KLEBER ROCHA DE OLIVEIRA

Proposta de um Catálogo de Padrões Aplicados ao Processo de Elicitação de Requisitos Para Software de Gestão Comercial

Tese apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do título de Doutor em Engenharia.

SÃO PAULO

2009

KLEBER ROCHA DE OLIVEIRA

Proposta de um Catálogo de Padrões Aplicados ao Processo de Elicitação de Requisitos Para Software de Gestão Comercial

Tese apresentada à Escola Politécnica da Universidade de São Paulo como parte do programa para obtenção do título de Doutor em Engenharia.

Área de Concentração: Engenharia de Produção

Orientador: Prof. Dr. Mauro de Mesquita Spínola

SÃO PAULO

2009

Este exemplar foi revisado e alterado em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.
São Paulo, 14 de Maio de 2009.

Assinatura do autor

Assinatura do orientador

FICHA CATALOGRÁFICA

Oliveira, Kleber Rocha de
Proposta de um catálogo de padrões aplicável ao processo de elicitação de requisitos para software de gestão comercial / K.R. de Oliveira. -- São Paulo, 2009. Edição Revisada.
158 p.

Tese (Doutorado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Produção.

1. Engenharia de software 2. Tecnologia da informação 3. Software I. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Produção II. t.

AGRADECIMENTOS

Ao professor Mauro de Mesquita Spínola, pela oportunidade, pelos ensinamentos, orientação, confiança e amizade.

Ao professor Marcelo Schneck de Paula Pessoa, pelos conselhos, conhecimento e estímulo.

Aos demais professores doutores membros da banca da defesa final André Leme Fleury e Wilson Yonezawa, em especial a professora doutora Rosana Braga que participou da qualificação e contribuiu com a evolução da pesquisa.

As mulheres da minha vida, Gisele minha dedicada esposa, Carmen minha Mãe, avós Nair e Emília (in memorian), minha tia Dirce, pela paciência, compreensão e incentivo, sendo que cada uma contribuiu de forma diferente e em tempos diferentes.

Meu pai e irmãos, Jair, Wellington e Robson, que me mostraram que existem coisas mais complexas do que se desenvolver uma Tese.

Pelas valiosas amizades que criei na Politécnica da USP, durante as reuniões de GTI e nas aulas do doutorado.

Aos meus Clientes pela compreensão e paciência durante minhas ausências.

Ao pessoal da FIB que sempre acreditaram no meu potencial e respeitaram minhas decisões e abdicação.

A Cristina da POLYCOPY da CAEP pela atenção e apoio, durante a geração dos exemplares.

Em especial ao meu avô Manoel Bento de Oliveira pelo exemplo de vida e dedicação a família.

E finalmente, às minhas maiores riquezas, minha filha Julia e meu filho Pedro.

RESUMO

Esta pesquisa do campo da Engenharia de Software explora a aplicação do conceito de padrões no tratamento dos problemas da fase inicial da produção de software. Essa fase visa à compreensão do problema com objetivo de sugerir uma solução viável ao projeto. A área que estuda esses fenômenos é conhecida como Engenharia de Requisitos, e a fase que compreende o levantamento das necessidades dos usuários e dos sistemas denomina-se "*elicitação*".

No desenvolvimento deste trabalho, é aplicada a pesquisa-ação como método de pesquisa. Foram selecionadas três empresas do ramo comercial em segmentos diferentes, através da técnica de observação e análise das atividades aplicadas na elicitação de requisitos, contidas no processo de construção de software nessas organizações. A abordagem teórica se limitou aos conceitos essenciais da Engenharia de Requisitos, com ênfase na fase de elicitação de requisitos, uma exploração sobre padrões, apresentando suas características e as diversas aplicações na padronização da solução geral para problemas complexos.

Essencialmente, a pesquisa sugere um catálogo de padrões candidatos, aplicável ao processo de elicitação de requisitos. Cada padrão é extraído dos documentos de requisitos construídos com base em estudo de campo realizado. São organizados por tipo de requisitos e organizados em um gabarito proposto pelo pesquisador. Posteriormente, são relacionados de acordo com suas afinidades e preocupações, transformando-os dessa maneira em um catálogo de padrões devido a sua classificação e sumarização.

Por fim, o pesquisador apresenta as conclusões e desenvolve as críticas acerca do catálogo de padrões, sugere melhorias, indica as limitações, e aponta as contribuições relativas à redução da complexidade na execução da atividade de elicitação de requisitos com a possibilidade de se antecipar ao problema que provavelmente o analista ou engenheiro de requisitos irá enfrentar.

Palavras-Chave: Elicitação, Engenharia de Requisitos, Padrões.

ABSTRACT

This research in the Software Engineering field explores the application of the patterns concept in the treatment of initial phase problems in software production. This phase aims at understanding the problem with the objective of suggesting a viable solution to the project. The area that studies those phenomena is known as Requirements Engineering and the phase that comprehends the detection of users needs and the systems needs is called "*elicitation*".

In the development of this work the research-action is applied as research method. Three commercial companies branch were selected in different segments, through the observation technique and analysis of applied activities in requirements elicitation, contained in construction process of software in these organizations. The theoretical approach was limited to essential concepts of Requirements Engineering with emphasis in the phase of requirements elicitation, an exploration on patterns, to show the characteristics and the several applications in patterns of general solution for complex problems.

Essentially, the research suggests a pattern catalog, containing candidates for requirements elicitation process. Each pattern is extracted from requirements documents built based on field studies implemented. They are organized by requirements type in a format proposed by the researcher. Then, they are related to each other according to their similitarities and concerns, transforming them in a pattern catalog due to their classification and summarization.

Finally, the researcher presents the conclusions and develops critics concerning the patterns catalog, suggesting improvements, establishing restrictions, as well as pointing out the relative contributions to the reduction of complexity in the execution of the requirements elicitation activity with possibility of anticipating problems that will be presumably be detected by the analyst or requirements engineer.

Keywords: Elicitation, Requirements Engineering, Patterns.

SUMÁRIO

Lista de Figuras

Lista de Quadros

Lista de Tabelas

CAPÍTULO 1 - INTRODUÇÃO

I know that my work is a drop in the ocean. But without him, the ocean would be smaller (Madre Teresa de Calcutá (Agnes Gonxha Bojaxhiu) (1910 - 1997) -Prêmio Nobel da Paz (1979).

Neste capítulo são apresentados: o contexto da pesquisa, as razões e motivação relacionados ao foco da pesquisa, uma abordagem sobre a importância e necessidade dos requisitos, os objetivos, a definição da metodologia aplicada, e por fim, a estrutura geral do trabalho.

1.1 Contexto

O processo produtivo pertinente à área denominada *Engenharia de Software* fundamenta-se em procedimentos sistematizados compostos de métodos, técnicas, normas e padrões, métricas e a garantia da qualidade do produto e do processo, a qual envolve muitos recursos de ordem tecnológica, financeira, e humana (KOTONYA; SOMMERVILLE, 1998; PRESSMAN, 2002).

Para a viabilização econômica do processo de produção de software, há uma oferta incessante de inovações tecnológicas que dão apoio a essas atividades, embora o conhecimento humano ao que se deve construir, isoladamente, é insuficiente para a elaboração do software (GREENFIELD; SHORT, 2004). Esta definição deverá estar fundamentada em um trabalho inicial de conhecimento do problema, para a proposição das alternativas de solução.

A abordagem sistemática deste conhecimento é apoiada por uma área específica da *Engenharia de Software*, denominada *Engenharia de Requisitos (ER)*[1], e a postura desta é de prover ao *engenheiro de software*, bases técnicas e ferramentas que auxiliem o processo de compreensão e registro dos requisitos que o software deve atender (LEITE *et al.*, 1997).

[1] Daqui para frente, será aplicada a abreviatura ***ER*** para referenciar Engenharia de Requisitos.

Para reduzir a complexidade deste processo, é conveniente a aplicação de um modelo que possa orientar o desafio da compreensão do problema, pois além de padronizar as atividades, melhora a precisão e a qualidade das especificações.

1.2 Motivação

Provavelmente, a primeira vez em que o termo "*Engenharia de Software"* ocorreu foi em uma conferência com esse nome, em 1968, na Alemanha. A conferência foi realizada por uma entidade que, a rigor, não possuía nenhuma ligação com a área: o Comitê de Ciência da **OTAN** (NAUR; RANDELL, 1969), apesar de, na época já haver instituições relacionadas com informática, principalmente como a **ACM**, criada em 1947 e que edita uma revista científica, *Communications of the* **ACM**, desde 1957.

O relatório da conferência da **OTAN** de 1968 e outros documentos produzidos na década de 1970 fornecem o seguinte diagnóstico (RANDELL; BUXTON, 1970):

- ✓ *cronogramas não observados;*
- ✓ *projetos com tantas dificuldades que são abandonados;*
- ✓ *módulos que não operam corretamente quando combinados;*
- ✓ *programas que não fazem exatamente o que era esperado;*
- ✓ *programas tão difíceis de usar que são descartados;*
- ✓ *programas que simplesmente param de funcionar.*

Em 1996, o projeto **ESPITI** realizou uma investigação sobre os principais problemas no desenvolvimento de software em nível europeu. Os resultados indicaram que os maiores problemas estavam relacionados com a elicitação, a especificação, a gestão e a documentação de requisitos (ESPITI, 1996).

Segundo Brooks (1995), a fase de concepção do software é essencial para o sucesso de um projeto, por isso, entender os problemas iniciais para saber o que se deve construir não deve ser ignorado. Os benefícios dessa atividade são percebidos em médio prazo, sendo que são necessários investimentos em curto prazo para sua efetivação. Assim, a atividade é, muitas vezes, tida como uma carga desnecessária à condução do processo. Contudo,

sua não utilização faz com que as economias de curto prazo sejam logo suplantadas pelos custos no longo prazo, verificadas com superação de custo e prazo nos projetos conduzidos (WEINBERG, 1994; PRESSMAN, 2002).

Na *ER*, a atividade que envolve o entendimento do problema e compressão das necessidades é denominada *Elicitação dos Requisitos* (LEITE *et al.*, 1997). Esta atividade antecede os processos de análise e especificação do software a ser implementado e deve reproduzir integralmente os requisitos observados.

Aparentemente simples, a atividade de elicitar requisitos poderá comprometer o projeto, elevando os riscos, caso sua implementação seja feita de forma não padronizada ou se não houver experiência adquirida necessária para executá-la na compreensão dos fatos e entendimento do problema.

Conforme pode ser visto na Figura 1.1, esta fase é a mais exaustiva do processo, pois é onde se busca o conhecimento sobre o domínio da informação e o macrosistema (DAVIS; HICKEY, 2002).

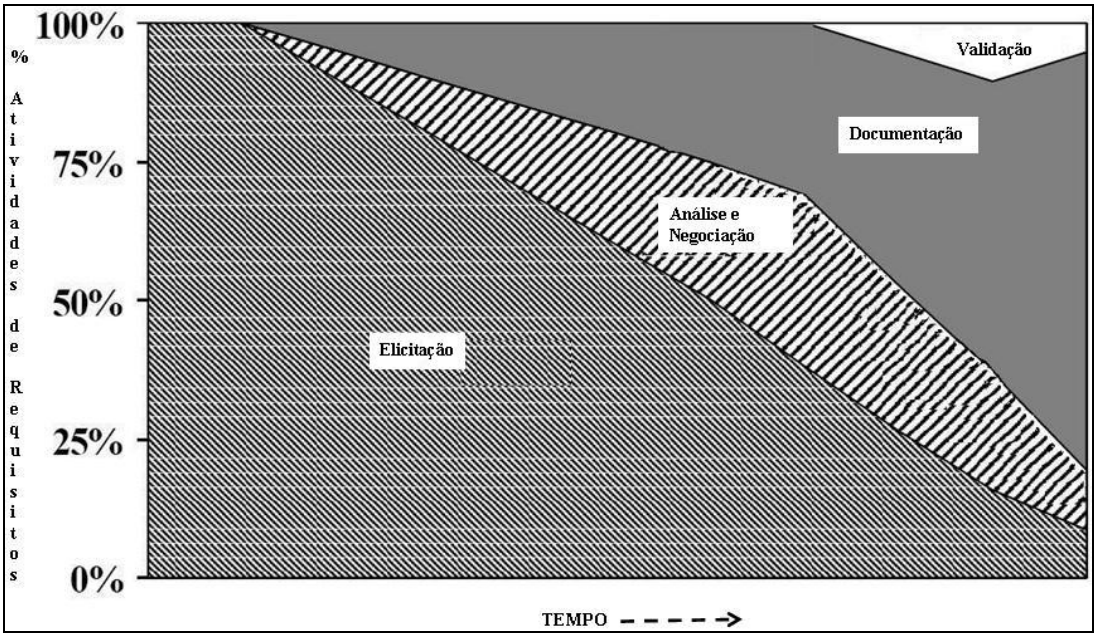

Figura 1.1 – Esforço do processo de requisitos (DAVIS; HICKEY, 2002).

Nota-se na figura 1.1 que o esforço no processo de elicitação é muito maior que nos outros processos que envolvem a análise e negociação, documentação e validação, além de estar presente em todo o ciclo de vida do tratamento dos requisitos.

1.3 Objetivos

A motivação apresentada demonstra surpreendentemente que, após décadas de pesquisas e apesar da evolução e criação de inúmeras ferramentas para construção de software, os processos aplicados apresentam resultados insatisfatórios para os clientes e usuários. Isto indica que os principais problemas que têm dado origem às crises do software residem nas primeiras etapas do desenvolvimento, quando são definidas as características do produto de software a ser desenvolvido. Comprovadamente, o custo das mudanças de requisitos, uma vez entregue o produto, fica entre 60 e 100 vezes superior ao custo representado na mesma mudança, durante as fases iniciais do desenvolvimento (BROOKS, 1995; PRESSMAN, 2002; SOMMERVILLE, 2003), por isso não é de se estranhar que, os projetos nos quais não são determinados corretamente os requisitos e estes mudam freqüentemente durante o desenvolvimento, superam consideravelmente seu custo inicialmente proposto.

Os erros sempre surgem a cada novo projeto e, naturalmente, há o questionamento da possibilidade de se produzir software de qualidade. Essa questão não é simples de ser respondida, mas é possível aventar algumas razões.

Um dos fatores que exerce influência negativa sobre a qualidade de um projeto é a complexidade, que está associada a uma característica bastante simples: o tamanho das especificações e suas relações (SOMMERVILLE; SAWYER, 1997; GABRIEL, 1996; NAUR; RANDELL, 1969). Em programas de computador, os problemas de complexidade e tamanho os tornam graves, em razão das interações entre os diversos componentes do sistema. O aspecto não repetitivo do desenvolvimento de software torna essa atividade mais difícil e, sobretudo, em boa medida imprevisível.

Com isso, surge a questão principal da Tese: *É possível catalogar padrões de requisitos, encontrados em documentos elaborados por analistas, comuns na construção de software para gestão comercial, e que possam contribuir na qualidade e redução da complexidade no processo de identificação dos problemas?* Isto resume a indagação inicial que envolve a

solução que se busca ao problema de levantamento dos requisitos iniciais de um software.

Uma vez definida a questão principal da Tese, é feita a delimitação do objetivo geral deste trabalho, que é c*onstruir um catálogo de padrões para serem aplicados no processo de elicitação de requisitos para software de gestão comercial.*

Para atender ao objetivo principal da pesquisa, são apresentados os seguintes objetivos específicos:

(i) Extrair e identificar os padrões candidatos;
(ii) Adequar os padrões aos tipos de requisitos;
(iii) Estabelecer os relacionamentos entre os padrões;
(iv) Analisar os padrões candidatos sob o ponto de vista da aplicação no processo de elicitação de requisitos e identificação dos problemas.

1.4 Delimitação do Escopo

Como os tipos de software encontrados no mercado são amplos e complexos na sua forma e concepção, é necessário delimitar a categoria deste produto nesta pesquisa. O pesquisador se limitou a estudar documentos de requisitos relacionados a Software para gestão comercial, deixando de fora Software relativos à inteligência artificial, Software embutidos, de missão criticas(?) e demais que não possuem aplicabilidade exclusivamente comercial.

Objetivamente, foram considerados como software para gestão comercial aqueles constituídos por um conjunto de aplicações integradas nas áreas de vendas, compras, recursos humanos, financeira, contabilidade, produção e gestão em geral em empresas comerciais, seja de serviços ou de produtos.

1.5 Processo de Trabalho

Com o intuito de organizar o trabalho de pesquisa a ser realizado, foi definido um processo inicial contendo fases, atividades e resultados esperados.

Fase 1 – **Preparação da Pesquisa**: fase que corresponde à delimitação da área de estudo, coleta e análise das referências bibliográficas, delimitação do tema e estabelecimento dos objetivos, questões e proposições.

Fase 2 – **Estruturação da Pesquisa**: fase de elaboração de um quadro referencial teórico, descrito nesta Tese como um conjunto de pontos de análise, com vista a apoiar o processo de investigação e o delineamento das conclusões. Seleção do método de pesquisa. Construção do roteiro de pesquisa e do protocolo de pesquisa.

Fase 3 – **Execução da Pesquisa**: fase da investigação em si, com coleta de dados em campo através da separação dos documentos de requisitos, análise do material, seleção de conteúdo, classificação dos elementos de acordo com as bases teóricas, bem como à consulta de outras fontes referenciais de apoio, como publicações do setor e outros documentos apresentados pelas empresas selecionadas.

Fase 4 - **Análise dos Resultados**: fase de aprendizado e análise dos dados de forma agregada, delineando os padrões candidatos e sugerindo práticas de melhoria na revisão contínua destes padrões.

Conforme ressalta Croom (2002), o processo de pesquisa é de natureza espiral e não linear, com incursões pela literatura, sucedendo refinamento de proposições e novamente retornando à literatura e às experiências profissionais.

Para uma representação mais realista deste processo, seria necessário representar a realimentação entre as fases e as atividades, e entre estas e os resultados obtidos. Isto tornaria a descrição desnecessariamente complexa, fugindo ao seu propósito principal que é o de oferecer uma visão geral do processo de pesquisa e construção do catálogo proposto nesta Tese.

1.6 Estrutura do Trabalho

Os capítulos estão estruturados de maneira a apresentar gradativamente os conceitos em padrões e *ER*, os processos e as técnicas aplicáveis em suas atividades.

Nos capítulos dois, três e quatro, as informações são de caráter genérico da área de *ER, elicitação de requisitos e padrões*. Neles estão

relatados os resultados das revisões bibliográficas que fundamentam o desenvolvimento do trabalho, além das informações de caráter mais específico, voltado para o conhecimento dos problemas em requisitos.

O capítulo cinco traz a estruturação da pesquisa, onde é dado o embasamento teórico e metodológico do estudo e a definição dos ciclos e fases de construção do catálogo que será adotada pelo pesquisador.

No capítulo seis, são apresentados os passos para criação do catálogo, a descrição dos padrões, definindo suas características, a finalidade e o relacionamento de cada elemento, classificando-os dentro do domínio da linguagem de requisitos, dentro de dois ciclos contínuos e iterativos da pesquisa-ação técnica.

No capítulo sete, é feita a crítica da proposta, avaliadas as dificuldades percebidas pelo pesquisador, assim como os benefícios e limitações dos padrões, e por fim, são transmitidas as conclusões deste estudo.

CAPÍTULO 2 - ENGENHARIA DE REQUISITOS

> *The hardest single part of building a software system is deciding precisely what to build. No other part of the conceptual work is as difficult as establishing the detailed technical requirements, including all the interfaces to people, to machines, and to other software systems. No other part of the work so cripples the resulting system if done wrong. No other part is more difficult to rectify later (F. Brooks, "No Silver Bullet", IEEE Computer, 1987).*

Nesta etapa da fundamentação, estão detalhadas as várias abordagens de tratamento dos requisitos em termos de processos e técnicas aplicáveis. Abordam-se os conceitos fundamentais, a definição e representação dos requisitos, conceitos tratados pela disciplina Engenharia de Requisitos (*ER*). Trata-se da importância do rastreamento de requisitos e da influência dos fatores humano, social e organizacional no descobrimento dos requisitos. Por fim, traz-se o foco para o conhecimento do problema relacionado ao tratamento dos requisitos.

2.1 Introdução

ER é um amplo e multidisciplinar campo de pesquisa, envolvendo ramos das ciências sociais, ciência cognitiva e das ciências exatas, uma vez que é relacionado com a tradução de observações informais em linguagens matemáticas de especificação formal (ZAVE; JACKSON, 1997).

A *ER*, sendo uma subárea da *Engenharia de Software*, estuda o processo de definição dos requisitos que o software deverá atender. A área surgiu em 1993, com a realização do *1st International Symposium on Requirements Engineering*. O processo de definição de requisitos é uma interface entre os desejos e necessidades dos clientes e a posterior implementação desses requisitos em forma de software.

Segundo Siddiqi e Shekaran (1996), desenvolvimentos em *ER*, tal como em desenvolvimento de sistemas, vêm em ondas, e que a próxima onda, a de técnicas e ferramentas, apontaria para o problema e o contexto de desenvolvimento, para atender à incompletude e reconhecer a natureza evolucionária da *ER*.

Segundo Berry e Lawrence (1998), a primeira onda focava sobre escrita de código; a segunda, sobre o desenvolvimento do ciclo de vida no qual a análise de requisitos era a primeira fase; a terceira focava sobre desenvolvimento evolucionário e a implicação de que os requisitos são sempre incompletos. Cada estágio envolve identificação de novos requisitos baseada na experiência dos estágios precedentes. Como resultado, hoje é reconhecido que *ER* tem seu próprio ciclo de vida, embora os debates sejam sobre quais atividades fazem parte dele.

Nota-se que existe muita coisa a discutir e que os estudos e pesquisas estão evoluindo rapidamente. Apesar disso, muitas ferramentas estão disponibilizadas, principalmente no que tange à automatização de captura de requisitos, mensuração de qualidade do processo e do produto, assistência automatizada para resolução de conflitos, técnicas de trazer à tona interações entre requisitos e gerenciamento de mudanças dos requisitos, permitindo a rastreabilidade das ocorrências históricas.

Esforço notável também ocorre na representação da informação, com métodos formais e ferramentas de checagem de precisão de especificações, embora exista pouco desenvolvimento na linha de padrões aplicados a requisitos, ou sua definição ainda é vaga, portanto sendo um campo a ser explorado (ARAUJO, 2005).

2.2 Uma Abordagem para o Descobrimento de Requisitos

Seguindo a abordagem de Zanlorenci (1999), para o melhor entendimento do que vem a ser requisito e qual a sua importância no contexto, é detalhada uma busca recursiva, ou seja, requisitos para o conhecimento dos requisitos de desenvolvimento de software. Pode ser entendida como um auto-exercício para aplicação dos conceitos tratados e apresentados na continuidade desta introdução.

O contexto de descrição de requisitos, conforme ilustrado na Figura 2.1, compreende a base para a abordagem no descobrimento de requisitos. Foram utilizadas figuras geométricas para diferenciar a representação e o significado dos elementos:

- *os octógonos representam os quatro elementos fundamentais, quais sejam, ambiente ou domínio da aplicação, problemas, requisitos e stakeholders;*
- *os retângulos e caixas de diálogos representam as características associadas aos elementos do modelo;*
- *o espiral representa os processos da ER e a aplicação das técnicas;*
- *a prancheta representa o produto resultante, o documento de requisitos.*

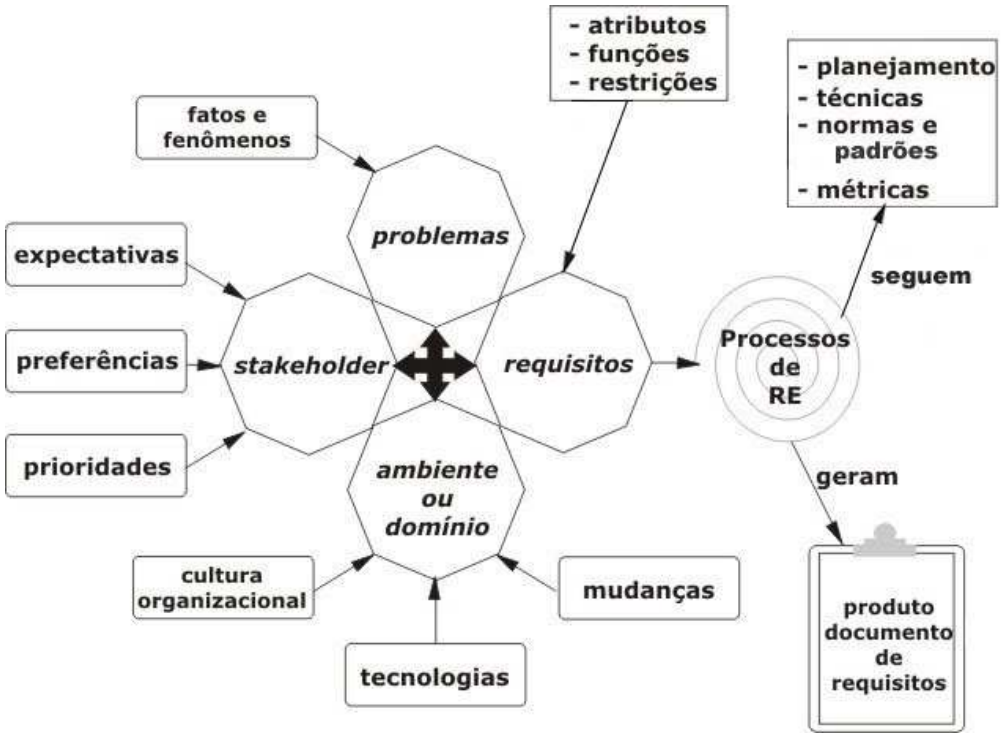

Figura 2.1 - Uma abordagem de requisitos (ZANLORENCI,1999).

a) Elementos Fundamentais da Abordagem de Requisitos

A descrição dos elementos da Figura 2.1 apresentada abrange alguns aspectos tratados no modelo proposto na tese. Os conceitos, características e aplicação dos processos e técnicas da *ER* são tratadas mais detalhadamente neste capítulo.

- **Ambiente ou Domínio da Aplicação**

O ambiente ou domínio da aplicação é onde ocorrem os fenômenos que caracterizam os problemas referentes aos requisitos particulares do cliente (JACKSON, 1995). O primeiro elemento a ser conhecido e representado pelo

engenheiro de requisitos, observando o contexto nos quais os fenômenos estão presentes e interagem:

- *a dinâmica social e organizacional do fator humano;*
- *a universalidade dos fatores econômicos e políticos, tendo como pano de fundo o avanço tecnológico, a competitividade dos fatores produtivos em questões básicas de sobrevivência organizacional do produtor de software, tais como a política de atendimento às mudanças de necessidades ou desejos do contratante do produto ou serviço durante o ciclo de vida do software;*
- *a exigência do mercado na obtenção de produto adequado, no momento certo e a custo reduzido. Isto se aplica necessariamente também às soluções de software particulares das organizações;*
- *os fatores de custos de produção e de benefícios associados aos investimentos em software, hardware e tecnologia de comunicação do ambiente de solução;*
- *o ambiente de contínuas mudanças, que podem ser traduzidas em oportunidades de negócios ou agregação de novos problemas e restrições.*

Portanto, a solução de software atual nem sempre estará adequada a demandas futuras. Conseqüentemente, a mudança será inevitável.

É evidente que satisfazer as necessidades vai depender da vontade de solução do problema pelo contratante, entendido como o responsável pela contratação do produto ou serviço. Nem sempre o produto ou serviço resultante é um novo software. Pode simplesmente ser uma característica adicional, uma substituição para uma versão mais eficaz da solução existente.

- **Problemas**

Um problema é a diferença de algo como desejado em relação a algo como percebido pela fonte de informação (GAUSE; WEINBERG, 1990).

Na área de *Engenharia de Software*, um problema que persiste é visualizar a tecnologia como solução de problema, sem anteriormente focar intensivamente o esforço em definição e entendimento do problema e a negociação de eventuais conflitos de interesses pela solução (JACKSON, 1995).

Para o *engenheiro de requisitos,* é possível identificar algumas causas de dificuldade na produção de software, que se tornam problema no decorrer do processo, por exemplo:

a) Como produzir software no tempo certo, do tamanho exato e a baixo custo, adequado às características de qualidade exigidas para o produto?
b) Como atender os *stakeholders*, tendo como cenário as dificuldades inerentes ao contexto organizacional, tais como:
   - *administração de conflitos de interesses e de poder;*
   - *definição do que é prioritário no interesse da organização;*
   - *delimitação da fronteira de aplicação da solução;*
   - *conhecimento do universo dos stakeholders atual e futuro;*
   - *gerenciamento contínuo da dinâmica dos requisitos para alinhamento às mudanças organizacionais;*
   - *restrições de recursos de produção, quer sejam humanos, financeiros ou tecnológicos;*
   - *como evitar erros típicos na produção de software, relativos a fatos incorretos, omissões, inconsistências e ambigüidades;*
   - *como captar as mudanças no ambiente em relação ao problema, no momento exato da ocorrência dos fatos, cujo impacto imediato poderá ser a revisão da alternativa de solução adotada e o reinício de estudo do problema.*

- **Requisitos**

Requisito é uma declaração descritiva de exigências, escrita do ponto de vista dos *stakeholders*, para a qual serão providos a tecnologia da informação e o compartilhamento de recursos na solução de problemas.

Para o *engenheiro de requisitos*, devem estar bem claros dois aspectos de sua área de atuação: ***definição de requisitos*** e ***condições para a definição de requisitos*** (SHELDON,1992).

Segundo Sheldon (1992), erros em definição de requisitos são os fatores mais comuns dos fracassos de projetos de software, acarretando custos e insatisfação do cliente. Os erros na fase de requisitos são extremamente caros de reparar.

