a pesquisa, o resultado será melhor;
- Se a pesquisa for devolvida pelo correio, facilitar esta tarefa providenciando um envelope pré-endereçado e selado;
- Evitar muitas perguntas abertas , pois elas irão aumentar o esforço da análise;
- Assegurar que as perguntas de múltipla escolha tenham apenas uma resposta;
- Ter certeza de que a pergunta está clara e concisa;
- Evitar perguntas duplas em uma única pergunta;
- Evitar perguntas negativas e complexas;
- Evitar perguntas que façam os pesquisados se sentirem desconfortáveis ou ofendidos. Por exemplo, é melhor perguntar a idade dos pesquisados através de faixas etárias do que idade exata;
- Para usuários com deficiência visual providenciar uma pesquisa em Braille, ou permitir que o pesquisado providencie suas respostas em entrevistas pessoais ou através de telefone;
- Usuários com deficiência motora podem ter problemas para preencher a pesquisa. Largas caixinhas na frente da reposta poderão ajudá-los. Novamente, eles podem preferir ser entrevistados;
- Revisar e assegurar que as perguntas são simples e compreensíveis.

Selecionar pessoas para preencher o questionário é uma tarefa difícil ou, como relata [WESSELS2002], "uma obra de arte":

- Ter certeza de deixar claro aos respondentes porque o questionário é importante e como ele será usado;
- Fixar claramente um prazo de entrega para os questionários e enviar lembretes antes da data devida;
- No caso de formulários impressos, entregá-los pessoalmente e contatar as pessoas também pessoalmente sobre a devolução do questionário.

6.5 Observação

6.5.1 A classificação na taxonomia

/ Representativo / Informal / Observação / Mista / Observação / Validação / Baixo / Registrar eventos / Alto / Ambos / Largura / Baixa /

6.5.2 Descrição da técnica

As pessoas geralmente acham difícil descrever o que elas fazem, pois isto é muito natural para elas. Às vezes, a melhor forma de compreender o que elas realmente fazem é observá-las no seu ambiente de trabalho [KOTONYA+1998]. A técnica de observação consiste em observar o usuário final executando uma tarefa em particular no 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. Segundo [MAGUIRE1998] a observação pode ser direta, quando o observador está presente durante a execução da tarefa em contexto relevante do mundo real, ou indireta, quando a tarefa é observada através de outros meios como, por exemplo, o uso de gravação em vídeo.

Na observação direta, o observador pode simplesmente acompanhar a pessoa que está sendo observada, sem solicitar que a atividade seja interrompida. Caso deseje uma explicação mais detalhada do processo, o observador pode solicitar uma interrupção das atividades para uma melhor análise.

Observar um processo em ação permite ao analista ver como o sistema realmente funciona. Quando pessoas descrevem suas ações, normalmente omitem muitas tarefas e não julgam adequadamente o tempo que desperdiçam em outras atividades. Entretanto, uma das dificuldades na observação é que a presença do analista pode alterar o comportamento das pessoas. Se as pessoas associarem o analista aos "olhos" da gerência, logicamente elas irão realizar suas tarefas diárias diferentemente de quando o analista não as estiver observando [WESSELS2002].

Uma abordagem particular da observação é denominada de etnografia. Este termo foi derivado da sociologia e antropologia e se aplica quando um observador fica inserido por um longo período de tempo em uma sociedade ou cultura que está sob estudo. A aplicação da etnografia se justifica pelo fato de que somente tornando-se intimamente envolvido com as situações do ambiente de trabalho, o observador pode obter uma completa compreensão das várias práticas, problemas e preocupações [BRAY2002], identificando requisitos não levantados através de outras técnicas. Um exemplo clássico dessa idéia foi fornecido por

[KOTONYA+1998] referenciando o trabalho de [GOGUEN+1993]: "é muito mais fácil demonstrar o processo de fazer um laço no cordão do sapato do que descrevê-lo" Entretanto, este processo de observação torna-se caro e pode ser efetivo somente para sistemas complexos e críticos.

6.5.3 Áreas típicas de aplicação da técnica e quando pode ser usada

Mais proveitosa no início da especificação para obter dados qualitativos. Útil para estudos atuais executados em tarefas e processos [MAGUIRE1998].

A observação está mais preocupada na interação entre o usuário e o sistema atual, embora ela também possa ser um meio importante de avaliar o protótipo de um sistema novo [BRAY2002]. Ela pode ser usada quando não é possível a análise de documentos, isto é, para sistemas sem documentação. Ela pode ser aplicada tanto para tarefas manuais, automatizadas ou para as tarefas complexas de serem descritas.

6.5.4 Benefícios/Vantagens

Permite que o observador visualize o que os usuários realmente fazem. A observação direta permite concentrar a atenção em áreas específicas de interesse. A observação indireta capta a atividade que de alguma forma não foi registrada ou notificada [MAGUIRE1998]. Um ponto forte da observação é que ela é simples de ser empregada e não requer muito treinamento e preparação. Uma outra vantagem é que os usuários quase não precisam explicar o trabalho deles ou articular as idéias. O observador pode capturar aspectos sociais e organizacionais, sem que os usuários falem, além de revelar detalhes que outras técnicas não podem fornecer.

6.5.5 Limitações/Desvantagens

A observação pode importunar e alterar o comportamento do usuário devido à presença de um observador. Anotações e gravações em vídeo precisam ser analisadas por quem as fêz, o que pode demandar tempo ao impossibilitar que a análise seja feita por diversas pessoas. Um ponto fraco é que ela pode capturar uma grande quantidade de informações irrelevantes. Além disso, há dificuldades em capturar tarefas cognitivas.

Observações consomem uma grande quantidade de tempo.

6.5.6 O que é necessário

- Uma pré-condição é ter acesso ao local de trabalho das pessoas;
- Tempo para análise dos resultados da observação;
- A observação indireta requer equipamentos audiovisuais; e
- A cooperação dos usuários é fundamental, de forma que as habilidades interpessoais do observador são importantes [MAGUIRE1998].

6.5.7 Processo

Segundo [BRAY2002], observação é mais arte do que ciência, mas alguns processos podem ser definidos. Uma decomposição hierárquica de tarefas é amplamente empregada e uma análise desta decomposição pode ser útil para planejar as sessões de observação.

Decidir o que observar é muitas vezes mais difícil do que a própria observação. Num primeiro momento devem-se estabelecer os objetivos e quais requisitos necessitam ser capturados e se estas informações devem ser obtidas em largura ou em profundidade. Em seguida, estabelecem-se horários, locais e pessoas que serão observadas. Os usuários devem ser avisados que estarão sob observação. A cooperação dos usuários com o processo de observação deve se dar natural e pacientemente. Deve-se atentar para o fato de que a legislação pode impedir a filmagem se as pessoas não consentirem por escrito.

Em seguida, deve-se decidir qual a técnica de gravação/documentação a ser usada: se se pretende fazer anotações a mão (tradicional), gravações em áudio, vídeo ou áudio-visual. É bom lembrar que quanto mais completa estiver a gravação/documentação, mais tempo se levará para analisá-la. É muito útil fazer algum tipo de análise no início da primeira gravação da observação [MAGUIRE1998].

Finalmente deve-se analisar, resumir e relatar/reportar a documentação de acordo com os objetivos estabelecidos no início do estudo. Na prática, a observação é freqüentemente usada como complemento ou para esclarecer as informações da entrevista, para que o analista tenha em primeira mão uma visão dos tópicos discutidos nas entrevistas [WESSELS2002].

Segundo [BRAY2002], a observação e a análise subseqüente têm um custo alto em termos de tempo e, conseqüentemente, de dinheiro. Por ser fácil produzir informação, pode-se acabar gerando mais informação do que o necessário. A recomendação é usar a observação de forma cautelosa e concentrá-la nas tarefas críticas e interativas, fazendo uma abordagem de amostragem onde for possível.

Como nas entrevistas, as anotações são provavelmente a principal maneira de se registrar as informações, mas o uso de gravações em vídeo vem aumentando. O observador

deve ter capacidade para registrar informações e saber usar equipamentos de vídeo. Em certos casos, é possível combinar entrevistas com a observação. O usuário pode falar sobre o que ele faz e, até mesmo, porque faz, e depois ele é observado realizando a tarefa.

A preparação para uma sessão de observação é de fundamental importância para a obtenção do máximo possível de informação. Diferentes visões de observação resultam em diferentes conjuntos de informações obtidas. Quando uma sessão de observação é desempenhada por uma equipe de desenvolvedores, deve ser atribuído antecipadamente um ângulo de observação para cada membro da equipe.

6.5.8 Procedimentos Práticos

Quando as pessoas observadas se sentem constrangidas, em muitos casos, não desempenham as tarefas da maneira como normalmente fazem. Isto é difícil de resolver devido à maneira de ser de cada um, mas algumas estratégias podem ser adotadas [BRAY2002]:

- Fazer uma completa explanação dos motivos da observação;
- Fazer observações mais longas, de forma que elas vão se tornando atividades normais e o usuário vai se acostumando; e
- Fazer as observações de forma "disfarçada", o que requer bastante cautela.

Outros procedimentos sugeridos por [MAGUIRE1998] são:

- Assegurar que as pessoas que estão sendo observadas conheçam a razão do levantamento e que não vejam o desenvolvedor ou observador de forma negativa. Isso é particularmente importante para usuários que tenham deficiência mental ou visual, pois podem se sentir perturbados por uma presença passiva, não se sentindo seguros a respeito;
- Realizar uma sessão piloto de observação para sentir qual é a expectativa e testar quaisquer formulários de observações. Isso irá ajudar também a determinar o tempo necessário para efetuar a observação. Assegurar que haverá tempo suficiente para permitir que esse tipo de interação aconteça;
- Tentar ser o menos inoportuno possível;
- Anotar quaisquer eventos que não sejam entendidos e tentar esclarecê-los com o usuário tão logo finalizada a sessão;
- Se possível, fotografar a área de trabalho dos usuários ou a área de operações porque isso servirá como um lembrete do contexto ambiental; e
- Após as observações, anotar as primeiras impressões antes do início da fase de análise.

6.6 Análise de Documentos

6.6.1 A classificação na taxonomia

/ Consultivo / Informal / Mapeamento / Mista / Documentos / Especificação e documentação / Baixo / Possuir habilidades analíticas / Médio / Sistema atual / Largura / Baixa /

6.6.2 Descrição da técnica

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 políticas e diretrizes, base de dados, contratos, documentos fiscais. Segundo [WESSELS2002], isto ajuda a dar uma idéia clara sobre os aspectos formais do sistema atual.

6.6.3 Áreas típicas de aplicação da técnica e quando pode ser usada

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 [WESSELS2002].

Útil para determinar as entradas e saídas do sistema e, por implicação, funções intermediárias e requisitos de armazenagem de dados [BRAY2002].

A análise de documentos é também útil como um ponto de partida no estudo da organização porque pode expor tanto a estrutura organizacional formal como os ideais ou filosofias. Declaração da missão, diagramas do *workflow*, hierarquias departamentais, descrições de trabalho ou funções, produtos, formulários etc., todos contêm informações que podem ajudar um analista a começar a compreender como uma organização funciona.

6.6.4 Benefícios/Vantagens

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 refletir as tarefas dinâmicas melhor do que a análise do trabalho efetivo feita pela observação.

6.6.5 Limitações/Desvantagens

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 [BRAY2002].

Um outro problema apontado por [BRAY2002] 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 é um problema, mas existe o perigo desta velha estrutura, geralmente com defeitos, ser transmitida para a especificação e, pior, para o novo sistema, sem as devidas considerações.

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

6.6.6 O que é necessário

Escolher um local tranqüilo para fazer a introspecção e obter todos os documentos relevantes do sistema atual.

6.6.7 Processo

Muitas vezes os documentos não são inteiramente exatos porque as organizações evoluem e estes documentos não são atualizados. Imprecisões podem freqüentemente ser um indício de problemas ou relações de poder que existem nas organizações e não devem ser descartados ou considerados como errados simplesmente. A análise da documentação pode ocorrer a qualquer momento durante um projeto, mas ela é também uma boa maneira de fazer uma introspecção sobre uma organização antes que a avaliação *in loco* ocorra. Analisando formulários padrão, desenhos e outros documentos, detalhes importantes do *workflow* são revelados. Por exemplo, alguns documentos requerem assinaturas autorizadas de um ou mais departamento indicando uma necessidade de esforços coordenados. Além disso, a análise dos documentos pode revelar inconsistências nos dados ou nos procedimentos.

Quando se está analisando os relatórios e os manuais, algumas questões devem ser elaboradas: "É desta maneira que as coisas são realmente feitas? Com que exatidão o atual processo (informal) segue o que está especificado formalmente?" A divergência entre o formal e o informal indica, freqüentemente, uma falha ou um ponto fraco no sistema

formal. Quando se está analisando os formulários, deve-se procurar por áreas que são comumente omitidas (indicando informação desnecessária ou casos especiais) e as áreas onde os erros são cometidos com mais insistência (indicando uma falta de clareza) [WESSELS2002].

Em combinação com outras técnicas, a análise de documentos possibilita, quase sempre, cruzar fontes múltiplas de dados, tal como quando dados primários (por exemplo, transcrições de uma entrevista) são comparados aos dados interpretativos (por exemplo, oriundos de um relatório).

As informações obtidas aqui serão utilizadas freqüentemente para análise dos dados e para ajudar a desenvolver o modelo de dados do domínio do problema.

6.6.8 Procedimentos Práticos

- A análise de documentos é usada geralmente em combinação com outras técnicas, mas pode ser usada individualmente para uma análise histórica;
- Se a documentação é revista durante a fase de elicitação e sumarizada pelo coordenador dos requisitos, por exemplo, os desenvolvedores poderiam usar a informação sumarizada sempre que houver necessidade de consultá-la, ao invés de ir novamente ao documento original;
- A quantidade de documentação para um domínio do problema pode ser substancial; assim pode ser apropriado para o engenheiro de requisitos encontrar-se com os usuários para determinar qual é a documentação mais importante; e
- Analisar menos documentação poderia gastar menos tempo, mas isto deve ser comparado com o risco que pode ser enfrentado ao se deixar de analisar informações importantes [McPHEE2001].