- **Stakeholders**

Stakeholders compreende o conjunto de pessoas ou objetos que, direta ou indiretamente, são afetados pela solução de sistema a ser construída (RYAN, 1998). É para quem o resultado do processo de desenvolvimento de software constitui interesse.

b) Características Associadas ao Ambiente ou Domínio da Aplicação

As características associadas ao ambiente ou domínio da aplicação referem-se aos aspectos culturais, à dinâmica das mudanças e aos impactos tecnológicos que afetam o ambiente organizacional.

- **Cultura Organizacional**

Relaciona às regras e normas que regulamentam a organização, comportamentos, hábitos e costumes. Também pode ser entendida como um conjunto de pressuposições básicas compartilhadas que os grupos de pessoas nela envolvida aprenderam como resolver seus problemas de adaptação externa e integração interna, que tem funcionado suficientemente bem para ser considerada válida (CROZATTI, 1998).

- **Mudanças**

Refere-se à dinâmica social e organizacional do elemento humano como agente de mudança no ambiente (RYAN, 1998).

- **Tecnologias**

Tratam dos avanços tecnológicos e dos impactos sobre o ambiente e a cultura organizacional.

Cabe ao *engenheiro de requisitos* identificar os fatores associados ao ambiente e fazer uso da multidisciplinaridade de conhecimento e domínio de técnicas no tratamento dessas informações (RYAN, 1998).

c) Características Associadas aos Problemas

Referem-se aos fatos ou fenômenos relativos a um contexto observado.

- **Fatos ou Fenômenos**

Um fato é uma verdade simples acerca do mundo (JACKSON, 1995). Um fenômeno refere-se à forma de ver o mundo, depende de interpretação do contexto e do impacto que causa, sob o ponto de vista de quem o interpreta (JACKSON, 1995). O conhecimento de ambos e a identificação de quem os relata são as bases que permitem o entendimento do problema.

Cabe ao *engenheiro de requisitos*, além de conhecer e documentar os fatos ou fenômenos que dizem algo da essência do problema, promover o relacionamento entre o problema e a declaração do requisito, como exigência ou condição a ser observada para solução do problema.

d) Características Associadas aos Requisitos

As características associadas aos requisitos referem-se à funcionalidade do requisito, aos atributos que compõem o produto ou serviço, observadas as restrições limitadoras próprias do ambiente do negócio.

- **Funções**

As funções são ações nas quais os requisitos são declarados. Especificam a produção de algo, a partir de um elemento de entrada e um resultado como produto. Descrevem o que fazer para atender à finalidade proposta. Por exemplo, os requisitos que descrevem os controles de nota fiscal de saídas possuem várias funções, entre elas: emissão da nota fiscal, baixa no estoque, verificação do *status* do cliente, quantidade suficiente em estoque, entre outras.

- **Atributos**

Os atributos são dimensões das características de funcionalidade e de qualidade dos requisitos. Estes devem ser consistentes, confiáveis e completos, com representatividade de pontos de vista das fontes de informação

para que se promova a garantia de qualidade do produto descrito. Por exemplo, na fase de validação de requisitos é analisada a consistência das informações extraídas dos *stakeholders*.

- **Restrições**

As restrições são limitações que delineiam o espaço de solução do problema. Tornam-se critérios de aprovação ou recusa para um produto (GAUSE; WEINBERG, 1990).

Um dos fatores restritivos mais complexos é que a solução de negócio passa muitas vezes por uma necessidade de novos procedimentos de trabalho, de reestruturação organizacional e por mudanças no relacionamento entre clientes e fornecedores. Por exemplo, a descrição da restrição de um requisito funcional poderia ser: a venda somente para clientes que não possuem débitos anteriores, enquanto que a descrição da restrição de um requisito não-funcional poderia ser: a solicitação de uma senha para uma determinada operação do sistema.

Cabe ao *engenheiro de requisitos* identificar os fatores associados às funções, atributos e restrições impostas ao produto ou serviço e promover a compatibilização dessas informações no conjunto dos requisitos.

e) Características Associadas aos *Stakeholders*

As características associadas aos *stakeholders* são as preferências pessoais por uma solução particular, as expectativas quanto à solução do problema e o critério de prioridade no tratamento da solução.

- **Preferências**

As preferências são condições desejáveis e particulares do cliente, porém opcionais ao produto de software. São condicionadas à definição prévia dos atributos e das restrições dos requisitos. Ou seja, são circunscritas no espaço de solução do problema (GAUSE; WEINBERG, 1990).

- **Expectativas**

As expectativas são declarações do cliente quanto à forma de ver atendida uma demanda. São originadas do conhecimento do problema e do ambiente, cuja satisfação refere-se à solução (GAUSE; WEINBERG, 1990).

Para o *engenheiro de requisitos*, as expectativas são expressas com o retorno da aplicação do modelo proposto em identificar falhas de representatividade de pontos de vista do universo de *stakeholders*, identificar ambigüidades nas declarações de prioridades de requisitos, obter a qualificação dos requisitos, checando a presença ou ausência de características essenciais de conteúdo, sob os aspectos da definição do domínio da aplicação solução (GAUSE; WEINBERG, 1990).

- **Prioridade**

A definição do que é prioritário pelos *stakeholders* é uma condição essencial no processo de desenvolvimento de software. O processo é essencialmente limitado pela disponibilidade de recursos (humanos, financeiros, tecnológicos... e pelo fator custo de produção.

Obter o nível de exigência de solução para produtos ou serviços mais prioritários, além de acelerar a entrega do resultado, fazendo antes o que é realmente essencial, contribui para a negociação da forma de trabalho e para a medição de resultado da satisfação do cliente de forma gradativa.

f) Aplicação dos Processos e Utilização de Técnicas

A aplicação dos processos e a utilização de técnicas de *ER* devem iniciar antes da definição do software a ser construído e ser baseado no conhecimento inicial do problema, fase identificada como de descobrimento de requisitos.

As técnicas aplicáveis à elicitação, que nesta parte da tese são apresentadas de uma forma superficial, serão discutidas com mais profundidade no próximo capítulo.

g) Produto da Abordagem de Descobrimento de Requisitos

O resultado da aplicação dos processos de *ER* caracteriza-se pelo documento de requisitos.

Cabe ao *engenheiro de requisitos* identificar se o resultado obtido com a documentação dos requisitos está compatível com o escopo definido para o trabalho; se os problemas identificados estão relacionados e devidamente associados aos requisitos dos *stakeholders* e se houve representação esperada dos mesmos em relação à finalidade do produto ou serviço objeto do trabalho.

Os documentos de requisitos, além de conter as funcionalidades e restrições dos requisitos (que reproduzem ações identificadas no contexto organizacional), devem também representar os requisitos não-funcionais (os quais refletem características de qualidade), identificando os atributos, restrições, preferências e expectativas.

2.3 Conceitos e Fundamentos da *ER*

Este tópico visa contextualizar a *ER*, enfatizar as diversas correntes de definição e uso do conceito de requisitos e de especificações, identificar os processos e as técnicas de *ER* e apresentar características do documento de requisitos. A descrição de cada item está organizada em três tópicos: conceito, característica e aplicação.

2.3.1 Engenharia de Requisitos

a) Conceito

Entende-se engenharia como aplicação de princípios matemáticos e científicos às construções, como técnica construtiva. Construções, no caso de requisitos, equivalem a descrições. É entendido por requisitos como a condição para conseguir certo fim, como exigência legal necessária para certos efeitos.

ER, segundo Zave e Jackson (1997), é o ramo da *Engenharia de Software* preocupada com os objetivos do mundo real, funções e condições sobre software. Também diz respeito ao relacionamento desses fatores para especificações precisas de comportamento de software, para evolução do software com o tempo e cruzamento de famílias de software.

ER, para Macaulay (1996), pode ser definida como o processo sistemático de desenvolvimento de requisitos por meio de um processo

iterativo e cooperativo de análise do problema, de documentação das observações resultantes em uma variedade de formatos de representação e de checagem da precisão do entendimento obtido.

ER, segundo Kotonya e Sommerville (1998), é um termo relativamente novo que foi inventado para cobrir todas as atividades envolvidas em descobrimento, documentação e manutenção de um conjunto de requisitos para um sistema baseado em computador.

ER, segundo Ryan (1998), é o processo de desenvolvimento e uso de tecnologia de custo efetivo (engenharia) para elicitação, especificação e análise dos requisitos dos *stakeholders* que serão satisfeitos pelo software.

O uso do termo engenharia implica que técnicas sistemáticas e repetitivas podem ser usadas para assegurar que os requisitos do software sejam completos, consistentes, relevantes etc.

Segundo Siddiqi e Shekaran (1996), o campo tradicionalmente conhecido como análise de sistemas foi primeiro aplicado a sistemas de informação e tinha uma orientação à aplicação e ao enfoque organizacional. O campo da *ER* pesquisa para incorporar uma orientação de engenharia dentro da análise de sistemas.

Como neste trabalho são tratados principalmente os aspectos relativos à elicitação dos requisitos, adota-se o conceito de Kotonya e Sommerville (1998) sob os aspectos da caracterização dos processos e atividades na *ER*.

b) Características

A *ER*, como área de pesquisa, identifica-se fundamentalmente com a fase que antecede o processo de desenvolvimento de software. Compreende a definição do que se quer produzir e quais as funções que o produto deve realizar (foco no entendimento do problema no domínio da aplicação), de que forma e sob quais atributos, restrições, preferências e expectativas do cliente, fatores determinantes que delimitam a abrangência do domínio da aplicação (foco na solução do problema com a tecnologia de software).

c) Aplicação

Cabe à *ER*, como subárea da *Engenharia de Software*, aperfeiçoar processos de manutenção de requisitos para o gerenciamento de seu ciclo de vida.

Deve também propor métodos, ferramentas e técnicas que promovam o desenvolvimento do documento de requisitos, para que este produto retrate o conhecimento do problema em conformidade à satisfação do cliente e aos padrões de qualidade, relacionados ao que se quer produzir com a tecnologia de software para solução do problema.

A *ER*, segundo Leite (1989), estabelece o processo de definição de requisitos como um processo no qual o que será feito deve ser elicitado, modelado e analisado. Este processo deve lidar com diferentes pontos de vista, e usar uma combinação de métodos, ferramentas e pessoal. O produto desse processo é um modelo, do qual um documento chamado requisitos é produzido (LEITE *et al.*, 1997). Este processo é perene e acontece em um contexto previamente definido a que chamamos de *Universo de Informações* (UdI) (LEITE; OLIVEIRA, 1995).

2.3.2 Requisitos e Especificações

Os requisitos são as expressões declaradas da maneira de ver e traduzir as necessidades ou desejos do cliente no ambiente de operação de software. As especificações são as representações do detalhamento de como implementar a solução de software.

a) Conceito

Requisito, segundo Macaulay (1996), simplesmente pode ser definido como *algo de que um cliente necessita*. Entretanto, do ponto de vista do *engenheiro de requisitos*, requisito pode também ser definido como *algo que necessita ser projetado*.

Existem inúmeras definições do termo requisitos; uma delas reporta-se a *IEEE padrão 610-1990* (*apud* MACAULAY, 1996), aplicável de forma geral e a situações que não as específicas de software:

- *uma condição ou capacidade necessária para um cliente resolver um problema ou realizar um objetivo;*

- *uma condição ou capacidade que deve ser satisfeita ou a propriedade de um software ou componente de software para satisfazer um contrato, padrão, especificação ou outro documento imposto formalmente.*

Em contraste ao padrão *IEEE*, o padrão *BSI* (*British Standards Institute*) enfatiza os requisitos do usuário. O *BS 6719* (*apud* MACAULAY, 1996), guia padrão 1986, aplicável a requisitos de usuários para sistemas baseados em computador, não provê uma definição de requisitos, mas uma base para descrição das necessidades e prioridades do cliente, ou seja, ele é específico para software.

Requisitos, segundo Sommerville, Sawyer e Viller (1998), são descrições de como o software poderá comportar-se, informações do domínio da aplicação, restrições sobre operação de software ou especificações de propriedade ou atributo de um software. Os requisitos são definidos durante os estágios iniciais do desenvolvimento de software como uma especificação do que poderá ser implementado. Requisitos, invariavelmente, contêm uma mistura de informação do problema, declarações de comportamento e propriedades do software, condições de projeto e restrições de construção.

Wieringa (1998) apresenta, em seu relatório comparativo de métodos de especificação de requisitos, a diferenciação de conceitos entre negócio, requisitos e especificações:

- *chama de negócio o ambiente social consistindo de pessoas que têm desejos e demandas relativas à forma de fazer o negócio;*
- *chama de requisito o desejo e demanda do ambiente do negócio. Um requisito é uma forma de trabalhar no negócio. Software e hardware são introduzidos no negócio para ajudar as pessoas a realizar em sua forma desejada de trabalho, ou seja, realizarem seus requisitos;*
- *a adequação do software desejado para realização do trabalho é feita através de especificações de software, que muitos denominam requisitos de software.*

Requisitos, segundo Zave e Jackson (1997), são fenômenos do domínio da aplicação. São exclusivamente todos os fenômenos do ambiente. É uma propriedade do domínio da aplicação ou ambiente, que o software deve

executar. Para representá-los exatamente, descrevem os relacionamentos acerca dos fenômenos do contexto do problema. Normalmente, são expressos em linguagem natural, diagrama informal ou usando alguma notação que é apropriada para o entendimento do problema.

Particularmente, esta última definição dá ênfase ao domínio do conhecimento sobre o problema, tendo como referência os fenômenos que ocorrem no ambiente de negócio do cliente, onde residem os problemas a serem resolvidos, seja com o uso de computador ou não. A Figura 2.2, extraída de (JACKSON, 1995), representa o contexto do problema.

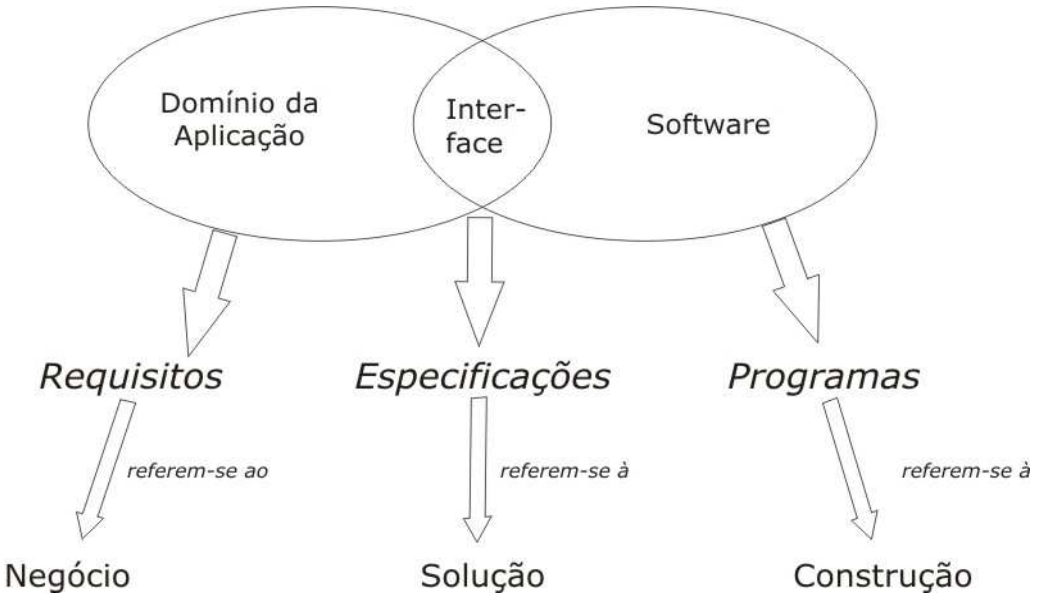

Figura 2.2 - Requisitos e especificações no contexto do problema (JACKSON,1995).

Os requisitos são descrições dos fenômenos do domínio da aplicação, as especificações são as declarações que descrevem as características de interface entre o ambiente e o software e os programas são as descrições do software.

Consolidando as idéias, requisito é uma declaração descritiva de fenômenos, escrita do ponto de vista do cliente, envolvendo o universo da fonte de informação para os quais serão providos a tecnologia da informação e o compartilhamento de recursos na solução de problemas.

b) Características

Os requisitos evoluem com o tempo e a dinâmica organizacional, e são caracterizados sob três classes: *funcional, não-funcional* e *inverso*. Os requisitos funcionais referem-se a condições e exigências de transformação de entradas em saídas. Para os requisitos funcionais, existem três classes de sentenças: entrada, saída e mudança de estado. Cada requisito tem que seguir uma das duas estruturas abaixo:

- *O sistema deve + [verbo + objeto | frase verbal] + [complemento de agente | null] + [condição | null]*.
- *O sistema deve + [verbo + objeto | frase verbal] + [complemento de agente | null] + {a) condição-1, b) condição-2,... condição-n}.*

O ***verbo*** é um verbo simples que expresse a funcionalidade daquele requisito. Dependendo do tipo do verbo, objeto pode ser um objeto direto ou um objeto direto seguido de um objeto indireto. A ***frase verbal*** é uma frase que expressa a funcionalidade do requisito. O ***complemento de agente*** é a identificação de um agente relacionado com o requisito. Algumas vezes esse complemento pode ser descrito pelo objeto indireto. Um agente pode ser uma pessoa, uma instituição, um grupo ou um dispositivo físico externo ao software. Uma ***condição*** é uma sub-sentença que reflete uma situação específica. Uma forma de adicionar mais estrutura, no que diz respeito a conectivos é apresentada abaixo.

- O *sistema* deve +verbo + objeto + "para" + complemento + ["quando" | "se"] + condição.

Já os *requisitos não-funcionais* podem ser classificados em: requisitos de processos, requisitos de produtos e requisitos externos (KOTONYA; SOMMERVILLE, 1998). Referem-se às especificações técnicas de padrões e métodos do processo produtivo, de qualidade do produto e características desejáveis e de políticas aplicáveis ao processo e ao produto gerado. Segundo Leite (2002), os requisitos não-funcionais podem ser expressos de duas maneiras: independentes ou associados a um requisito não funcional. No caso de um requisito não-funcional associado tem-se a seguinte estrutura:

- O *sistema* deve + [verbo + objeto | frase verbal] + [complemento de agente | null] + [condição | null] + [restrição | null].

Nesse caso, ***restrição*** é uma frase que qualifica a funcionalidade restringindo-a. É uma frase não verbal onde às vezes o núcleo da frase é um adjetivo. Os requisitos não-funcionais independentes têm geralmente as seguintes estruturas:

- O *sistema* deve + [“ter” | “manter” | “possuir” | “atender” | “fazer” + objeto] + frase-adjetiva.
- O *sistema* deve + [“ter” | “manter” | “possuir” | “atender” | “fazer” + objeto] + [frase-adjetiva | null].
- *Condição* + o sistema deve + [“ter” | “manter” | “possuir” | “atender” + “fazer” + objeto] + [complemento de agente | null] + [frase-adjetiva | null].

O software é caracterizado por sua funcionalidade (o que faz) e por sua qualidade (como se comporta com respeito a alguns atributos observáveis como performance, reusabilidade, confiabilidade...), conforme ilustrado na Figura 2.3. Este ponto de vista, colocado por Macedo (1999), tem o objetivo de esclarecer seu ponto de vista acerca da não-funcionalidade e distinguir três conceitos fundamentais: *atributo não-funcional, comportamento não-funcional e requisito não-funcional*. Esta conceituação é utilizada para colocar informação não-funcional na definição da arquitetura de software.

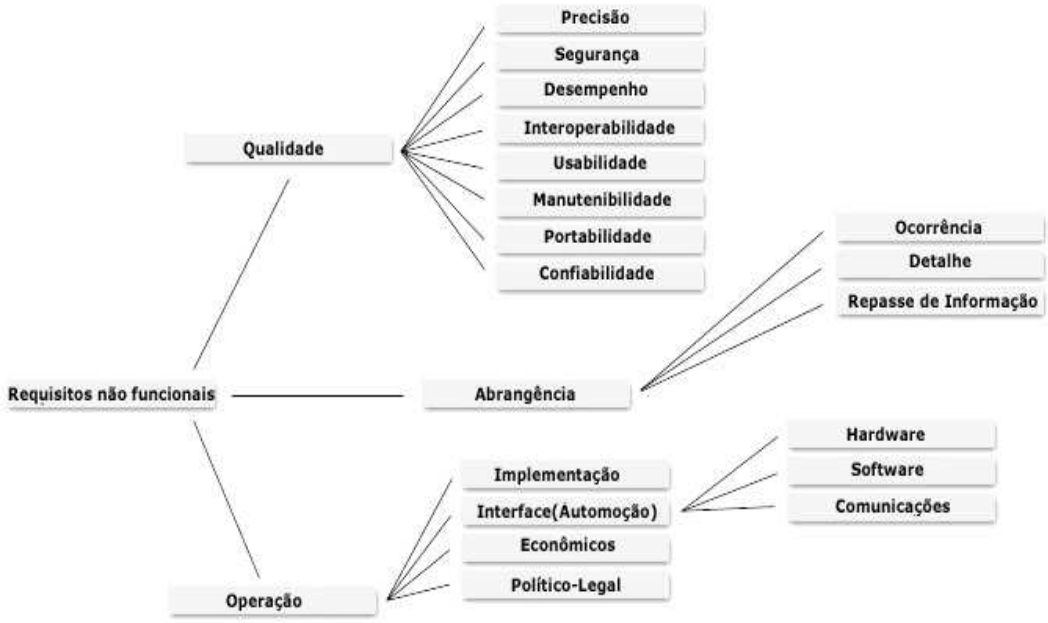

Figura 2.3 - Classificação dos RNF, segundo Macedo (1999).

Por fim, os requisitos inversos, representados por R-1 , são situações que não podem ocorrer em situação alguma. São de certa maneira restrições de alcance geral.

- O sistema ***não*** pode + [frase verbal].

c) Aplicação

O propósito do levantamento de requisitos é determinar o que deve ser feito, a base na qual as futuras ações serão propostas no desenvolvimento de um software. A primeira fase de requisitos termina com acordo obtido sobre os requisitos iniciais com o cliente antes da fase de projeto, mas o trabalho de requisitos evolui e pode ser iterativo até que o produto esteja concluído. Requisitos completos, concisos e consistentes são o que se quer obter, mas todo o esforço envolvido no processo deve estar alerta à dinâmica das mudanças organizacionais e ambientais onde o software irá atuar.

2.3.3 Processos de ER

Os processos de *ER* compreendem a sistematização dos procedimentos de descrição de requisitos.

a) Conceito

Um processo, segundo Kotonya e Sommerville (1998), é um conjunto organizado de atividades que transforma entradas em saídas. Processos são partes dos aspectos da vida e um mecanismo essencial para reprodução com complexidade. Descrições de processos são muito importantes porque permitem conhecimento para reuso.

Uma vez que se tenha trabalhado como resolver um problema, documentadas as formas em que a solução foi derivada como um processo, isto ajuda outras pessoas a tratarem problemas similares e iniciar suas próprias soluções. Processos são fundamentais para atividades humanas e as pessoas certamente comunicam detalhes dessas atividades pela descrição de processos associados.

O processo de *ER*, segundo Kotonya e Sommerville (1998), é um conjunto estruturado de atividades para conhecer requisitos, validar e mantê-los

em um documento de requisitos. Essas atividades incluem elicitação, análise, negociação e validação de requisitos. Uma descrição completa inclui quais atividades são destacadas, a estruturação ou escalonamento destas atividades, quem é o responsável, as entradas e/ou saídas para/de e as ferramentas usadas para dar suporte à *ER*. É um conjunto estruturado de atividades que conduz à produção de um documento de requisitos que especifica um software.

O processo de *ER*, para Macaulay (1996), pode ser entendido como uma série de atividades consistindo de: articulação do conceito inicial, análise de problema, viabilidade e escolha de opções, análise e modelagem e documentação de requisitos. Cada atividade pode resultar em um produto. O produto deverá ser capaz de ser mantido e estar sujeito a controle de qualidade. Cada atividade do processo requererá o uso de técnicas específicas. Diferentes situações requererão diferentes modelos de processos.

b) Características

Na prática, as atividades do processo de *ER* são intercaladas e existe uma grande interação e realimentação de uma para outra atividade. As atividades já citadas são agregadas ao processo de *ER*, a atividade de documentação e de gerenciamento de requisitos, conforme apresentado na Figura 2.4:

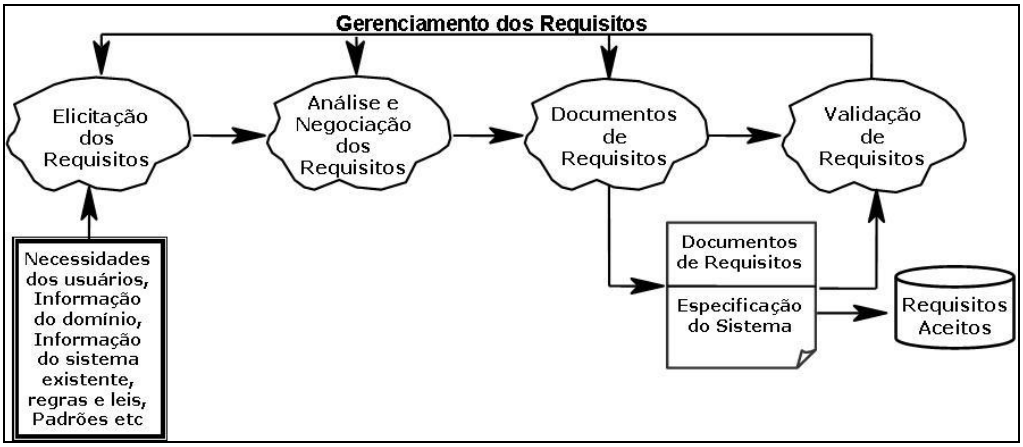

Figura 2.4 – Fases da *ER* (KOTONYA; SOMMERVILLE, 1998).

- **Elicitação** – Processo pelo qual os requisitos são descobertos por meio de consultas aos *stakeholders*, documentos, domínio do conhecimento e estudos de mercado, o que favorece um número maior de soluções, pelo fato de se conhecer mais sobre o problema a ser resolvido. Durante muito

tempo, requisitos de qualidade apareciam em fases tardias do ciclo de vida de desenvolvimento o que, muitas vezes, implicava um grande esforço para implementação, obrigando a mudanças no projeto. É um processo cuidadoso de interação com pessoas aliado à necessidade de avaliação da organização, do domínio da aplicação e dos processos de negócio em que o sistema operará. Pode parecer apenas um processo de transferência de conhecimento entre pessoas, e seria, se o cliente soubesse exatamente o que está precisando, fato que raramente encontramos no mundo real. Segundo Kotonya e Sommerville (1998), as quatro dimensões para o descobrimento dos requisitos são:

- ***Domínio da aplicação****: Conhecimento do domínio da aplicação consiste em conhecer o negócio sobre o qual o sistema será aplicado. Isto significa que para entender os requisitos de um sistema, é necessário conhecer o negócio em questão.*
- ***Problema****: O conhecimento dos detalhes específicos do problema do cliente é fundamental para a definição correta dos requisitos verdadeiros do sistema. Para isso, é necessário o conhecimento da forma de operação do negócio da empresa.*
- ***Negócio****: Conhecer como o sistema afetará as diferentes partes do negócio do cliente, e as contribuições que o mesmo fará, é fundamental no descobrimento dos requisitos verdadeiros.*
- ***A necessidade e restrições do cliente e seus usuários****: Entender a necessidade e as restrições das pessoas afetadas pelo sistema implica entender os processos do negócio que serão suportados pelo sistema e o papel dos sistemas existentes nesses processos de negócio.*

A qualidade do produto a ser construído dependerá diretamente da qualidade dos requisitos identificados. Isto quer dizer que construir coisas de forma correta, não significa que elas sejam as coisas certas. A implicação desta afirmação torna essencial um processo eficiente para identificação do problema e suas possíveis soluções, pelo fato de que se forem elicitados os requisitos errados, ainda que o produto atenda tais requisitos, ele não será o produto que o cliente espera receber.

- **Análise e Negociação** - Os requisitos são analisados em detalhe e diferentes *stakeholders* negociam para decidir sobre quais requisitos serão aceitos. Este processo é necessário porque existem, inevitavelmente, conflitos entre os requisitos de diferentes fontes, informações podem estar incompletas ou os requisitos descritos podem estar incompatíveis com as restrições ambientais. A utilização de um *checklist* facilita esta fase, tendo em vista ser este processo trabalhoso e que envolve pessoas experientes da equipe de trabalho, o que passa a implicar alto custo. Segundo Sommerville e Sawyer (1997), alguns itens que devem ser verificados nesta fase:
  - *Existe alguma informação em relação à forma como o sistema será implementado?*
  - *A descrição dos requisitos descreve um único requisito ou existem requisitos que podem ser desmembrados?*
  - *Os requisitos são realmente necessários?*
  - *Os requisitos estão de acordo com as metas do negócio do cliente?*
  - *Os requisitos podem ser interpretados de maneira diferente por diferentes usuários (ambigüidade)?*
  - *Os requisitos são possíveis de ser implementados no sistema?*
  - *Os requisitos podem ser testados para verificar se são satisfeitos pelo sistema?*

  Outra ferramenta bastante útil para suportar o processo de análise é a utilização de um protótipo, em razão dos seguintes aspectos:
  - *Pode ser considerado como uma versão inicial do sistema;*
  - *É algo bastante concreto para os envolvidos no projeto, principalmente para o usuário final e o cliente, que poderão "ver" o produto desejado, antes de ser empenhado os altos custos de desenvolvimento;*
  - *Foca basicamente a funcionalidade desejada. Esta característica torna a sua construção relativamente rápida.*

  Esta fase termina com a geração do documento dos Requisitos verdadeiros do Cliente, ou seja, aqueles que devem guiar o processo de implementação, também conhecido como requisitos alocados.
- **Validação** - Após a documentação dos requisitos ter sido produzida, inicia-se o processo de validação, buscando checar se os requisitos estão certos,

ou seja, descritos de forma apropriada, procurando eliminar problemas de incompleteza, ambigüidade ou inconsistência. A preocupação maior desta fase é com a qualidade do documento de requisitos produzido. Para esta fase também é recomendado o uso de *checklist* que deve identificar, entre outras coisas:

- *Os requisitos podem ser entendidos claramente?*
- *Os requisitos não possuem informação repetida desnecessariamente?*
- *Os requisitos atendem completamente às necessidades do cliente?*
- *Existe alguma informação faltante que deveria estar descrita no documento?*
- *Os requisitos podem ser interpretados de forma diferente por diferentes usuários?*
- *Os requisitos não geram contradição entre si?*
- *Os requisitos estão organizados de forma adequada?*
- *Os requisitos estão em conformidade com os padrões estabelecidos?*
- *Os requisitos podem ser rastreados, possuem ligações com outros requisitos que possuem relação e a razão de sua existência está documentada?*

O objetivo final desta fase é liberar o documento de requisitos que deve representar de forma clara e consistente o que deve ser implementado. Tal documento será o guia para as etapas seguintes do processo de desenvolvimento do sistema. Porém, enquanto problemas forem detectados nesta fase, o documento de saída será a lista de problemas encontrados. Esta lista de problemas retornará para as etapas anteriores do ciclo de vida do processo de desenvolvimento de requisitos, ficando neste *loop* até que o documento atinja o padrão de qualidade adequado.

- **Gerenciamento** - Responsável por controlar a evolução dos requisitos de um sistema, seja por constatação de novas necessidades, seja por constatação de deficiências nos requisitos registrados até o momento (THAYER; DORFMAN, 1997). Os requisitos são a base para comunicação entre os *stakeholders* e o *engenheiro de requisitos*. Esses são difíceis de capturar, analisar, rastrear, gerenciar e, portanto, mudam freqüentemente.

Sempre que os requisitos alocados forem alterados, os planos de software, os artefatos e as atividades afetadas devem sofrer ajustes para continuarem consistentes. O ponto chave é que os requisitos são ativos e estão em uso durante todo o ciclo de vida sendo a base para a modelagem do sistema.

Uma das grandes verdades do desenvolvimento de sistemas é: “os requisitos mudam”. Existem várias razões que motivam a mudança, mas sem dúvida, o principal fato é que durante o desenvolvimento os envolvidos vão entendendo melhor suas necessidades, refletindo em “mudanças”.

Por este motivo, agilidade no processo de tratamento das mudanças é fundamental. Congelar os requisitos após a etapa de validação é inviável, já que os negócios não são estáveis. Como eles se adaptam às mudanças, os sistemas também têm de se adaptar. A capacidade de adaptação do processo de desenvolvimento é hoje um diferencial estratégico entre as empresas de software. Esta capacidade de adaptação é mérito em grande parte do processo de gerenciamento de requisitos. Esta agilidade se torna possível a partir do momento em que podemos rastrear os requisitos (LEFFINGWELL; WIDRIG, 2001)..

Gerenciar o ciclo de vida de requisitos é uma atividade essencial para recuperação das mudanças e avaliação de impacto das ocorrências no software. Portanto, constitui uma fase fundamental na *ER* e a facilidade de recuperação das informações irá depender da pré-definição de atributos gerenciáveis dos requisitos durante a fase de descobrimento e definição dos mesmos.

c) Aplicação

Poucas organizações têm um processo de *ER* (KOTONYA; SOMMERVILLE, 1998) padronizado e definido explicitamente. A aplicação varia de uma organização para outra, mas muitos processos envolvem as atividades de: elicitação, análise e negociação, documentação e validação de requisitos. O funcionamento do processo ocorre em forma de espiral, é iterativo e envolve repetição das atividades na geração de versões do documento de requisitos.

Segundo Kotonya e Sommerville (1998), existem inúmeros fatores que contribuem para a variabilidade de processos de *ER*, dentre os quais são notórios:

- *maturidade técnica: as tecnologias e métodos usados para ER variam de uma organização para outra;*
- *envolvimento multidisciplinar: os tipos de disciplinas de engenharia e de gerenciamento envolvidos em ER variam de uma organização para outra;*
- *cultura organizacional: a cultura de uma organização tem um efeito importante sobre todos os processos de negócios e, como a cultura varia, variam também os processos de ER;*
- *domínio da aplicação: diferentes tipos de sistemas de aplicação necessitam de tipos diferentes de processos de ER.*

A atividade de levantamento de requisitos varia de acordo com cada projeto e depende das informações disponíveis e do nível de conhecimento dos entrevistados. Por essa razão é mais adequado a organização iniciar com um modelo mais genérico das atividades que envolvem a *ER*, e instanciá-lo para um modelo mais detalhado que seja apropriado as suas próprias necessidades.

2.3.4 Técnicas de ER

As técnicas de *ER* são os recursos utilizáveis como suporte aos processos de *ER.*

a) Conceito

Entende-se técnica como o conjunto dos processos de uma arte, de um ofício ou de uma ciência. Em *ER*, as técnicas referem-se a um conjunto de métodos e de ferramentas aplicáveis às atividades dos processos de elicitação, análise, validação e gerenciamento de requisitos.

b) Características

Macaulay (1996) apresenta uma série de abordagens para o entendimento do problema de requisitos. São elas:

- *Marketing – interessada no relacionamento entre requisitos e o sucesso de um produto no mercado;*
- *Psicologia e Sociologia – interessada no relacionamento entre requisitos e necessidades de pessoas como seres inteligentes e sociais;*
- *Análises Orientadas a Objetos (OOA - Object-oriented Approaches) – interessada no relacionamento entre requisitos e o processo de desenvolvimento do software, iniciado de uma perspectiva de objetos do mundo real;*
- *Análises de Sistemas Estruturados (SSA - Structured Systems Analysis) – interessada no relacionamento entre requisitos e o processo de desenvolvimento de software, iniciado de uma perspectiva de processos e dados;*
- *Projeto Participativo (PD - Participatory Design) – interessada em requisitos como parte de um processo que permite o envolvimento ativo do usuário no projeto de software que afeta seu próprio trabalho;*
- *Interação Computador e Humanos (HCI - Human-Computer Interaction) e Fatores Humanos (HF - Human Factors) – interessada nos fatores humanos de aceitabilidade de software pelas pessoas, a usabilidade do software, a adequação da interface apresentada, o relacionamento entre requisitos e avaliação do software em uso;*
- *Sistemas Soft (SSM - Soft Systems Method) – interessada no relacionamento entre requisitos e como as pessoas interagem fazendo parte de um sistema organizacional;*
- *Qualidade, (por exemplo, QFD - Quality Function Deployment) – interessada no relacionamento entre requisitos e a qualidade de um produto, em relação ao processo de aperfeiçoamento que conduz à satisfação do cliente;*
- *Representação Formal Ciência da Computação – interessada no relacionamento entre requisitos e a necessidade de precisão da Engenharia de Software, dependendo da natureza do negócio.*

As características técnicas serão ampliadas e discutidas com mais propriedade no próximo capítulo que tratará da elicitação de requisitos com mais rigor.

c) Aplicação

O papel das técnicas de *ER* pode ser sumarizado como necessário para suportar as diferentes fases e atividades do processo de *ER*, tendo como foco a abordagem do relacionamento entre produtor e fornecedor.

As várias técnicas constituem um elenco de alternativas para seleção da técnica específica ou de um conjunto de técnicas adaptáveis ao processo de *ER*, à comunicação humana, ao desenvolvimento do conhecimento do problema, à documentação e ao gerenciamento dos requisitos.

2.3.5 Documento de Requisitos

O documento de requisitos é o produto final do processo de descobrimento de requisitos que reúne necessidades e propósitos demandados pelos *stakeholders*.

a) Conceito

É uma declaração formal de requisitos de clientes, usuários finais e desenvolvedores de software (SOMMERVILLE; SAWYER, 1997; KOTONYA; SOMMERVILLE, 1998).

É uma especificação *do que* é requerido a um software fazer (*não como fazer*) (MACAULAY, 1996).

b) Características

Dependendo da organização, o documento de requisitos pode ter diferentes nomes e ser de vários tipos ou níveis de detalhamento. Um documento de requisitos pode ter diferentes papéis e diferentes formas e conteúdo.

Um documento de requisitos, segundo norma *IEEE-STD-1233* (IEEE *apud* SOMMERVILLE; SAWYER, 1997), deverá conter declarações não ambíguas e ser completo, verificável, consistente, modificável, rastreável e usável durante todas as fases do ciclo de vida do requisito.

c) Aplicação

Esse documento tem por finalidade formalizar os requisitos e ser utilizado como referência para as outras fases do ciclo de vida do software. Segundo Ryan (1998), o documento de requisitos é o meio utilizado para descrever as restrições quanto às características do produto e quanto ao processo de desenvolvimento, a interface com outras aplicações, a informação acerca do domínio da aplicação e informações de suporte ao conhecimento do problema, tais como: modelos, termos especializados do negócio, recuperação e gerenciamento de informações em mudança.

Segundo Kauppinen e Sulonen (1997), o documento de requisitos tem três principais objetivos: i) estabelecer um acordo entre cliente e fornecedor sobre o que o software deverá fazer; ii) prover a base de especificação de software e projeto e; iii) suportar a verificação e validação do software. O documento de requisitos também suporta o gerenciamento do projeto. O esforço de estimativa preliminar pode ser feito tão logo os requisitos tenham sido definidos. Outro importante benefício é a redução do tempo total e esforço requerido para o desenvolvimento do software. Definindo bem os requisitos antes do início do projeto, é possível evitar posterior reprojeto, recodificação e refazer testes.

2.3.6 O Léxico Ampliado da Linguagem

a) Conceito

O principal objetivo do *Léxico Ampliado da Linguagem* (*LAL*) é registrar a linguagem utilizada pelos atores do UdI, sem contudo se preocupar com a funcionalidade (LEITE; FRANCO, 1993).

b) Características

O *LAL* do *UdI* é composto por entradas, onde cada entrada está associada a um símbolo (palavra ou frase) da linguagem do UdI. Cada símbolo pode possuir sinônimos e é descrito através de noções e impactos. As noções descrevem o significado e as relações fundamentais de existência do símbolo com outros símbolos (denotação). Os impactos descrevem os efeitos do uso ou ocorrência do símbolo no *UdI* ou os efeitos de outras ocorrências de símbolos

no *UdI* sobre o símbolo (conotação). Dependendo do símbolo que descrevem, as entradas podem ser classificadas como sujeito, verbo, objeto e estado (predicativo).

c) Aplicação

Ao descrever entradas do *LAL*, deve-se obedecer aos princípios de vocabulário mínimo e de circularidade. O princípio de vocabulário mínimo demanda que a utilização de símbolos externos ao *LAL* do *UdI* seja minimizada ao máximo. O princípio de circularidade implica na maximização da utilização de símbolos do *LAL* do *UdI* na descrição de símbolos (NETO, 2000). Conforme se observará na figura 2.5, o processo de construção do *LAL* é contínuo, sendo retomado sempre que alterações surgirem no UdI que tragam novos símbolos para o domínio ou pela descoberta de novos símbolos durante a elicitação de um dado símbolo (princípio da circularidade).

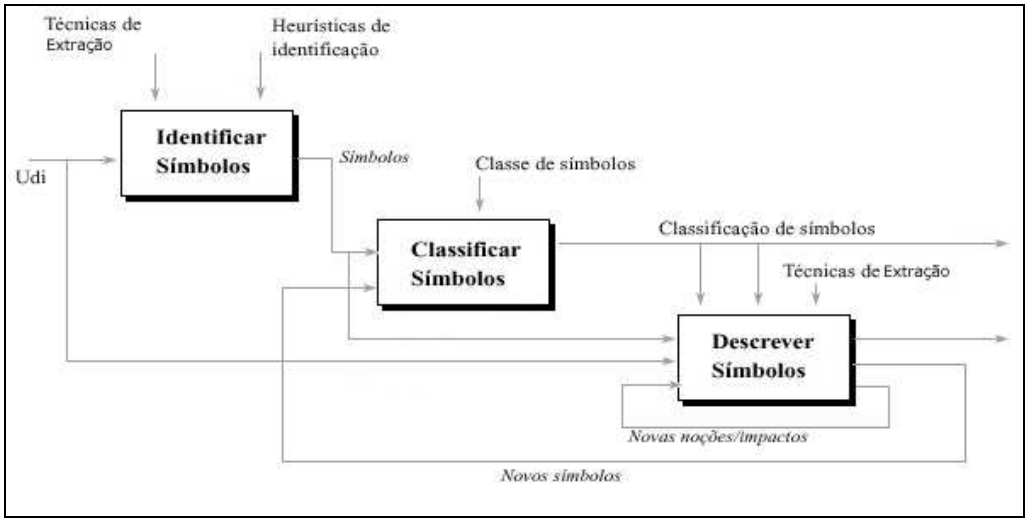

Figura 2.5 - Representação do processo de entradas do *LAL* (NETO, 2000).

2.4 Definição de Requisitos

Segundo Leite e Oliveira (1995) as condições para definição de requisitos são:

- *conhecer o domínio da aplicação, o ambiente onde os requisitos dos clientes são encontrados, a definição do alvo do problema e a abrangência do domínio da solução;*

- *identificar o problema a resolver, promovendo o entendimento, a especialização e o domínio do conhecimento, compreendendo: o quê, com quê, para quê e para quem;*
- *delimitar o contexto do negócio do cliente, estabelecendo objetivos, dominando assuntos organizacionais, fatores políticos, conflitos de poder, relações de influência, entendendo o histórico e estrutura organizacional e a organização do conhecimento acerca da organização;*
- *identificar qual o universo de fontes de informações (stakeholders);*
- *identificar as exigências e condições para satisfação das necessidades ou desejos do ponto de vista dos stakeholders;*
- *validar as informações obtidas e o relacionamento entre elas e compatibilizar idéias;*
- *administrar, revisar e negociar conflitos de interesse no atendimento às necessidades;*
- *priorizar a demanda pela solução dos problemas;*
- *documentar os requisitos, com recuperação acessível do ciclo de vida;*
- *aperfeiçoar o processo de comunicação;*
- *gerenciar as mudanças nos requisitos.*

Para o *engenheiro de requisitos*, o primeiro princípio é conhecer o universo atual de fonte de informação e o potencial futuro; o segundo é contatar e obter informação, se impossível do universo, mas de uma amostra representativa deste universo. Por amostra representativa entendem-se as pessoas ou organizações que representam o interesse direto na definição dos requisitos e características do produto ou serviço e na abrangência da aplicação.

2.5 Representação dos Requisitos

A representação dos requisitos é tema permanente de discussão e responsável por grande parte da ambigüidade da terminologia na área. Davis (1993) afirma que um requisito, qualquer que seja a linguagem de sua especificação:

- *define um objeto, uma função ou um estado;*

- *limita ou controla as ações associadas a um objeto, uma função ou um estado;*
- *define relacionamentos entre objetos, funções ou estados.*
- *um objeto é qualquer entidade do mundo real, com interface bem definida e com relevância para a solução do problema. Uma função, sob a perspectiva de requisitos, é uma tarefa, serviço, processo, função matemática ou atividade que é:*
- *executada no mundo real;*
- *executada pelo sistema que está sendo especificado para resolver um problema no mundo real.*

Um estado, do ponto de vista de requisitos, é uma condição de algo que capta parte de sua história e é usada para determinar como ele deve se comportar em circunstâncias específicas. Este algo pode ser o sistema, um objeto ou uma função. Segundo Zave e Jackson (1997) requisitos devem descrever aquilo que pode ser observado na interface entre o ambiente e o sistema, e nada sobre o sistema.

Os autores afirmam ainda que todas as declarações feitas no contexto da *ER* são declarações sobre o ambiente, podendo ser de dois tipos:

- *indicativas - descrevendo o ambiente como ele é, na ausência do sistema ou sem considerar suas ações;*
- *optativas - descrevendo o ambiente tal como gostaríamos que ele fosse quando o sistema estiver conectado a ele.*

Zave e Jackson (1997) alertam para a tendência à implementação a que podem estar sujeitas especificações baseadas em estados. Ao especificar estados internos do sistema, esta sendo descrito algo que não é observável na interface entre ele e o ambiente. Diversas abordagens utilizam técnicas baseadas em modelos, representando o estado do sistema através de estruturas matemáticas como conjuntos, relações e duplas. Tais técnicas correm maior risco de polarizar a especificação em favor da implementação.

A polarização da especificação em favor da implementação é indesejável, pois pode restringir inadequadamente a solução, fazendo com que o modelo utilizado na especificação seja adotado no projeto, sem que outras

alternativas sejam analisadas criteriosamente e que, talvez, pudessem visar de forma mais adequada às características desejadas de um bom projeto.

2.6 Rastreamento de Requisitos

O rastreamento de requisitos refere-se à habilidade de descrever e seguir a vida de um requisito (GOTEL; FINKELSTEIN, 1997; HAMMER et al., 1997), desde a fonte de origem até o desenvolvimento e especificação. É uma técnica que permite a visibilidade do relacionamento de dependência entre os requisitos, identificando relacionamentos hierárquicos do tipo "pai-filho" entre eles. Além da dependência entre os requisitos, deve ser possível identificar a dependência entre os requisitos e os artefatos que eles afetam.

Segundo Pinheiro e Goguen (1996), a motivação para o rastreamento de requisitos é justificada pelas seguintes observações:

- *os requisitos evoluem durante a vida de um software;*
- *os requisitos são contextualizados e dependem de detalhes da situação particular do contexto em que surgem;*
- *o rastreamento de requisitos também é contextualizado. Isto implica que um rastreamento poderá produzir objetos significativos para determinada situação;*
- *os requisitos são parte intrínseca do processo de desenvolvimento e o rastreamento de artefatos do projeto é útil por todo o ciclo de vida dos requisitos e do desenvolvimento.*

2.7 Fatores Humano, Social e Organizacional

A *ER* envolve representantes de diferentes grupos de usuários, com diferentes experiências profissionais e pessoais e também objetivos pessoais e organizacionais distintos. A atividade de requisitos é tarefa secundária no contexto organizacional e em geral não será a prioritária. Além disso, cada um dos representantes dos usuários deverá tentar influenciar de forma diferente os requisitos produzidos. Tal influência é na maioria das vezes exercida por "*status*" político e não através de argumentos racionais. Em uma organização, diferentes departamentos e indivíduos têm diferenciados graus de influência política. Tal influência depende dos indivíduos, das prioridades da organização

e do sucesso tanto da organização quanto dos usuários na realização de seus objetivos particulares. As pessoas tentam influenciar os requisitos de forma a manter ou aumentar sua influência política na organização. A identificação de tais movimentos é difícil e muitas vezes um requisito é inserido no documento de requisitos apenas para que os *engenheiros de requisitos* se livrem da influência exercida por um representante específico.

Tal procedimento deve ser evitado e uma das formas de se identificar tais requisitos é a adoção de técnicas de levantamento de requisitos baseadas em perspectivas (SOMMERVILLE *et al*, 1993).

2.8 Problemas em *ER*

O conhecimento do problema é para onde devem convergir os esforços do *engenheiro de requisitos* na busca de oportunidades de negócios para a produção de software em atendimento à demanda do cliente.

Um problema que persiste na área de *Engenharia de Software* é visualizar a tecnologia como solução de problema, sem anteriormente focar intensivamente o esforço em definição, entendimento do problema e a negociação de eventuais conflitos de interesses pela solução (JACKSON, 1995).

A falta de habilidade para discutir problemas tem sido uma das mais flagrantes deficiências da teoria e da prática em software. Muitos autores de métodos de desenvolvimento afirmam oferecer uma proposta de análise de problema, quando, de fato, oferecem somente um contorno de solução, deixando o problema inexplorado e inexplicável (JACKSON, 1995).

O tema é abordado em relação à definição e características do problema, aos fatos e fenômenos do ambiente, à origem e à essência do problema; quem tem e de quem é a responsabilidade pelo mesmo; por que e para que é necessário conhecê-lo e solucioná-lo. Abrange também a necessidade de domínio de algumas técnicas de descrição de requisitos, suas características, tipos e tecnologia de uso e uma abordagem sobre o gerenciamento de requisitos.

2.9 Foco no Conhecimento do Problema

O problema no contexto de elicitação de requisitos é a razão principal para o entendimento, a especialização e o domínio do conhecimento. Identificar o que é o problema, qual é a definição do problema, quem tem o problema e qual é a essência do problema, sob o ponto de vista de quem o tem, caracteriza a complexidade do processo.

Segundo Jackson (1995), é necessário distinguir claramente um processo de definição do problema (conhecimento dos requisitos) de um processo de solução do problema (aplicação de ferramentas de software como solução). Já que a fonte de problemas está intrínseca no comportamento das pessoas, é importante identificar qual é o desejo de ter resolvido o problema e se existe realmente o desejo de uma solução.

O foco no conhecimento do problema (figura 2.6), segundo Gause (1998), envolve o cliente e o *engenheiro de requisitos* e deve ser observado sob quatro aspectos:

- *O que é essencial - conhecido por ambos;*
- *O que é expectativa do engenheiro de requisitos - desconhecido pelo cliente e que, portanto, torna-se fator surpresa para o cliente e oportunidade de negócio para o fornecedor;*
- *O que é expectativa do cliente - desconhecido pelo engenheiro de requisitos e que, portanto, torna-se oportunidade perdida de negócio para o fornecedor;*
- *O que nunca será realizável - desconhecido por ambos, cliente e engenheiro de requisitos.*

Figura 2.6 - Foco no conhecimento do problema (GAUSE, 1998).

Conhecer o problema viabiliza o incremento de qualidade na solução de software para o cliente e abre o espaço de oportunidade de negócio para o fornecedor. Reduzir a área de desconhecimento do problema, referente ao quadrante de nunca realizável, conforme apresentado na Figura 2.6 extraída de (GAUSE, 1998), é para onde devem convergir os esforços de ambos: *engenheiros de requisitos* e o cliente. Por isso, tenta-se ampliar o universo de realizações no atendimento aos requisitos do cliente.

2.9.1 Definição e Características

Segundo Gause e Weinberg (1990), um problema é a diferença entre algo como desejado e algo como percebido pelo demandante.

Cabe ao *engenheiro de requisitos* a perspicácia de entender a discrepância entre o que é desejado na realidade e como está a situação de acordo com a percepção individual.

A palavra “problema” é de origem grega. Segundo Jackson (1995), os gregos foram os primeiros no mundo ocidental a pensar sistematicamente em como resolver problemas. Muitas palavras usadas para referenciar solução de problema, como análise, síntese, método, sistema, teorema, heurística, são derivadas de palavras gregas. Para os gregos, problemas matemáticos são divididos em duas classes: problemas para provar, e problemas para pesquisar.

Esta classificação nem sempre é clara e objetiva, pois depende de interpretação do que se quer obter. O problema para pesquisar requer do resolvedor encontrar ou construir algo.

O matemático Polya escreveu o livro *How To Solve It* (*apud* JACKSON, 1995), no qual expõe e amplia idéias para solução de problemas matemáticos. As idéias são sugestões interessantes para resolução de problemas de desenvolvimento de software.

- Quanto aos problemas para provar algo, Polya sugere:
  - *tentar pensar acerca de um teorema familiar com a mesma conclusão ou similar;*
  - *perguntar se a conclusão, dada à hipótese, é mais próxima a ser falsa ou verdadeira;*
  - *considerar que outras conclusões seguem das hipóteses.*
- Quanto aos problemas para pesquisar soluções ou idéias, Polya sugere:
  - *dividir a condição em partes;*
  - *pensar em um problema familiar comparando a um desconhecido similar;*
  - *checar que estão sendo usados todos os dados;*
  - *checar que estão sendo usadas todas as condições;*
  - *pensar em uma variação do desconhecido ou dos dados ou de ambos, a fim de que o novo desconhecido e o novo dado estejam próximos.*

O princípio metodológico de entendimento do problema é a utilização de método sensível a problemas que ele pode ajudar a resolver. Se um método não fala sobre problema, como pode ajudar na solução? Alguns métodos não são sensíveis aos problemas, porque são expressos em termos de solução mais que em problemas (JACKSON, 1995).

2.9.2 Contexto, Fatos e Fenômenos

O contexto do problema (JACKSON, 1995) é a parte do mundo em que o software será instalado, na qual os efeitos e benefícios do software serão sentidos e avaliados. Pensar em contexto do problema, inicialmente, é pensar no domínio da aplicação ou ambiente que é a parte do mundo em que o cliente

é interessado, a parte que é relevante para um problema particular. Para entender o problema, deve ser entendido o ambiente. Então, dado um contexto de problema, tem-se que descobrir e descrever os requisitos do problema.

A significância do contexto do problema é a abrangência de ambos, o software e o ambiente. Deve-se concentrar a atenção sobre o ambiente para aprender o que o problema é e quais são seus requisitos. A distinção entre domínio da aplicação e software é relacionada a o quê e a como. O quê o sistema faz é buscado no ambiente, enquanto que como o sistema faz é buscado no software (JACKSON, 1995). O ambiente abrange fatos e fenômenos (JACKSON, 1995):

- *Um fato é uma verdade simples acerca do mundo. É a menor unidade de observação ou o menor fenômeno; envolve indivíduos ou instâncias. Qualquer coisa pode ser um indivíduo: uma pessoa, um número, um evento, uma data, uma cor, uma emoção. O que for escolhido para ser tratado como indivíduo dependerá de como se opta para olhar o mundo e para que propósito.*
- *Um fenômeno é o que parece estar presente, caso se observe o mundo ou parte de um domínio. A forma de ver o mundo é personificada através de uma linguagem. A linguagem é adaptada para expressar o que se vê, e a visão é condicionada aos conceitos familiares da linguagem aplicada.*

2.9.3 Origem e Essência

A origem do problema, segundo Gause e Weinberg (1990), é natural (procede da natureza) ou é da natureza humana e nada existe a ser feito acerca disto.

Problemas que procedem da natureza são piores por duas razões: primeiro, o desamparo para fazer algo que é visto vindo de uma fonte remota e desconhecida; segundo, a indiferença do agente natureza em relação aos seres componentes.

Problemas que são de origem humana estão associados a uma fonte humana, um objeto real ou uma ação em que é possível visualizar uma alternativa de solução e ter motivação para associar a um problema.

A essência do problema está relacionada à razão de existir de um negócio ou de uma situação ou ao fato observado em um contexto organizacional.

2.9.4 Descrição de Requisitos

A todo o momento, o indivíduo defronta-se com a tarefa de representar, na forma falada ou escrita o que quer comunicar, no atendimento às necessidades físicas, emocionais, psíquicas, sociais, econômicas, enfim, existenciais no mundo em que vive (JACKSON, 1995).

Comunicar é o fato de tornar comum. É o entendimento recíproco da idéia entre emissor e receptor. O sucesso nem sempre é obtido nessa comunicação. Em sua maioria, a causa é o não entendimento da mensagem pelo receptor, ocorrendo equívoco de destinatário ou, principalmente, o uso de linguagem e simbologia não apropriadas (JACKSON, 1995).

Na *Engenharia de Software*, com o processo de representação do conhecimento não é diferente. As diferentes formas de linguagens de comunicação quer sejam escritas ou gráficas, dependem fundamentalmente do conhecimento e de uma série de regras comuns de linguagem, para se produzir mensagens comunicativas. Representar o mundo real, o ambiente para o qual se quer construir uma solução, requer o conhecimento de técnicas de descrição por parte do desenvolvedor (JACKSON, 1995).

É fato incontestável e admitido por uma grande maioria de desenvolvedores de software a dificuldade de tratar o detalhamento da representação do conhecimento. Isto é justificado pela falta de habilidade na formulação da escrita dos fenômenos que ocorrem no ambiente (JACKSON, 1996).

É mais fácil e rápido partir para uma fase de criar modelos, projetar protótipos de software e tentar a comunicação do resultado com o cliente a aprofundar o conteúdo com o conhecimento da abrangência do problema ou da demanda requerida. Isto custa tempo e recurso e implica negociação complexa com o cliente, porque também depende da disponibilidade e do comprometimento do mesmo. A não ocorrência desta fase implica a queima de

etapas, o que não permite a maturidade para a ação de proposta de alternativas de solução (JACKSON, 1996).

O domínio da aplicação ou ambiente é a porção relevante do mundo real para o projeto de desenvolvimento de software. O software é uma parte da solução baseada em computador que será construída e conectada ao ambiente, como resultado do processo de desenvolvimento (JACKSON, 1995; JACKSON, 1996).

2.9.5 Qualificação de Requisitos

Para a qualificação de requisitos (ZANLORENCI; BURNETT, 1998), é necessário contextualizar vários fatores e considerações relatados nas pesquisas bibliográficas, que são de interesse para o esclarecimento do processo:

- *A caracterização de um problema, como a diferença entre o que é desejado e o que é percebido pelo demandante; portanto o problema é algo que tem origem na exteriorização dos sentidos humanos;*
- *Partindo do princípio anterior, a análise e avaliação de um problema dependem da reunião de múltiplas perspectivas de pontos de vista;*
- *O entendimento dos pontos de vista está associado ao contexto que a informação representa e, particularmente, descrito sob a perspectiva de um cenário, envolvendo atores, episódios e os relacionamentos entre eles;*
- *A participação dos atores nos episódios é definida por papéis que possuem ações definidas por uma linguagem;*
- *A adequação dos desejos ou necessidades dos atores em relação ao resultado é determinada pelo nível de exigência dos requisitos dos atores no domínio da atuação respectivo;*
- *A linguagem de comunicação que permitirá o entendimento entre os stakeholders.*

A qualidade dos requisitos documentados deve estar associada à confiabilidade e representatividade da fonte de informação (stakeholders), pelo papel exercido, pelo nível de ocupação no contexto organizacional e pela definição do nível de exigência pela informação.

Ao mesmo tempo, o requisito deve ser qualificado pela funcionalidade, pela origem da demanda e relação de dependência entre eles. Com estes fatores devidamente caracterizados, é possível, em uma fase posterior, facilitar a definição de prioridade de implementação dos requisitos.