6.7 Prototipação

6.7.1 A classificação na taxonomia

/ Apoio geral / Informal / Observação / Mista / Mista / Validação / Alto / Desenvolvimento de software ou construção de modelos / Médio / Sistema novo / Profundidade / Alta /

6.7.2 Descrição da técnica

Em determinadas situações, "os usuários podem entender e expressar melhor suas necessidades através da comparação com um produto de software que sirva de referência. Quando tal produto não existe, a prototipação pode ser usada para criar um produto que ilustre as características relevantes, através do qual os usuários podem descobrir as suas reais necessidades" [CARVALHO+2001].

Os protótipos permitem aos desenvolvedores criar um modelo do software que deve ser construído. Desenvolvedores e usuários se reúnem e definem os objetivos globais do software, identificam todos os requisitos que são conhecidos até então e decidem em quais áreas será necessário aprofundar as definições. Logo em seguida, faz-se um projeto rápido, que objetiva mostrar uma representação dos aspectos do software que serão visíveis ao usuário como, por exemplo, formatos de entradas e saídas. Este projeto leva à construção do protótipo que será validado pelo usuário e utilizado para refinar os requisitos do sistema a ser construído. Através de interações o protótipo é moldado para satisfazer as necessidades do usuário ao mesmo tempo em que o desenvolvedor obtém uma melhor compreensão do problema.

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].

Segundo [KOTONYA+1998] existem principalmente dois tipos de prototipação:

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

De maneira geral, é possível considerar que todo o ciclo de vida de um software pode ser visto como uma série incremental de protótipos acumulativos. A prototipação evolutiva é baseada no princípio de que os requisitos dos usuários mudarão e sob o ponto de vista evolutivo do ciclo de vida do software pode-se considerar a primeira entrega como um protótipo inicial. Modificações e melhorias subseqüentes resultam em novas entregas de protótipos mais maduros. Este processo continua até que se tenha desenvolvido o produto final.

Para a implementação rápida de um protótipo de sistema, existem três abordagens possíveis[KOTONYA+1998]:

Prototipação no papel, onde uma simulação do sistema é desenvolvida e usada para fazer experimentos, testes e simulação sobre o sistema. Prototipação no papel é uma maneira barata e surpreendentemente efetiva de se construir protótipos. Nenhum software executável necessita ser desenvolvido. São desenhadas versões das telas em papel e apresentadas aos usuários e vários cenários podem ser montados. Analistas e usuários finais trabalham nestes cenários e simulam como o sistema deve ser usado.

Prototipação "mágico de OZ", na qual uma pessoa simula as respostas do sistema para alguns *inputs* do usuário. É também relativamente barato e também não requer que se desenvolva um software. O usuário interage com uma pessoa representando o sistema que simula as respostas do sistema. Esta abordagem é útil quando um novo sistema tem de ser desenvolvido como uma extensão de um sistema existente. Usuários já estão familiarizados com a interface e podem ver as interações entre esta e a funcionalidade do sistema, simuladas pelo "mágico de OZ". O único software requerido é o da interface com o usuário.

Prototipação automatizada (ou baseada em software), na qual uma linguagem de quarta geração, ou outro ambiente rápido de desenvolvimento, é usado para implementar um protótipo executável. É uma maneira mais cara do que as anteriores. Softwares são escritos para simular a funcionalidade do sistema a ser produzido. Devido à necessidade do desenvolvimento rápido, uma linguagem de programação de alto nível deve ser usada para desenvolver o protótipo.

6.7.3 Áreas típicas de aplicação da técnica e quando pode ser usada

Em sistemas de hardware, protótipos são freqüentemente desenvolvidos para testar e fazer provas de projetos; em sistemas de software, protótipos são mais usados para ajudar a elicitar [KOTONYA+1998], entender e validar os requisitos do sistema. Ainda segundo [KOTONYA+1998], os usuários podem experimentar o novo sistema e apontar os pontos fortes e pontos fracos, pois eles têm algo concreto para criticar.

Quando usada apropriadamente, esta técnica pode ser útil para superar as várias dificuldades inerentes ao processo de elicitação de requisitos, especialmente as dificuldades de comunicação e de articulação de necessidades pelo usuário [CARVALHO+2001]. Prototipação é útil para a determinação dos requisitos quando [MAGUIRE1998]:

- Os requisitos do usuário não são bem ou claramente compreendidos;
- Poucos usuários estão envolvidos com o sistema;
- Os possíveis projetos são complexos e requerem uma forma concreta para validá-los;
- Existem problemas de comunicação entre usuários e analistas; e
- As ferramentas e os dados estão prontamente disponíveis.

Em geral, o processo de prototipação é rápido. O desenvolvimento de uma simulação ou protótipo de um futuro sistema pode ser de grande valia, permitindo que os usuários visualizem o sistema e providenciem *feedback* sobre o sistema. Assim, ele pode ser usado para esclarecer opções de solicitações do usuário. Mais adiante no processo, ele pode também ser usado para especificar detalhes de interface do usuário para ser incluído no futuro sistema. [MAGUIRE1998]

6.7.4 Benefícios/Vantagens

- A prototipação é benéfica somente se o protótipo puder ser construído substancialmente mais rápido que o sistema real [CARVALHO+2001];
- Construir um protótipo permite que o usuário veja um modelo de sua descrição verbal, possibilitando que revisões a este modelo sejam feitas;
- Para [KOTONYA+1998] o principal benefício de desenvolver um protótipo na fase de elicitação é permitir aos usuários finais do sistema testá-lo através do software e descobrir o que realmente eles precisam, compreendendo como o sistema pode ser usado para apoiar o trabalho deles;
- Normalmente as pessoas encontram dificuldades em visualizar como uma especificação textual de requisitos será transformada em um software executável. Com um protótipo para demonstrar os requisitos que eles não compreendem completamente e os usuários descobrem que é mais fácil encontrar problemas e sugerem como os requisitos podem ser melhorados [KOTONYA+1998]; e
- Proporciona aos usuários uma demonstração tangível do que trata o sistema [MAGUIRE1998]

Outras vantagens relatadas por [KOTONYA+1998] são:

- O protótipo pode ajudar a estabelecer todas as viabilidades e utilidades do sistema antes que altos custos de desenvolvimento sejam contraídos;
- Prototipação é a única maneira efetiva de desenvolver intefarces de usuários. Se um protótipo foi desenvolvido como parte de um processo de requisitos, pode-se reduzir custos futuros de desenvolvimento para o sistema;
- É possível usar o protótipo para desenvolver testes de sistema que podem ser usados mais tarde no processo de validação do sistema; e
- A implementação de protótipos requer um estudo cuidadoso dos requisitos. Isto muitas vezes revela inconsistências, omissões e requisitos incompletos.

6.7.5 Limitações/Desvantagens

Existem custos e problemas associados à prototipação. [KOTONYA+1998] relata seis deles:

- Custo de treinamento: Se uma organização não possui experiência com o desenvolvimento de protótipos, os desenvolvedores devem ser treinados em algum ambiente de prototipação ou em alguma técnica discutida anteriormente;
- Custo de desenvolvimento: Depende do tipo de sistema a ser prototipado e da técnica a ser aplicada. Os custos podem variar de poucas pessoas por dia, para sistemas pequenos, a várias pessoas por ano para grandes sistemas;
- Prorrogação de cronogramas: em alguns casos, desenvolvimentos de protótipos podem causar uma prorrogação de cronograma, com conseqüente adiamento da data de entrega do produto. Por outro lado, o tempo gasto na prototipação pode ser recuperado, evitando refazer tarefas; e
- Protótipos incompletos: Algumas vezes, protótipos podem somente simular somente a versão final do sistema, sem estágios intermediários. Isto oferece pouca ajuda na determinação de requisitos que poderiam surgir na prototipação.

Outras desvantagens da prototipação são:

- Prototipação não é benéfica quando seu uso demonstra que há uma tendência de evitar a criação da documentação formal do sistema;
- Um dos grandes problemas enfrentados com o uso dessa técnica está no fato de os usuários sentirem-se satisfeitos com o protótipo e decidirem pela não construção da versão final do sistema [KOTONYA+1998];
- Embora normalmente seja mais rápida, em alguns casos a técnica pode levar mais tempo que outras abordagens [MAGUIRE1998];
- Os recursos exigidos para a prototipação baseada em software e hardware são bem maiores do que os recursos para a prototipação em papel [MAGUIRE1998].

6.7.6 O que é necessário

Requer habilidades para desenvolvimento de software ou construção de modelos por parte dos desenvolvedores. Uma condição fundamental para a utilização da técnica de prototipação é que o protótipo tem que ser desenvolvido rapidamente, de modo que ele possa ser usado no processo de desenvolvimento [KOTONYA+1998]. Para tanto, é necessária a utilização de ferramentas que proporcionem esta agilidade, sobre as quais o desenvolvedor deve ter total domínio. Os processos para obtenção de *feedback* do usuário

também incluirão um certo custo em termos de tempo e esforços [MAGUIRE1998].

6.7.7 Processo

O processo de prototipação começa com um estudo preliminar dos requisitos dos usuários. A seguir, tem início um processo iterativo de construção do protótipo e avaliação com os usuários. Cada repetição permite que o usuário entenda melhor os requisitos, inclusive as implicações dos requisitos articulados nas interações anteriores. Eventualmente, um conjunto final de requisitos pode ser formulado e o protótipo é descartado [CARVALHO+2001].

Um procedimento geral para adotar o método de prototipação rápida é o seguinte [MAGUIRE1998]:

- Primeiramente, deve haver tempo suficiente para criar o protótipo. Se o protótipo for avaliado com usuários, então é necessário um tempo para projetar tarefas relevantes, recrutar usuários, avaliar o protótipo e relatar os resultados;
- Montar o equipamento necessário, incluindo o hardware e o software e as ferramentas necessárias para criar um protótipo interativo;
- Desenvolver o protótipo;
- Selecionar usuários apropriados para testar o protótipo, tentando cobrir uma gama de usuários dentro de uma população alvo. Um facilitador também poderá ser solicitado para instruir os usuários e proceder à avaliação;
- Preparar tarefas reais para serem feitas pelos usuários enquanto eles trabalham com o protótipo;
- Conduzir o procedimento da avaliação e assegurar que o protótipo possa ser usado para cumprir as tarefas;
- Cada sessão deve ser conduzida pelo facilitador, que fornece as instruções aos usuários para trabalhar nas tarefas alocadas, interagindo e respondendo ao sistema apropriadamente;
- Se necessário, informação adicional pode ser obtida através de entrevista com os usuários à medida que eles utilizem o protótipo;
- Agradecer ao usuário;
- Analisar a informação obtida e então resumir as observações e avaliações do usuário;
- Resumir as implicações para o projeto e recomendações para melhorias e dar o *feedback* para a equipe do projeto; e

- Verificar onde é necessário refinar o protótipo e repetir o processo descrito acima.

6.7.8 Procedimentos Práticos

- Evitar desperdiçar muito tempo no desenvolvimento do primeiro protótipo. Similarmente, tentar não gastar muito esforço em características particulares (por exemplo, animações), que podem ser desnecessárias [MAGUIRE1998];
- Evitar fazer o protótipo muito bem elaborado, pois isso pode fazer com que os usuários o visualizem como finalizado [MAGUIRE1998];
- Evitar colocar características que irão aumentar as expectativas dos usuários, mas que são improváveis de serem implementadas em um sistema real (por exemplo, tempo muito rápido de resposta ou gráficos muito sofisticados) [MAGUIRE1998];
- O rápido desenvolvimento de protótipos é essencial, por isso eles devem estar disponíveis logo na fase inicial do processo de elicitação [KOTONYA+1998].

6.8 Construção de Cenários

6.8.1 A classificação na taxonomia

/ Apoio geral / Informal/ Levantamento estruturado / Mista / Mista / Análise / Médio / Trabalhar em grupo / Médio / Ambos / Largura / Alta /

6.8.2 Descrição da técnica

É uma técnica de elicitar requisitos a partir de indivíduos, compreender os requisitos e analisá-los para descrever antecipadamente os comportamentos esperados do sistema. [KOTONYA+1998] explica que, de uma maneira geral, os usuários preferem relacionar exemplos do seu dia-a-dia ao invés de relatar descrições abstratas das funções realizadas por um sistema. Por este motivo, é freqüentemente útil desenvolver um conjunto de interações de cenários e usá-lo para elicitar os requisitos. Os usuários simulam as suas interações usando o cenário e os desenvolvedores as utilizam para formular os requisitos reais do sistema. Ainda segundo [KOTONYA+1998], os cenários podem ser pensados como sendo "estórias que explicam como o sistema é usado".

Cenários são, portanto, caracterizações de usuários e suas tarefas dentro de um contexto específico. Os cenários oferecem representações concretas de um usuário

trabalhando com um sistema computacional para atingir um objetivo particular. Os objetivos primários da construção do cenário são gerar requisitos finais do usuário, expor interações possíveis do sistema e revelar as facilidades de que o sistema pode precisar [KIRAKOWSKI1997]. Cenários podem ser escritos de diferentes maneiras, mas de acordo com [KOTONYA+1998] eles deveriam incluir pelo menos as seguintes informações:

- Uma descrição do estado do sistema no início do cenário;
- O fluxo normal de eventos no cenário;
- Exceções ao fluxo normal de evento (o que pode sair errado e como é tratado);
- Informações sobre outras atividades concorrentes; e
- Uma descrição do estado do sistema no final do cenário.

6.8.3 Áreas típicas de aplicação da técnica e quando pode ser usada

Os cenários são criados pelos membros de uma equipe de desenvolvimento, e discutidos com usuários para estabelecer como eles gostariam ou não de interagir com o sistema [MAGUIRE1998]. Cenários são úteis primariamente para adicionar detalhes a uma descrição de requisitos [KOTONYA+1998];[SOMMERVILLE+1997]. Podem ser usados também para descrever como um sistema responde à ocorrência de um evento. Os cenários são descrições do uso prático do sistema e, portanto, bastante úteis na elicitação de requisitos, podendo ser usados para sistemas novos como para discutir melhorias em sistemas já em produção.

Este trabalho trata a construção de cenários como uma técnica que pode ser empregada de maneira independente para elicitar requisitos, pois como será visto no próximo item, 6.9, cenários também são partes ou instâncias de casos de uso que, por sua vez, são partes básicas da análise orientada a objeto. [SOMMERVILLE+1997] relata que cenários podem ser usados para coletar requisitos de maneira mais informal, cabendo ao desenvolvedor e ao usuário identificar cenários e seus detalhes e, caso se deseje fazer uma abordagem mais estruturada, os casos de uso podem ser aplicados.

6.8.4 Benefícios/Vantagens

Alguns benefícios são relatados por [KIRAKOWSKI1997]:

- Os usuários acham fácil compreender e descrever os requisitos através de cenários;
- Cenários podem ajudar a identificar tempos prováveis de conclusão das tarefas;
- Um mínimo de recursos é necessário para gerar os cenários;

- A técnica também pode ser aplicada por desenvolvedores que possuem pouca experiência no assunto;
- Auxiliam o entendimento dos requisitos e facilitam o trabalho dos usuários, simulando tarefas reais. A construção dos cenários favorece o descobrimento de detalhes que não poderiam ser observados de outra maneira [KOTONYA+1998];
- Uma vez desenvolvidos, os cenários podem ser reutilizados através do próprio sistema ou em outros sistemas nos quais se aplique o mesmo contexto. [KOTONYA+1998]; e
- Facilita a compreensão dos requisitos e expõe possíveis interações do sistema.

6.8.5 Limitações/Desvantagens

- Cenários não são apropriados para considerar os detalhes de interface e *layout* [MAGUIRE1998];
- A construção de cenários pode consumir bastante tempo porque sistemas complexos, usualmente, exigem um grande número de cenários [KOTONYA+1998]; e
- O custo total do desenvolvimento dos cenários também depende da experiência da equipe de desenvolvedores.

6.8.6 O que é necessário

Os recursos necessários são mínimos e os cenários devem ser produzidos de forma rápida. Para sistemas simples, talvez apenas poucas horas sejam necessárias. Um moderador experiente é recomendado para as sessões nas quais o cenário é explorado e até algumas horas por sessão podem ser exigidas [MAGUIRE1998].

Necessita do trabalho em conjunto do desenvolvedor e do usuário. O desenvolvedor toma nota dos comentários do usuário, problemas e sugestões. O usuário simula o uso do sistema, seguindo o cenário, apontando onde o cenário está correto ou incorreto [KOTONYA+1998]. Requer um baixo esforço e tempo para elicitar os requisitos.

6.8.7 Processo

Cenários podem ser imaginados como estórias que explicam como o sistema deverá ser usado. Uma vez conhecida a idéia básica das facilidades que o sistema deve prover, um cenário é construído para mostrar estas facilidades. Os cenários podem ser identificados através das discussões iniciais com os usuários que interagem com o sistema [KOTONYA+1998]. O desenvolvedor pergunta sobre vários pontos das tarefas dos usuários - como são desempenhadas, quem está envolvido com elas e o que acontece se elas forem executadas de alguma maneira alternativa. Durante este levantamento de requisitos, são acrescentados detalhes para fazer uma descrição completa desta interação [SOMMERVILLE+1997].

[MAGUIRE1998] e [KIRAKOWSKI1997] descrevem alguns passos para empregar esta técnica:

- Encontrar um facilitador experiente junto à equipe de desenvolvimento, outros usuários relevantes ou qualquer pessoa envolvida;
- Identificar os usuários candidatos, suas tarefas e o contexto geral. Essas informações irão prover a base para os cenários serem criados pela equipe de desenvolvimento;
- Decompor funcionalmente os objetivos do usuário em operações necessárias para alcançá-los;
- Atribuir estimativas de tempo e critérios para a conclusão das tarefas;
- A sessão pode ser gravada em vídeo para uma posterior revisão ou transcrição para distribuição entre os envolvidos;
- Os resultados das sessões de construção do cenário podem ser usados para planejar avaliações com os usuários.

6.8.8 Procedimentos Práticos

- Gerar cenários para cobrir diversas situações, não somente as mais comuns ou aquelas de maior interesse para a equipe de desenvolvimento [MAGUIRE1998];
- Incluir situações problemáticas, que irão testar o conceito do sistema e não somente cenários simples ou sem dificuldades; e
- Trabalhar em cenários integrais para ter meios de analisar o sistema ao invés de tentar alterar o sistema na metade do caminho.

6.9 Casos de Uso

6.9.1 A classificação na taxonomia

/ Apoio geral / Informal / Levantamento estruturado / Mista / Mista / Especificação / Alto / Capacidade de escrever ou redigir / Alto / Ambos / Largura / Alta /

6.9.2 Descrição da técnica

Casos de uso foram originalmente propostos por [JACOBSON+1992] como uma técnica para especificação de requisitos funcionais e atualmente faz parte da proposta da UML (*Unified Modeling Language*) e é uma característica fundamental desta notação usada para descrever modelos de sistemas orientados a objetos [SOMMERVILLE+1997]. Casos de uso têm uma aplicação muito ampla e aqui o objetivo é dar uma visão conceitual e mostrar a sua aplicabilidade como técnica de elicitação de requisitos.

Um caso de uso pode ser entendido basicamente como a descrição de um conjunto de seqüências de interações entre o sistema e um ou mais atores na qual se considera o sistema como uma caixa preta, fornecendo resultados observáveis que têm valores para os atores [JACOBSON+1992];[TORO+2000]. Cada caso de uso é um curso de eventos completo no sistema, visto sob a perspectiva do usuário [JACOBSON+1992]. O conjunto de casos de uso mostra todas as formas possíveis de se utilizar o sistema e é representado por **diagramas de casos de uso**.

Os modelos de diagramas de casos de uso servem para proporcionar uma visão global de todos os casos de uso de um sistema, dos atores e dos casos de uso em que estes participam. Nestes diagramas os atores são representados por pequenos bonecos ("homens de pau ou palito") e os casos de usos são mostrados através de elipses contidas em um retângulo que representa o sistema. O nome de um caso de uso é um verbo no infinitivo, seguido pelo objeto ou entidade afetada pelo caso de uso e é dado sempre a partir da perspectiva do usuário e não do sistema. [TORO+200];[SOMMERVILLE+1997]. A figura 7 mostra a representação gráfica dos diagramas de casos de uso. Modelos de casos de uso são utilizados para mostrar os requisitos de um novo sistema e também para desenvolver novas versões de um sistema já existente. Na construção de um diagrama de casos de uso deve-se ter claros três conceitos básicos:

1. Atores

Os atores podem ser pessoas, sistemas ou hardware que interagem com o sistema cujos requisitos funcionais estão sendo descritos pelos casos de uso. Um usuário pode agir como vários atores e o papel de um ator pode ser desempenhado por vários usuários; o termo ator representa o perfil genérico do usuário do sistema. O nome que se dá a um ator deverá refletir o papel que ele terá para o sistema [JACOBSON+1992]. Podemos ter a seguinte situação: suponhamos que estamos especificando um sistema no qual os clientes podem incluir pedidos interativamente no sistema, por exemplo, uma empresa que vende produtos pela internet; cada cliente que incluir um pedido no sistema estará desempenhando o papel de um ator. São vários clientes (usuários) fazendo a mesma coisa no sistema.

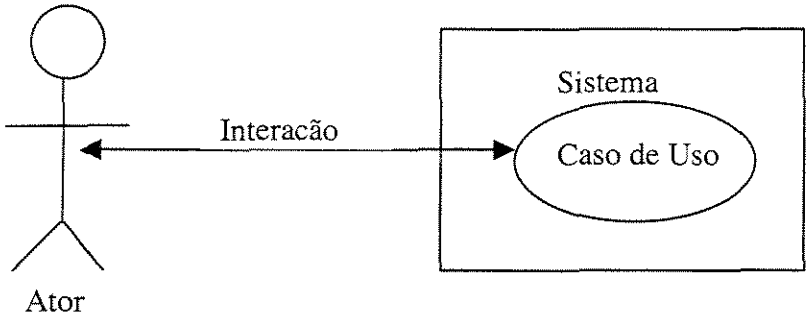

Figura 7 - Diagrama de caso de uso

Um outro sistema que de alguma forma interage com o que está sendo construindo também é um ator. Por exemplo, se o novo sistema deverá gerar ordens de pagamentos para ser processadas pelo sistema financeiro, este sistema será um ator que usa os serviços de novo sistema.

Através da identificação dos atores é possível:

- Definir os limites do sistema (o que faz parte do sistema e o que não faz).
- Desenvolver um sistema orientado para o usuário que contemple todas as funcionalidades esperadas pelos diferentes atores.

Atores podem ser identificados através de alguns questionamentos sugeridos por [JACOBSON+1992] e [FURLAN1998]:

- Quem utilizará a principal funcionalidade do sistema?
- Quem irá usar, manter, administrar e fazer com o que o sistema permaneça operando?
- Quem dará suporte ao sistema em sua operação diária?

- Quem tem interesse nos resultados produzidos pelo sistema ou em algum requisito funcional específico?
- Quais os recursos de hardware exigidos pelo sistema?
- Quais outros sistemas irão interagir com o novo sistema?

Apesar da identificação inicial dos atores, este processo deve se repetir, conforme os casos de usos vão sendo descobertos, pois ao conhecer mais detalhes do sistema podem aparecer novos tipos de usuários

2. Casos de uso

Como já mencionado anteriormente, um caso de uso é uma seqüência de interações entre um sistema e alguém ou algo que usa algum de seus serviços. Reflete o uso que os atores farão do sistema; através deles podem ser mostradas tanto as funcionalidades que o sistema oferecerá como os diferentes comportamentos possíveis às diversas situações contempladas (cenários). Os casos de uso são escritos com a finalidade de expressar o que deve fazer o sistema a ser desenvolvido, sem levar em consideração como fazê-lo.

Segundo [WESSELS2002], com o aumento do uso de técnicas orientadas a objeto, tanto para análise como para o projeto, existe uma ênfase maior na idéia de casos de uso como um meio de obter informações dos usuários de um sistema. A idéia dos casos de uso é estabelecer um conjunto de atividades desempenhadas por um sistema para atingir algum resultado específico. Isso é tipicamente uma abordagem utilizada para descrever sistemas, feita inteiramente sob a perspectiva do usuário. Por exemplo, se for observado um sistema de informação de uma oficina mecânica, pode-se ter um caso de uso que descreva a seqüência de eventos quando um cliente quer agendar uma nova revisão. Isso é inteiramente descrito sob a perspectiva deste usuário/cliente e o objetivo é colher os casos de uso considerando cada aspecto observável do sistema. O desenvolvedor deve moldar o sistema de acordo com as necessidades dos usuários e não fazer com que os usuários se adaptem ao sistema. Da coleta e junção dos casos de uso forma-se um completo modelo do sistema.

A partir do momento que um ator inicia um caso de uso, ele e outros atores trocam dados ou controles com o sistema, tornando-se participantes desse caso de uso. Os casos de uso são identificados através dos atores e para cada curso de eventos completo iniciado por um ator é identificado um caso de uso. Para identificar casos de usos deve-se ler a especificação de requisitos da perspectiva do usuário e discutir com todos aqueles que atuarão como atores [JACOBSON+1992].

[JACOBSON+1992] sugere algumas questões que podem ser feitas para ajudar a

identificar os casos de usos:

- Quais são as principais tarefas de cada ator no sistema?
- Quais informações o sistema deve produzir?
- O que cada ator necessita fazer?
- O ator precisa ler, criar, apagar, modificar ou armazenar algum tipo de informação no sistema?
- O ator necessita informar o sistema sobre mudanças ocorridas externamente?
- O ator necessita ser informado sobre mudanças ou eventos inesperados ocorridos no sistema?
- Haverá ganho de eficiência, mudanças, simplificações no trabalho diário do usuário através de novas funções no sistema?
- Existem problemas com o atual sistema?
- Quais resultados são esperados do novo sistema e que entradas e saídas o sistema requer?
- Todos os requisitos funcionais podem ser desenvolvidos pelos casos de usos?

3. Relacionamentos

Os atores comunicam-se com o sistema através do envio e recebimento de mensagens [JACOBSON+1992]. Os casos de usos podem estar relacionados com atores ou outros casos de uso e uma relação entre estes e/ou atores relacionados é representada, graficamente, por uma linha sólida. A extremidade desta linha dependerá do tipo da relação, isto é, a linha conterá ou não uma seta. Existem, basicamente, quatro tipos possíveis de relação: comunicação, uso, extensão e generalização.

Comunicação: É a relação de um ator com o caso de uso com o qual ele está interagindo. É representada por uma linha sólida que conecta o símbolo de ator ao símbolo de caso de uso.

Uso (*includes, uses*): É representado por uma seta apontando no sentido da relação, ou seja, do caso de uso que faz o uso ao caso de uso que é usado e a seta é etiquetada com o rótulo <<*includes*>>. É uma simples relação de inclusão, ou seja, os cenários ou situações possíveis detalhados em um caso de uso estão incluídos em outro caso de uso. Quando dois ou mais casos de uso têm comportamento comum - uma seqüência comum de interações - esse comportamento pode ser modelado ou descrito em um outro caso de uso, que é utilizado por outros casos [JACOBSON+1992].

Na elaboração de um sistema é comum que a mesma funcionalidade do sistema seja

acessada por vários casos de uso. Por exemplo, a funcionalidade de "Validar Cliente" pode ser acessada quando:

- um novo cliente estiver sendo cadastrado;
- um novo pedido estiver sendo feito; ou
- uma consulta dos pedidos de um cliente for solicitada, etc.

Esta relação serve para evitar a repetição deste caso de uso nos outros casos de uso, onde é necessário também "Validar Cliente". Podemos ler da seguinte maneira: o caso de uso "Consultar pedidos de um cliente" usa o caso de uso "Validar Cliente". A figura 8 mostra um exemplo deste relacionamento.

Este conceito não é novo, podendo ser comparado ao conceito de sub-rotina ou subprograma e estas são algumas características das relações de uso:

- Aparecem como funcionalidade comum ao invés de especificar vários casos de uso;
- Os casos <<include>> usados são casos de uso por si só, isto é, existem primariamente como um caso de uso;
- O caso de uso <<include>> é usado sempre que o caso de uso principal é executado. Isto faz a diferença com as extensões, que são opcionais.