A qualificação dos requisitos possibilita a efetivação dos processos iterativos de negociação de conflitos e resolução de inconsistências na análise do problema. Na Figura 2.7, é feita uma representação do relacionamento dos fatores representativos no contexto da qualificação dos requisitos.

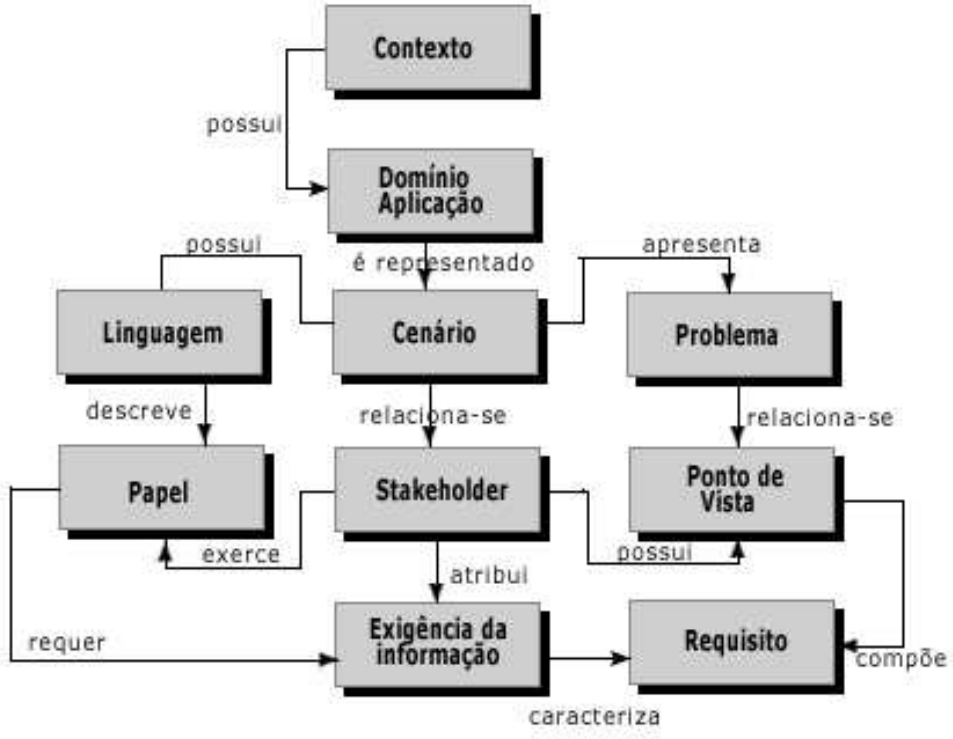

Figura 2.7 - Qualificação de requisitos (ZANLORENCI; BURNETT,1998).

2.10 Resumo

Neste capítulo, foram detalhadas as várias abordagens de tratamento dos requisitos em termos de processos e técnicas. Foram explorados os conceitos fundamentais, a definição e representação dos requisitos, conceitos tratados pela disciplina Engenharia de Requisitos (*ER*). A importância do rastreamento de requisitos e da influência dos fatores humano, social e organizacional no descobrimento dos requisitos sendo apresentada na forma de sua qualificação e aplicação. Encerrando, apresentou o foco no conhecimento do problema relacionado ao tratamento dos requisitos.

CAPÍTULO 3 - ELICITAÇÃO DE REQUISITOS

Excelência é uma habilidade conquistada através de treinamento e prática. Nós somos aquilo que fazemos repetidamente. Excelência, então, não é um ato, mas um hábito (Aristóteles, 384-322 a.C.).

Nesta etapa da fundamentação, estão detalhadas as várias abordagens de tratamento dos requisitos em termos de avaliação, processos e técnicas aplicáveis à fase de *elicitação de requisitos,* além de uma consistente e ampla conceituação sobre o tema, incluindo os problemas encontrados durante o processo.

3.1 Introdução

O processo de elicitação de requisitos de um sistema é tão complexo que torna imprescindível uma avaliação cuidadosa das necessidades que o futuro sistema precisa satisfazer. É preciso dizer *"o porquê"* da necessidade do sistema, baseado em condições atuais ou previstas, as quais podem ser operações internas ou externas, elucidar as características do sistema, na qual irão servir e satisfazer esse contexto, e ainda é preciso expor como o sistema será construído (LAMSWEERDE, 2000). Para atingir os objetivos é necessário se atentar às seguintes concepções:

Quadro 3.1 – Relação das concepções, segundo Lamsweerde (2000).

Concepção	Descrição
Amplo escopo	O sistema final não compreende somente uma peça de software, é preciso levar em consideração o ambiente, pessoas, dispositivos, outros Software etc.
Preocupações com requisitos não-funcionais e técnicos	É necessário preocupar-se com os itens relacionados à qualidade do software como segurança, flexibilidade, custos, manutenção etc.
Conflitos entre stakeholders	Pelo fato de estarem envolvidas várias pessoas com diferentes conhecimentos e interesses, pode haver pontos de vista conflitantes.
Deficiências na especificação de requisitos	Erros que podem causar efeitos desastrosos na fase de desenvolvimento e ter impacto na qualidade do software (contradições, ambigüidade, incompatibilidades entre as necessidades reais e as percebidas/observadas, etc.).

Essencialmente, essa fase do processo é relacionada à obtenção dos requisitos do sistema e compreensão das necessidades dos usuários. Assim, analistas e engenheiros de software trabalham com clientes e usuários finais, para descobrir o problema a ser resolvido, os serviços do sistema, o desempenho necessário, restrições de hardware e outras informações. Kotonya e Sommerville (1998) apresentam alguns papéis essenciais para a execução da atividade de elicitação de requisitos:

Quadro 3.2 - Os papéis no processo de *ER* (KOTONYA; SOMMERVILLE, 1998).

Papel	Descrição
Especialista do domínio	Responsável por prover informações sobre o domínio de aplicação e do problema específico a ser resolvido naquele domínio
Usuário final	Responsável pelo uso do sistema após a entrega
Engenheiro de Requisitos	Responsável por identificar e especificar os requisitos do sistema
Engenheiro de Software	Responsável por desenvolver o protótipo do sistema
Gerente de Projeto	Responsável pelo planejamento do projeto

Ainda segundo Kotonya e Sommerville (1998), a atividade de elicitação de requisitos é apresentada em quatro dimensões de compreensão:

i) **Entendimento do domínio da aplicação** - Expressa o entendimento geral da área em que o sistema será aplicado;

ii) **Entendimento do problema** - Reproduz o entendimento dos detalhes do problema específico a ser resolvido com o auxílio do sistema a ser desenvolvido. Segundo Davis (1993) denomina-se *Entendimento do Problema* a atividade relacionada ao aprendizado sobre o problema a ser resolvido, ao entendimento das necessidades dos usuários, à correta identificação dos usuários e ao entendimento de todas as restrições aplicáveis à solução;

iii) **Entendimento do negócio** – Define o entendimento da contribuição do sistema para que sejam atingidos os objetivos gerais da organização;

iv) **Entendimento das necessidades e das restrições dos *stakeholders*** – em que surge o entendimento detalhado:

- *das necessidades de apoio a serem providas pelo sistema à realização do trabalho e aos interesses de cada um dos stakeholders;*
- *dos processos de trabalho a serem apoiados pelo sistema e;*
- *do papel de eventuais sistemas existentes na execução e condução dos processos de trabalho.*

Nenhuma dessas definições fornece, contudo, a real dimensão da dificuldade na condução da atividade. Tal dificuldade surge da natureza menos técnica e mais social da atividade de *ER*, como destacam Goguen e Linde (1993). Segundo suas pesquisas, poucos esforços foram conduzidos no sentido de utilizar mais profundamente os conhecimentos desenvolvidos na área das ciências sociais no contexto da *ER*.

A forte influência das questões sociais acaba por introduzir problemas nos requisitos levantados. Tais problemas precisam ser identificados para que possam ser tratados. Entre os problemas comuns enfrentados na atividade de elicitação Kotonya e Sommerville (1998) citam:

- *A forma dispersa como são encontrados os requisitos (em livros, manuais, conhecimento de pessoas específicas, etc.);*
- *A terminologia específica do domínio da aplicação que precisa ser entendida para garantir o entendimento do problema no contexto do domínio da aplicação;*
- *A tarefa de auxiliar no levantamento de requisitos é, via de regra, secundária no contexto de trabalho dos stakeholders constituindo uma barreira à execução do trabalho de requisitos, culminando com o não envolvimento dos stakeholders no processo de requisitos;*
- *Questões organizacionais e fatores políticos que exercem grande influência sobre os requisitos. Tais fatores nem sempre são identificados pelos usuários finais e podem passar despercebidos pelo engenheiro de requisitos.*

Além desses problemas, a possibilidade de automação altera a perspectiva dos *stakeholders* sobre o próprio trabalho, fazendo com que não tenham uma correta percepção sobre os requisitos do sistema (KRUCHTEN, 2000).

3.2 Descobrimento de Requisitos

Gause e Weinberg (1990), em seu livro *Exploring Requirements*, fazem a seguinte citação:

> *The discovery is nothing; the discovering (the exploring) is everything*. Traduzindo-se, "A descoberta não é nada; o descobrimento (*a exploração*) é tudo".

O uso do termo descobrimento, no texto citado, compreende o processo exploratório inicial para capturar e descrever os requisitos, passando por uma atividade de planejamento, cujo foco é o conhecimento do problema.

Segundo Millard, Lynch e Tracey (1998), a *ER* tem tradicionalmente focado ferramentas e técnicas, envolvendo, em grande escala, desenvolvimento de sistemas organizacionais. Métodos convencionais de elicitação de requisitos certamente assumem que o usuário sabe exatamente o que deseja do futuro sistema e conhece o sistema de forma que, uma vez implementado, absorve os impactos sobre a forma de trabalho. Concentram-se sobre funcionalidades do sistema em razão de considerar o contexto em que operam.

Na verdade, qualquer sistema que envolve intervenção humana possui características de ser volátil, não previsível e complexo. Impor hierarquia e rigor matemático para reduzir esta complexidade pode distorcer o entendimento e os elementos não estruturados do sistema técnico-social podem ser deixados de lado.

Uma das grandes dificuldades na exploração inicial de requisitos é o conhecimento do que perguntar. Este conhecimento fundamenta-se sob três aspectos fundamentais: do que se trata (fatos e fenômenos), a quem diz respeito (fonte de informação) e o estabelecimento de uma forma de comunicação entre os *stakeholders* (linguagem para a elicitação e documentação dos requisitos).

A identificação de fatos ou fenômenos vai depender do conhecimento do ambiente ou domínio da aplicação pelo *engenheiro de requisitos*. É onde os requisitos dos *stakeholders* são reconhecidos e, para tal, é necessário determinar o foco e a abrangência do tratamento dos fatos. Neste processo, podem ser utilizadas várias técnicas de elicitação de requisitos, que serão detalhadas à frente (LEITE; LEONARDI, 1998).

A identificação da fonte de informação vai depender do conhecimento do universo de abrangência da informação. A estratégia para comprometer os *stakeholders* no processo de descobrimento é conhecer não só o volume atual, mas fazer a projeção futura potencial para que a alternativa de solução seja estruturada de forma a acompanhar e estar preparada para as mudanças previstas ou tendências.

Em Zanlorenci e Burnett (1998), é proposta uma forma de identificação da fonte de informação como fator de qualificação de agente consumidor ou produtor da informação, observando o ponto de vista em relação ao requisito e ao grau de exigência do mesmo. Esse critério é utilizado para avaliação de riscos de implementação de requisitos, em função da representatividade do universo dos *stakeholders* e a exigência do requisito.

Além disso, o esforço de representação dos requisitos também está orientado no registro de demanda de negócio e na identificação das regras do negócio.

Pela própria natureza dos requisitos e pelos relacionamentos entre eles, embora tenha disponíveis as mais variadas técnicas, a geração do documento de requisitos poderá conter informações que reflitam conflitos de requisitos, omissões, inconsistências e, principalmente, o caráter anacrônico dos fatos.

Mas a forma apropriada de representar os requisitos deve atender principalmente ao princípio básico da comunicação do conteúdo e de entendimento comum, na linguagem dos *stakeholders*.

3.3 Técnicas de Elicitação de Requisitos

Para uma boa elicitação de requisitos, existem técnicas que proporcionam melhor entendimento e comunicação entre clientes e analistas, para que problemas não ocorram, ou se ocorrerem, que sejam mais facilmente resolvidos.

Segundo Jackson (1995), uma técnica deve explorar as características específicas do problema e como as características do problema variam muito, faz-se necessário um repertório de métodos de classe de problema (SIDDIQI,1997). O problema da elicitação de requisitos não pode ser

resolvido com uma abordagem puramente tecnológica, porque nesta fase o contexto social é mais crucial do que na fase de programação, especificação e projeto.

A seguir são apresentadas as principais técnicas de elicitação de requisitos. Serão descritas suas características de forma simplificada, embora possa variar o grau de eficiência das técnicas dependo do contexto, tipo do requisito, amplitude do universo de informação e características do sistema a ser construído.

3.3.1 Casos de Uso (Use Cases)

Casos de Uso é uma técnica baseada em cenários para obtenção de requisitos, que foram introduzidos pela primeira vez no método *Objectory* (KRUCHTEN, 2000). Eles se tornaram uma característica fundamental da notação em UML para descrever modelos de sistemas orientados a objetos (KRUCHTEN, 2000).

Para criar um *Caso de Uso*, o analista deve primeiro identificar os diferentes tipos de pessoas (ou dispositivos) envolvidos no sistema ou produto. Esses *atores*, na verdade, representam papéis que as pessoas (ou dispositivos) desempenham quando o sistema opera. Definido de um modo um tanto mais formal, um ator é qualquer coisa que se comunica com o sistema ou produto e que não faz parte dele. No exemplo abaixo (Figura 3.1), o *Vendedor* está relacionado ao caso de uso *Emissão de Nota Fiscal*; isto indica que de alguma forma esse *stakeholder* tem participação direta ou indireta neste caso.

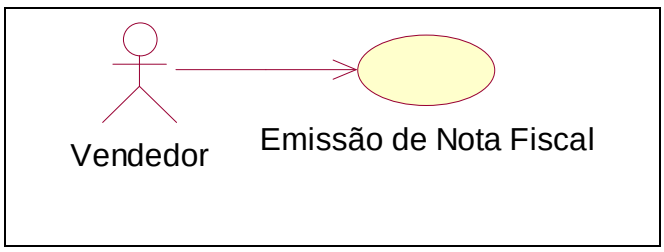

Figura 3.1 - Exemplo de Caso de Uso

3.3.2 Observação

A *Observação* possibilita um contato pessoal e estreito do pesquisador com o fenômeno pesquisado, o que apresenta uma série de

vantagens (DAMIAN, 1997). As técnicas de observação são extremamente úteis para “descobrir” aspectos novos de um problema. Isto se torna penoso nas situações em que não existe uma base teórica sólida que oriente a coleta de dados. Ao mesmo tempo em que o contato direto e prolongado do observador com a situação pesquisada apresenta as vantagens mencionadas, envolve também uma série de problemas. Algumas críticas são feitas ao método de observação, primeiramente por provocar alterações no ambiente ou o comportamento das pessoas observadas. Outra crítica é a de que este método é baseado, excessivamente, na interpretação pessoal (DAMIAN, 1997).

Embora seja o método mais aplicado na elicitação dos requisitos, é preciso cautela e inteligência por parte do *engenheiro de requisitos*, para não levar a uma visão distorcida dos fenômenos ou a uma representação parcial da realidade.

3.3.3 Entrevista

Entrevista é uma técnica de elicitação de requisitos também muito usada. O *engenheiro de requisitos* discute com diferentes usuários e, a partir da discussão elabora um entendimento de seus requisitos. Há basicamente, segundo Kotonya e Sommerville (1998), dois tipos de entrevista:

a) Entrevistas fechadas, em que o engenheiro de requisitos procura as perguntas para um conjunto pré-definido de questões;

b) Entrevistas abertas, em que não há agenda pré-definida e o engenheiro de requisitos discute de modo aberto, o que os usuários querem do sistema.

Entrevistas podem ser positivas para desenvolver um entendimento do problema e para elicitar muitos requisitos gerais do sistema.

Portanto, as entrevistas são efetivas para o entendimento do domínio da aplicação e das questões organizacionais que afetam os requisitos.

3.3.4 Análise de Protocolo

A *Análise de Protocolo* é uma técnica que pode ser usada por formar uma compreensão de usuários, pois é usualmente eficiente para descrever seus trabalhos e as dificuldades que enfrentam de forma relativamente natural, entretanto podem ter expectativas não realistas sobre o suporte que o computador dará.

Pede-se aos usuários para se engajarem em alguma tarefa e falar sobre esta, explicando seu pensamento do processo, enquanto um observador registra os dados verbais. Esse tipo de linguagem pode ser considerado uma "verbalização direta do processo cognitivo específico" (ERICSSON; SIMON, 1993).

Entretanto, a análise do protocolo não é um guia confiável sobre o que as pessoas estão pensando, pois ela está sujeita a problemas de interpretações pelo *engenheiro de requisitos*. Segundo Goguen (1997), a restrição em estudar protocolos é que as pessoas podem produzir linguagens que oferecem um perfil de atividade cognitiva autônoma. É sabido que os clientes têm conhecimento sobre negócios e necessidades organizacionais, enquanto a equipe ligada aos requisitos tem conhecimento sobre as possibilidades técnicas. Por isso é exigida, por parte do *engenheiro de requisitos*, certa experiência sobre o contexto observado.

Depois de cada sessão realizada, os dados verbais são analisados, e o processo de pensamento cognitivo dos usuários na execução das tarefas pode ser comparado ao real processo do sistema. Assim, é possível serem identificadas as forças e as fraquezas dos sistemas (ERICSSON; SIMON, 1993).

3.3.5 Joint Application Development (JAD)

JAD é uma marca registrada da IBM. O objetivo principal é colocar autoridades representativas e gerenciais juntas dentro de um workshop estruturado para promover decisões. Segundo Damian (1997), consiste em cinco fases: *definição do projeto; pesquisa; preparação para a sessão JAD; sessão JAD, e documento final.*

As fases de definição de projeto e pesquisa no processo *JAD* lidam com a coleta de informações. São usadas para validar as informações recolhidas nas fases anteriores. O processo concentra-se na sessão *JAD*, e deste modo, contribui para a elicitação de requisitos como significado para validar as informações já colhidas. Em cada sessão, as pessoas certas têm que estar envolvidas, e a presença de um facilitador pode ajudar a manter a sessão focalizada e minimizar ataques e defesas emocionais improdutivas. *JAD*

promove cooperação, entendimento, e trabalho em grupo entre os vários grupos de usuários e o pessoal de sistemas de informação (DAMIAN, 1997).

3.3.6 Participatory Design (PD)

Tradicionalmente, valores democráticos, os quais têm sido levados em conta no processo de projeto, têm sido somente aqueles concentrados em fatores técnicos e econômicos. Mas com o uso do *PD*, fatores técnico-sociais têm sido levados em conta. O projeto em que se aplica *PD* deve ser feito com o usuário, pois o aprendizado mútuo deve ser parte importante do trabalho em um grupo de projeto, não sendo meramente uma visita aos laboratórios de pesquisa (DAMIAN,1997). Usuários e clientes são inteligentes e criativos, colaboradores produtivos para as organizações, desde que sejam encorajados a expressarem seus desejos, aplicarem sua esperteza e exercitarem suas capacidades de tomar decisões, assumindo responsabilidades do impacto de suas ações.

3.3.7 Quality Function Deployment (QFD)

O termo qualidade é definido como *um conjunto de meios para produzir economicamente produtos ou serviços, os quais satisfaçam os requisitos do cliente*. *QFD* é um conceito que provê meios de interpretar requisitos do cliente em requisitos técnicos, apropriados para cada estágio do desenvolvimento e produção do produto (DAMIAN, 1997). As fases iniciais do *QFD* podem ser descritas como sendo simplesmente um sistema de identificação e priorização das necessidades do cliente obtidas de cada recurso avaliado.

Tal conceito é aplicado para a elicitação de requisitos, especialmente em um modelo de elicitação onde a voz do cliente é o guia para a criação de requisitos.

3.3.8 Cooperative Requirements Capture (CRC)

Como definido em Sommerville e Sawyer (1997), *CRC* é uma sessão de grupo, que é similar ao *JAD*, em que os papéis dos participantes e o papel do facilitador são claramente definidos. Em *CRC*, participantes consistem não somente de usuários e facilitador, mas também de outras pessoas

envolvidas indiretamente no sistema. É diferente de *JAD* e *QFD*, pois foca o usuário operativo.

3.3.9 Prototipação

A técnica de *Prototipação* é utilizada para a avaliação e validação do usuário. O conjunto inicial de requisitos é usado como base para criar o protótipo de interface do usuário com o sistema inicial (simplificado). Para essa criação, o projetista precisa manter o protótipo tão simples quanto possível. O ponto forte desta atividade é apresentar muitas alternativas para o usuário antes de gastar muito esforço para qualquer protótipo em particular. Após a aceitação do protótipo pelos usuários, os desenvolvedores precisam criar um documento de especificação dos requisitos paralelo ao protótipo de interface (SOMMERVILLE, 2003).

3.3.10 Scenarie (?) Requirements Analysis Method - (SCRAM)

Usuários finais e outros agentes do sistema acham a utilização de cenários mais fácil para relacionar aos exemplos da vida real do que descrições abstratas das funções realizadas pelo sistema. Por essa razão, torna útil desenvolver um conjunto de interação dos cenários, e usar estes para elicitar os requisitos do sistema. Cenários são exemplos de sessões de interação as quais são concentradas com um tipo único de interação entre um usuário final e o sistema. A interação dos usuários finais através da aplicação de cenários intensifica o grau de conhecimento, ao time de *engenheiros de requisitos*, sobre o que estão fazendo e a informação que eles precisam do sistema para descrever a tarefa expressa no cenário (SUTCLIFFE, 1998).

Em *SCRAM* são criados enredos para representar caminhos de possível comportamento de um caso de uso e estes são investigados para elicitar os requisitos. É importante ressaltar que os artefatos gerados por esta técnica também podem ser utilizados na documentação definitiva dos requisitos (SUTCLIFFE ; RYAN, 1998).

3.3.11 User Stories (Extreme Programming - XP)

Uma *User Story* é a menor quantidade de informações (uma etapa) necessárias para que o cliente defina um caminho através do sistema. São

através das *User Stories* que são elicitados os requisitos do sistema. Todas as funcionalidades do sistema são levantadas através dessa técnica que são registradas em pequenos cartões (BECK, 2000; ASTELS; MILLER ; NOVAK, 2002). Segundo Beck (2000), esses cartões são escritos pelo cliente, por algumas razões:

a) *Ao escrever uma User Story, o cliente é forçado a pensar melhor na funcionalidade, pois ele formaliza o pensamento e analisa melhor o assunto sobre o qual irá tratar, sendo recomendáveis User Stories simples e curtas, com apenas o suficiente para transmitir a idéia desejada;*
b) *Quando o cliente escreve o cartão, cria-se um vínculo. Trata-se de um vínculo psicológico, mas de grande importância, fazendo o cliente sentir responsabilidade sobre aquilo que está sendo solicitado;*
c) *Através do cartão, o cliente compreende que existe um custo em tudo o que é pedido, ou seja, tenha consciência deste custo para que saiba selecionar aquilo que deve ou não ser solicitado a cada momento, ou que possa ser priorizado.*

Um conjunto inicial de *User Story* é criado no início do projeto. Em seguida, novas *User Stories* podem ser criadas como resultado de novos requisitos. Elas também podem objetivar o refino dos itens a serem entregues à medida que novas expectativas de funcionalidades existentes são descobertas.

A equipe de desenvolvimento utiliza os cartões para saber quais são as funcionalidades desejadas pelo cliente (TELES, 2004). Os desenvolvedores escolhem as *User Stories* que irão implementar a cada dia de trabalho. Contudo, quando o projeto é muito grande, os cartões podem representar um grande esforço para serem implementados. Neste caso, é comum dividir os cartões em *tarefas,* que são registradas em novos cartões; assim os desenvolvedores implementam *tarefas* ao invés de *User Stories* (TELES, 2004).

3.3.12 Brainstorming

O *Brainstorming* é uma das diversas técnicas de reuniões de grupo, provavelmente a mais antiga e a mais conhecida. O princípio 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 (KIRAWOWSKI, 1997). As idéias sugeridas e exploradas nestes encontros não devem ser criticadas ou julgadas. A técnica pode ser aplicada no início da fase de elicitar requisitos quando pouco do projeto é conhecido e são necessárias idéias novas. O resultado de uma sessão de *Brainstorming* bem sucedida é um conjunto de boas idéias, e propicia a sensação de que todos participaram da solução do problema.

É uma técnica particularmente efetiva para ser aplicada à concepção de um sistema ou na exploração e entendimento do potencial de mercado para produto ou sistema a ser construído (KIRAWOWSKI, 1997).

3.3.13 Pesquisa Contextual (Contextual Inquiry - CI)

A *CI* ou Pesquisa Contextual é uma técnica utilizada para examinar a compreensão dos usuários sobre um sistema já utilizado ou a ser desenvolvido, seu ambiente de trabalho, as tarefas por ele realizadas, suas preferências e assuntos de interesse. Pode ser usada para conhecer e analisar as necessidades dos usuários e realizar a análise de tarefa (HOLTZBLATT; JONES, 1993).

A aplicação desta técnica é apropriada sempre que o projetista precisar desenvolver ou comunicar a interpretação dos usuários sobre um sistema existente ou proposto. Administrar aplicação da *CI* requer visita a vários usuários no local de trabalho, observando-os, levando em consideração a análise das tarefas e o registro dos dados resultantes em documento (HOLTZBLATT; JONES, 1993).

É um método relacionado ao campo da entrevista estruturada, por juntar dados organizados da prática de trabalho, enquanto gera conhecimento desarticulado sobre o trabalho de forma explícita ao entendimento do *engenheiro de requisitos*, aproximando-se dos detalhes de baixo nível relacionada aos trabalhos habituais e na maioria das vezes invisíveis. *CI* está baseada em alguns princípios básicos:

- *Entender o contexto no qual um produto é usado;*
- *Considerar o usuário como um membro do projeto;*
- *Deve haver um foco para processo de projeto relativo à usabilidade;*

- *Interpretação dos fatos.*

Contextual Inquiry é mais um processo de descoberta do que um processo evolutivo. Também é mais um processo de aprendizado do que uma forma de testar ou validar dos requisitos (BEYER; HOLTZBLATT, 1998).

3.3.14 Etnografia

A abordagem etnográfica tem o objetivo de entender uma cultura não familiar - o conhecimento, técnicas e práticas que a constituem - de forma a traduzi-la de maneira que possa ser entendida e usada por outros. O *engenheiro de requisitos* tem a necessidade de documentar o domínio do sistema e a sua relação com a atividade de cada pessoa envolvida no seu funcionamento (SOMMERVILLE *et al.*, 1993).

Para que se consiga extrair o máximo de conhecimento possível das pessoas, é necessário se comunicar com os usuários utilizando a sua própria linguagem e não uma linguagem técnica de engenharia de software que é incompreensível e intimadora para a maioria delas. Posteriormente, a equipe de desenvolvimento deve ser capaz de usar todos os dados obtidos para que possa desenvolver o produto realmente apropriado, correspondente com a informação recolhida, que se adapte completamente às necessidades dos utilizadores e seja perfeitamente integrado no seu ambiente (MILLEN, 2000).

Essa técnica também colabora na identificação das interfaces do sistema e no agrupamento de requisitos dos usuários quando o sistema é amplo e complexo.

Embora tenha sua eficiência, o estudo etnográfico requer muito mais tempo do que as técnicas de identificação de requisitos mais comuns, logo todos os recursos financeiros e temporais, devem ser utilizados da forma mais otimizada possível (SOMMERVILLE *et al.*, 1993).

3.3.15 Soft System Methodology (SSM)

O método *SSM* estabelece uma compreensão do problema dentro de uma organização, estruturando a situação sob as perspectivas de todos os sistemas pertinentes. Entre esses sistemas encontram-se pessoas, procedimentos, políticas, hardware e software (WILSON; DUFFY, 2001).

Esta ampla perspectiva faz dessa abordagem utilizável em elicitação de requisitos, combinado com outros métodos de análise dos requisitos, facilitando a identificação do problema e na sugestão de soluções (KOTONYA; SOMMERVILLE, 1998).

3.4 Técnicas de Elicitação Orientadas pelo Ponto de Vista

Algumas técnicas de elicitação se orientam pela observação e pelo ponto de vista dos engenheiros de requisitos. A noção de pontos de vista como meio de organizar e estruturar a atividade de *ER* tem sido proposta por diversos métodos (LEITE, 1989; LAPPE et AL, 2004). Contudo, cada um dos métodos adota uma conceituação diferente para ponto de vista.

Em seguida, algumas das principais abordagens serão caracterizadas de acordo com suas potencialidades no apoio às atividades relacionadas à *ER*, e que contribuíram nas observações do pesquisador para captura dos padrões candidatos.

Segundo a proposta de Kotonya e Sommerville (1996), pontos de vista podem ser classificados em duas grandes categorias:

- **Pontos de vista diretos** - Correspondem aos clientes recebendo serviços do sistema e enviando-lhe dados e informações de controle. Representam operadores/usuários ou subsistemas que interagem com o sistema analisado;
- **Pontos de vista indiretos** - Têm interesses em alguns ou todos os serviços oferecidos pelo sistema, porém não interagem diretamente com ele. Pontos de vista indiretos geram requisitos que restringem os serviços oferecidos a pontos de vista diretos (geram requisitos não funcionais). São difusos, indo desde pontos de vista de engenharia (cujo foco de interesse são: o projeto e a implementação do sistema), passando por pontos de vista organizacionais (que focalizam a influência do sistema na organização) até pontos de vista externos (com preocupações relacionadas às influências do sistema em seu ambiente externo). Exercem influência significativa no contexto organizacional e, freqüentemente, representam a força que decide sobre a continuidade do sistema ou grandes alterações a que devam ser submetidos

os sistemas antes de entrarem em operação. Assim, segundo Kotonya e Sommerville (1996), o processo de *ER* estará incompleto caso sejam desconsiderados os pontos de vista indiretos.