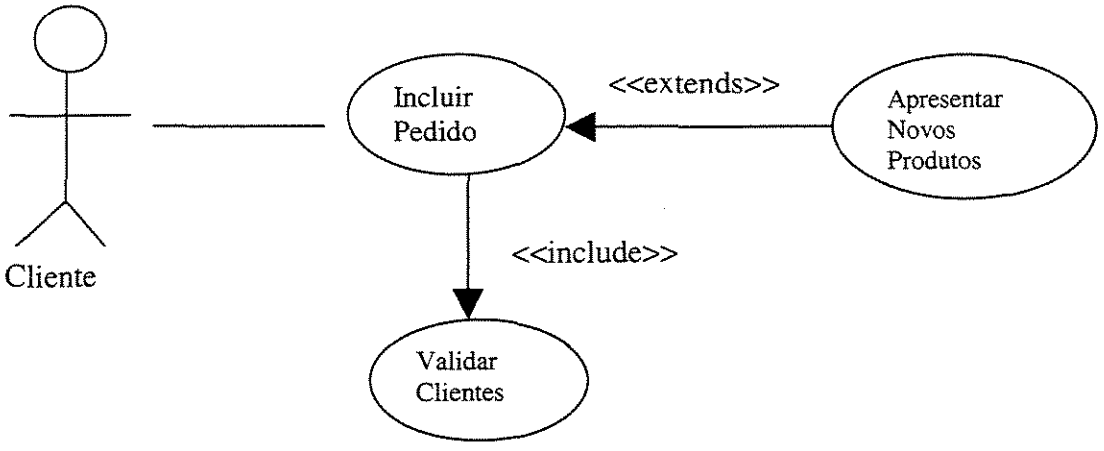

Figura 8 - Relacionamentos em caso de uso

que uma rota particular pode compreender um cenário particular.

[SOMMERVILLE+1997] explica ainda que, freqüentemente, existe confusão para determinar se um caso de uso é ou não um cenário em si; pode-se dizer que um caso de uso representa um conjunto de cenários semelhantes e cada cenário é um traço isolado dentro do caso de uso, havendo um cenário para a interação normal e os cenários para descrever cada variação ou exceção. Pode-se afirmar, então, que casos de uso são técnicas baseadas em cenários nos quais são identificados atores e sua interação com o sistema; um conjunto de casos de uso descreve todas as possíveis interações com o sistema. Pode-se também afirmar que um caso de uso é um cenário e um cenário pode ser um conjunto de casos de usos. Os cenários construídos podem, portanto, ser usados nos casos de uso.

Os cenários e os casos de uso permitem ao desenvolvedor do sistema "caminhar através" de uma seqüência particular de ações ou de eventos que descrevem uma parte do comportamento do sistema, além de fornecer um mecanismo para compreender e comunicar a seqüência de ações, incluindo as variantes, que ocorrem nesta parcela/porção particular do comportamento do sistema [MCPHEE2001].

6.9.4 Áreas típicas de aplicação da técnica e quando pode ser usada

Apesar de ter nascido como técnica voltada primariamente para a validação e especificação dos requisitos do sistema que já foram previamente elicitados por alguma outra técnica, casos de uso podem ser usados tanto para elicitação como para especificação de requisitos, pois na validação podem surgir novos requisitos. Portanto, eles também podem ser usados, secundariamente, para elicitar requisitos.

São também úteis para projetar testes/provas que assegurem a qualidade dos requisitos e propiciam uma compreensão mútua dos requisitos tanto por parte dos usuários como dos desenvolvedores. Neste sentido eles facilitam a compreensão por parte dos usuários, pois eles modelam o sistema sob o ponto de vista do usuário.

Uma vez que os requisitos são validados, os casos de usos geram uma versão inicial da documentação e de manuais do sistema. São usados também para identificar quem interagirá com o sistema e o que o sistema irá fazer, bem como quais interfaces deverão ter o sistema.

Cada caso de uso é documentado através de uma representação gráfica e de um texto com a descrição das situações ou cenários que os usuários podem encontrar em sua interação com o sistema. Também são úteis para ser utilizados como confirmação de entendimento dos requisitos por parte da equipe de desenvolvimento.

Eles também são boa base para um desenvolvimento incremental e interativo,

Extensões: Este tipo de relação reflete situações ou comportamentos particulares de exceção e casos especiais que aumentariam a quantidade de casos de uso no modelo. Em outras palavras, muitas vezes a funcionalidade de um caso de uso inclui um conjunto de passos que incidem somente em algumas oportunidades. Os casos de uso nesta situação poderiam ser tratados (estendidos) por outros casos de uso, especificando como o comportamento definido para o primeiro caso pode ser inserido no comportamento definido para o segundo [JACOBSON+1992]. Esta relação é representada também por uma seta que é etiquetada com o estereótipo <<*extends*>>, do caso de uso que fornece a extensão para o caso de uso básico [FURLAN1998].

No caso da empresa que vende produtos pela Internet podemos ter a seguinte situação: dentro da funcionalidade da inclusão de pedidos o usuário pode solicitar ao sistema que lhe faça uma apresentação sobre os novos produtos disponíveis, suas características e preços. Neste caso, tem-se uma exceção dentro do caso de uso "Incluir Pedido". A exceção consiste em interromper o caso de uso e passar a executar o caso de uso "Apresentar Novos Produtos". Nesta situação dizemos que o caso de uso "Apresentar Novos Produtos", estende o caso de uso "Incluir Pedido". A figura 8 mostra um exemplo deste relacionamento.

As extensões têm as seguintes características:

- Representam uma parte da funcionalidade do caso que nem sempre ocorre; e
- Não necessariamente advêm de um erro ou exceção.

Generalizações: Trata-se do conceito de herança, comum nos diagramas de classes, porém também aplicado entre casos de uso e atores. São representadas por uma seta com um triângulo vazio cuja ponta assinala o sentido da relação. Uma generalização ocorre quando um caso de uso é um caso particular de outro, herdando toda a funcionalidade do caso de uso. Deve ser utilizada quando dois ou mais casos de uso possuem muitas funcionalidades em comum.

6.9.3 Casos de uso e cenários

[BRAY2002] argumenta que o termo cenário é anterior ao caso de uso e tem sido usado mais ou menos com o mesmo significado por alguns, mas outros significados, sutilmente diferentes, foram a ele atribuídos, incluindo o ambiente no qual um caso de uso pode ser executado. Alguma imprecisão ainda persiste, mas cenário é agora usado para indicar uma instanciação de um caso de uso. Casos de uso simples são lineares e tem somente uma rota, mas muitos casos de usos têm rotas alternativas; é nestas circunstâncias

principalmente, se for utilizado junto com algum tipo de prototipação. [JACOBSON+1992] propôs que os casos de usos fossem usados para fazer a reengenharia de processos do negócio.

6.9.5 Benefícios/Vantagens

- Casos de usos podem propiciar maior credibilidade por parte dos usuários, pois logo na fase inicial do desenvolvimento eles podem ter uma idéia clara dos limites do sistema;
- Os casos de uso apresentam certas vantagens sobre uma descrição meramente textual dos requisitos funcionais, pois facilitam a elicitação de requisitos e são facilmente compreendidos pelos usuários, além de servir de base de provas do sistema e de documentação para os usuários [TORO+2000];
- Um caso de uso bem escrito/modelado descreve somente o que é pertinente na interação entre o ator e o sistema; e
- Casos de usos oferecem uma possibilidade de validar requisitos logo no início do processo de ER.

6.9.6 Limitações/Desvantagens

- O tempo gasto para definir todos os casos pode ser alto, principalmente se o sistema for grande e/ou complexo;
- A análise de qualidade depende da qualidade com que foi feita a descrição dos casos de uso;
- Somente o uso de casos não fornece informações suficientes para se descobrir todos os requisitos de um sistema; e
- Não se deve esperar que os casos de uso capturem todos os requisitos. Eles capturam somente os funcionais e podem reforçar as regras do negócio.

6.9.7 O que é necessário

Para ter casos de uso bem definidos são necessários, segundo [HAM1998], os seguintes componentes:

- descobrir os atores que interagem com o sistema;
- descobrir os eventos e as regras de negócios associadas com os quais os atores interagem;

- descobrir o fluxo de informações que são enviadas e retornadas no curso de cada interação;
- ter definido claramente o contexto no qual os casos de uso acontecem.

As fontes de informações para os casos de uso podem ser:

- documentação de especificações do sistema;
- leituras referentes ao domínio;
- entrevistas com especialistas do domínio;
- o conhecimento que a equipe de desenvolvimento tem sobre o domínio; e
- sistemas legados.

Para elaborar os casos de usos são necessárias pessoas que possuem habilidades adequadas e não simplesmente alocar as pessoas que estão disponíveis. Basicamente, dois tipos de pessoas são requeridos:

- Especialistas do domínio; e
- Analistas (desenvolvedores, engenheiros de requisitos).

Os especialistas do domínio são as pessoas que fornecem o material "bruto" contendo as regras do negócio para compor as descrições dos casos de usos. São as pessoas que devem compreender o negócio como ele é e, mais importante, que sejam capazes e estejam dispostos a definir como será o novo sistema.

Os analistas facilitam o desenvolvimento dos casos de uso e são eles que analisam e escrevem os casos de uso. Os analistas devem pensar abstratamente, ter alta tolerância para lidar com incompletitudes e necessitam compreender inteiramente o processo e os objetivos do desenvolvimento dos casos de uso. Os casos de uso são textos e não são derivados como teoremas; eles são compostos. Escrever um caso de uso não tem nada em comum com escrever códigos de programa; está mais próximo do estilo de escrever uma estória curta do que escrever uma especificação técnica. Não são, portanto, documentos técnicos; para que os casos de uso tenham sucesso eles devem ser bem escritos por pessoas que têm as habilidades para expressar exatamente o que acontece entre o sistema e seus atores.

6.9.8 Processo

Os casos de uso podem ser construídos tanto para o sistema existente quanto para um sistema proposto. Para o processo ser efetivo, é importante identificar todas as pessoas que interagem com o sistema e todas as atividades que elas executarão através do sistema. Geralmente é nas conversas iniciais que se tentará identificar todos (ou quase todos) os casos de uso, antes de iniciar a construção detalhada da informação de cada um. Uma vez que os casos de uso tenham sido identificados, o próximo passo é obter informações sobre cada um deles usando, por exemplo, algumas técnicas de elicitação descritas anteriormente. À medida que essas informações vão sendo obtidas, pode-se verificar se alguns casos de uso foram deixados de lado, incorporando-os quando necessário..

Como sempre, esse é um processo interativo. Nos primeiros passos tenta-se identificar os principais elementos de dados e as principais atividades associadas com cada caso de uso; nos passos seguintes refina-se tudo para obter detalhes mais precisos.

Quando existem muitos casos de uso, pode-se dividi-los em grupos relacionados ou pacotes. Cada pacote pode ser dividido em sub-grupos, de forma que cada sub-grupo seja reduzido a um número baixo de casos de uso, facilitando a documentação, análise e compreensão.

Uma vez expostos os principais tipos de relação que vamos encontrar nos diagramas de casos de uso, deve-se fazer referência à descrição que acompanha cada caso de uso. Até agora foi explicada e levada em consideração, principalmente, a representação gráfica dos casos de uso; porém, separado desta representação, um diagrama de casos de uso terá associada uma descrição textual, em forma de fluxos de eventos, de cada caso de uso representado. Podemos considerar aqui o momento em que cenários serão utilizados para ilustrar o comportamento do sistema, ou seja, o cenário será uma instância do caso de uso. Surgem aqui dois tipos de fluxos, ou cenários, para ser considerados:

- Fluxo de eventos principal (Cenário Principal): Trata-se de uma descrição dos eventos que vão acontecendo no uso habitual, ou seja, quando não se apresenta nenhum tipo de problema e tudo corre bem.

- Fluxo de eventos excepcional (Cenário Secundário): Permite uma seqüência diferente de eventos em relação ao cenário principal, sendo que para cada um destes cenários atípicos será definido o fluxo de eventos correspondente. São os casos onde podem ocorrer exceções, erros ou qualquer outra ação possível e diferente.

Um exemplo do fluxo de eventos do cenário principal do Caso de Uso "Incluir Pedido", no caso da empresa que vende produtos pela Internet é mostrado a seguir:

1. O cliente acessa o sítio da empresa;
2. O cliente seleciona "Incluir Pedido"
3. O cliente entra com seus dados pessoais (CPF, nome, endereço)
4. O sistema valida os dados do cliente
5. O cliente seleciona os produtos desejados a partir de uma lista
6. O sistema fornece descrição e preço do produto
7. O cliente confirma os produtos desejados
8. O cliente entra com a informação da forma de pagamento
9. O cliente submete o pedido
10. O sistema verifica todas as informações do pedido
11. O sistema apresenta o valor total, a data da entrega e gera um ID do pedido
12. O cliente confirma ou não o pedido
13. Quando o pagamento é confirmado o sistema avisa a transportadora para enviar o pedido ao cliente.

Este caso de uso pode ter vários cenários secundários, entre eles:

- Pedido incompleto;
- Cliente não cadastrado;
- Endereço de entrega incompleto;
- Produto não disponível em estoque;
- Apresentar novos produtos; e
- Pagamento com cheque.

Algumas perguntas podem ser feitas para descobrir cenários secundários [BRAY2002]:

- Existe outra maneira de realizar esta tarefa?
- O ator tem uma escolha de próximos passos?
- Quais erros podem surgir durante a execução da tarefa?

Um exemplo do fluxo de eventos do cenário secundário do caso de uso "Apresentar Novos Produtos" seria:

- O cliente solicita que o sistema apresente os novos produtos; e
- O sistema mostra uma lista dos novos produtos ou avisa que não há novos produtos.

6.9.9 Procedimentos Práticos

- Para que os desenvolvedores possam ter uma melhor comunicação com os usuários os casos de uso podem ser escritos utilizando a língua natural. Isto proporciona um ganho significativo na comunicação, pois os usuários não precisam entender notações gráficas empregadas pelo desenvolvedor. Entretanto, este uso da língua natural deve ser feito com bastante cuidado;
- Casos de uso bem escritos em língua natural, além de ser uma forma proveitosa de comunicação entre os usuários e desenvolvedores, podem ter também sua qualidade assegurada, desde que escritos por analistas com as habilidades requeridas e experientes;
- Deve-se manter coerência de estilo de escrita para todos os casos de uso do sistema;
- Um dos pontos mais críticos é verificar cada caso de uso com uma quantidade razoável e variada de pessoas, para se assegurar que os passos sejam descritos corretamente e que detalhes importantes não sejam omitidos [WESSESLS2002].

O processo de descrição de um caso de uso básico é simples. Para cada caso de uso candidato deve-se:

- Expressar claramente o objetivo do caso de uso;
- Produzir um resumo da descrição do caso de uso no cenário principal, ou seja, em que tudo funciona perfeitamente e o objetivo do caso de uso é alcançado;
- Elaborar uma estrutura/esquema simples da descrição do caso de uso para refletir sobre o cenário principal. Com esta estrutura é possível elaborar uma exata, mas não detalhada, descrição do caso de uso. O caso de uso ainda não está "colorido";
- Adicionar na estrutura/esquema da descrição do caso de uso mais detalhes sobre negócio ou processo para o cenário principal (não fornecer detalhes da tecnologia). Agora a descrição do caso de uso tem profundidade e personalidade, ele está "colorido".

6.10 Reuso de Requisitos

6.10.1 A classificação na taxonomia

/ Apoio geral / Informal / Levantamento estruturado / Tecnológica / Documentos / Especificação / Alto / Aproveitar análises anteriores / Baixo / Sistema novo / Profundidade / Baixa /

6.10.2 Descrição da técnica

É geralmente boa prática na ES aproveitar o máximo possível o conhecimento já existente na organização para a construção de um novo sistema. Na elicitação também é uma boa prática reutilizar a especificação de requisitos ou, pelo menos, parte dela, desenvolvida para outros sistemas. Enquanto existem situações onde são claras as distinções de requisitos para cada sistema, há outros casos nos quais o reaproveitamento ou reuso de requisitos é viável e possível [KOTONYA+1998]. O reuso de requisitos compreende a tarefa de identificação de requisitos que estão contemplados no sistema atual e que poderão ser reaproveitados no novo sistema. Ao se adotar esta técnica, economizam-se tempo e esforço, pois requisitos reutilizados já foram analisados e validados em outros sistemas ganhando, com isto, produtividade e diminuição dos custos de elicitação de requisitos.

Reuso de requisitos é uma abordagem para usar sistematicamente os requisitos existentes visando reduzir o esforço geral de desenvolvimento do ciclo de vida do software. O uso sistemático de requisitos para desenvolver softwares requer duas ações específicas. A primeira compreende a definição de uma maneira adequada de modelar e armazenar as especificações para reuso na fase de desenvolvimento. A segunda envolve a definição de um processo para comparar e adaptar os requisitos reusáveis na fase de desenvolvimento do novo software [LAGUNA+2001].

6.10.3 Áreas típicas de aplicação da técnica e quando pode ser usada

Existem algumas situações em que a reutilização dos requisitos pode ser aplicada [KOTONYA+1998]:

- Requisitos que especifiquem informações sobre um domínio de aplicação podem ser compartilhados por todos os sistemas no mesmo domínio. Numa mesma área

de aplicação, apenas um baixo percentual de requisitos de um novo sistema é realmente novo e exclusivo dele;

- Requisitos que especifiquem características de interface podem ser reutilizados na construção de diferentes sistemas. Dentro de um mesmo contexto organizacional os usuários já estão acostumados com uma interface para todos os sistemas. A reutilização proporciona, portanto, uma consistência de estilos entre as aplicações e isto faz com o que os erros dos usuários sejam menores ao navegar de um sistema para outro; e
- Requisitos corporativos, que refletem políticas da organização, tais como políticas de segurança, podem ser reutilizados em todos os sistemas da organização.

Segundo [KOTONYA+1998], para determinados sistemas, aproximadamente 50% dos requisitos pertencem as categorias acima descritas e apresentam grande potencial de reuso e de diminuição de custos no seu desenvolvimento.

6.10.4 Benefícios/Vantagens

- Reuso de requisitos eleva a qualidade, pois os requisitos já estão validados;
- Aumenta a produtividade da equipe de desenvolvimento, pois elimina esforços de elicitação, análise e especificação;
- Reuso em sistema economiza tempo, esforço e dinheiro, principalmente para sistemas similares;
- Propicia uma rápida elicitação a um custo relativamente baixo, com uma conseqüente entrega mais rápida do produto final;
- Há redução no risco produzir requisitos difíceis de implementar [SOMMERVILLE+1997]; e
- Reuso é ainda um processo informal, contudo um reuso de requisitos de forma mais sistemática economizaria mais esforços [KOTONYA+1998].

6.10.5 Limitações/Desvantagens

- Quando a reutilização é aproveitada num contexto diferente do originalmente desenvolvido podem surgir problemas inesperados [KOTONYA+1998];
- Encontrar os requisitos reutilizáveis nos sistemas atuais;
- Pode envolver direitos de propriedade (intelectual/legal/contratos); e
- Pode haver dificuldade na reutilização dos requisitos sem fazer algum tipo de modificação.

6.10.6 O que é necessário

O desenvolvedor deve conhecer ou pesquisar os sistemas existentes para verificar se existe algum requisito que possa atender o novo sistema e deve, ao mesmo tempo, ter a capacidade de aproveitar as análises anteriores.

As especificações validadas devem estar prontamente disponíveis.

6.10.7 Processo

Podemos ter duas situações de implementação de reuso de requisitos segundo [SOMMERVILLE+1997]:

Uso direto: Neste caso, os requisitos de um sistema são reaproveitados por outro, com um mínimo de mudanças. O uso direto dos requisitos pode ser executado através dos seguintes passos:

- Identificar características comuns entre o novo e o sistema atual;
- Identificar os requisitos no sistema atual que são potencialmente reutilizáveis e que são relevantes para ter suas características identificadas;
- Avaliar se os requisitos potencialmente reutilizáveis também são válidos no sistema a ser desenvolvido; e
- Verificar com os usuários se estes requisitos vão ao encontro de suas necessidades.

Uso indireto: Usa os requisitos já existentes como base de discussão com os usuários sobre requisitos novos ou específicos. Alguns passos podem ser seguidos para implementar o reuso indireto de requisitos:

- Identificar os requisitos que têm alguma proximidade com o novo sistema;
- Apresentar estes requisitos para os usuários e explicar o seu significado;
- Perguntar aos usuários se estes requisitos são adequados ou não;
- Reescrever os requisitos de acordo com as sugestões dadas pelos usuários; e
- Repetir o processo até que os usuários estejam satisfeitos.

6.10.8 Procedimentos Práticos

- No desenvolvimento de um novo sistema, o desenvolvedor deve, sempre que possível, reusar requisitos previamente definidos para a mesma área de domínio;
- O processo de reuso de requisitos só deve ser aplicado quando a equipe de desenvolvedores estiver suficientemente treinada;
- Requisitos podem ser reusados com ou sem modificação;
- Requisitos individuais de um domínio são mais prováveis de ser reusados do que especificações completas de requisitos; e
- A abordagem do uso indireto pode ser eficientemente combinada com a técnica baseada em cenários. É muito mais simples de implementar do que o uso direto; pode haver, por exemplo, diferenças de domínio sutis e importantes entre os sistemas ou algum tipo de confidencialidade [SOMMERVILLE+1997].

Alguns pontos devem ser levados em consideração para uma boa implementação do reuso de requisitos [SOMMERVILLE+1997]:

- Como representar os requisitos;
- Como organizar uma biblioteca de requisitos reutilizáveis;
- Como recuperar os requisitos apropriados; e
- Como customizar os requisitos recuperados.

6.11 Metodologia *Soft Systems*

6.11.1 A classificação na taxonomia

/ Representativo / Informal / Mapeamento / Sócio-organizacional / Mista / Análise / Alto / Ser neutro / Alto / Ambos / Largura / Alta/

6.11.2 Descrição da técnica

Métodos estruturados de análise de requisitos, como análise estruturada e métodos orientados a objetos, dentre outros, não são particularmente úteis nas fases iniciais da análise, quando o domínio da aplicação, o problema e os requisitos organizacionais devem ser compreendidos. Eles são baseados em modelos ou métodos *hard* de sistemas, tais como Modelo Entidade Relacionamento, Diagramas de Fluxo de Dados e outros. Estes modelos são inflexíveis, têm seu uso limitado e estão voltados para sistemas automatizados. Por

outro lado, modelos ou métodos *soft* consistem na construção de modelos menos formais de um sistema completo, considerando três aspectos-chave: o sistema automatizado, as pessoas e a organização. Vários métodos foram desenvolvidos nesta linha e, dentre eles, a Metodologia *Soft Systems* [KOTONYA+1998].

A Metodologia *Soft Sytems* foi originalmente desenvolvida por Peter Checkland [KOTONYA+1998] como uma abordagem sistemática para resolver problemas. Ela fornece um conjunto de diretrizes para analisar e examinar uma organização a fim de determinar em quais pontos os problemas existentes podem ser tratados de forma mais produtiva. Esta metodologia indica uma maneira de lidar com situações de problema em que há um alto componente social, político e humano envolvido. Neste processo, surgem não somente os requisitos, mas também podem surgir os objetivos organizacionais, a compreensão do ambiente, as estruturas de tarefas, idéias para mudar a estrutura da organização, dentre outras.

MSS não foi projetada especificamente para ser uma técnica de elicitação de requisitos para sistemas computadorizados; como já mencionado ela foi desenvolvida para ajudar no entendimento dos problemas dentro de uma organização e está preocupada com sistemas técnico-sociais num sentido amplo, incluindo pessoas, procedimentos, políticas, software, hardware e outros. Esta ampla perspectiva a torna utilizável na elicitação de requisitos combinada com outras técnicas. A essência da MSS é o reconhecimento de que sistemas estão enraizados num contexto humano e organizacional mais amplo. Ela oferece meios para compreender os requisitos abstratos de sistemas através da análise do contexto organizacional, do entendimento do problema a ser resolvido e do conhecimento dos sistemas existentes. Foi uma das primeiras técnicas a introduzir a noção de ponto de vista, ou perspectivas, ou seja, procura considerar uma organização (pessoas, máquinas, políticas, etc.) como um todo. Diferentes pontos de vista apontam para diferentes percepções do problema e da solução dos requisitos [KOTONYA+1998].

A idéia básica na MSS é começar com um exame da situação do problema no mundo real, construir um modelo de melhorias da situação e compará-lo com o modelo do mundo real. MSS envolve, então, a comparação entre a situação atual e a situação ideal.

6.11.3 Áreas típicas de aplicação da técnica e quando pode ser usada

MSS permite a identificação de requisitos a partir de um amplo contexto humano e organizacional em que o sistema funciona. Ela se encaixa melhor em situações nas quais o problema a ser resolvido pelo novo sistema tem, inicialmente, uma vaga definição.

Embora ela também possa ser usada para analisar qualquer problema ou situação, é

mais apropriada para a análise dos sistemas que contêm componentes humanos e tecnológicos [KASSABOVA+2000].

MSS e outras metodologias *soft systems* não são técnicas para a elicitação detalhada de requisitos; elas são mais efetivas para ajudar a compreender o problema, compreender a situação organizacional no qual o problema existe e as limitações da solução do problema. Elas são particularmente úteis quando existem incertezas sobre que tipo de sistema é realmente necessário num determinado contexto. Produzem requisitos abstratos que necessitam ser refinados por outras técnicas de elicitação [KOTONYA+1998].

6.11.4 Benefícios/Vantagens

Alguns benefícios são:

- O processo é independente da tecnologia que será empregada na implementação. Conseqüentemente, há maior flexibilidade no processo de decisão por não ser influenciado pela tecnologia;
- Devido à ampla estrutura da metodologia que pode ser aplicada em diversas situações, não somente ao desenvolvimento de sistemas, é mais fácil que os usuários não técnicos a compreendam;
- A aplicação da metodologia pode trazer benefícios adicionais para a organização, aumentado a compreensão dos fatores que a afetam e do seu desempenho, por exemplo; e
- Contribui para uma reflexão sobre a situação atual da organização e para identificar o potencial de mudança da organização.

6.11.5 Limitações/Desvantagens

- Potenciais problemas com a metodologia podem ser evitados ao assumir que a MSS pode ser adaptada para as necessidades da organização no sentido de resolver melhor qualquer problema. Entretanto, esta estrutura geral pode ser considerada como uma limitação, porque ela pode não oferecer as diretrizes suficientes para se iniciar com a metodologia e usá-la durante a elicitação de requisitos [CHRISTEL+1992];
- Deve-se tomar cuidado para que a discussão não se desvie para assuntos não técnicos, fora do escopo do sistema ou da organização; e
- O processo de consultar pessoas para obter uma melhor compreensão sobre as organizações pode ser longo, aumentando o tempo de análise.

6.11.6 O que é necessário

Implica em envolver os usuários do sistema e os desenvolvedores num debate sobre o que são soluções desejáveis ao invés de impor uma solução que os desenvolvedores acreditam ser a correta. Para tanto, exige um facilitador para garantir pontos de vistas imparciais.

Essencialmente, a abordagem MSS visa construir um modelo conceitual idealizado a partir do ambiente do problema. Com esta idealização são provocados debates para determinar quais mudanças são possíveis [KOTONYA+1998].

6.11.7 Processo

O processo clássico da MSS é composto por sete fases que permitem estruturar o problema, analisando-o sob duas perspectivas:

- uma relacionada às atividades do mundo real onde as pessoas exercem suas atividades e se relacionam com este mundo; e
- a outra relacionada ao pensamento sistêmico, no qual o analista usa do seu conhecimento sobre sistemas para entender o problema do usuário, existente neste mundo real.

Estas fases não são, necessariamente, desempenhadas progressivamente na ordem descrita. Não são "receitas de bolo" estáticas. Segundo [KOTONYA+1998] as fases são:

1. **Avaliação da situação do problema**

Consiste em descobrir uma situação problemática e desestruturada na organização que precisa ser resolvida. A solução do problema não existe ou é vaga e a informação tem que ser extraída de várias fontes. É necessário, então, descobrir as pessoas envolvidas, as percepções destas pessoas sobre a situação, os processos existentes, etc. para se obter uma compreensão geral do problema e dos processos da organização.