3.4.1 CORE (Controlled Requirement Expression)

CORE é um método aplicável na necessidade de expressar os requisitos elicitados, que é apoiado por notações e diagramas simples, que podem representar muitos pontos de vista vitais dos requisitos de sistema. Define os passos em produção de uma especificação de requisitos, com ênfase particular em "*começo - para cima"* e a ligação entre os passos. Para cada passo, *CORE* define também as atividades envolvidas e as verificações a ser aplicadas (Mullery, 1979). Assim como outras técnicas, o *CORE* torna-se útil no agrupamento dos requisitos elicitados, dando melhor compreensão às informações e aos dados obtidos por outras abordagens.

3.4.2 SADT (Structured Analysis and Design Technique)

É uma metodologia de engenharia de software proprietária para análise e representação de requisitos e projetos de software. *SADT* é composto de uma linguagem gráfica e de um método para seu uso. Um modelo *SADT* é organizado por diagramas de seqüências, cada um com suporte textual (MARCA; McGOWAN, 1988).

SADT também define os papéis das pessoas envolvidas em um projeto de software (MARCA; McGOWAN, 1988). Em *ER* , *SADT* também é usado para análise e documentação de requisitos de software.

3.4.3 VOSE (Viewpoint-Oriented Software Engineering)

Em *VOSE,* os pontos de vista são organizados em configurações (coleções de pontos de vista relacionados) para solucionar conflitos entre requisitos dos sistemas. Modelos e *templates* são usados para descrever os pontos de vista.

Uma configuração para um problema hipotético no domínio pode consistir em modelos com diferentes estilos (observando a mesma partição do

domínio de problema) ou modelos com o mesmo estilo (observando partições diferentes do domínio de problema) (KOTONYA ; SOMMERVILLE, 1998).

O sistema final é construído depois de combinar todas as configurações com possíveis soluções para o problema. Em *ER*, *VOSE* é mais eficiente na execução das análises de alto nível com requisitos iniciais.

VOSE é um *framework* baseado em ponto de vistas para integração de métodos de desenvolvimento. A técnica existente em *VOSE* é aquela em que o desenvolvimento de software envolve a participação de muitos especialistas em vários aspectos do desenvolvimento de software e da área de aplicação. Neste modelo, são usados pontos de vista para capturar o papel e a responsabilidade de determinados participantes em fases específicas do desenvolvimento do sistema (FINKELSTEIN, 1992).

3.4.4 VORD (Viewpoint-Oriented Requirement Definition)

VORD pode ser usado por analisar um sistema por múltiplas perspectivas para capturar todos os requisitos possíveis. Primeiramente, os pontos de vista pertinentes são identificados, e detalhados em *"sub"* pontos de vista. Então, cada ponto de vista elicitado é organizado, avaliado por grau de importância, documentado, analisado e especificado. *VORD* também pode ser utilizado para documentar os requisitos do software e de todo o sistema de informação (KOTONYA, 1999).

O modelo adotado pelo ponto de vista *VORD*, é orientado a serviços, em que pontos de vista são análogos para os clientes em um sistema de arquitetura cliente-servidor. O sistema entrega os serviços aos pontos de vista e os pontos de vista repassam as informações de controle e de parâmetros, associados ao sistema. *VORD* define dois tipos principais de pontos de vista; diretos e indiretos. Pontos de vista diretos correspondem diretamente aos clientes em como eles recebem os serviços do sistema. Pontos de vista indiretos correspondem a entidades que não interagem diretamente com o sistema proposto, mas têm algum interesse no processo ou em alguns dos serviços providos (KOTONYA; SOMMERVILLE, 1996).

3.4.5 VORV (Viewpoint-Oriented Requirements Validation)

VORV usa pontos de vista como um meio para validação de requisitos iniciais. *VORV* está baseado na noção que requisitos podem ser validados através da construção de visões do sistema de pelo menos duas perspectivas diferentes de analistas, usando notações diferentes e então comparando os resultados (LEITE, 1989).

O objetivo do método é identificar e classificar problemas relacionados à completeza e perfeição.

3.5 Problemas Encontrados Durante a Elicitação de Requisitos de Software

Durante a elicitação de requisitos, e até mesmo no início da elicitação, aparecerão muitos problemas. No início da elicitação, há o problema da escolha dos usuários a serem entrevistados, pois uma escolha errada pode levar à perda de tempo e prejuízos financeiros para o desenvolvedor e para os clientes. Já durante a elicitação aparecerão problemas de escopo, de entendimento e de volatilidade.

Além dos problemas citados, existem dois grandes grupos de problemas: *problemas acidentais e problemas essenciais* (FAULK, 1997).

a) **Problemas Acidentais**: são aqueles oriundos da falta de controle sobre aquilo que precisa ser construído, dentre os quais podem ser destacados: pouco esforço despendido no levantamento de informações junto ao usuário; documentação pobre sobre os requisitos obtidos, pouca revisão dos requisitos obtidos; especificações incorretas e tendência em iniciar logo o processo de desenvolvimento do software (MARTINS, 1999), o que leva o usuário a não estar satisfeito com o resultado final do projeto. As dificuldades acidentais, segundo Faulk (1997), são:

   - *Documentação escrita como uma reflexão tardia: isto continua sendo uma prática comum, em que a documentação de requisitos é desenvolvida depois que o software tenha sido escrito. Tal procedimento inevitavelmente viola o princípio da definição do "o que" o sistema deve fazer antes de "como".*
   - *Não projetada para ser proveitosa: freqüentemente, na vontade de implementar; logo, pouco esforço é despendido para projetar, escrever, checar, ou gerenciar a administração da criação e evolução*

do documento de especificação. O resultado mais óbvio é uma organização pobre do software. O documento resultante não é uma referência técnica efetiva. A especificação é difícil para usar e difícil para manter.

b) **Problemas Essenciais**: são aqueles inerentes à elicitação de requisitos, dentre os quais podem ser destacados: dificuldades do usuário em saber efetivamente o que ele quer, dificuldade de comunicação entre usuários e desenvolvedor e a natureza mutante dos requisitos (MARTINS, 1999). Segundo Faulk (1997), os problemas essenciais seriam causados por dois fatores primordiais:

- *Compreensão: pessoas não sabem o que elas querem. Isto não significa que as pessoas não tenham uma idéia geral do que o software fará. Ao invés disso, ele não começa com um preciso e detalhado entendimento de que as funções pertencem ao software, o que as saídas devem fazer para cada possível entrada, quanto tempo cada operação deveria levar, como uma decisão afetaria a outra, e assim por diante. Muitas decisões ainda não foram tomadas e, expectativas mudarão na medida em que o problema é mais bem entendido.*
- *Comunicação: requisitos de software são difíceis para se comunicar efetivamente. A dificuldade inerente da comunicação é composta pela diversidade de pessoas e audiências para a especificação de requisitos.*

3.6 Diferenças Entre os Problemas

É possível afirmar que os problemas acidentais são mais fáceis de ser evitados, dependendo apenas da utilização das fases da ER. Porém, os problemas essenciais envolvem a comunicação entre pessoas e o processo deve levar em conta o contexto, as habilidades pessoais do entrevistado, o lado psicológico, ou seja, apenas uma abordagem tecnológica não poderá solucionar esses problemas, pois os aspectos sociais assumem grande importância na elicitação dos requisitos, por isso os aspectos cognitivos devem ser considerados como uma ferramenta para se evitar acidentes. (MARTINS, 1999).

3.7 Resumo

Neste capítulo, foram fundamentadas as várias abordagens de tratamento dos requisitos em termos de avaliação, processos e técnicas aplicáveis à fase de *elicitação de requisitos,* além de uma consistente e ampla conceituação sobre o tema, incluindo os problemas encontrados durante o processo. Também foram apresentadas as várias visões correspondentes à observação e busca dos requisitos durante o processo de elicitação de requisitos.

CAPÍTULO 4 - PADRÕES

...the key to reusable software is to reuse analysis and design; not code (James M. Neighbors Software Construction Using Components University of California, Irvine, 1980).

Nesta etapa da fundamentação, estão detalhadas as várias abordagens sobre padrões. Trata da linguagem e dos elementos essenciais de padrões. Aborda o descobrimento e a classificação de padrões relativas às fases de produção de software. Explora os conceitos de reutilização de padrões aplicados aos estudos da engenharia e de requisitos de software. Por fim, o autor apresenta os caminhos e fontes para busca de conhecimento sobre padrões aplicados a projetos e requisitos.

4.1 Introdução

Padrões podem ser definidos como o encapsulamento da descrição abstrata e estruturada de uma solução satisfatória para um problema que ocorre repetidamente dentro de um contexto, dado um conjunto de forças ou restrições que atuam sobre o problema (GAMMA *et al*., 1995).

Padrões visam a melhorar e formalizar a documentação de experiência, de conhecimento. Têm o objetivo de registrar de maneira estruturada as soluções que pessoas mais experientes encontraram ao se defrontar com problemas repetitivos, para que possam ser transmitidas de maneira mais eficiente a pessoas menos experientes, para evitar que essas passem pelas mesmas dificuldades, ou que cometam os mesmos erros que os primeiros, mais experientes, cometeram. Padrões são identificados por nomes, que, com o tempo, vão sendo incorporados no vocabulário das pessoas (LARMAN, 2004).

4.2 Padrão e Padronização

A organização racional do trabalho não se preocupa somente com a análise do trabalho, estudo dos tempos e movimentos, fadiga dos funcionários, divisão e especialização do trabalho, e com os planos de incentivos salariais. Vai mais além, ou seja, preocupa-se também com a padronização dos métodos

e processos de trabalho, com a padronização das máquinas e equipamentos, ferramentas e instrumentos de trabalho, matérias-primas e componentes, no intuito de reduzir a variabilidade e a diversidade no processo produtivo e, a partir daí, eliminar o desperdício e aumentar a eficiência (CHIAVENATO, 2003).

Com isso, a padronização passa a ser vital para a administração científica na melhoria da eficiência. A padronização conduz à simplificação na medida em que a uniformidade reduz a variabilidade e as exceções que complicam o processo produtivo (CHIAVENATO, 2003).

No âmbito da computação, padrão é uma unidade de medida adotada e aceita comumente como critério. Já a padronização é a aplicação de padrões em uma organização para obter uniformidade e redução de custos.

Segundo Gama *et al (2005)*, um padrão descreve um problema recorrente e a essência da solução geral para o problema a ser aplicada em um contexto particular, acessível ao público em geral, através de publicações, expressa a relação entre certo contexto, e contém um sistema de forças que ocorre repetidamente nesse contexto, conforme mostra a Figura 4.1.

Figura 4.1 - Descrição sobre o problema e solução

A "**solução geral para o problema"** significa que cada *padrão* identifica e descreve uma solução e o problema que aquela solução resolve. A "**essência da solução"** significa que apenas os elementos essenciais são descritos, deixando os aspectos específicos da geração para a pessoa que utiliza o *padrão*. Esses aspectos específicos tendem a ser dependentes da situação específica. O "**problema recorrente"** significa que o problema deve ter ocorrido várias vezes de modo a viabilizar o esforço a ser investido na descrição da solução. E "**em um contexto"** significa que aquela solução é reconhecida como válida em um contexto particular.

Na *Engenharia de Software*, padrões encapsulam as melhores soluções baseadas em anos de desenvolvimento de aplicações, observação e experiência. Para encontrar a melhor solução, o desenvolvedor deve entender o problema, o contexto e as forças que governam esse problema (HARRISON; FOOTE; ROHNERT, 1999), sendo que nenhum padrão é uma entidade isolada e cada um é apoiado por outros padrões (ALEXANDER, 1979). Dessa forma, os padrões ajudam na construção de sistemas confiáveis, seguindo os passos de outras construções de sistemas de sucesso (HARRISON; FOOTE; ROHNERT, 1999).

4.3 Conceito de *Design Patterns*

Os conceitos de padrões e linguagens de padrões começaram a ser introduzidos na comunidade de software no fim dos anos 80 e no início dos anos 90, mas foram realmente conhecidos e popularizados com a publicação dos *Design Patterns*, da chamada Gang-of-Four (GoF), formada pelos pesquisadores Erich Gamma, Ralph Johnson, Richard Helm e John Vlissides (*GAMMA et al.*, 1995). Eles criaram um catálogo de padrões para o projeto de software orientado a objetos, documentando as suas experiências na resolução de problemas de projeto independentes do domínio da aplicação. O catálogo registra diversas soluções satisfatórias que encontraram durante o desenvolvimento de *frameworks*, relacionados principalmente por problemas de reusabilidade, flexibilidade, modularidade etc.

4.4 Elementos e Formas de Padrões

As três formas mais utilizadas pela comunidade estão descritas nos subitens a seguir. A forma Alexander é mais textual e mais apropriada para padrões mais abstratos, enquanto a forma Gamma é mais granulada e mais apropriada para padrões mais próximos de projeto e implementação de software. A forma Portland é mais referencial e documental, e a forma Coplien é intermediária entre os formatos apresentados.

4.4.1 Forma Alexander

A descrição de um *padrão* é iniciada com a reprodução de **uma fotografia** que descreve um exemplo de uma aplicação do *padrão*. Em

seguida, descreve-se um parágrafo inicial que estabelece o **contexto** do *padrão*. Este parágrafo explica como o padrão ajuda a completar outros padrões mais abrangentes (os padrões "maiores"). Em negrito, é descrito então um parágrafo com a **essência do problema**. Em seguida é descrito o problema, através de uma discussão como evidências empíricas de sua validade, as várias formas nas quais o *padrão* pode se manifestar nas construções e outros aspectos.

Posteriormente, é descrita a **solução**. Esta solução é formada pelos relacionamentos físicos e sociais que são necessários para resolver o problema no contexto mencionado. A descrição é sempre na forma de instruções. Esta solução é então ilustrada com um **diagrama da solução**. Finalmente, a descrição é terminada com a relação dos **padrões "menores"** que são necessários para completar ou embelezar o *padrão* (ALEXANDER, 1979).

A maior parte dos padrões Alexandrinos é descrita da seguinte maneira:

> *"SE você encontra em um **CONTEXTO**,*
> *por exemplo, **EXEMPLO**, com o **PROBLEMA***
> *submetido a determinadas **FORÇAS***
> ***ENTÃO** por algumas **RAZÕES***
> *aplique **REGRAS** e/ou **FOMATOS DE PROJETO***
> *para construir a **SOLUÇÃO***
> *convergindo para um **NOVO CONTEXTO** e **OUTROS PADRÕES".***

4.4.2 Forma Portland

Apresenta um conjunto de páginas descrevendo linguagens de padrões e traz referências a outras páginas. Contém três seções (PORTLAND, 2007):

- **Documento de Linguagem**: contém um sistema de padrões que operam em conjunto;
- **Parágrafo do Padrão**: segue uma estrutura bem natural de descrição de padrão, como: "Deparando-se com este problema..."; "O problema existe por causa disto e as forças a serem resolvidas são..."; Portanto: "Aja aproximadamente da seguinte maneira..."; "Agora você está preparado para tratar do seguinte problema...".

- **Sumário**: dentro da linguagem, existem grupos de padrões com idéias similares e, nos sumários, são explicitadas as entrelinhas da criação do padrão.

4.4.3 Forma Coplien

Esta forma apresenta seis seções (COPLIEN, 1998):

- **Problema:** descreve o problema a ser resolvido;
- **Contexto:** descreve o contexto no qual a solução descrita resolve o problema;
- **Forças:** apresentam o conjunto de forças que atuam no problema;
- **Solução:** descreve a solução do problema descrito;
- **Contexto Resultante:** descreve o contexto resultante após a aplicação da solução;
- **Racionalidade:** descreve uma racionalidade e exemplos que justificam a solução.

4.4.4 Forma Gamma

É o mais extenso, e obedece a um gabarito com os seguintes itens (GAMMA *et al, 1995)*:

- **Nome e Classificação do Padrão**: o nome expressa a essência do padrão de forma sucinta. Um bom nome é vital porque ele se tornará parte do vocabulário de projeto. A classificação do padrão é uma forma de categorizar sua aplicação.
- **Intenção:** descreve a intenção do *padrão*. Esta descrição deve responder perguntas, tais como, o que o *padrão* faz e qual o problema que ele resolve.
- **Motivação:** descreve um cenário que ilustra o problema e como a estrutura de classes e objetos do *padrão* resolve o problema. Normalmente é descrito também um exemplo concreto para motivar a importância do *padrão*.
- **Aplicabilidade:** descreve as situações nas quais o *padrão* deve ser aplicado. Detalha também exemplos de problemas com outras soluções para o problema resolvido pelo *padrão*.

- **Estrutura:** descreve uma representação gráfica da estrutura de classes e objetos que compõem o *padrão*. Normalmente é aplicado o padrão da OMT.
- **Participantes:** descrevem as classes e objetos que participam do *padrão* e suas respectivas responsabilidades.
- **Colaborações:** descrevem como os participantes colaboram para a solução do problema.
- **Conseqüências:** descrevem as conseqüências do *padrão*, inclusive possíveis limitações da solução.
- **Implementação:** descreve sugestões para a implementação do *padrão*.
- **Usos conhecidos:** descrevem exemplos de utilização do *padrão* em sistemas de software conhecidos.
- **Padrões relacionados:** bem parecido com a idéia de linguagem de padrões do formato Alexandrino. Quais os *padrões* mais intimamente são relacionados entre si.
- **Exemplo de código:** fragmentos ou blocos de códigos para ilustrar a implementação.
- **Também conhecido como:** os apelidos e sinônimos do *padrão*, se existir.

4.4.5 Outros Elementos

Outros elementos podem ser adicionados na descrição de padrões, para melhorar seu entendimento ou simplesmente para fornecer informações adicionais que possam ser relevantes para um determinado contexto ou domínio da aplicação. Alguns desses elementos seriam:

- **Exemplos**: ilustrações de contexto e problema concretos, situações reais em que o problema genérico é encontrado. Exemplos descrevem também a aplicação da solução concreta, e também porque outras soluções que poderiam ser usadas não são melhores do que a aplicada. Exemplos descrevem a parte concreta de um padrão, o que é importante, visto que as pessoas se sentem mais à vontade com elementos concretos do que abstrações. Além disso, segundo John Vlissides (VLISSIDES, 1998), *(...) as pessoas parecem entender melhor os conceitos quando eles são*

primeiramente apresentados em termos concretos, e depois em termos abstratos.

- **Justificativa**: justificativa da escolha da solução para o problema, mostrando como a solução resolve as forças impostas sobre o problema. Pode também citar outras possíveis soluções que são menos satisfatórias do que a escolhida.
- **Relacionamentos**: dentro de uma linguagem, padrões estão relacionados de diversas maneiras, para resolver o problema mais genérico endereçado. Padrões podem também estar ligados com outros que estão fora do contexto de uma linguagem, pertencentes a outras linguagens ou descritos isoladamente. Esses relacionamentos são dados de diversas maneiras (MESZAROS; DOBLE, 1997), por exemplo, um padrão pode levar a outro que resolve um problema gerado pelo primeiro ou que resolve um problema mais específico.

Ainda em Meszaros e Doble (1997), outros elementos opcionais algumas vezes encontrados em padrões são *Indications* (sintomas do problema), *Code Samples* (amostras de implementação da solução, como sugerido por Gamma *et al* (1995)), *Aliases* (outros "nomes" pelos quais um padrão é conhecido, na mesma idéia de "também conhecido como" (GAMMA *et al* ,1995)), *Acknowledgements* (agradecimentos a pessoas que contribuíram na elaboração do padrão).

A utilização de qualquer formato apresentado deve levar em conta, durante o momento da análise do domínio da aplicação, o que exatamente o sistema faz, e o que se pretende construir.

4.5 Classificação de Padrões

Diversos padrões encontrados na literatura foram estudados e analisados. Para facilitar o estudo e a aplicação desses padrões em um processo de desenvolvimento, eles foram agrupados em categorias. Para os padrões de Engenharia de Software, estendem-se aqui as categorias citadas por Buschmann *et al.* (1996), incluindo as categorias padrões de processo, requisitos, análises, arquiteturais, projetos, implementação (idiomas), persistência de dados e de testes:

Quadro 4.1 – Tipos de Padrões.

Tipos de Padrões	**Descrição dos Padrões**
Processos	Conduzem o desenvolvimento de software, descrevendo uma abordagem ou série de ações aprovadas e de sucesso para o desenvolvimento de software (AMBLER, 1998).
Requisitos	É um *framework* para requisitos que suporta as necessidades de um determinado produto (software), minimizando a distância da compreensão que poderão causar falhas no sistema, além de suportar diversos projetos e implementações (WITHALL, 2007). Também é capaz de suportar várias técnicas de elicitação e especificação de requisitos. O *framework* disponibiliza ferramentas que suportam a reusabilidade de artefatos utilizados em outros projetos, de acordo com as necessidades do projeto atual e da experiência do *engenheiro de requisitos* (ROBERTSON, 2006).
Análises	Expressam grupos de conceitos que representam uma construção comum na modelagem de negócio. Eles podem ser relevantes para um domínio, ou para vários domínios (FOWLER, 1996).
Arquiteturais	Expressam uma organização estrutural ou esquemas para sistemas. Como exemplos, o Model-View-Controller (MVC) (BUSCHMANN *et al.*, 1996).
Projetos	Refinam subsistemas ou componentes de um sistema, ou a relação entre eles. Nessa categoria têm-se os padrões identificados por Gamma *et al.* (1995).
Persistência de Dados	Descrevem mecanismos para mapear objetos persistentes para um banco de dados que facilita a implementação de sistemas orientados a objetos (OO) com banco de dados relacionais (FOWLER, 1996).

Implementação ou Idiomas	Específicos de linguagens de programação, esses padrões descrevem como implementar aspectos particulares dos componentes ou a relação entre eles utilizando as características da linguagem (BUSCHMANN *et al.*, 1996).
Testes	Descrevem diferentes métodos de testes de sistemas (WITHALL, 2007).

4.6 Coletâneas de Padrões

De acordo com Appleton (1997), padrões podem ser agrupados em diversos tipos de coletâneas, tais como coleções, catálogos, sistemas e linguagens de padrões, de acordo com forma de organização e a necessidade de recuperação para sua aplicação, conforme apresentado no Quadro 4.2:

Quadro 4.2 – Coletâneas de Padrões.

Coletâneas de Padrões	**Descrição dos Padrões**	**Exemplos**
Coleção de padrões	Simplesmente uma coletânea qualquer de padrões que não possuem nenhum vínculo entre si e, em geral, nenhuma padronização no formato de apresentação. Em uma coleção os padrões podem estar reunidos por terem sido apresentados em um mesmo congresso, por terem sido propostos pelo mesmo autor, ou por se referirem a um mesmo domínio.	[Coplien et al., 1995; Vlissides et al., 1996; Martin et al., 1998;
Catálogo de padrões	Coleção de padrões relacionados que, em geral, subdivide os padrões em um pequeno número de categorias abrangentes e pode incluir algumas referências cruzadas entre os padrões	Gamma et al. (1995)

Sistema de padrões	Conjunto coeso de padrões correlacionados que trabalham juntos para apoiar a construção e evolução de arquiteturas completas, estando organizado em grupos e subgrupos relacionados em múltiplos níveis de granularidade	[Buschmann et al., 1996] e [Schmidt et al., 2000]
Linguagem de padrões	Coleção estruturada de padrões que se apóiam uns nos outros para transformar requisitos e restrições numa arquitetura. Seus padrões constituintes cobrem todos os aspectos importantes de um dado domínio e pelo menos um padrão deve estar disponível para cada aspecto da construção e implementação de um sistema de software.	[Cunningham et al, 1996], [Meszaros e Doble, 1997]

4.7 Descobrimento de padrões

Um modelo, para ser considerado um padrão, deve ser um fenômeno recorrente. Existe uma convenção, definida como *rule of three*, que indica que um determinado modelo deve ocorrer em pelo menos três sistemas diferentes para ser considerado um padrão (GAMMA et al, 1995).

Segundo Meszaros e Doble (1997), outros critérios utilizados para se definir um padrão estão entre os seguintes:

- *Um padrão resolve problemas, ou seja, descrevem soluções e não estratégias ou princípios abstratos;*
- *Um padrão representa um conceito provado na prática (não é teoria nem especulação);*
- *Um padrão descreve uma solução que não é aparentemente óbvia;*
- *Um padrão explica porque a solução apresentada é útil e funcional.*

4.8 Conceito de Reutilização

Para que os processos sejam mais reutilizáveis, organizações precisam expressar elementos comuns e variáveis dentro de um processo. *Frameworks* fornecem um mecanismo para obter esta reutilização e são bem

apropriados para domínios onde várias aplicações similares são construídas várias vezes, partindo-se apenas de idéias (HOLLENBACH; FRAKES, 1996). Intensivamente, pesquisas voltadas para padrões têm mostrado que *frameworks* são ferramentas efetivas para a reutilização (GAMMA *et al.*, 1995), entretanto seu uso para processos ainda é pouco difundido.

Um considerável número de *frameworks* tem sido desenvolvido durante os últimos anos, mas sabe-se que, projetar um de alta qualidade ainda é uma tarefa difícil. Há uma série de problemas em aberto, quanto ao seu estudo, desde problemas no projeto, até sua instanciação e evolução. A aplicação ainda é realizada isoladamente, por exemplo, com uma definição não muito clara e precisa do que realmente são *frameworks* e padrões para processos de *ER*, e não contam com técnicas específicas para o seu desenvolvimento (HOLLENBACH; FRAKES, 1996).

4.9 Reutilização de Padrão

Reutilização de padrão refere-se ao uso de abordagens publicamente documentadas para solução de problemas comuns. Dependendo do tipo, padrões são freqüentemente documentados como um simples diagrama de classes e tipicamente compreendem de uma a cinco classes. Em sua reutilização, não se está reutilizando código, ao contrário, está se reutilizando a inteligência que está por trás do código. É uma forma de reutilização muito alta, que experimentará uma longa expansão de vida – pelo menos entre as linguagens de computação que estão sendo utilizadas e potencialmente pelo próprio paradigma de orientação a objetos.

Por exemplo, o padrão de "*Ponto de Contato*" (AMBLER, 1998) foi modelado com o uso de um diagrama de classe da UML 1.1, e mostra um padrão comum para rastrear os pontos de contato entre uma empresa e outras entidades de negócio. Esse padrão mostra que é possível tratar endereços de e-mail, endereços de superfície e números de telefone com o mesmo tipo de objetos – pontos de contatos através do qual sua organização interage com outras entidades de negócio (clientes, empregados, fornecedores e assim por diante). Esse padrão aumenta a flexibilidade de suas aplicações, possibilitando não somente postar uma carta para um cliente, mas também enviar-lhe um e-

mail ou um fax. Em vez de enviar um CD-ROM ou uma fita de vídeo para um endereço, o produto pode ser transmitido eletronicamente. O padrão “Ponto de Contato” é uma chave de habilitação para fazer essas coisas. Tem sido implementado com sucesso em várias aplicações, reutilizando a parte mais difícil de um modelo – o pensamento que ficou por trás dele. A reutilização de padrões provê uma reutilização de alto nível que pode ser implementado em muitas linguagens e plataformas. Padrões encapsulam o aspecto mais importante do desenvolvimento – a inteligência que está embutida na solução. Aumentam a capacidade de manutenção e de incrementar sua aplicação usando abordagens comuns para problemas que são reconhecidos por qualquer desenvolvedor experiente em orientação a objetos.

4.10 Conceito de Reuso Aplicado à Engenharia

Reuso de software em geral tem sido um objetivo primordial em *ER*. A própria utilização do termo *reuso* é uma demonstração do quanto é essencial e não está sendo atingido. Todas as tradicionais disciplinas de engenharia (como por exemplo, engenharia civil, elétrica, mecânica e produção) estão tão intrinsecamente baseadas na grande quantidade de reuso de elementos, que o termo reuso nem é mencionado. Afinal, reuso é parte integrante de praticamente tudo que se faz em engenharia. Estas cláusulas comuns formam uma boa definição de engenharia. Engenharia está relacionada com a criação de soluções eficientes, para problemas práticos, através da aplicação de conhecimento científico, construindo coisas a serviço da condição humana (SAWYER; SOMMERVILLE; VILLER, 1997).

Para atingir este objetivo, uma engenharia utiliza conhecimentos científicos sobre domínios tecnológicos que estão codificados de uma forma que seja diretamente útil para um engenheiro. Deste modo, este conhecimento codificado provê respostas para questões que ocorrem comumente na prática. Ou seja, este conhecimento deve ser reutilizado para a geração de soluções (SAWYER; SOMMERVILLE; VILLER, 1997).

Engenharia também pode ser entendida pela distinção entre trabalho criativo e trabalho rotineiro. Trabalhos rotineiros são aqueles que envolvem a solução de problemas conhecidos e, portanto, facilitam a

reutilização de grande parte de outras soluções já aplicadas a problemas similares (SAWYER; SOMMERVILLE; VILLER, 1997) e, Engenharia está fortemente relacionada com trabalhos rotineiros.

Para uma disciplina atingir a condição de engenharia, algumas condições são necessárias. Uma delas é o discernimento sobre o que é essência e o que é acidente. Outra condição é o conhecimento de quais elementos são estáveis. Ou seja, entre várias soluções, de problemas similares, quais elementos estão presentes nestas soluções e quais são importantes para a geração das soluções. O reuso efetivo depende do entendimento e representação dos pontos de pressão da estabilidade e variabilidade do domínio da aplicação (SAWYER; SOMMERVILLE; VILLER, 1997).