2. **Descrição e expressão da situação do problema**

Nesta fase são descritos os problemas e as percepções do mundo real a respeito da situação, para que se tenha uma definição clara e uma maior compreensão da realidade que se deseja transformar. A situação do problema é descrita utilizando alguma técnica de análise para mostrar todos os elementos do sistema e os seus relacionamentos. Uma das

principais técnicas utilizadas é a *rich picture*[9], que é simplesmente um esboço ou retrato, desenvolvido colaborativamente, das características da organização e/ou da situação sob estudo, de modo que seja compreensível para todos os envolvidos no estudo. Entretanto, ela não é somente um esquema ou diagrama; o seu valor está no processo colaborativo para sua produção, que é projetado para encorajar uma compreensão articulada entre os participantes da situação do problema. A representação da *rich picture* não depende somente de um conjunto limitado de símbolos para descrever a situação: pode-se usar qualquer figura apropriada, tais como mapas, logotipos, desenhos esquemáticos, etc.

3. Definição abstrata do sistema a partir de pontos de vistas selecionados

Na terceira fase são identificadas as "definições-raiz dos sistemas", isto é, os objetivos essenciais da situação do problema. A definição-raiz é estabelecida a partir de um ponto de vista particular da visão de mundo, ou seja, uma interpretação das percepções e valores dos envolvidos sobre como o sistema deve ser para desempenhar as suas funções.

Uma definição-raíz contém ou implica em seis partes/tipos de informações representadas pelos elementos do mnemônico inglês CATWOE [CHRISTEL+1992]:

- cliente (***Customer***), quem se beneficia do sistema;
- ator (***Actor***), quem está envolvido;
- transformação (***Transformation***), a transformação das entradas em saídas;
- visão de mundo (do alemão, ***Weltanschauung*** ou *word view*), a percepção a partir da qual a definição foi produzida;
- proprietário (***Owner***) , detentores do problema; e
- restrições do ambiente (***Environment constraints***), restrições ambientais que podem afetar o sistema.

A identificação dos elementos serve para averiguar do que consiste o sistema realmente.

4. Modelagem conceitual do sistema

Nesta etapa devem-se modelar os sistemas identificados na etapa anterior estritamente de acordo com as respectivas definições-raíz. A elaboração do modelo conceitual consiste na descrição dos meios necessários para que o sistema represente a situação desejada. O modelo construído é chamado de "modelo das atividades humanas" e, como regra geral, as atividades do modelo devem ser baseadas nos verbos contidos na

[9] Pode ser definida como "uma rica descrição do mundo real"

definição-raiz. Deve existir pelo menos um modelo para cada definição raiz.

5. Comparação do modelo conceitual com o mundo real

O modelo conceitual produzido é comparado com a situação atual existente no mundo real (ou na realidade), descrita na fase dois. Para cada atividade no modelo conceitual, devem ser feita perguntas como "esta é uma atividade que existe no mundo real?", "como isto pode ser medido?". Consiste, essencialmente, na utilização do modelo conceitual para questionar a situação do mundo real e, assim, revelar áreas onde as mudanças podem ser feitas.

6. Identificação de mudanças

A comparação feita na fase anterior serve de base para a discussão sobre as mudanças passíveis de ser implementadas para transformar a realidade. As possíveis mudanças no sistema são identificadas e analisadas entre os envolvidos, que fazem os seus diagnósticos e apresentam sugestões para aproximar a situação problemática do pensamento sistêmico. Uma lista contendo as mudanças acordadas é gerada. Nesta análise, é levado em conta o que é desejável e o que é viável de se introduzir na organização. Também é analisado se a mudança é realista em termos políticos e de custos.

7. Recomendações para ação e implementação das mudanças

As mudanças identificadas são avaliadas e as possíveis implementações são apontadas através de um conjunto de ações. Esta é a fase de execução das ações para melhorar a situação do problema.

A figura 9 dá uma visão geral da metodologia MSS.

6.11.8 Procedimentos Práticos

- MSS deve examinar o ambiente dentro do qual o sistema funciona;
- Não diminuir o escopo da investigação;
- Deve-se considerar porque o sistema existe, ou deveria existir e como ele se ajusta ao ambiente; e
- Uma vez proposto o sistema, analisar as diferenças entre o sistema atual e o proposto;
- MSS usa modelos conceituais para comparar os estados desejados com os estados reais. Para [McPHEE2001] o processo clássico da MSS visa a uma mudança revolucionária entre o sistema atual e o sistema futuro, isto é, determina o estado

final e como atingi-lo, ao invés de analisar as melhorias incrementais do estado atual. Já [CHRISTEL+1992] argumenta que esta abordagem funciona bem para o desenvolvimento de um novo sistema. Entretanto, em outros cenários de elicitação, em que a revisão de um sistema existente deve ser feita, há um custo significativo associado ao redesenho radical do sistema atual. Incorporar as limitações do sistema existente na definição dos requisitos seria uma abordagem com melhor relação custo/benefício e potencialmente mais atrativa do que trabalhar a partir de um estado desejado e tentar alcançá-lo.

[CHRISTEL+1992] continua argumentando que é difícil, contudo, identificar:

- quando as limitações do sistema existente devem ser mantidas; e
- quando uma evolução para um estado ideal ou desejado deve ser almejada independentemente do sistema existente.

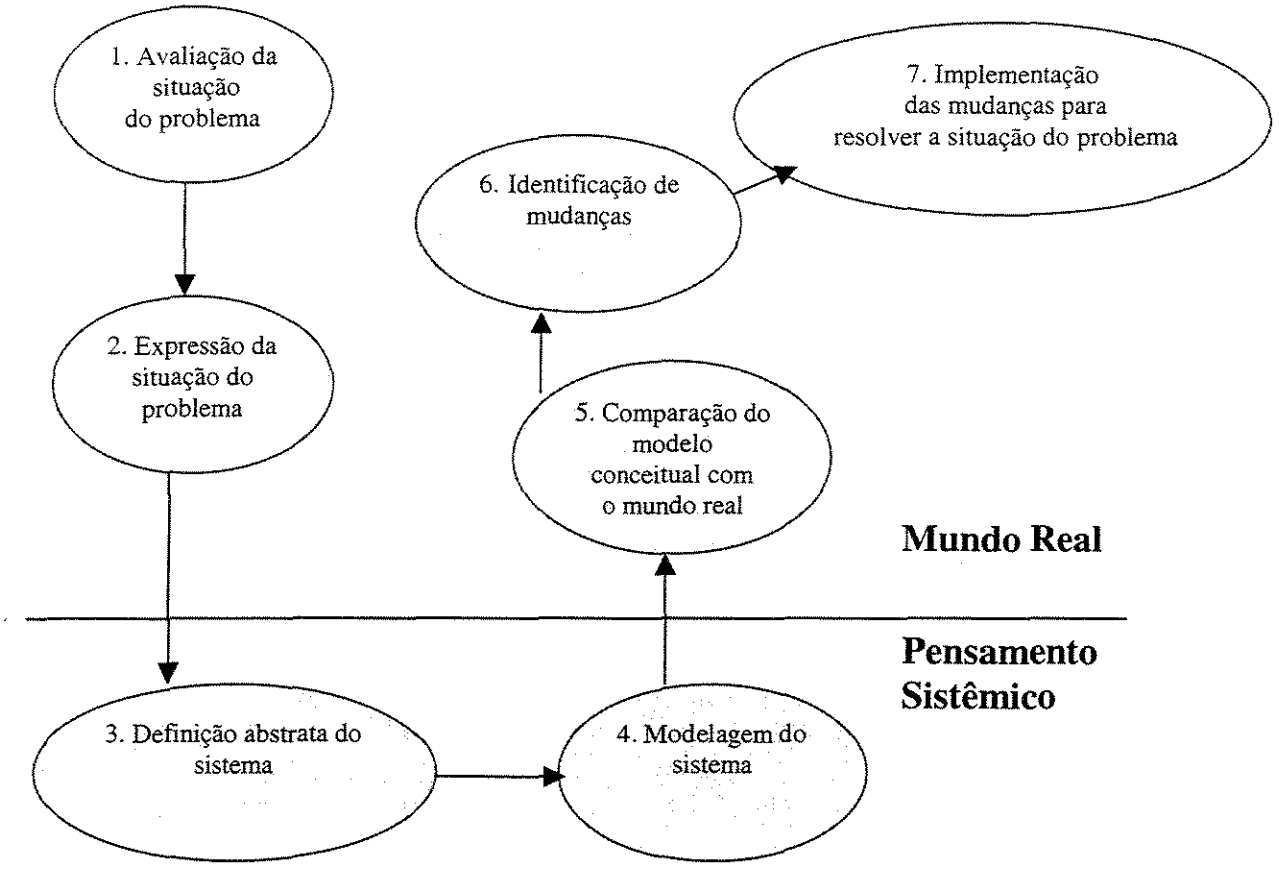

Figura 9 - Visão geral da metodologia MSS

6.12 Tabelas com Resumo da Taxonomia Proposta

Na tabela 14 é mostrado o esquema de classificação facetado proposto. Cada técnica está classificada de acordo com os termos propostos nas facetas. Na tabela 15 é apresentado um resumo da classificação das técnicas de elicitação de requisitos. A duas tabelas fornecem uma visão geral da taxonomia e podem servir como um guia rápido de consulta.

6.13 Roteiro para Escolha das Técnicas de Elicitação

Para começar um processo de elicitação é necessário que a equipe de desenvolvimento tenha um mínimo de conhecimento sobre o domínio do problema sendo necessário empregar alguma técnica para o levantamento inicial. É importante identificar as facetas mais importantes do problema e escolher as técnicas apropriadas. Este primeiro levantamento tem que ser rápido e para tanto pode-se utilizar a entrevista não estruturada para conhecer o sistema que se deseja construir e identificar quem serão os usuários principais. Em seguida, pode-se fazer uma sessão de *brainstorming* com os envolvidos e com isto, é possível ter uma idéia de suas necessidades destas pessoas (assume-se que a equipe saiba aplicar estas duas técnicas). Após coletar as informações iniciais é possível determinar algumas informações ou facetas:

- o papel exercido pelos usuários;
- as fontes de obtenção dos requisitos; e
- se é um sistema novo ou manutenção no sistema atual (finalidade da informação coletada).

Em seguida, a equipe deve se reunir para analisar outras facetas:

- categorias de aplicação, fazendo um levantamento estruturado ou empregando técnicas de observação, por exemplo;
- os aspectos organizacionais encontrados no levantamento inicial para decidir o tipo de abordagem a ser empregada na extração de requisitos; e
- quanto tempo os usuários estarão disponíveis, para definir o nível de participação do usuário.

Com estes tópicos definidos, algumas técnicas já se destacam. Para concluir a escolha, a equipe deverá continuar analisando os seguintes aspectos:

- Das técnicas candidatas, quais delas são de conhecimento da equipe. Com isto pode-se determinar o nível de treinamento dos desenvolvedores para empregar a técnica;
- Descobrir se os membros da equipe possuem as habilidades exigidas no uso de cada técnica;
- Estudar se na extração dos requisitos é mais importante utilizar uma técnica que capture informações em profundidade ou largura;
- Para encurtar o tempo total da fase de ER, a equipe pode escolher uma técnica que também pode ser empregada em outra fase, como por exemplo, usar a mesma técnica para elicitar e validar os requisitos; e
- Avaliar o tempo e esforço demandado pela técnica. Tempo e recursos geralmente limitam a quantidade de sessões de trabalho que podem ser empreendidas.

Como já dito anteriormente, com o uso de uma só técnica não é possível descobrir satisfatoriamente todos os requisitos dos usuários e do sistema. Então, este processo de escolha de técnicas deve resultar em pelo menos duas técnicas selecionadas. Durante a elicitação, os analistas devem rever a técnica que está sendo utilizada e decidir se vale a pena continuar com ela, utilizando os passos acima descritos.

Tabela 14 - Esquema de classificação facetado proposto para técnicas de elicitação de requisitos

{Papel exercido pelo usuário}	{Formalidade}	{Categorias de aplicação}	{Abordagens organizacionais}	{Fontes de obtenção dos requisitos}	{Técnicas aplicáveis à diferentes fases da ER}	{Nível do Treinamento do desenvolvedor na técnica}	{Habilidades exigidas do desenvolvedor}	{Custo da técnica}	{Finalidade da informação coletada}	{Quantidade informação coletada}	{Nível d participação do usuário}
onsultivo ıtrevistas *rainstorming* uestionários nálise de ocumentos **epresentativo** AD bservação ISS **ecisório** **polo geral** rototipação enários asos de uso euso de quisitos	**Formal** **Semi-formal** **Informal** Entrevistas *Brainstorming* JAD Questionários Observação Análise de documentos Prototipação Cenários Casos de uso Reuso de Requisitos MSS	**Observação** Observação Prototipação **Levantamento não-estruturado** Entrevistas *Brainstorming* **Mapeamento** Análise de documentos MSS **Levantamento estruturado** JAD Questionários Cenários Casos de uso Reuso de requisitos	**Tecnológica** Questionários Reuso de requisitos **Sócio-Organizacional** *Brainstorming* MSS **Mista** Entrevistas JAD Observação Análise de Documentos Prototipação Cenários Casos de uso	**Indivíduo** Questionários **Grupo** *Brainstorming* JAD **Mista** Entrevistas Prototipação Cenários Casos de uso MSS **Documentos** Análise de Documentos Reuso de requisitos **Observação** Observação	**Entendimento do domínio** Análise Cenários MSS **Especificação e Documentação** Análise de documentos Casos de uso Reuso de requisitos **Validação** Entrevistas JAD Observação Prototipação **Gerencia-mento** **Controle de qualidade**	**Baixo** Observação Análise de documentos **Médio** Entrevistas *Brainstorming* Cenários **Alto** JAD Questionários Prototipação Casos de uso Reuso de Requisitos MSS **Domínio Completo**	**Motivador do grupo** *Brainstorming* **Ser neutro** Entrevistas MSS **Registrar eventos** Observação **Apresentar e capturar idéias** JAD **Possuir habilidades analíticas** Análise de documentos **Desenvolvimento de *software* ou construção de modelos** Prototipação **Saber montar projetos e analisar pesquisas** Questionários **Capacidade de redigir ou escrever** Casos de uso **Trabalhar em grupo** Cenários **Aproveitar análises anteriores** Reuso de requisitos **Saber especificar/modelar**	**Baixo** Reuso de requisitos **Médio** Entrevistas *Brainstorming* Questionários Análise de documentos Prototipação Cenários **Alto** JAD Observação Casos de uso MSS	**Sistema novo** Prototipação Reuso de requisitos **Sistema atual** Análise de documentos **Ambos** Entrevistas *Brainstorming* JAD Questionários Observação Cenários Casos de uso MSS	**Profundi-dade** *Entrevistas* JAD Prototipação Reuso de requisitos **Largura** *Brainstorming* Questionários Observação Análise de documentos Cenários Casos de uso MSS	**Baixa** *Brainstorming* Questionários Observação Análise de documentos Reuso de requisitos **Média** Entrevistas **Alta** JAD Prototipação Cenários Casos de uso MSS