4.11 Padrão e a Produção do Software

No âmbito da engenharia, o arquiteto Alexander observou a semelhança nas soluções pertinentes à arquitetura urbanista, o que foi suficiente para que pudesse postular o famoso conceito de padrões através da trilogia *A timeless way of building, A pattern language e The Oregon Experiment* que constituem o ideário coeso do arquiteto, no qual defendia a seguinte proposição:

> *(…) Cada padrão descreve um problema que se coloca, vez por outra, em nosso entorno, e traz em si mesmo o núcleo da solução para esse problema, de tal forma que se possa utilizar essa solução mais de um milhão de vezes, sem necessidade de repeti-la nunca da mesma maneira (ALEXANDER et al,1979).*

O conceito foi adotado e expandido posteriormente por Gamma *et al* na obra *Design Patterns (*GAMMA *et al.,* 1995*)*, onde são abordadas tais doutrinas para solucionar problemas relacionados à análise orientada a objetos na construção de software. Pesquisas direcionadas à produção de software baseadas nos estudos de padrões têm sido amplamente difundidas, visto os resultados obtidos pelo cientista computacional Richard Gabriel, que vislumbrou caminhos na aventura teórica de Alexander.

No elástico mundo do conhecimento, não é incomum a migração de enunciados e de princípios de um campo da ciência para explicar fenômenos

de outra origem epistemológica. A concepção arquitetural se alicerça em analogias, e parece razoável admitir uma interpretação inversa, onde certo paradigma teórico do pensamento arquitetônico possa estimular a interpretação de problemas de projetos de outra área da criação contemporânea (BUSCHMANN *et al.*, 1996).

No prefácio do livro de Gabriel (1996), Christopher Alexander deixa transparecer surpresa com o fato, mas também alguma mágoa pela incompreensão de seus pares:

> *(...) O que teve de fascinante para mim, na verdade, inteiramente surpreendente, foi que no ensaio dele (Gabriel), um cientista da computação, para mim um desconhecido, e com quem nunca havia me encontrado, me pareceu entender mais sobre o que tenho feito e o que venho tentando em meu próprio campo, do que meus próprios colegas arquitetos (Alexander, C. prefácio em Gabriel, 1996).*

Gabriel (1996) percebe e põe em evidência naquele conjunto de ensaios a relação possível entre o método gerador de formas e estruturas de Alexander e a oportunidade de propor uma acepção no campo dos sistemas orientados a objetos. Isso, em essência, é dado pelo reconhecimento de que, em um caso e outro, o processo é o de associação de "entidades" que funcionam como blocos de uma linguagem. Os dois casos estão diante de processos que projetarão por analogias. Após a detalhada interpretação do pensamento de Alexander, o autor lança algumas bases para o desenvolvimento de uma teoria própria, de certa maneira apontando já os caminhos de uma interface cognitiva distinta, isto é, uma forma diferente de construir o conhecimento através de um processo de simulação que, no caso, se vale do isomorfismo entre entidades arquitetônicas e partes de uma linguagem computacional. Na concepção de linguagem (informática) de Gabriel estão presentes, entre os principais aspectos da "teoria alexandriana":

i) A capacidade de geração de padrões como partes intercambiáveis, capazes de metamorfoses conforme a atividade e a posição geométrica que ocuparem no programa;

ii) A geração de Software caracterizados pela autonomia semântica entre suas partes;

iii) A construção de Software habitáveis, ou seja, configurados por linhas de código compreensíveis por um grande número de pessoas da comunidade de informática;

iv) O desenvolvimento de Software que possam evoluir a partir de um crescimento incremental e;

v) Quando necessários, Software complexos poderiam ser estruturados através de conexões com outros programas pré-existentes.

De muitas maneiras, a idéia sintetizada como catálogo de padrões, vis-à-vis o sentido de totalidades, sugere um processo de sucessivos acoplamentos, de partes maiores ou menores, na conformação da estrutura do software ou de um sistema. Pode-se então falar que isso revela um modo onde estrutura e organizações convergem para a idéia de rede, que é realizada nos muitos planos de coordenação e controle do processo.

Até mesmo em jogos, os conceitos de padrões são aplicáveis. Wright (2001), criador do *SimCity*, afirma que sua inspiração original para o jogo foram as 256 regras de *Design* contidas no livro "*A Pattern Language*", de Alexander (1979), cada qual baseada em um aspecto do comportamento humano. A idéia básica é a de que o projeto de construção deve refletir aspectos desse comportamento em diferentes escalas.

A aplicação de padrões para o entendimento do problema e captura dos requisitos de um sistema de informação vem sendo difundida entre diversos grupos de pesquisas (DEUTSCHES, 2006; LAPPE et al, 2004), onde inclusive é produzida razoável quantidade de artigos sobre o tema e suas aplicabilidades, de forma satisfatória. Um desses grupos que tem forte destaque nesta linha de pesquisa é a alemã *Deutsches Elektronen-Synchrotron* (DESY). Sócios da Associação de Helmholtz, é um centro de pesquisa nacional apoiado por fundos de dívida pública, localizada em Hamburg e Zeuthen (Brandenburg), reconhecidamente um dos principais centros de aceleração gravítica (prótons e elétrons) no mundo (DEUTSCHES, 2006).

4.12 Resumo

Neste capítulo, foram detalhadas as várias abordagens sobre padrões. O texto desenvolvido abordou a classificação e o descobrimento de padrões relativos às fases de produção de software. E por fim, o autor apresenta os caminhos e fontes para busca de conhecimento sobre padrões aplicados a projetos e requisitos.

CAPÍTULO 5 - ESTRUTURAÇÃO DA PESQUISA

> *A pesquisa é um procedimento formal, com método de pensamento reflexivo, que requer um tratamento científico e se constitui no caminho para conhecer a realidade ou para descobrir verdades parciais (Lakato & Marconi, 2005).*

Embora o objetivo deste capítulo não seja o de discorrer extensamente sobre questões de metodologia científica, o que já é objeto de inúmeras obras conceituadas, faz-se necessário estabelecer conceitos relevantes acerca de certos termos, de forma a apoiar a justificativa da seleção da abordagem metodológica definida, conferindo-lhe lógica e coerência com os objetivos estabelecidos.

5.1 Introdução

Segundo Karlsson (2002), no editorial da edição especial sobre metodologia de pesquisa em Engenharia de Produção, do periódico *International Journal of Operations and Production Management – IJOPM*:

> *A metodologia está lá (na tese) para dar credibilidade ao leitor que você planejou e conduziu seu estudo assim como analisou e estabeleceu conclusões de uma forma que nós possamos acreditar no que você escreveu. Não é uma seção do texto onde você exibe suas considerações metodológicas favoritas ou demonstra que você leu algum material sobre metodologia. A idéia é a garantia da qualidade da pesquisa.*

Seguindo os preceitos de Karlsson (2002), esta seção visa a tão somente posicionar o leitor quanto às opções metodológicas adotadas, apoiando-as com conceitos importantes para sua compreensão, e não um longo discurso favorecendo uma ou outra abordagem.

5.2 Definição de Pesquisa

A pesquisa científica ou investigação científica pode ser entendida como *um procedimento reflexivo sistemático, controlado e crítico, que permite descobrir novos fatos ou dados, relações ou leis, em qualquer campo do conhecimento,* conforme define ANDER-EGG apud (LAKATOS; MARCONI, 2005).

Conceito similar pode ser visto em (GIL, 2002) em que o autor define pesquisa como sendo *o procedimento racional e sistemático que tem como objetivo proporcionar respostas aos problemas que são propostos.*

Para uma seleção adequada do método a ser adotado no desenvolvimento de determinada pesquisa científica, faz-se necessário ainda compreender os tipos de pesquisa e a respectiva correlação entre os diferentes métodos disponíveis. Existem várias taxonomias para caracterizar os tipos de pesquisa e a primeira dificuldade é encontrar uma que seja aplicada ao campo específico de atuação do pesquisador. A grande maioria dos textos sobre este assunto se baseia nas ciências sociais e nas considerações sobre o que é necessário conhecer e aplicar em cada tipo específico de investigação.

5.3 Caracterização da Pesquisa

Uma pesquisa não se encaixa de forma precisa em uma ou outra tipologia. Em especial, porque o processo de construção do conhecimento não se dá de forma linear ao longo do tempo, mas, sim, de forma iterativa e incremental. Neste processo, mesclam-se métodos e ferramentas de acordo com a etapa do ciclo da pesquisa e, ainda, as características do objeto investigado e a evolução da compreensão do pesquisador acerca do fenômeno estudado.

Isto é claramente constatado por Simon Croom (2002) ao propor um modelo normativo para as etapas de pesquisa: *(...) nós admitimos abertamente que o processo de pesquisa é totalmente caótico, envolvendo re-iteração entre os diversos estágios do processo.* Ainda, no mesmo texto, o autor prossegue, citando Bechhofer apud (CROOM, 2002): *(...) o processo de pesquisa não é uma seqüência bem definida de procedimentos que seguem um*

padrão organizado, mas uma iteração desorganizada entre o mundo conceitual e prático, dedução e indução ocorrendo ao mesmo tempo.

Tendo isto em conta, é possível, quanto ao objetivo, classificar esta pesquisa como sendo um estudo de pesquisa-social uma vez que, alicerçado no conhecimento teórico, busca uma redução na complexidade do processo de elicitação de requisitos, relevante ao cenário do contexto definido, especificamente de empresas do ramo comercial. Em sua fase inicial, faz uso de um estudo exploratório, uma vez que levanta e tenta compreender como o reuso de software está sendo praticado em diversos segmentos e instâncias organizacionais. Esta etapa, de caráter inicial, tem como finalidade a aproximação e compreensão do tema e visa ao embasamento teórico imprescindível à identificação dos conceitos que determinam o modelo de pesquisa.

Quanto à fonte de dados utilizada, é possível classificá-la como uma pesquisa com multiplicidade de meios uma vez que inicialmente se apóia no levantamento em fontes bibliográficas e em vivência do pesquisador no ambiente sendo estudado, e posteriormente, envereda pela pesquisa de campo, através de procedimentos observacionais diretos e indiretos, valendo-se das observações e análises documentais.

A condução deste trabalho assumiu algumas características que ultrapassam os conceitos tradicionais de pesquisa, como a interação do pesquisador com usuários, o fenômeno em estudo e o propósito de resolução de um problema prático. Tais condições ocasionam um direcionamento à metodologia de *pesquisa-ação*

E, finalmente, considerando a natureza dos dados coletados e os procedimentos de análise, pode ser considerado um estudo qualitativo, embora não esteja isento de coletas e evidências de natureza quantitativa.

5.4 Pesquisa-Ação Como Método de Pesquisa

A pesquisa-*ação* é um tipo de pesquisa social com base empírica vinculada com uma ação ou com a resolução de um problema, no qual os pesquisadores e os participantes representativos da situação estão envolvidos de modo cooperativo ou participativo (Thiollent, 2005).

> *A pesquisa-ação é ao mesmo tempo um trabalho de pesquisa e assessoria, possuindo então um propósito duplo de auxiliar a reflexão, formulação ou implementação da ação e de desenvolver, enriquecer ou testar quadros referenciais teóricos ou modelos relevantes ao fenômeno em estudo.*

A *pesquisa-ação* diferencia-se da pesquisa científica convencional, que utiliza regras de metodologia quantitativa, enquanto ela se caracteriza pela utilização da argumentação e interpretação no processo de investigação.

Para Thiollent (2005), a pesquisa-ação é uma estratégia metodológica de pesquisa social que se caracteriza por: a) interação explícita entre pesquisadores e as pessoas implicadas na situação investigada; b) as prioridades dos problemas a serem pesquisados e as soluções a serem encaminhadas sob forma de ação concreta surgem dessa interação; c) o objeto de investigação não são as pessoas e sim a situação social e os problemas a ela associados; d) O objetivo da pesquisa-ação consiste em resolver ou esclarecer os problemas da situação observada, ou até mesmo reduzir sua complexidade, e e) a pesquisa não se limita a uma forma de ação, pois pretende aumentar o conhecimento dos pesquisadores e o conhecimento ou o "*nível de consciência*" das pessoas que participam da situação.

5.5 Modalidades de Pesquisa-Ação

A participação não é o único determinante do tipo de projeto de pesquisa-ação que se está executando. Existe uma dialética entre escolha do tópico e participação, variações que dão origem a diferentes modalidades de pesquisa-ação, termo cunhado por Grundy (1983). Nesta linha, sugere fazer a distinção entre pesquisa-ação técnica e a pesquisa-ação prática. Deve-se pensar sobre a natureza de um projeto de pesquisa-ação. Então, Grundy (1983) propõe 5 modalidades de pesquisa-ação:

Pesquisa-ação técnica: Constitui uma abordagem pontual na qual o pesquisador toma uma prática existente de algum outro lugar e a implementa em sua própria esfera de prática para realizar uma melhora. Ela é “técnica”, porque o pesquisador está agindo de modo inteiramente mecânico. Um bom

exemplo de pesquisa-ação técnica é a construção de uma linguagem de padrões ou a criação de um modelo para gerenciamento de requisitos.

Pesquisa-ação prática: A pesquisa-ação prática é diferente da técnica pelo fato de que o pesquisador escolhe ou projeta as mudanças feitas. Nesse caso, as duas características distintivas são: primeiro, é mais como a prática de um ofício – o artífice pode receber uma ordem, mas o modo como alcança o resultado desejado fica mais por sua conta de sua experiência e de suas idéias; e segundo, porque o tipo de decisões que ele toma sobre o quê, como e quando fazer são informadas pelas concepções profissionais que tem sobre o que será melhor para seu grupo. Os artífices estabelecem seus próprios critérios para qualidade, eficácia, durabilidade e assim por diante.

Pesquisa-ação política: Quando se começa a tentar mudar ou analisar as limitações dessa cultura sobre a ação, é preciso engajar-se na política, porque isso significa trabalhar com ou contra outros para mudar "o sistema". Só se pode fazer isso pelo exercício do poder e, assim, tal ação torna-se política. Naturalmente, há muitos tipos de poder e muitos modos de exercê-lo. Por exemplo, há o poder de conseguir fazer as pessoas trabalharem juntas, o poder de fazer coisas quando os outros não estão olhando, o poder de superar as objeções dos outros e assim por diante.

Pesquisa-ação socialmente crítica: Essa é, realmente, uma modalidade particular de pesquisa-ação política e ambas se sobrepõem porque, quando se trabalha para mudar ou para contornar as limitações àquilo que você pode fazer, isso comumente é resultado de uma mudança em seu modo de pensar a respeito do valor último e da política das limitações. Você não está buscando como fazer melhor alguma coisa que você já faz, mas como tornar o seu pedaço do mundo um lugar melhor em termos de mais justiça social. Geralmente, isso é definido na literatura por mudanças tais como: aumento de igualdade e oportunidade, melhor atendimento às necessidades das pessoas, tolerância e compreensão para com os outros, cooperação maior e mais eficiente, maior valorização das pessoas (de si mesmo e dos outros) e assim por diante. Essas são as "grandes idéias" de uma sociedade democrática. A pesquisa-ação socialmente crítica passa a existir quando se acredita que o

modo de ver e agir “dominante” do sistema, dado como certo relativamente a tais coisas, é realmente injusto de várias maneiras e precisa ser mudado.

Pesquisa-ação emancipatória: Essa é outra variação da pesquisa-ação política, que tem como meta explícita mudar o *status quo* não só para si mesmo e para seus companheiros mais próximos, mas de mudá-lo numa escala mais ampla, do grupo social como um todo e constitui assim, necessariamente, um esforço participativo e colaborativo, o que é socialmente crítico pela própria natureza. Não é preciso dizer que a pesquisa-ação emancipatória ocorre muito raramente.

5.6 O Ciclo da Investigação-Ação

É importante que se reconheça a pesquisa-ação como um dos inúmeros tipos de investigação-ação, que é um termo genérico para qualquer processo que siga um ciclo no qual se aprimora a prática pela oscilação sistemática entre agir no campo da prática e investigar a respeito dela. Normalmente o ciclo compreende as seguintes ações: Planejar, Agir, Observar e Revisar. O ciclo para melhoria contínua pode ocorrer invariavelmente dependendo dos objetivos ou do que se pretende atingir.

A solução de problemas, por exemplo, começa com a identificação do problema, o planejamento de uma solução, sua implementação ou construção, seu monitoramento ou verificação, e a avaliação de sua eficácia.

Evidentemente, aplicações e desenvolvimentos diferentes do ciclo básico da investigação-ação exigirão ações diferentes em cada fase e começarão em diferentes lugares. Entre alguns dos diversos desenvolvimentos do processo básico de investigação-ação, estão (todos citados (apud) em Tripp (2005)) a pesquisa-ação (LEWIN, 1946), a aprendizagem-ação (REVONS, 1971), a prática reflexiva (SCHÖN, 1983), o projeto-ação (ARGYRIS, 1985), a aprendizagem experimental (KOLB, 1984), o ciclo PDCA (DEMING, 1986), PLA, PAR, PAD, PALM, PRA1 etc. (CHAMBERS, 1983), a prática deliberativa (MCCUTCHEON, 1988), a pesquisa práxis (WHYTE, 1964; 1991), a investigação apreciativa (COOPERRIDER; SHREVASTEVA, 1987), a prática diagnóstica (genérica em medicina, ensino corretivo etc.), a avaliação-ação (ROTHMAN, 1999)2 , a metodologia de sistemas flexíveis

(CHECKLAND; HOLWELL, 1998) e a aprendizagem transformacional (MARQUARDT, 1999). Além das encontradas pelo pesquisador desta Tese: gerenciamento de risco no processo de gerenciamento de software (IVERSEN;MATHIASSEN;NIELSEN,2004) , processo de gerenciamento de requisitos (NGUYEN;SWATMAN,2003), pesquisa-ação em engenharia de produção (WESTBROOK, 1994) e métodos de pesquisa empírica em *ER*. (EASTERBROOK, 2007).

Há várias razões para a produção desses muitos tipos diferentes de investigação-ação, porque algumas pessoas reconheceram e possivelmente conceituaram o ciclo sem conhecimento das demais versões já existentes e denominaram o mesmo ciclo e suas etapas de muitos modos diferentes (TRIPP, 2005). Houve também quem desenvolvesse versões sob medida para utilizações e situações particulares, porque há muitos modos diferentes de utilizar o ciclo e executar cada uma das suas quatro atividades. Assim, tipos diversos de investigação-ação tendem a utilizar processos diferentes em cada etapa e obter resultados diferentes que provavelmente serão relatados de modos diferentes para públicos diferentes.

Qual tipo de processo se utiliza e como ele é utilizado depende dos objetivos e circunstâncias. Até com objetivos e circunstâncias similares, pesquisadores diferentes podem ter diferentes habilidades, intenções, cronogramas, níveis de apoio, modos de colaboração e assim por diante. Tudo isso afetará os processos e os resultados. O ponto importante é que o tipo de investigação-ação utilizado seja adequado aos objetivos, práticas, participantes, situação (e seus facilitadores e restrições).

No caso desta pesquisa, foi adaptado o ciclo proposto por Westbrook (1994), pelo fato de abranger a área de Engenharia de Produção e demonstrar finalidades próximas aos objetos desta pesquisa de características pesquisa-ação técnica, como demonstrada na figura 5.1:

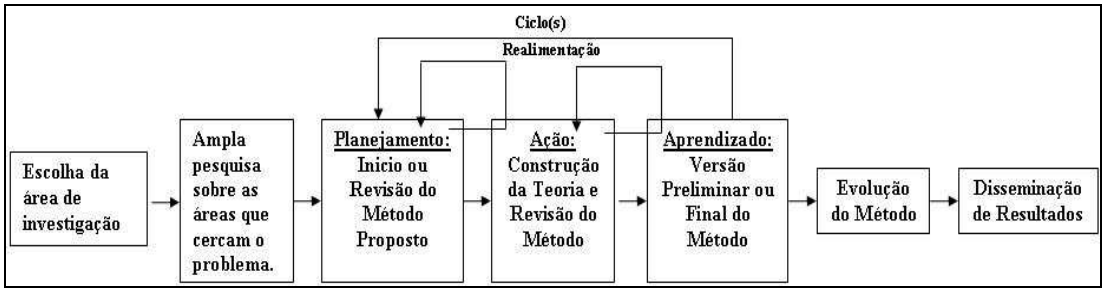

Figura 5.1 - Pesquisa-ação Para Criação de Teoria ou Métodos, segundo Westbrook (1994).

5.7 As Formas de Raciocínio e Argumentação

A pesquisa-ação está afastada das preocupações metodológicas relacionadas com a formalização ou com questões de lógica em geral. Para a lógica formal é difícil lidar com conhecimentos informais e emergentes de situações comunicativas ou interativas. Mesmo assim, esses conhecimentos – nem sempre logicamente claros ou coerentes – são potencialmente resgatáveis. A pesquisa não perde a sua legitimidade científica pelo fato de ela estar em condição de incorporar raciocínios imprecisos, dialógicos ou argumentativos acerca de problemas relevantes. Processar a informação e o conhecimento obtidos em situações interativas não constitui em si mesmo uma infração contra a ciência social (Thiollent, 2005).

5.8 Objetivo da Pesquisa e Objetivo da Ação

Segundo Thiollent (2005):

a) O objetivo prático da pesquisa consiste em contribuir para o melhor equacionamento possível do problema considerado como central da pesquisa com levantamento de prováveis soluções e ações correspondentes para auxiliar as pessoas envolvidas na sua atividade transformadora da situação. No caso particular tratado neste estudo, o objetivo prático foi o de reduzir a complexidade no processo de elicitação de requisitos de software.

b) O objetivo de conhecimento, por sua vez, consiste em obter informações relevantes e aumentar o conhecimento sobre determinados aspectos e situações. No caso deste trabalho, tal objetivo consistiu em explorar o conceito de padrões e elicitação de requisitos para gerar conhecimentos sobre a padronização do processo de compreensão dos problemas nas organizações.

Segundo Thiollent (2005), na relação entre os dois objetivos, com maior conhecimento melhora a capacidade para conduzir a ação. Na prática, deve existir um balanço ou equilíbrio entre as duas ordens de preocupações.

5.9 Critérios Para Avaliação do Valor Científico de um Projeto de Pesquisa

Segundo Eco (1977), um estudo é científico quando responde aos seguintes requisitos:

a) O estudo debruça sobre um objeto reconhecível e definido de tal maneira que seja igualmente reconhecível pelos outros. O termo objeto não tem necessariamente um significado físico. Definir um objeto significa então definir as condições sobre as quais possamos falar, com base em certas regras que são estabelecidas ou que outros pesquisadores estabeleceram anteriormente. Nessa ótica, o objeto de estudo do presente trabalho é *o processo de elicitação de requisitos.*

b) O estudo deve dizer do objeto algo que ainda não foi dito, ou rever sob uma óptica diferente ao que já se disse. Não é de conhecimento do autor desta pesquisa a existência de algum outro trabalho abordando a aplicação de padrões no processo de elicitação de requisitos. Se assim for, a condição "b" fica também cumprida.

c) O estudo deve ser útil aos demais. A questão colocada em pauta é de indubitável utilidade. A pesquisa, por sua vez, pretende auxiliares os profissionais que desenvolvem o trabalho de levantamento de requisitos e identificação dos problemas organizacionais.

d) O estudo deve fornecer elementos para a verificação e contestação das hipóteses apresentadas e, portanto, para uma continuidade pública. Com respeito a esta pesquisa, pelo tipo de pesquisa qualitativa realizada num contexto e circunstâncias específicas, a repetição comprobatória dos mesmos fatos apenas é possível através dos ciclos da pesquisa-ação ou até mesmo pelo estudo de casos. No caso particular da pesquisa social os fenômenos não possuem o caráter de perfeita repetição, como no caso dos fatos mecânicos, e além do mais o papel do pesquisador nunca é neutro dentro do campo observado (Thiollent, 2005). Portanto, esta pesquisa servirá para uma

aplicação prática ou para enriquecimento teórico de pesquisas relacionadas à área de estudo, conseqüentemente, satisfaz a condição "d".

5.10 Questões de estudo e proposições

Segundo Tripp (2005), uma proposta de pesquisa geralmente se concentra numa questão muitas vezes derivada dedutivamente de uma teoria, mas quando é possível pré-especificar o objetivo da pesquisa-ação, ele sempre será do tipo *como posso/podemos melhorar essa prática?*

Sobre as questões de estudo, Yin (1994) destaca a propriedade de estar relacionado mais fortemente às questões do tipo “como” e “porque”, conforme discorrido na seção anterior, e ressalta a importância da definição adequada das questões como norteadoras do estudo a ser desenvolvido empiricamente através dos casos. No entanto, apesar de norteadoras das investigações, não é incomum que as questões evoluam ao longo do curso da pesquisa e que os construtos e as variáveis inicialmente estabelecidos sejam modificados, desenvolvidos ou abandonados em função das descobertas (VOSS; TSIKRIKTSIS; FROHLICH, 2002).

Conforme estabelecido no Capítulo 1, a principal questão que esta tese se propõe a investigar é a seguinte: - *É possível catalogar padrões de requisitos, encontrados em documentos elaborados por analistas, comuns na construção de software para gestão comercial, e que possam contribuir na qualidade e redução da complexidade no processo de identificação dos problemas?* A ela se segue outra, que a refina e embasa e que auxilia no detalhamento das proposições: *Como construir um catálogo de padrões que possa contemplar os principais elementos do processo de elicitação de requisitos? e Como documentar esses padrões, identificando os problemas e as soluções além dos métodos para se aproximar dos melhores resultados?* Ao estabelecer estas questões iniciais, delimitou-se o campo de estudo e o objeto que será investigado.

Sobre as proposições, o autor discorre sobre a importância destas como meio de direcionar a atenção do pesquisador sobre o que efetivamente será examinado dentro do escopo da pesquisa-ação baseando no em ações de pesquisas.

Uma vez que este não se trata de um estudo exploratório, mas sim de um estudo descritivo, conforme definido neste capítulo, é possível estabelecer proposições, com base nas reflexões acerca da literatura. Estas visam a estreitar e nortear o campo de investigação.

- **Proposição-1**: É possível definir um catálogo de padrões para elicitação de requisitos para software de gestão comercial;
- **Proposição-2**: É possível classificar os requisitos de acordo com as preocupações envolvidas na construção do software;
- **Proposição-3**: O relacionamento entre os padrões favorece a compreensão sobre a importância de cada requisito;
- **Proposição-4**: Os requisitos, assim como os padrões, possuem forças opostas que se conflitam na solução dos problemas.

Esta pesquisa estabelece dois ciclos de aprendizado seguindo o modelo teórico da pesquisa-ação, e utiliza para a construção destes conhecimentos 3 (três) empresas na qual o pesquisador teve um envolvimento direto. Essa ação ocorre em dois momentos concomitantes, onde o pesquisador observar as ações dos participantes e analisa os documentos gerados, e paralelamente criar os padrões no exercício de mineração e refinamento dos elementos contidos nos documentos gerados pela atividade de elicitação de requisitos.

No modelo de construção dos padrões desta Tese, não há um plano validação ou aplicação dos padrões gerados, limitando-se apenas no refinamento dos padrões candidatos.

5.11 Resumo

Neste capítulo, definiu-se o que se trata por pesquisa e sua caracterização. Foram abordados os principais conceitos sobre pesquisa-ação, método escolhido pelo pesquisador para conduzir este trabalho. Foram esclarecidas as modalidades da pesquisa-ação e o ciclo de investigação, principal característica deste método de pesquisa. Reforçou-se a questão principal de estudo - que estimula a busca de resultados e uma solução através

do aprendizado - e as proposições visando a estreitar e nortear o campo de investigação.

CAPÍTULO 6 - CONSTRUÇÃO DO CATÁLOGO DE PADRÕES

O valor da Tecnologia de Objetos está fundamentalmente na sua capacidade de lidar com problemas complexos e criar sistemas compreensíveis e gerenciáveis, que podem acompanhar uma complexidade crescente e ser facilmente adaptáveis, se habilidosamente projetados (Craig Larman.).

Nesta etapa, é feita a execução da pesquisa, propriamente dita através de ações de pesquisa voltados para a elicitação de requisitos em projetos dos quais o pesquisador participou como *engenheiro de requisitos*. Esses estudos servirão de experiência adquirida para a construção do catálogo de padrões. Serão transcritos alguns aspectos relevantes observados e formalizados durante as atividades de extração e análise dos requisitos.

6.1 Introdução

Esta pesquisa fez uso do estudo de três casos, envolvendo contextos distintos para analisar a situação da relação da elicitação de requisitos e processo de desenvolvimento de software nas empresas. Conforme Thiollent (2005), a pesquisa-ação justifica-se quando permite investigação de eventos da vida real com a participação efetiva do pesquisador. Neste caso específico, trata-se de uma pesquisa na área de Engenharia de Software, com estudos de aspectos de levantamento das necessidades e identificação dos problemas relacionados as expectativas dos usuários, podendo revelar certas situações e contribuir com esta busca para o conhecimento, proporcionando experiências e lições aprendidas.

Pela dimensão do objeto de estudo, um padrão apenas é insuficiente; então para lidar com a complexidade envolvida, sugeriu-se a construção de um catálogo de padrões. A construção é também justificada pelo fato de alguns padrões serem aplicados apenas em circunstâncias particulares e específicas, além de existirem soluções alternativas para partes desses problemas mais complexos.

6.2 Ciclo da Pesquisa

Conforme Stringer (1996), é possível construir uma pesquisa-ação através de uma rotina composta por três ações principais: observar, para reunir informações e construir um cenário; pensar, para explorar, analisar e interpretar os fatos; e agir, construindo e avaliando as ações.

Nesta teoria, foram feitos dois ciclos de pesquisa-ação identificados como **Mineração** e **Refinamento** para a construção do catálogo conforme Figura 6.1

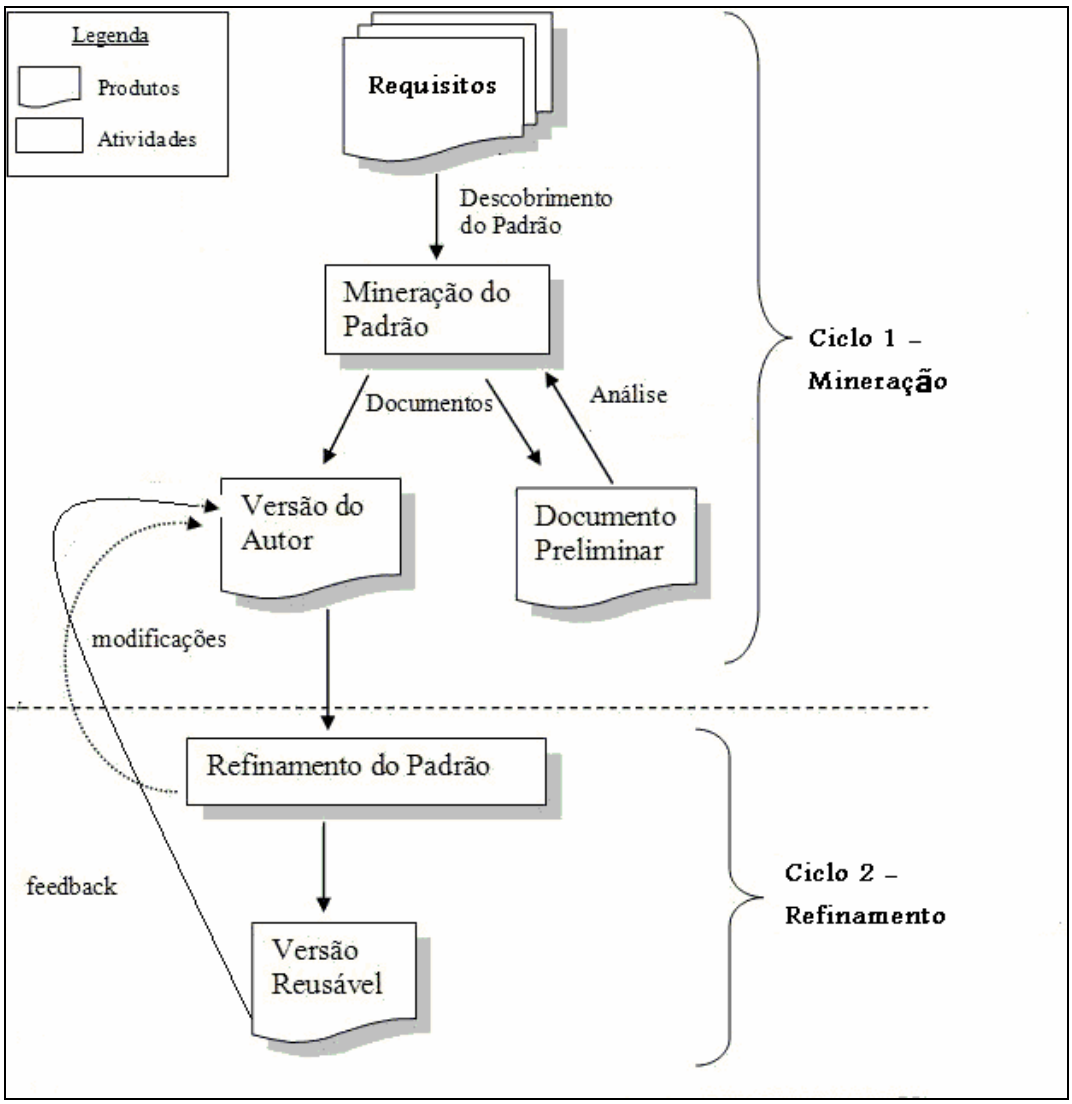

Figura 6.1 - Ciclo de vida da Pesquisa-Ação Para Construção do Catálogo de Padrões

Os ciclos consistem nas seguintes atividades:

Fase 1 - *Mineração*: Nesta fase, é feita uma seleção dos documentos de requisitos que serão analisados pelo pesquisador. A partir daí,

os requisitos serão analisados e separados conforme o tipo de requisito que o pesquisador for encontrando. Normalmente os requisitos são separados por tipo de problemas ou contexto, o que torna a busca mais cognitiva.

Fase 2 - *Refinamento*: Nesta fase, os requisitos são analisados por profissionais e pesquisadores experientes através de Conferências, sessões técnicas e grupos de pesquisas, além de alunos de graduação.

6.3 CICLO 1 – MINERAÇÃO

No primeiro ciclo é feita a construção dos padrões. É considerada a pesquisa bibliográfica sobre o tema, com intenção de se estabelecerem as práticas e possíveis métodos para auxiliar no processo de construção do catálogo de padrões, aplicados ao processo de elicitação de requisitos. A inquietação que motiva este ciclo pode ser definida da seguinte maneira: “Quais são as etapas para se extrair cada padrão candidato?”.

Após a reflexão sobre os métodos de construção de padrões encontrados na literatura, o pesquisador adotou os passos sugeridos pela DESY ((DEUTSCHES, 2006; LAPPE et al, 2004), que envolve as seguintes atividades: *observação->mineração->documentação*.

6.3.1 Minerando os Padrões de Requisitos

O catálogo de padrões foi surgindo através do exercício de observação de fatos e informações relativas às atividades de elicitação de requisitos. Foram derivados padrões através de três observações de aspectos semelhantes, identificadas em projetos diferentes, conforme Figura 6.2.

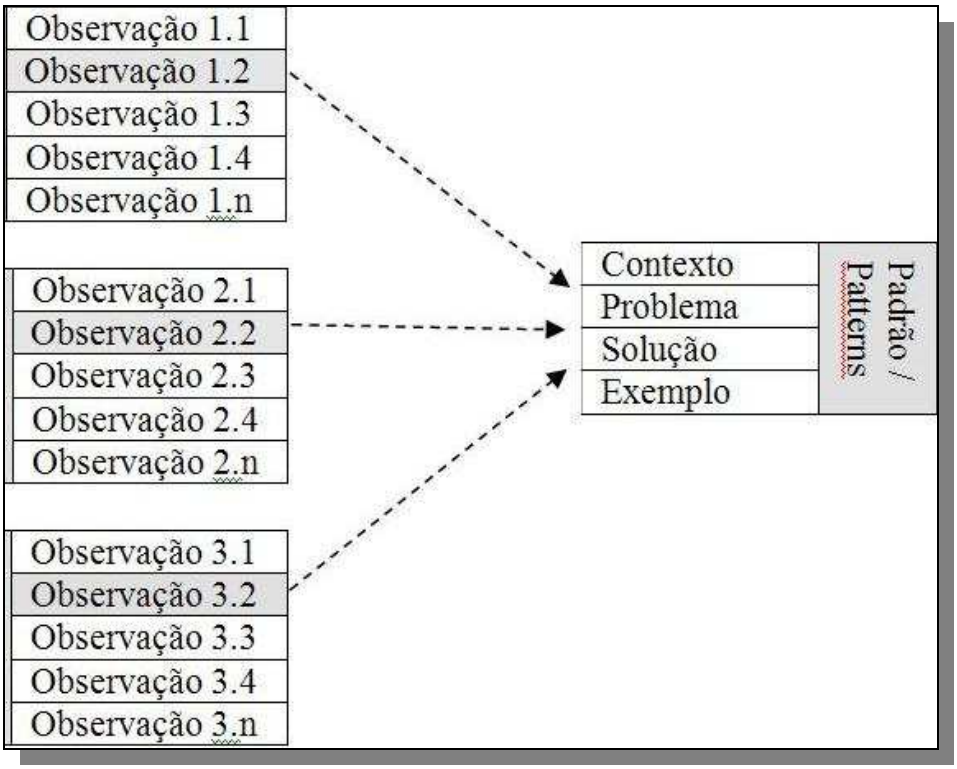

Figura 6.2 - Captura de padrões através da observação.

Para minerar os padrões dessas observações foram considerados cinco passos essenciais:

a) *Revisar documentos de requisitos validados, gerando conhecimento sobre informações do mundo real. Tais documentos contêm dados de importantes eventos e trazem relevantes experiências de outros projetos;*
b) *Os casos são analisados e reorganizados através de exercícios de observações, que descrevem as informações dos eventos no formato do vetor padrão;*
c) *Para identificar padrões, são feitos exercícios de observações de forma completa, analisando características similares em projetos distintos, seja na essência ou contexto. Nessas observações, são identificados os padrões candidatos.*
d) *Os padrões candidatos são construídos e descritos através uma estrutura padrão.*
e) *Passam por uma sessão técnica especifica da área de padrões (Writer´s Workshop), e outras atividades relacionadas a esse tipo de ação.*

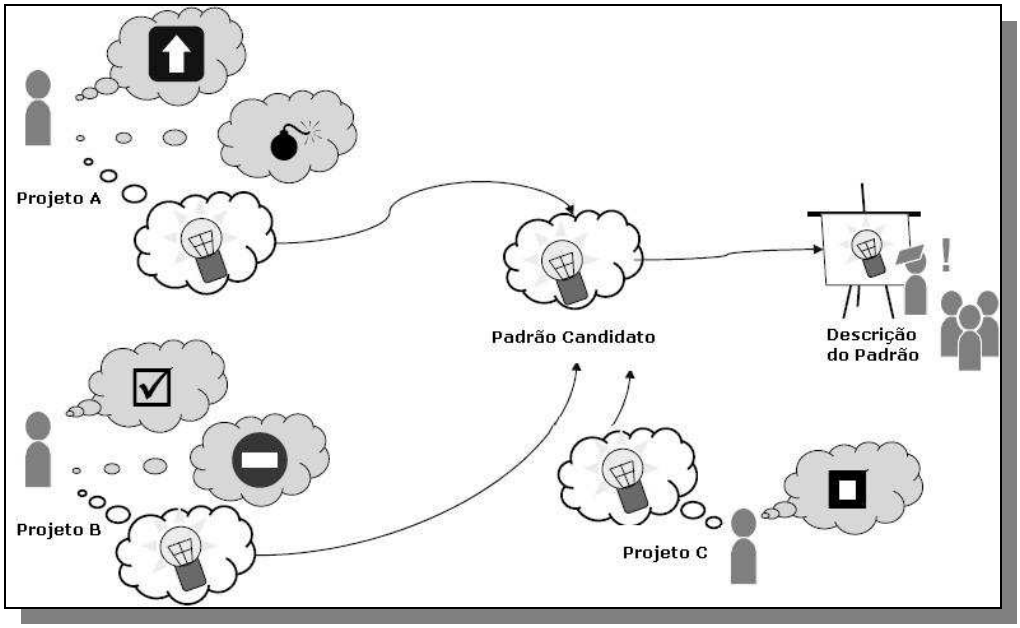

Figura 6.3 - Minerar os padrões candidatos.

O exercício de observação é uma atividade constante, e durante a construção desta Tese, foram feitas inúmeras observações aos vários documentos de requisitos dos projetos selecionados, e extraídos padrões candidatos de requisitos, distintos e relacionados a algum determinado tipo de problema destes projetos.

6.3.2 Separação por Preocupação

Os requisitos relacionados aos apontamentos, artefatos, configuração e projetos encontram-se criados fisicamente e organizados em módulos durante a fase de implementação, sendo tarefa de um desenvolvedor implementá-los.

O trabalho de organizar os requisitos de software em nichos de funcionalidades, interesse, preocupações e responsabilidades, denomina-se separação de preocupações. Os agrupamentos dos requisitos segundo um critério estabelecido é uma das atividades indispensáveis para desenvolvedor de software orientado a aspectos. Engenheiros de Software devem ter como premissa básica a separação de preocupações ou interesses e aplicá-la em todas as fases do processo de desenvolvimento de software.

Esta teoria tem levado a área de Engenharia de Software a refletir sobre a filosofia do dividir um problema em partes menores para resolvê-lo. Na prática, ela tem dado um novo ânimo para se investigar métodos, técnicas e

ferramentas capazes de reduzir a complexidade do desenvolvimento de software.

Contudo, atualmente, nem todos os requisitos são transformados em um módulo (ou componente) físico e independente. Isto impede que a substituição de um requisito seja equivalente à substituição de um ou mais componentes. A teoria de separação de preocupações, à primeira vista, lembra a teoria de desenvolvimento de software baseado em componentes. Nela, divide-se o software em partes menores denominadas componentes que, depois de integrados, originam o sistema planejado. Isolar estas partes não é tarefa trivial, pois diferentes profissionais da área podem ter percepções de arranjos diferentes do software.

Os padrões candidatos extraídos durante o processo de observação serão classificados conforme suas preocupações dentro da construção do software.

Seguindo as premissas do conceito citado, e de forma cognitiva, através da definição das forças, os padrões candidatos serão identificados e classificados dentro dos critérios contemplados pela literatura utilizada neste trabalho.

6.3.3 Definição das Forças

Padrões normalmente contêm pares, de problemas e soluções, em que são especificadas declarações de problema explicitamente em termos de duas (ou mais) forças contraditórias. As forças são diretamente relacionadas a condições ou situações como elas são encontradas pelos engenheiros de requisitos. A solução é provida em termos de uma ação proposta que foi observada para resolver o conflito, compensando as forças ou parte delas.

Uma estrutura formal chamada de *vetor padrão* (LAPPE *et al.*, 2004*;* HAGGE; LAPPE, 2006) foi adaptada para capturar a essência de um padrão [*P*] de *ER.*e contém parte do requisito ou problema [*R*] que precisa ser conduzido, as duas forças contraditórias [$F^{\leftarrow}$] e [$F^{\rightarrow}$] que caracterizam o problema, a ação ou método de elicitação [*A*] que pode contribuir com a solução [*S*], resolver o problema ou atender ao requisito. A seguir, a fórmula simplificada da estrutura.

P = (R, $F^{\leftarrow}$, $F^{\rightarrow}$, S, A) ou ***"se $F^{\leftarrow}$ mas $F^{\rightarrow}$ então S por A para R"***

6.3.4 Extraindo Padrões dos Casos Observados

Para solucionar problemas complexos com soluções específicas, Meszaros e Doble (1997) recomendam o uso de um dialeto de padrões para dividir o problema em um número maior de problemas menores e suas respectivas soluções. Dessa maneira, cada padrão deve resolver um problema especifico dentro do contexto compartilhado pela linguagem. Para que os padrões não fiquem isolados formando apenas uma coleção de padrões, após a criação é feita a organização e o relacionamento destes padrões, transformando dessa maneira, a proposta em um catálogo ou até uma linguagem de padrões.

Para criação dos padrões, será aplicado o *vetor padrão*, mas este poderá sofrer adaptações dependendo do tipo do padrão e, portanto, podendo variar a técnica aplicada, seguindo a proposta de Meszaros e Doble (1997).

Os padrões candidatos extraídos foram classificados nas formas apresentadas no capítulo anterior, podendo também variar os elementos utilizados de acordo com a finalidade do padrão.

Como os elementos envolvidos na compreensão da *ER* são extremamente amplos, o que os tornam altamente coesos e pouco acoplados, cada tipo de requisitos identificado na literatura foi tratado de modo particular, sem perder sua essência e a relação com o tipo de solução esperada.

A construção dos padrões se encontra no **Apêndice A – Extração dos Padrões Candidatos – Requisitos Funcionais**, **Apêndice B – Extração dos Padrões Candidatos – Requisitos Não-Funcionais e Apêndice C – Extração dos Padrões Candidatos – Requisitos Técnicos.**

6.3.5 Organização dos Padrões

Existem diversos formatos ou *templates* para a descrição de padrões de requisitos de Software. Alguns são quase puramente textuais escritos em prosa livre, enquanto outros são mais estruturados (GAMMA *et al.*,1995). Embora haja tantas opções, não existe um formato padronizado, principalmente porque diferentes tipos e domínios de padrões podem exigir

diferentes maneiras de apresentar tais padrões. Mesmo assim, há certo consenso geral sobre elementos essenciais que devem ser contemplados e comunicados por qualquer padrão, independentemente do formato utilizado (GAMMA *et al.*,1995 ; SHALLOWAY; TROTT, 2004).

Os padrões também podem variar de acordo com sua granularidade e no nível de abstração, uma vez que cada padrão tem suas peculiaridades, tornando-se necessário organizá-los de maneira a fazer sentido sua aplicação.

Nesta pesquisa, os padrões foram classificados pelos tipos de requisitos identificados em qualquer ambiente. Os padrões podem, nessa visão, ter características de ser funcional, não-funcional e técnico. Os padrões funcionais estão relacionados com os elementos dinâmicos de um software. Os padrões não-funcionais estão relacionados com os elementos estáticos do software e, de alguma maneira, exercem influência na sua qualidade. Já os padrões de características técnicas, estão ligados aos elementos de estrutura, de arquitetura e de apoio.

Além disso, os padrões foram moldados por uma visão baseada na idéia de *framework,* conforme pode ser visto na Figura 6.4.

Requisito	Funcional	Não-Funcional	Técnico
PADRÃO	*User*	*Interface*	*Dictionary*
	Process	*Security*	*Implementation*
	Context	*Component*	
	Module		
	Inverse		

Figura 6.4 - Catálogo dos padrões de elicitação de requisitos.

Estes padrões refletem o que tem sido aprendido sobre projetos de alta qualidade para solucionar problemas específicos, e encapsulam soluções identificadas na maioria dos estudos referentes à Engenharia de Software.

6.3.6 Intenções dos Padrões

A relação dos padrões de requisitos catalogados neste modelo, assim como sua classificação e intenções são listados na tabela 6.1:

Tabela 6.1 – Catálogo de padrões de requisitos.

REQUISITO	PADRÃO	INTENÇÕES
Funcional	User	Reconhecer e controlar usuários que terão direito de uso do sistema da informação.
	Process	Identificar maneiras pelas quais se realiza uma operação, segundo determinadas normas, métodos ou técnicas.
	Context	Descrever idéias de uma funcionalidade, expondo o grau de formalidade ou de intimidade entre as fases.
	Module	Organizar unidades planejadas a determinadas proporções e destinada a reunir ou a ajustar a outras unidades análogas, de várias maneiras, formando um todo homogêneo e funcional.
	Inverse	Exprimir possíveis variações das exceções ocorridas nos fluxos das informações em função do contexto.
Não – Funcional	Interface	Tratar sobre dispositivos, físicos ou lógicos, que façam a adaptação entre o sistema e o usuário, a respeito da usabilidade.
	Security	Ilustrar sobre dispositivos, físicos ou lógicos, relativos à segurança dos dados (ativos intangíveis).
	Component	Propor uso de componentes de conexão aos dispositivos lógicos e físicos do sistema.
Técnico	Dictionary	Definir glossário de termos, sigla, nomenclaturas aplicados aos processos de negócios.
	Implementation	Escolher tecnologias (ferramentas e recursos), o método e o processo de implementação do sistema.

Como existem diversas maneiras de se organizar os padrões, e para justificar a categoria de um catálogo de padrões, a maioria dos padrões propostos nessa Tese, devem ser sumarizados e usados conjuntamente mesmo que nem todos os padrões sejam usados num mesmo projeto. Por exemplo, o padrão *User* é freqüentemente usado com o *Process* e o *Interface*.

Outros padrões resultam em requisitos semelhantes, embora tenham intenções diferentes. Alguns são alternativos: O *Implementation* pode ser um padrão alternativo para o *Component*.

Outra forma de organizar os padrões é de acordo como eles mencionam outros padrões no modelo de relacionamento. A Figura 6.5 ilustra estes relacionamentos graficamente.

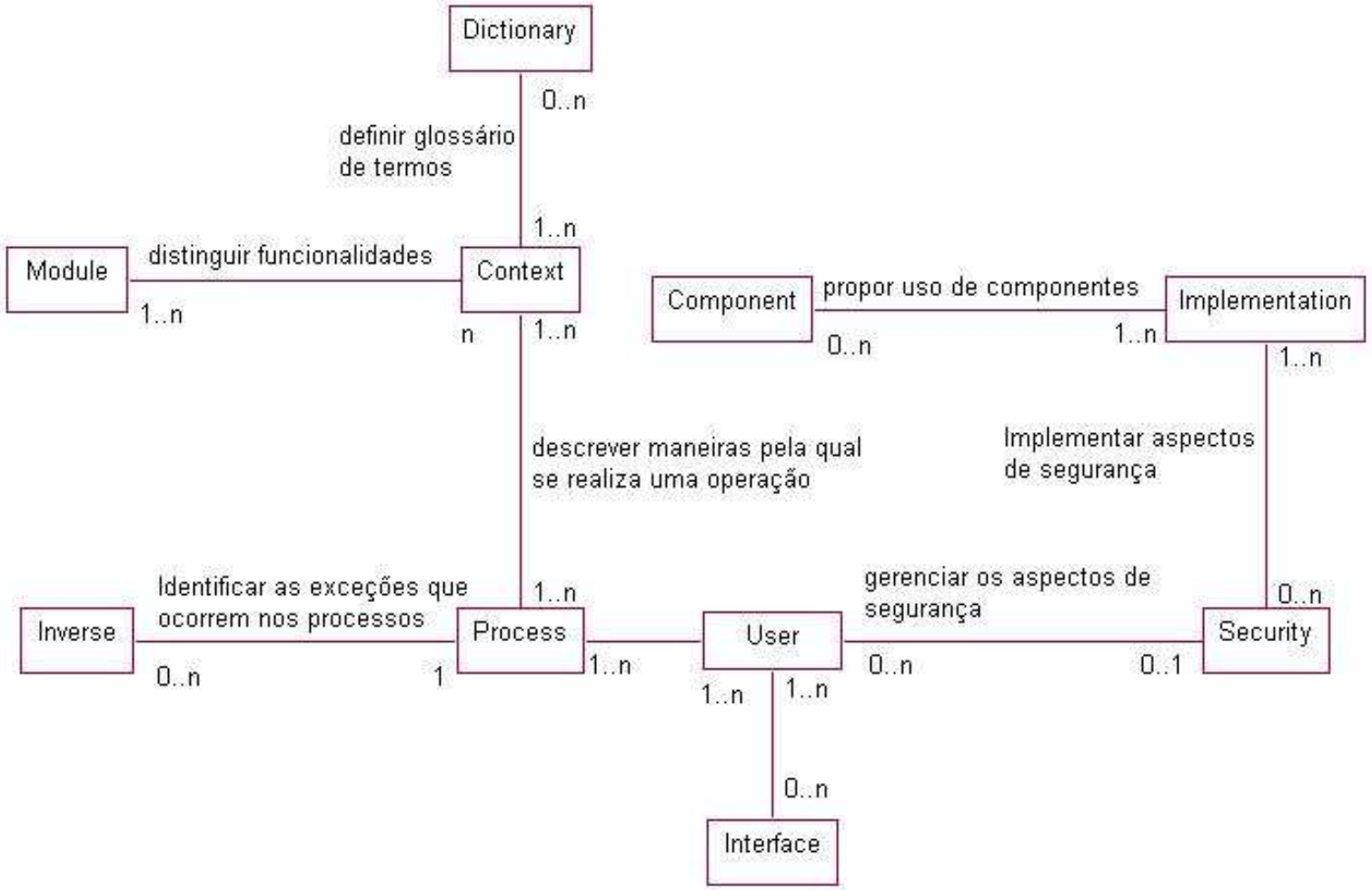

Figura 6.5 - Relacionamento entre padrões de requisitos.

6.3.7 Descrição dos padrões

As notações gráficas, embora sejam importantes e úteis, não são suficientes. Elas simplesmente capturam o produto final do processo de projeto, portanto, assim como as formas e elementos apresentados sobre *padrões*, serão descritos os padrões de requisitos usando um formato consistente e textual.

Seguindo as propostas apresentadas no capitulo anterior sobre as formas e elementos dos padrões, nesta Tese, cada padrão também será dividido em seções de acordo com o gabarito apresentado na tabela 6.2.

O gabarito fornece uma estrutura uniforme às informações, tornando os padrões de requisitos mais simples para ser compreendido, comparado e aplicado:

Tabela 6.2 – Forma e elementos utilizados na Tese.

ITEM	DESCRIÇÃO
Nome	Todo padrão tem um nome significativo, que lhe dê identidade, permitindo que ele seja facilmente incorporado no vocabulário dos usuários do padrão. O nome sugere a solução encontrada para o problema tratado.
Contexto	Contém os pré-requisitos para a aplicação do padrão, define o ambiente em que o problema é encontrado e onde a solução apresentada é válida ou satisfatória. O contexto determina a aplicabilidade do padrão, ou seja, as situações onde o problema é encontrado e os mesmos podem ser aplicados.
Problema	O problema que o padrão está tentando resolver.
Forças	Define um conjunto de forças e restrições que atuam sobre o problema, e que influenciam a decisão da solução. São critérios, normalmente contraditórios, que são levados em consideração na elaboração de uma solução.
Método de Elicitação	Propõe o(s) método(s) de elicitação que poderá/poderão facilitar a atividade de identificação dos problemas para a possível solução.
Solução	Como o padrão provê uma solução para o problema no contexto em que ele aparece.

ITEM	DESCRIÇÃO
Exemplos	Ilustrações de contexto e problema concretos, situações reais onde o problema genérico é encontrado. Exemplos descrevem também a aplicação da solução concreta, e também porque outras soluções que poderiam ser usadas não são melhores do que a aplicada. Exemplos descrevem a parte concreta de um padrão, o que é bastante importante, visto que as pessoas se sentem mais à vontade com elementos concretos do que abstrações. Alguns exemplos serão demonstrados por notações gráficas.

Segue abaixo a descrição dos padrões candidatos, com as descrições gabaritadas com o modelo acima.

6.3.7.1 Padrão Candidato I

- **Nome:** USER
- **Contexto:** Na identificação dos usuários que terão direito de acesso e uso do sistema da informação.
- **Problema:** Acesso ao sistema.
- **Forças:** “padronizar os processos”; “nenhum usuário poderá ter acesso à senha alheia”; “customizado de acordo com a função de cada funcionário”; “deve ser flexível aos usuários em geral”; “poderá utilizar terminais diferentes”; “flexibilidade de um usuário do mesmo nível utilizar os privilégios de outro usuário”.
- **Método Elicitação:** Caso de uso, observação e análise de protocolo.
- **Solução:** Criar um controle de acesso aos usuários com senhas.

6.3.7.2 Padrão Candidato II

- **Nome:** PROCESS
- **Contexto:** Na identificação dos processos contidos nos módulos de implementação.
- **Problema:** Identificação dos processos.
- **Forças:** “os processos podem se repetir em momentos distintos”; “processos duplicados pode ser reflexos de canais diferentes”; “informações dos processos depende de etapas ou processos anteriores”
- **Método Elicitação:** Caso de Uso, User Stories, QFD.
- **Solução:** Buscar a identificação de cada processo.

6.3.7.3 Padrão Candidato III

- **Nome:** CONTEXT
- **Contexto:** Descrição de idéias das funcionalidades, expondo o grau de formalidade ou de intimidade entre as fases.
- **Problema:** Descrição dos contextos.
- **Forças:** “descrever as variações de contexto”; “identificar suas variações e ligações”.
- **Método Elicitação:** Caso de Uso, Prototipação , User Stories.
- **Solução:** Descrever as variações do contexto, suas idéias, ligações e formalidades.

6.3.7.4 Padrão Candidato IV

- **Nome:** MODULE
- **Contexto:** Organização de unidades planejadas a determinadas proporções e destinada a reunir ou ajustar a outras unidades análogas, de várias maneiras, formando um todo homogêneo e funcional.
- **Problema:** Distinguir módulos.
- **Forças:** “representar os possíveis módulos do sistema”; “os módulos podem ser intersecionais”;“Mostrar as ligações dos módulos e o tipo de relacionamento”; “identificar suas variações e ligações”.
- **Método Elicitação:** Caso de Uso; Prototipação ; User Stories ; SSM.
- **Solução:** Identificar e distinguir os módulos do sistema.

6.3.7.5 Padrão Candidato V

- **Nome:** INVERSE
- **Contexto:** Expressão de possíveis variações das exceções ocorridas nos fluxos das informações em função do contexto.
- **Problema:** Identificar exceções.
- **Forças:** “mapear todas as restrições dos processos”; “identificar se há pontos condicionais”; “relacionar as restrições a cada módulo”; “identificar os pontos em cada processo com restrição”.
- **Método Elicitação:** Caso de Uso, Prototipação, User Stories.
- **Solução:** Identificar os cenários e representar os departamentos e suas responsabilidades.

6.3.7.6 Padrão Candidato VI

- **Nome:** INTERFACE
- **Contexto:** Tratamento de dispositivos, físicos ou lógicos, que façam a adaptação entre o sistema e o usuário, a respeito da usabilidade.
- **Problema:** Tratar as conexões dos dispositivos físicos ou lógicos.
- **Forças:** “identificar os métodos de entradas de dados”; “identificar os meios de entradas de dados feitos por atalhos”; “criar controles visuais e operacionais na entrada de dados”; “padronizar as ações de teclas de acordo com as funcionalidades do sistema”; “mostrar aos usuários todos os atalhos relativos a entradas de dados”; “escolher cores e teclas de funções dentro das limitações do equipamento e abrangências dos programas”
- **Método Elicitação:** Entrevista e Brainstorming.
- **Solução:** Estabelecer meios para redução dos esforços na operação do sistema.

6.3.7.7 Padrão Candidato VII

- **Nome:** SECURITY
- **Contexto:** Ilustração de dispositivos, físicos ou lógicos, relativos à segurança dos dados (ativos intangíveis).
- **Problema:** Ilustrar sobre a segurança dos dispositivos.
- **Forças:** “identificar do usuário em cada processo”; “os processos podem se repetir em momentos distintos”; “registrar a primeira identificação”; “informações dos processos depende de etapas ou processos anteriores”.
- **Método Elicitação:** Caso de Uso , User stories e Etnografia.
- **Solução:** Entender as formas de entradas de dados e sobre os equipamentos.