Tabela 15 - Resumo da classificação proposta para técnicas de elicitação de requisitos

Técnicas / Facetas	Entrevistas	*Brainstorming*	JAD	Questionários	Observação	Análise de documentos	Prototipação	Cenários	Casos de Uso	Reuso de Requisitos	Metodologia *Soft Systems*
Papel exercido pelo usuário	Consultivo	Consultivo	Represen-tativo	Consultivo	Represen-tativo	Consultivo	Apoio geral	Apoio geral	Apoio geral	Apoio geral	Representativo
Formalidade	Informal	Informal	Informal	Informal	Informal	Informal	Informal	Informal	Informal	Informal	Informal
Categorias de aplicação	Levantamento não estruturado	Levantamento não estruturado	Levantamen-to estruturado	Levantamento estruturado	Observação	Mapeamento	Observação	Levantamento estruturado	Levan-tamento estruturado	Levanta-mento estruturado	Mapeamento
Abordagens organizacionais	Mista	Sócio-organizacional	Mista	Tecnológica	Mista	Mista	Mista	Mista	Mista	Tecnológica	Sócio-organizacional
Fontes de obtenção dos requisitos	Mista	Grupo	Grupo	Indivíduo	Observação	Documentos	Mista	Mista	Mista	Documentos	Mista
Técnicas aplicáveis à diferentes fases da ER	Validação	Null	Validação	Null	Validação	Especificação e documentação	Validação	Análise	Especifi-cação	Especifi-cação	Análise
Treinamento do desenvolvedor na técnica	Médio	Médio	Alto	Alto	Baixo	Baixo	Alto	Médio	Alto	Alto	Alto
Habilidades exigidas do desenvolvedor	Ser neutro	Motivador do grupo	Apresentar e capturar idéias	Saber montar projetos e analisar pesquisas	Registrar eventos	Possuir habilidades analíticas	Desenvolvi-mento de software ou construção de modelos	Trabalhar em grupo	Capacidade de escrever ou redigir	Aproveitar análises anteriores	Ser neutro
Custo da técnica	Médio	Médio	Alto	Médio	Alto	Médio	Médio	Médio	Alto	Baixo	Alto
Finalidade da informação coletada	Ambos	Ambos	Ambos	Ambos	Ambos	Sistema atual	Sistema novo	Ambos	Ambos	Sistema novo	Ambos
Quantidade informação coletada	Profundidade	Largura	Profundidade	Largura	Largura	Largura	Profundidade	Largura	Largura	Profundidade	Largura
Nível de participação do usuário	Média	Baixa	Alta	Baixa	Baixa	Baixa	Alta	Alta	Alta	Baixa	Alta

Capítulo 7

Conclusões

Neste trabalho propusemos uma maneira de classificar as técnicas de elicitação de requisitos com o objetivo de facilitar o processo de elicitação.

7.1 Considerações

No momento de fazer a elicitação, os desenvolvedores, desempenhando o papel de engenheiros de requisitos, devem decidir quais técnicas serão utilizadas para extrair os requisitos do sistema. Este mecanismo pode ser diferente de organização para organização, pois o grau de conhecimento sobre o uso das técnicas pode variar devido a fatores como o histórico de uso das técnicas na organização, o conhecimento de cada técnica por parte dos engenheiros de requisitos, o uso ou não de uma ferramenta de auxílio, o grau de maturidade no uso da técnica, o esforço e tempo demandados pelo uso da técnica. Enfim, muitos aspectos têm que ser levados em consideração. Às vezes uma técnica não é escolhida simplesmente pelo fato de não se conhecer detalhes mínimos sobre a sua aplicação; assim são mantidas as técnicas tradicionalmente usadas, não havendo a natural reciclagem. Outro ponto importante a ser considerado é a necessidade dos engenheiros de requisitos com pouca experiência na elicitação de requisitos ter parâmetros que possam ajudá-los na escolha das técnicas corretas.

Propor uma taxonomia através do uso da classificação facetada não é uma tarefa trivial, visto que a definição das facetas é muito subjetiva; pode haver divergência de opinião com relação à classificação de uma técnica em um termo de uma faceta, assim como omissão de facetas ou de termos. Também pode haver dúvida quanto à associação da técnica a mais de um termo da faceta, por exemplo, a habilidade requerida do desenvolvedor para elaborar bons casos de uso; obviamente ele tem que ter a capacidade de saber entrevistar e conduzir reuniões; porém, escolhemos "Capacidade de redigir ou escrever" como a mais significativa das habilidades, pois os casos de uso devem ser bem escritos para que sejam facilmente compreendidos. Também não foi possível preencher todos os termos das facetas. Por exemplo, o termo "Decisório" da faceta "Papel Exercido pelo Usuário" e o termo "Formal" da faceta "Formalidade" não tiveram nenhuma técnica associada. Esse fato se deve, principalmente, ao número reduzido de técnicas estudadas.

Mesmo com estes problemas, entendemos que a criação de uma taxonomia para uma melhor compreensão da abrangência das técnicas de elicitação proporciona melhores condições de escolha da técnica mais apropriada a ser aplicada no processo de elicitação.

7.2 Trabalhos futuros

O objetivo de escolher as técnicas adequadas para o processo de elicitação é garantir melhorias no processo da ER como um todo, assegurando uma fase de projeto com o menor número de problemas possíveis e, para tal, é necessário criar meios para medir ou analisar este processo. Uma maneira de medir o uso das técnicas na organização é a criação de algum indicador, métrica ou parâmetro. Sem a definição de critérios objetivos fica difícil estabelecer parâmetros que garantam eficiência na utilização da técnica. Sem uma análise estatística fica difícil saber se está havendo ou não melhorias no processo através do uso da técnica adequada. A criação da métrica de avaliação pode ser tratada em trabalhos futuros.

Um dos principais desafios da ER nos próximos anos, segundo [NUSEIBEH+2000], é a reutilização de requisitos. Trabalhos futuros devem ser desenvolvidos no sentido de criar modelos de referências de requisitos nos domínios de aplicação da organização, de maneira que se possa reduzir drasticamente o esforço de desenvolvimentos de novos requisitos. Um bom processo de elicitação deve ser capaz de produzir uma boa especificação e documentação de requisitos para facilitar a recuperação dos requisitos nos próximos projetos. Neste caso também o desenvolvedor deve ter discernimento para num primeiro momento saber selecionar a técnica que produza uma boa documentação e num segundo momento saber usar a técnica de reuso de requisitos.

Trabalhos futuros poderiam também abordar o desenvolvimento e aprimoramento de técnicas de elicitação que procuram estudar os aspectos humanos e sociais que envolvem a elicitação de requisitos. Nestes termos, a metodologia MSS é uma técnica que pode ter seu uso ampliado nas organizações através de uma maior divulgação e treinamento.

7.3 Contribuições

A identificação das técnicas mais adequadas a ser aplicadas na elicitação não é uma atividade fácil. Depende de vários fatores, entre eles, o tipo sistema a ser construído, os resultados que se deseja obter ou oferecer ao usuário, as expectativas dos usuários, os recursos computacionais existentes na organização, a disponibilidade dos usuários e de informações, etc. Com a taxonomia proposta, o engenheiro de requisitos poderá descobrir como cada técnica se encaixa no perfil da organização. Fica fácil localizar-se no contexto

organizacional e, assim, a escolha das técnicas pode ser feita com critério e não empiricamente. Através de uma rápida análise das tabelas 14 e 15, o desenvolvedor pode descobrir as características de cada técnica, selecionar a que lhe parece mais apropriada e, em seguida, estudá-la com mais detalhes utilizando o material contido no Capítulo 6.

Um fator importante é o que diz respeito à equipe de desenvolvimento. Ela deve estar em sintonia para que o domínio no uso de uma técnica seja divulgada a todos os membros através de treinamentos e reciclagem. Estes treinamentos podem ser feitos tanto numa organização externa como aplicados pela própria equipe de desenvolvimento. O desenvolvedor pode, e talvez deva, conhecer diversas técnicas, mas também tem que saber quando deve ou não usá-las; deve também saber combinar o uso das técnicas quando necessário. A equipe de desenvolvimento deve ir adquirindo aos poucos o domínio sobre as técnicas, não tentando aprender ou aplicar todas ao mesmo tempo, mas sim ir adquirindo maturidade no uso da técnica.

Acreditamos que a proposta de [KOTONYA+1998] para utilizar um modelo espiral é a maneira mais adequada para abordar o processo de ER, pois não é possível obter um conjunto completo de requisitos em uma só interação; usar sempre a mesma técnica nas diversas interações pode tornar o processo de elicitar requisitos enfadonho e repetitivo. Se nas interações seguintes forem utilizadas outras técnicas, o processo pode ficar mais dinâmico, interessante e produtivo. Mais uma vez os desenvolvedores terão que ser capazes de escolher técnicas que melhor se adaptem à cada interação.

Uma contribuição deste trabalho é a descrição das técnicas contida no Capítulo 6. Em nosso entender é uma ajuda para quem tem pouco conhecimento no assunto ou deseja conhecer mais sobre técnicas de elicitação de requisitos.

7.4 Conclusão

Existem diversas técnicas, mas um número reduzido delas são comprovadamente efetivas. A princípio, a equipe de desenvolvimento deve procurar usar técnicas que são conhecidas, familiares e provadas na organização. Entretanto, é sempre interessante inovar buscando novas técnicas que possam realmente trazer bons resultados à organização.

Muito tem sido escrito sobre as técnicas, mas talvez muito pouco seja realmente implementável nos projetos. Não existe uma técnica universal feita sob medida para cada processo de elicitação. Existem técnicas que são mais indicadas ou apropriadas para uma situação específica. A equipe de desenvolvimento, em última instância, deve ter meios para selecionar a que for mais conveniente; fornecer estes meios foi a principal proposta deste trabalho. Com isto, os desenvolvedores poderão definir sistemas que melhor atendam as

necessidades dos usuários, evitando aumento de custos e prolongamento de cronogramas. Neste sentido acreditamos que a nossa proposta oferece a criação de condições e mecanismos para melhorar o processo de ER contribuindo para que as possíveis falhas no desenvolvimento de sistemas fiquem num patamar aceitável e tolerável.

Referências Bibliográficas

[ALENCAR1999] ALENCAR, Fernanda M. R. *Mapeando a Modelagem Organizacional em Especificações Precisas.* Tese de Doutorado. Centro de Informática Universidade Federal de Pernambuco, Recife, 1999.

[BELGAMO+2000] BELGAMO, Anderson; MARTINS, Luiz E. G. *Um Estudo Comparativo sobre as Técnicas de Elicitação de Requisitos do* Software, XIX CTIC (Concurso de Trabalhos de Iniciação Científica) no XX Congresso Brasileiro da Sociedade Brasileira de Computação, Curitiba, 2000

[BRACKETT1990] BRACKETT, John. W. *Software Requirements.* Technical Report SEI-CM-19-1.2, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, USA, 1990. Disponível em http://www.sei.cmu.edu/publications/documents/cms/cm.019.html. Consulta em 10/09/2002.

[BRAY2002] BRAY, Ian K. *An Introduction to Requirements Engineering.* London: Addison Wesley, 2002.

[BUREN+1998] BUREN, Jim V., COOK, David A. *Experiences in the Adoption of Requirements Engineering Technologies.* Crosstalk: The Journal of Defense Software Engineering, 1998. Disponível em http://www.stsc.hill.af.mil/crosstalk/1998/dec/cook.pdf. Consulta em 05/09/2002.

[CARVALHO+2001] CARVALHO, Ariadne M. B. R.; CHIOSSI, Thelma C.S. *Introdução à Engenharia de Software.* Campinas,SP: Editora da Unicamp, 2001.

[CARVALHO+2001a] CARVALHO, Ana E. C.; TAVARES, Helena C.;CASTRO, Jaelson B. *Uma Estratégia para Implantação de uma Gerência de Requisitos Visando a Melhoria dos Processos de software.* Workshop on Requirement Engineering – WER. Buenos Aires, 2001. Disponível em http//www.inf.puc-rio.br/~wer01/Pro-Req-3.pdf. Consulta em 17/09/2002.

[CHRISTEL+1992] CHRISTEL, Michael G.; KANG, Kyo C. *Issues in Requirements Elicitation.* Technical Report CMU/SEI-92-TR-12. Software Engineering Institute, Carnegie Mellon University Pittsburgh, USA, September, 1992. Disponível em http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.012.html. Consulta em 14/09/2002.

[COUPRIE+1997] COUPRIE, Dale; GOODBRAND. Alan, LI. Bin; ZHU. David. *Soft Systems Methodology*. Technical Report, Department of Computer Science, University of Calgary, Canadá,1997. Disponível em http://sern.ucalgary.ca/courses/seng/613/F97/grp4/ssmfinal.html. Consulta em 15/05/2003.

[CYBULSKI1996] CYBULSKI, Jacob L. *The Formal and the Informal in Requirements Engineering*. Technical Report 96/7, Department of Information Systems, The University of Melbourne, November, 1996. Disponível em http://www.dis.unimelb.edu.au/staff/jacob/publications/informal-reqs5CFormal-Informal-IEEE-Fmt.pdf. Consulta em 15/01/2003.

[DAVIS+2000] DAVIS, Alan M.;YOURDON, Ed.; ZWEIG, Ann S. *Requirements Management Made Easy.* PM Network Magazine, pp. 61-63, December, 2000. Disponível em http://www.omni-vista.com. Consulta em 12/08/2002.

[DEBORTOLI1998] De BORTOLI, Lis A. *Um Método de Trabalho para Auxiliar a Definição de Requisitos*, III Semana Acadêmica do Programa de Pós-Graduação em Computação da UFRGS, Porto Alegre, Agosto,1998. Disponível em http://www.inf.ufrgs.br/pos/SemanaAcademica/Semana98/lis.html. Consulta em 15/08/2002.

[FERREIRA1986] FERREIRA, Aurélio B. H. *Novo Dicionário da Língua Portuguesa.* Rio de Janeiro: Editora Nova Fronteira Ltda, 1986.

[FIORINI+1998] FIORINI, Soeli T.; LEITE, Julio C.S.P.; LUCENA, Carlos J.P. *Organizando Processos de Requisitos.* Workshop de Engenharia de Requisitos WER'98, Maringá, Outubro, 1998 Disponível em http://www.inf.puc-rio.br/~wer98/artigos/1.html Consulta em 15/08/2002

[FURLAN1998] FURLAN, José D. *Modelagem de Objetos através da UML-the Unified Modeling Language.* São Paulo: Makron Books, 1998.

[GANE1988] GANE, Chris. *Desenvolvimento Rápido de Sistemas.Tradução de Newton Dias de Vasconcelos.* Rio de Janeiro: LTC - Livros Técnicos e Científicos Ltda., 1988.

[GOGUEN+1993] GOGUEN, Joseph.A., LINDE, Charlotte., *Techniques for requirements elicitation.* In Proceedings of the IEEE International Symposium on Requirements Engineering- RE'93 San Diego-USA, IEEE Computer Society Press, pp. 152-164, 1993. Disponível em http://www.cs.ucsd.edu/users/goguen/pps/rqelit.ps. Consulta em 18/09/2002.

[HAM1998] HAM, Gary A. *Four Roads to Use Case Discovery: There is a Use (and a Case) for Each One.* Crosstalk: The Journal of Defense Software Engineering pp. 17-19, 1998. Disponível em http://www.dimap.ufrn.br/~jair/ES/artigos/FourCases.pdf. Consulta em 05/09/2002.

[HUSSEIN1997] HUSSEIN, Alfred. *Requirements Engineering Notes.* The Department of Computer Science, University of Calgary, Canada,1997. Disponível em http://pages.cpsc.ucalgary.ca/~mildred/SENG/Al-Notes611.1.html. Consulta em 13/09/2002.

[KARLSSON1996] KARLSSON, Joachim. *Software requirements prioritizing.* Proceedings of the 2nd International Conference on Requirements Engineering (ICRE '96), Los Amitos. IEEE CSP, pp. 100-116, 1996. Disponível em ftp://ftp.ida.liu.se/pub/labs/aslab/people/joaka/icre.ps.zip. Consulta em 23/09/2002.

[KASSABOVA+2000] KASSABOVA, Diana; TROUNSON, Rachel. *Applying Soft Systems Methodology for User-Centred Design.* Proceedings of the NACCQ 2000, Wellington, New Zeland, 2000. Disponível em www.naccq.ac.nz/papers/kassabova159.pdf. Consulta em 02 de Junho de 2003.

[KIRAKOWSKI1997] KIRAKOWSKI, J. *Requirements Engineering and Specification in Telematics: Methods for User-Orientated Requirements Specification.* Human Factors Research Group, Cork, Ireland,1997. Disponível em http://www.ejeisa.com/nectar/respect/3.2/. Consulta em 16/03/2003.

[KOTONYA+1998] KOTONYA, Gerald; SOMMERVILLE, Ian. *Requirements Engineering: Processes and Techniques.* Chichester,UK: John Wiley & Sons, 1998.

[KUCHMISTAYA2001] KUCHMISTAYA, Sophia B. *Incorporation of Joint Application Design (JAD) in Systems Requirements Determination*. University of Missouri, Information Systems Analysis Notes, November,2001. Disponível em http://www.umsl.edu/~sauter/analysis/488_f01_papers/kuchmistaya.htm. Consulta em 13/09/2002.

[JACOBSON+1992] JACOBSON, Ivar; CHRISTERSON, Magnus; JONSSON, Patrik; ÖVERGAARD, Gunnar. *Object–Oriented Software Engineering: A Use Case Driven Approach*. Addison–Wesley, 1992.

[LAGUNA+2001] LAGUNA, Miguel A; VILLEGAS, Oscar L. *Requirements Reuse for* Software *Development*. In Fifth IEEE International Symposium on Requirements Engineering, Toronto, Canada, Agosto:2001. Disponível em http://giro.infor.uva.es/docpub/Doc-Workshop.pdf. Consulta em 23/05/2003.

[LEISE2001] LEISE, Fred. *Using Faceted Classification to Assist Indexing*. Key Words, vol 9, no.6, pp. 178-179, Novembro/Dezembro, 2001 Disponível em http://www.contextualanalysis.com/publications-usingfacets.htm. Consulta em 25/01/2003.

[LEITE+2002] LEITE, Julio S. P.; EBERLEIN, Armin. *Agile Requirements Definition: A View from Requirements Engineering*. Proceedings of the International Workshop on Time Constrained Requirements Engineering(TCRE'02), Essen, Alemanha, 2002. Disponível em http://www-di.inf.puc-rio.br/~julio/tcre-site/p4.pdf. Consulta em 23/09/2002.

[LEITE+1999] LEITE, Jair. C.; SOUZA, C. S. *Análise e Modelagem de Usuários e Tarefas em Projeto de Interfaces de Usuário* – Notas de Aula, editado por Jair C Leite, DIMAP-UFRN, Natal, RN, 1999. Disponível em http://www.dimap.ufrn.br/~jair/piu/apostila/cap6.pdf. Consulta em 13/03/2003.

[LOPES2002] LOPES, Paulo S. N. *Uma taxonomia da pesquisa na área de Engenharia de Requisitos*. Dissertação de Mestrado, IME/USP, São Paulo, 2002.

[MAGUIRE1998] MAGUIRE, Martin C. *RESPECT Deliverable D 5.3*. RESPECT User-Centred Requirements Handbook. HUSAT Research Centre, Loughborough,1998. Disponível em http://www.ejeisa.com/nectar/respect/5.3/. Consulta em 10/10/2002.

[MAIDEN+1996] MAIDEN, Neil.A.M; RUGG,Gordon., *ACRE: Selecting Methods For Requirements Acquisition*, Software Engineering Journal vol.11, no.3, pp. 183-192, Maio, 1996. Disponível em http://www.haumer.net/rational/files/SEJ96Paper.pdf. Consulta em 14/11/2002.

[MARTIN+1991] MARTIN, James; McCLURE, Carma. *Técnicas estruturadas e CASE; tradução de Lúcia Faria Silva.* São Paulo: Makron, McGraw-Hill, 1991.

[McPHEE2001] McPHEE, Christopher. *Requirements Engineering for Projects with Critical Time-To-Market.* Dissertação de Mestrado, Department of Electrical and Computer Engineering, University of Calgary, Canadá, 2001.

[MOURA2001] MOURA, Arnaldo V. *Especificação em Z: Uma Introdução.* Campinas, SP: Editora da Unicamp, 2001.

[MURRAY2002] MURRAY, Philip C. *Faceted Classification of Information.* The Knowledge Management Connection Company. 2002. Disponível em http://www.kmconnection.com/DOC100100.htm#why_faceted. Consulta em 10/10/2002.

[NUSEIBEH+2000] NUSEIBEH, Bashar; EASTEBROOK, Steve. *Requirements Engineering: A Roadmap.* The Future of Software Engineering, 22nd International Conference on Software Engineering, pp.35-46, June 2000. Disponível em http://www.cs.ucl.ac.uk/staff/A.Finkelstein/fose/finalnuseibeh.pdf. Consulta em 19/09/2002.

[OBERG+2000] OBERG, Roger; PROBASCO, Leslee; ERICSSON, Maria. *Applying Requirements Management With Use Cases*,Technical Paper TP505, Rational Software Corporation, Cupertino, CA, 2000.

[PASTORE2000] PASTORE, José. *O trabalho na virada do século*, Folha de São Paulo, São Paulo: 2 de Janeiro de 2000, Empregos, p. 9.

[PORTELLA1994] PORTELLA, Cristiano R. R. *Técnicas de prototipação na especificação de requisitos e sua influência na qualidade do software.* Dissertação de Mestrado, Instituto de Informática PUC-Campinas, Campinas, 1994.

[PRESSMAN1995] PRESSMAN, Roger S. *Engenharia de Software.* Tradução José Carlos Barbosa dos Santos. São Paulo: Makron Books, 1995.

[PRIETO-DÍAZ+1987] PRIETO-DÍAZ, Rubén.; FREEMAN, P. *Classifying Software for Reusability*. IEEE Software, v. 4, n. 1, p. 7-16, 1987. Disponível em http://www.cs.jmu.edu/users/prietorx/RubenPubs/publications/ClassSoftReusa.pdf. Consulta em 15/11/2002.

[PRIETO-DÍAZ1991] PRIETO-DÍAZ, Rubén. *Implementing Faceted Classification for Software Reuse*. Communications of the ACM, v.34, pp. 88 – 97. New York, 1991. Disponível em http://www.cs.jmu.edu/users/prietorx/RubenPubs/publications/ImpFacClassSoftReuse PrietoDiaz.doc. Consulta em 15/11/2002.

[PRIETO-DÍAZ2002] PRIETO-DÍAZ, Rubén. *A Faceted Approach to Building Ontologies*. 21st International Conference on Conceptual Modeling-ER2002, Tampere, Finland, October 7-11, 2002. Disponível em http://www.cs.jmu.edu/users/prietorx/RubenPubs/publications/BulidOntologiesRPD-ER2002.doc. Consulta em 15/11/2002.

[RAGHAVAN+1994] RAGHAVAN, Sridhar; ZALENISK, Gregory; FORD, Gary. *Lecture Notes on Requirements Elicitation.* Educational Material CMU/SEI-94-EM-10. Software Engineering Institute, Carnigie Mellon University, Pittsburg, USA, 1994. Disponível em http://www.cs.concordia.ca/~faculty/paquet/teaching/342/SEI2.pdf. Consulta em 14/09/2002.

[REIMER2001] REIMER, Ulrich. *Tutorial on Organizational Memories for Capturing, Sharing and Utilizing Knowledge*. International Conference on Enterprise Information Systems, ICEIS 2001, Setubal, Portugal, Julho, 2001. Disponível em http://research.swisslife.ch/~reimer/OM_Tutorial/index.html. Consulta em 15/11/2002;

[ROCHA1987] ROCHA, Ana R.C. *Análise e Projeto Estruturado de Sistemas.* Rio de Janeiro: Campus., 1987.

[ROCHA2000] ROCHA, Álvaro. *Influência da Maturidade da Função Sistema de Informação na Abordagem à Engenharia de Requisitos.* Tese de Doutorado em Tecnologias e Sistemas de Informação, Universidade do Minho, Guimarães, Portugal, 2000.

[ROCHA+2002] ROCHA, Álvaro, CARVALHO; João A. *Pendor da Condução da Atividade de Engenharia de Requisitos: Modelo para Diagnóstico e Reflexão.* Atas da 3ª Conferência da Associação Portuguesa de Sistemas de Informação, Coimbra, Portugal, 20 a 22 de Novembro de 2002.

[SOMMERVILLE+1997] SOMMERVILLE, Ian; SAWYER, Pete. *Requirements Engineering: A good practice guide*. John Wiley & Sons, Ltd. England:1997

[SPRINGL1998] SPRINGL, Milan. *Requirements Elicitation - Lecture Notes*. Software Engineering Research Network, University Calgary, 1998. Disponível http://sern.ucalgary.ca/~springl/Seng611. Consulta em 14/03/2003.

[STANDISH1995] THE STANDISH GROUP, *Chaos*, Standish Group Report, 1995. Disponível em http://www.standishgroup.com. Consulta em 10/09/2002.

[TORO+2000] TORO, Amador D; JIMÉNEZ, Beatriz B. *Metodología para la Elicitación de Requisitos de Sistemas de Software.* Informe Técnico LSI-2000-10. Facultad de Informática y Estadística Universidad de Sevilla, Outubro, 2000. Disponível em http://www.lsi.us.es/~informes/lsi-2000-10.pdf. Consulta em 24/09/2002.

[TROPE1999] TROPE, Alberto. *Organização Virtual: Impactos do Teletrabalho nas Organizações.* São Paulo: QualityMark Editora, 1999.

[ZANLORENCI+1998] ZANLORENCI, Edna P; BURNETT, Robert C. *Engenharia de Requisitos: Processos e técnicas no contexto organizacional.* Revista Bate Byte 80, Curitiba, Outubro, 1998. Disponível em http://www.pr.gov.br/celepar/celepar/batebyte/edicoes/1998/bb80/engenha.htm. Consulta em 14/08/2002.

[WESSELS2002] WESSELS, David. *Requirements and analysis. Lectures Notes* for *Systems Analysis* - Computing Science Department, Faculty of Science and Technology Malaspina University-College, 2002. Disponível em http://csciun1.mala.bc.ca:8080/~wesselsd/csc375/notes.html. Consulta em 16/09/2002.

[WIEGERS1999a] WIEGERS, Karl E., *Software Requirements:What and Why*. Microsoft Press: 1999. Disponível em http://www.processimpact.com/reqs_book. Consulta em 03/09/2002

[WIEGERS1999b] WIEGERS, Karl E., *Writing Quality Requirements*, Software Development , vol. 7, no. 5. May, 1999. Disponível em http://www.sdmagazine.com/documents/s=758/sdm9905c/9905c.htm. Consulta em 14/08/2002.

[WIEGERS2000a] WIEGERS, Karl E., *10 Requirements Traps to Avoid*, Software Testing & Quality Engineering, vol. 2, no. 1. January-February, 2000. Disponível em http://www.processimpact.com/articles/reqtrap.pdf. Consulta em 14/08/2002

[WIEGERS2000b] WIEGERS, Karl E. *When Telepathy Won't Do:Requirements Engineering Key Practices*. Cutter IT Journal, vol. 13, no. 5. May, 2000. Disponível em http://www.processimpact.com/articles/telepathy.pdf. Consulta em 24/09/2002.