6.3.7.8 Padrão Candidato VIII

- **Nome:** COMPONENT
- **Contexto:** Proposição ao uso de componentes de conexão aos dispositivos, lógicos e físicos do sistema.
- **Problema:** Propor uso de componentes de implementação.
- **Forças:** “elencar os componentes”; “definir quais componentes serão usados”; “aprender sobre cada um deles”; “adquirir componentes proprietários”; “fazer adaptações no software para uso”
- **Método Elicitação:** Caso de Uso, Prototipação e User Stories.
- **Solução:** Identificar os pontos de uso de componentes.

6.3.7.9 Padrão Candidato IX

- **Nome:** DICTIONARY
- **Contexto:** Definição glossário de termos, sigla, nomenclaturas aplicados aos processos de negócios.
- **Problema:** Definir glossário de termos.
- **Forças:** “entender os termos do ambiente”; “mapear todas as siglas e termos”; “captar os significados com quem conhece e aplica”; “deixar a disposição para atualizações”.
- **Método Elicitação:** Entrevista, User Stories; Participatory Design.
- **Solução:** Criar um glossário com os termos do contexto.

6.3.7.10 Padrão Candidato X

- **Nome:** IMPLEMENTATION
- **Contexto:** Escolha tecnologias (ferramentas e recursos), o método e processo de implementação do sistema.
- **Problema:** Determinar tecnologias de implementação.
- **Forças:** “implementar para as bases especificadas”; “criar mecanismo de implementação por componentes”;” “Custo da tecnologia”; “Encontrar profissionais que conheçam a tecnologia aplicada”.
- **Método Elicitação:** SCRAM, Prototipação.
- **Solução:** Criar um documento de cenário e um protótipo de operações.

6.3.8 Encerramento do Ciclo 1

Neste primeiro ciclo, foram extraídos os padrões candidatos e organizados em gabarito uniforme para documentação de padrões.

Não obstante, foi possível verificar as preposições sugeridas pelo pesquisador, embora seja necessária a execução do próximo ciclo para se confirmar a qualidade dos padrões, pois como se trata de uma pesquisa participativa onde o pesquisador se utiliza de documentos relacionados a projetos em que participou, e pode ter havido alguma influência na extração dos padrões candidatos.

Foi possível responder a questão chave desta etapa com as etapas de para construção preliminar do catálogo.

6.5 CICLO 2 - REFINAMENTO

Neste segundo ciclo, é feito o refinamento dos padrões com a revisão dos padrões candidatos. É um momento de aprendizado e de *refatorar*[2] dos padrões construídos no primeiro ciclo da pesquisa, através da exposição do catálogo em simpósio específico da área, aplicação prática com alunos de graduação, revisão dos documentos de requisitos utilizados no primeiro ciclo e sessão técnica com profissionais da área.

6.5.1 Refinando os Padrões de Requisitos

O planejamento deste ciclo é feito com a estratégia de expor ao máximo os padrões em eventos relacionados à área, com o objetivo de ser criticado pelos pares das áreas, pesquisadores e futuros profissionais que poderão utilizar esta ferramenta num futuro.

Esse ciclo ocorre em partes paralelamente e de forma iterativa com o ciclo anterior, o que é possível pelo tipo de pesquisa proposto pelo autor desta Tese.

O ciclo que se inicia com a indagação principal deste ciclo é: *Os padrões propostos nesta Tese está dentro da conformidade e clareza sugerido pela comunidade cientifica, acadêmica e profissional*? Para responder a essa indagação, cada método de refinamento teve um objetivo especifico na construção dos catálogos.

Para isso, o pesquisador fracionou a Tese em artigos que ,de certa forma, pudessem ser analisados pelo profissional citado.

6.5.2 Primeira Ação

Apresentação da pesquisa nas reuniões do grupo de pesquisa GTI – Gestão da Tecnologia da Informação, onde mensalmente ocorre encontro dos professores, alunos, pesquisadores e convidados, com o objetivo de trocar experiências e conhecimentos relativos às pesquisas que estão em desenvolvimento pelos alunos da USP/Poli, especificamente do departamento

[2] Do inglês *Refactoring* , é o processo de modificar um sistema de software para melhorar a estrutura interna do código sem alterar seu comportamento externo. Neste caso o autor aplicou os mesmos princípios no processo de construção do catálogo de requisitos.

de Engenharia de Produção e que tratam de assuntos direcionados a esta linha de pesquisa.

Esta ação teve a oportunidade de ocorrer em duas ocasiões diferentes, uma no inicio da pesquisa, e outra próxima ao fim da construção do catálogo.

As contribuições nesse caso foram de ordem metodológica e estruturação da pesquisa. Até o momento a pesquisa possuía um perfil próximo ao estudo de caso, mas devido à participação efetiva do pesquisador e a forma de contribuição que se deseja como resultado, a pesquisa assumiu os métodos e práticas da pesquisa-ação.

6.5.3 Segunda Ação

A publicação do artigo na *6º Conferência Latino-Americana de Linguagens de Padrões de Programação - SugarLoafPlop'2007*, teve o objetivo de discutir a aplicabilidade dos padrões no processo de elicitação de requisitos.

Esse evento que ocorre anualmente tem a participação de alunos, professores e pesquisadores das principais instituições de ensino do Brasil, inclusive com a participação de renomados pesquisadores internacionais.

Esse evento conta com tutoriais e palestra convidada, além de três seções diferentes com artigos sobre padrões de software:

- *Writers' Workshop (WW) - artigos que documentam padrões (padrões de projeto, padrões de processo, etc.).*
- *Pattern Applications (PA) - artigos que explorem aplicações de padrões, tanto no mercado quanto no ensino, ferramentas para facilitar o uso de padrões, comparações de produtividade usando padrões, e assim por diante.*
- *Writing Patterns (WP) - artigos de novatos na área de padrões que desejam aprender como melhor elaborar uma idéia que se assemelha bastante a um padrão. Essa sessão será uma espécie de tutorial prático sobre a escrita de padrões.*

Uma ação positiva do evento é o processo que ocorreu antes da publicação, em que o artigo previamente selecionado passa por uma pré-análise e acompanhamento de um membro do comitê técnico do evento chamando de *shepherd* e, dentro de um cronograma, são feitos os ajustes e aperfeiçoamentos necessários para a apresentação e publicação final dos artigos selecionados.

Finalizada a etapa de *shepherding*, no evento, o artigo relativo a esta Tese foi analisado, discutido e criticado na seção *PA*, e se teve a oportunidade de esclarecer dúvidas, ouvir sugestões e criticas sobre a construção e aplicabilidade destes padrões numa eventual ferramenta ou através de *templates* que pudessem auxiliar o analista ou o engenheiro de software na sua atividade de compreensão dos problemas das empresas.

Essa atividade foi importante, pois o pesquisador obteve auxilio do *shepherding* na construção de alguns padrões.

6.5.4 Terceira Ação.

A publicação do artigo na *Euro American Conference on Telematics and Information Systems (EATIS'07)*, que teve como objetivo discutir a clareza e compreensão dos padrões por profissionais de áreas e que atuam no mercado como analistas ou engenheiros.

O *EATIS'07* é um evento que ocorre anualmente, cada vez em um País diferente e, neste ano, ocorreu em Faro - Portugal. Similarmente ao evento anterior, o artigo pré-selecionado passa por uma análise de pesquisadores que também sugerem alguns ajustes e melhorias nos artigos enviados. Nesta oportunidade, o artigo desta Tese foi selecionado entre os 10 melhores artigos oriundos de pesquisados pertencentes a países em desenvolvimento.

Foram sugeridas algumas mudanças na forma de apresentação dos padrões que contribuíram para consolidação do catálogo.

6.5.5 Quarta Ação.

A Sessão Técnica (*Writers' Workshop*) ocorrida na Escola Politécnica da USP, e teve como objetivo mostrar a forma de escrita para profissionais que trabalham com padrões ou requisitos. O pesquisador procurou

manter características parecidas com a proposta do evento *SugaLoafPlop*. Esta sessão foi sugestão da Professora Doutora Rosana Terezinha Vaccari Braga na oportunidade de sua participação na banca de qualificação deste trabalho.

Foram convidados profissionais da área de *Engenharia de Software* que possuíssem conhecimentos relativos a padrão e requisitos.

O evento teve a participação dos senhores **Jair Rillo Junior**, analista da IBM com certificações *Sun Certified Java Programmer, Sun Certified Java Associate, Sun Certified Web Component Developer, Sun Certified Business Component Developer e IBM SOA Associate*, **Rodrigo Aparecido Marques** analista do Correios com certificações *IBM Certified Solution Designer Rational Unified Process V7.0 e Certified Specialist for Requirements Management with Use Cases*, **Luis Fernando Guaido** também analista dos Correios com a certificações *Certified ITIL® Foundation V2 e Sun Certified Java Programmer*, além do professor doutor **Mauro de Mesquita Spínola** , orientador deste trabalho, que atuou como coordenador da sessão.

Detalhes sobre as ocorrências, críticas e sugestões de como foram escritos os padrões estão na ata da sessão anexa ao final deste trabalho.

6.5.6 Quinta Ação.

A utilização pelos alunos em um projeto de graduação, aplicando os padrões no processo de elicitação de requisitos de um software de biblioteca nas disciplinas Modelagem de Sistemas e na Linguagem de Programação Orientada a Objeto utilizando a linguagem Java.

Este procedimento serviu para o pesquisador perceber o grau de compreensão, clareza e aplicação que os padrões ofereciam para usuários menos experientes no assunto em questão.

Os alunos foram separados em dois grupos de 10 alunos, denominados de Grupo A e Grupo B. Inicialmente foi passada aos alunos a proposta do trabalho, apenas ao Grupo A foram passados os padrões propostos no catálogo desta Tese. Quanto ao Grupo B, foram passados os princípios e conceitos da AOO e UML.

Neste primeiro teste, os resultados foram expressivos, sendo que o Grupo A apresentou um número maior de casos de usos e documentos de requisitos como segue na tabela abaixo:

Tabela 6.3 – Primeiro Comparativo Quantitativo dos Requisitos Especificados.

Grupo	Use Cases	Documentos Requisitos	Outros de Documentos
A	22	13	5
B	11	8	0

Num segundo momento, foram passados aos alunos do Grupo B os padrões dos catálogos apresentados da mesma forma ao Grupo A e, novamente, foram para execução de levantamento dos requisitos. E os mesmos apresentaram os seguintes resultados. Aqui o Grupo B, na segunda execução é chamado de B2:

Tabela 6.4 – Segundo Comparativo Quantitativo dos Requisitos Especificados.

Grupo	Use Cases	Documentos Requisitos	Outros de Documentos
A	22	13	5
B	11	8	0
B2	24	17	7

Como foi possível observar, os padrões proviram condições de melhorar a cognição dos participantes, favorecendo na busca e partição dos problemas e compreensão dos fatos, de forma objetiva.

6.5.7 Encerramento do Ciclo 2.

Com o encerramento do ciclo 2, o pesquisador obteve um catálogo consolidado. As ações de revisão do catálogo reforçaram as respostas às preposições iniciais.

- **Proposição-1**: Com a criação dos padrões candidatos através do processo de observação e extração, esta proposição se cumpre.
- **Proposição-2**: Cada padrão foi classificado conforme o aspecto que procura resolver, portanto é possível classificá-lo por preocupação.

- **Proposição-3**: O pesquisador, baseado na finalidade de cada padrão, relacionou os padrões conforme sua aplicabilidade e afinidade na solução dos problemas, confirmando está proposição.
- **Proposição-4**: A construção do catálogo baseado na observação e análise dos documentos de requisitos mostrou que as forças são diretamente relacionadas a condições ou situações como elas são encontradas nestes documentos. A solução é provida em termos de uma ação proposta que foi observada para resolver o conflito, compensando as forças ou parte delas. O que também confirma esta proposição.

6.5.8 Próximo Ciclo.

Embora o trabalho tenha, dentro dos ciclos da pesquisa-ação, executado apenas dois ciclos, não uma definição da quantidade exata de ciclos necessários para se chegar a uma teoria próxima do ideal, por isso são definidos como ciclos de aprendizado.

Uma das principais contribuições da pesquisa-ação é a geração do chamado conhecimento intermediário. Este é um conhecimento de finalidade prática que, apesar de às vezes não ser muito valorizado no plano cultural-simbólico, é muito útil na resolução de problemas do mundo real. Esse conhecimento não se dá espontaneamente na prática, mas é produzido e adaptado no processo de interação entre o pesquisador e os interlocutores representativos dos problemas abordados. Imediatamente aplicáveis ou não aos problemas tratados, as soluções encontradas são também instrumentos de sensibilização e conscientização (Thiollent, 1998).

Portanto, o pesquisador cumpre com o objetivo de construção sugerida pela pesquisa-ação técnica, e contribui com a formação de uma teoria e sugere novas práticas para solucionar problemas relativos a área de *ER*.

CAPÍTULO 7 - CONCLUSÃO

As conquistas mais preciosas da vida não advêm de conquistas materiais (Albert Einstein).

Nesta etapa da dissertação, é feita a conclusão do trabalho, associando considerações e relacionando trabalhos futuros.

7.1 Relevância do estudo

O mundo da tecnologia tem evoluído tão rápido que, muitas vezes, é difícil se manter atualizado. As pessoas, bem como as organizações, estão, a cada dia, demandando novas tecnologias e técnicas para que possam, mais rapidamente, solucionar seus problemas. O tempo fica cada vez mais curto e mais coisas devem, e precisam ser feitas em um intervalo de tempo pequeno.

Diante desta situação, há uma demanda cada vez maior por software, o que tem obrigado as empresas a produzirem Software similares para situações parecidas, ou muitas vezes mais software – incluindo as mudanças ou manutenções – em menos tempo. Isto representa um grande desafio para as organizações produtoras de software. Manter ou até aumentar a qualidade do software, baixando o tempo de seu desenvolvimento o que não é uma tarefa tão trivial.

Como foi comentada pelos participantes da sessão técnica, a grande dificuldade da elicitação de requisitos é que na maioria das vezes o problema não ocorre, embora esteja presente no ambiente em que se está levantando informação, e diferentemente de outros padrões existentes, que, ao se deparar com o problema, é aplicado o padrão, nos padrões para requisitos, essa aplicação tem que ser antecipada.

Há algumas pesquisas relativas a padrões de requisitos, mas encapsulam requisitos prontos para reuso (ROBERTSON,2006;WITHALL, 2007;DEUTSCHES,2006), e não para serem encontrados no momento de elicitação. Há possibilidades claras de esses padrões trabalharem em conjunto com os padrões sugeridos nessa Tese.

7.2 Contribuições da pesquisa

Esta pesquisa procurou contribuir com parte da solução dos problemas citados no item anterior, sugerindo padrões que podem de maneira não singular, reduzir a complexidade de se compreender o que é necessário produzir.

Não se pode perder tempo tentando desvendar quais os problemas que cercam determinados contextos e, para isso, esta Tese mostrou que alguns elementos que fazem parte da solução na maioria dos problemas podem ser estruturados e colocados à disposição do Analista ou Engenheiro de Requisitos, favorecendo a qualidade dos requisitos identificados, e na redução do tempo de execução e, conseqüentemente, na diminuição do custo desta operação.

A utilização de padrões no desenvolvimento de software representa o uso de um conhecimento sobre a construção de software adquirido por muitas pessoas durante anos. A tendência é de que a cada dia mais pessoas conheçam padrões e os utilizem no desenvolvimento de seus projetos. Um software desenvolvido com a utilização de padrões terá uma boa documentação e permitirá que as manutenções no futuro se tornem mais fáceis, caso necessário, facilitando a compreensão de sua documentação. Além do mais, uma equipe de desenvolvimento de uma nova empresa, por exemplo, poderá através do uso de padrões desenvolver aplicações com a mesma experiência ou pelo menos próxima de uma equipe de outra empresa que tem anos de trabalho nesta área. A tendência é que pequenas e novas empresas ou até mesmo equipes de trabalho de empresas mais antigas utilizem padrões desfrutando dos seus benefícios, os quais já foram apresentados anteriormente.

Efetivamente o sucesso nessa fase do desenvolvimento trará benefícios a fases seguintes do desenvolvimento do software.

7.3 Limitações

Até pela complexidade, o pesquisador se limitou a trabalhar para um tipo especifico de software, o que pode comprometer a aplicabilidade deste

catálogo a outros tipos de software, mas sem comprometer a originalidade do processo nem os resultados obtidos no tipo de software proposto nesta Tese.

Neste trabalho também não foi abordado o conceito de anti-padrões, em função da sua real aplicabilidade e a relação com os padrões sugeridos nessa pesquisa. Anti-padrões ainda são recentes, e o foco maior na comunidade continua sendo sobre padrões, mas a cada dia vêm crescendo mais as atenções sobre anti-padrões.

Alguns anti-padrões, além de descrever uma solução ruim, mostram como se pode sair dela e chegar a uma boa solução para o problema. É muito importante, assim como conhecer as boas soluções, conhecer como não se devem fazer as coisas.

Aqui, mais uma vez, perceberemos que é possível aprender a partir de experiências passadas. Não precisamos repetir os erros dos outros ou refazer as coisas, caso já tenham sido feitas ou descobertas, assim como para os padrões existem referências que documentam alguns anti-padrões.

Duas publicações recentes sobre anti-patterns, são os livros dos autores *Brown, Malveau, McCormick e Mowbray,* intitulados *Anti-patterns: Refactoring Software, Architecture and Projects in Crisis* e *Anti-patterns in Project Management.*

Portanto, a documentação de soluções ruins é muito importante, mas mais importante que isto é não somente identificar estas soluções, mas sim mostrar como se pode sair delas. É esta última que as pessoas mais buscam em anti-padrões.

7.4 Comparação com Outros Padrões Relacionados a Software

Embora haja vários padrões relacionados a software, cada modelo procura tratar problemas diferentes, como arquitetura, requisitos, testes, projetos etc. A seguir é feito uma breve análise dos principais catálogos ou linguagens de padrões com os padrões apresentados pelo pesquisador nesta Tese:

7.4.1 Modelo da Deutsches Elektronen-Synchrotron (DESY)

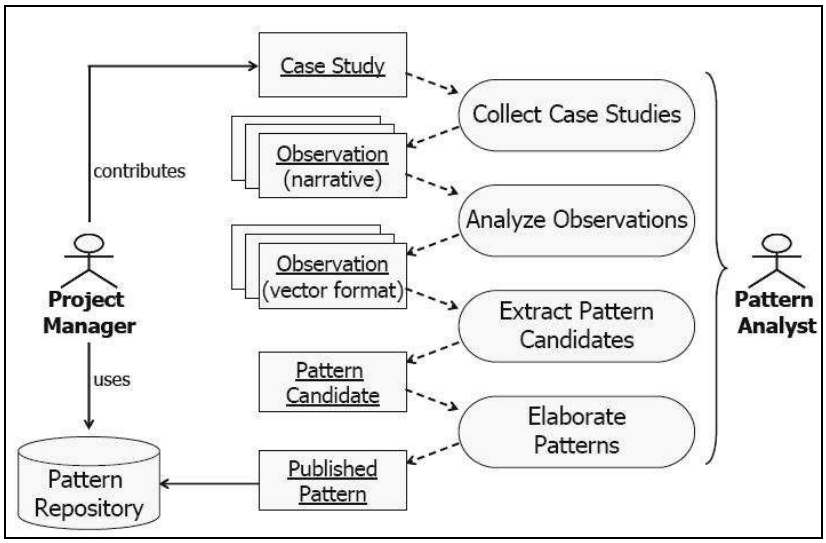

Figura 7.1 - Padrões da Deutsches Elektronen-Synchrotron (DESY).

O modelo da DESY se refere a um repositório de padrões aplicados na área Física de forma similar. Diferentemente dos padrões proposto na pesquisa desta Tese, não tem as características de instanciação.

7.4.2 Modelo de James Robertson & Robertson de Requirements Patterns

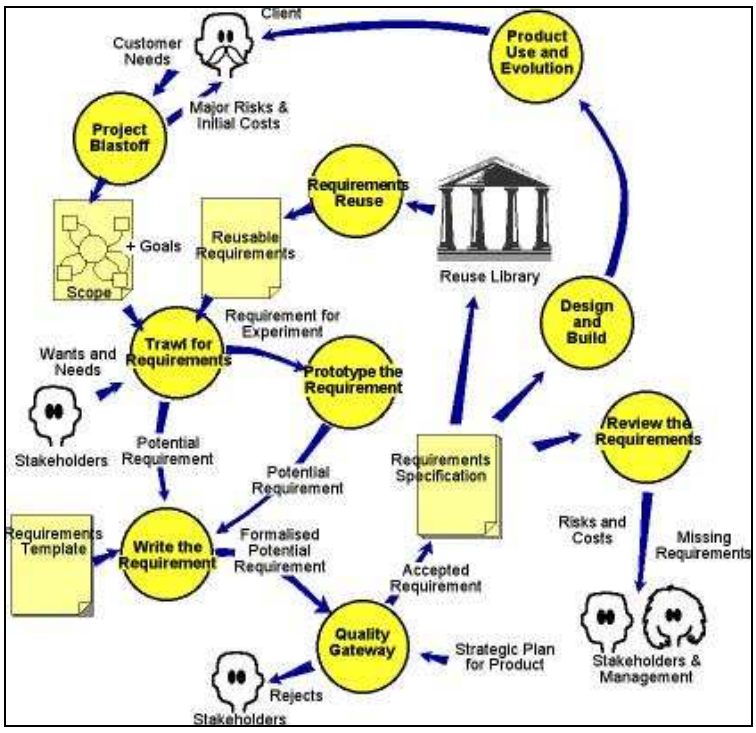

Figura 7.2 - Padrões de James Robertson & Robertson.

Assim como os padrões das DESY, os padrões de Robertson & Robertson, propõem *templates* que podem ser utilizados na aplicação dos padrões, entretanto neste caso é possível instanciá-los, similarmente aos padrões apresentados nesta Tese.

7.4.3 Modelo de Gof4 – Design Patterns

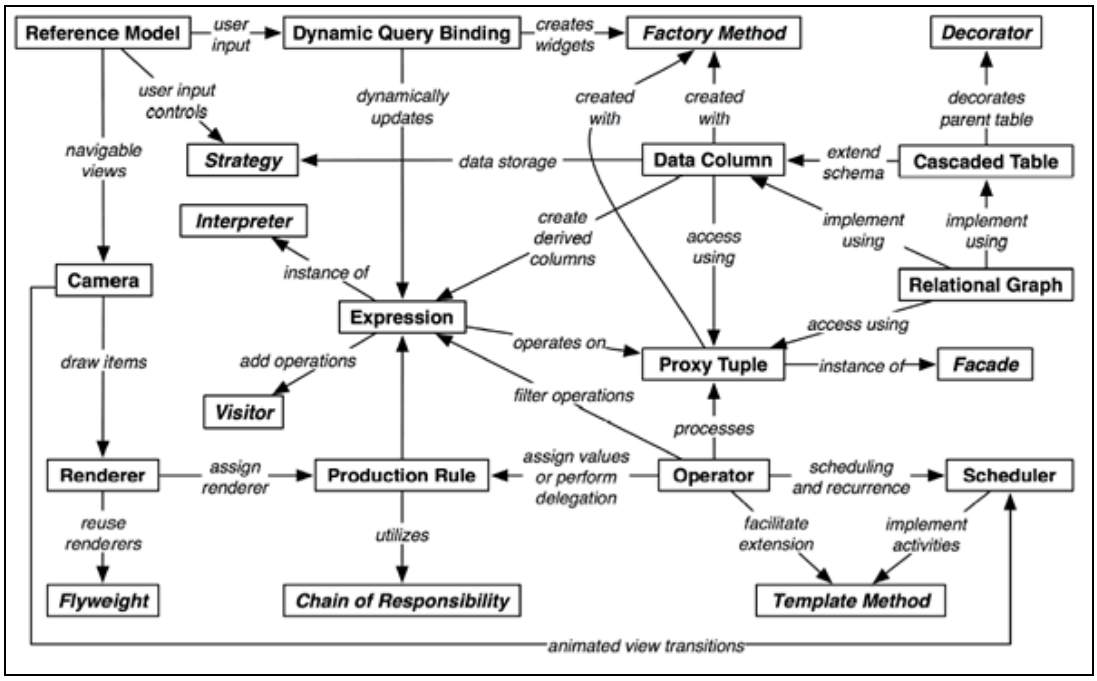

Figura 7.3 - Padrões da Design Patterns – Gamma (GoF).

Padrões que trazem os mesmo conceitos de aplicação dos padrões candidatos desta Tese, embora sejam destinados a projetos e o desta pesquisa, para requisitos. Ambas apresentam padrões instanciáveis.

7.4.4 Modelo de Withall

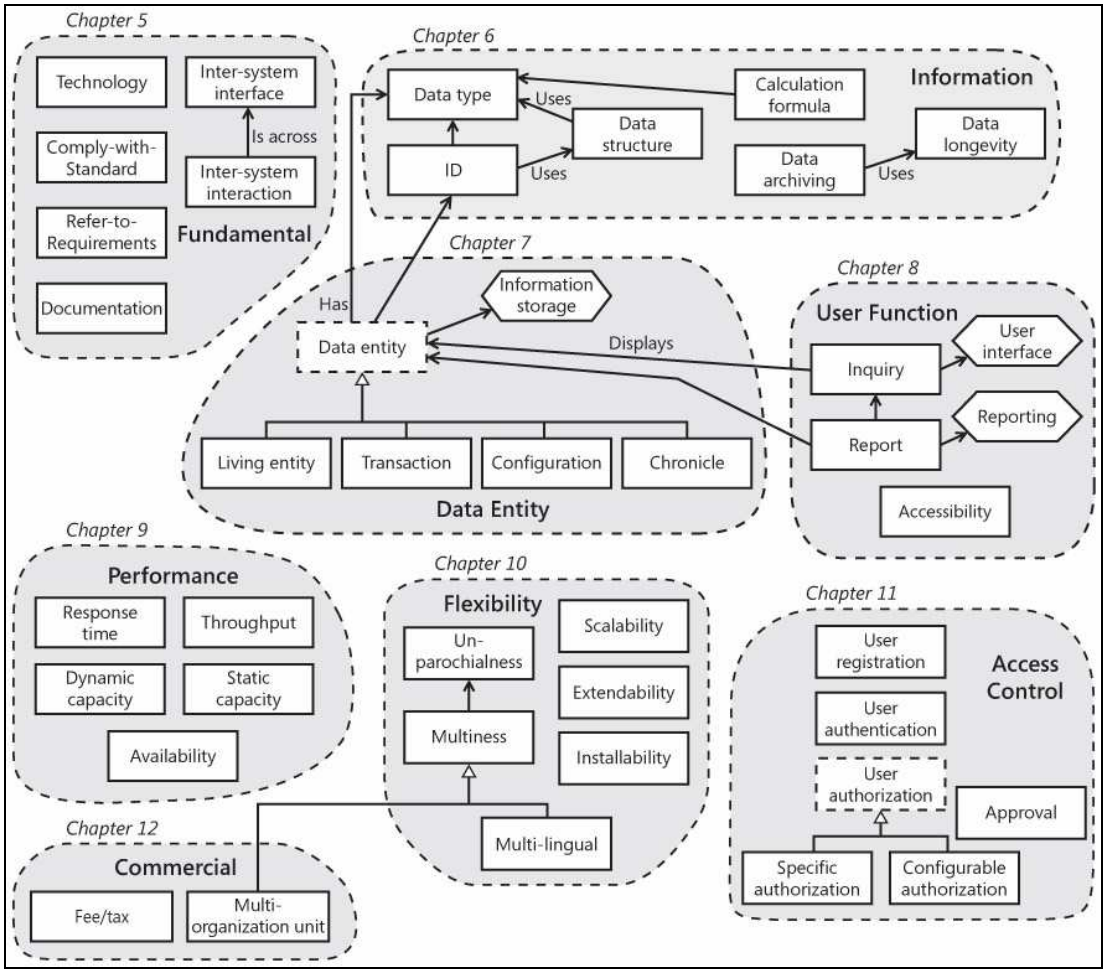

Figura 7.4 - Padrões de Software de Withall.

É um dos trabalhos publicado em livros, mais recente (2007). Está relacionado aos padrões de elementos e recursos organizacionais, como impostos, cargos e funções, controle de acesso etc, que contribuirão com a construção do software. Assim como a maioria das referências de padrões utilizadas pelo autor nesta Tese, Withall não deixa claro seu método de criação desses padrões, nem mesmo sua aplicabilidade.

7.5 Considerações Finais

A pesquisa mostrou que embora haja uma lacuna entre o que se pretende construir com o que é construindo, as pesquisas na área vêm

crescendo e em breve haverá ferramentas que suportam - ou pelo menos suprirão a maioria – número de variáveis existentes em diversos contextos de negócio.

A adaptação deste catálogo para software de outras áreas como, por exemplo, *inteligência artificial, sistemas críticos e embarcados*, não é uma tarefa simples, mas que poderá ser aprimorado e conseqüentemente utilizado com eficiência, mas, para isso, será necessária a contribuição de profissionais experientes, e que saibam, empiricamente, quais os elementos importantes de cada área e se repetem com freqüência dentro de um contexto habitual.

7.6 Trabalhos Futuros

Embora o pesquisador já tenha dado início ao desenvolvimento, o trabalho favorece a criação de uma ferramenta CASE para auxiliar no processo de levantamento de requisitos, consistindo as informações levantadas no momento da elicitação de requisitos, fazendo inclusive o relacionamento entre os diversos padrões automaticamente.

Outro trabalho que pode ser desenvolvido é a ampliação do catálogo para outras áreas de negócio e de tipos de sistemas, como os excluídos citados nesta Tese.

Finalmente, fazer o relacionamento dos elementos do catálogo de padrões com os documentos de requisitos gerados, ao processo de gerenciamento de requisitos em seu ciclo de vida. Para isso, é necessária a agregação de alguns atributos de temporalidade aos requisitos, nominação e referência da origem da informação, pessoa e área, além da geração da base de informações e da ferramenta para registro das mudanças para que a operação possa ser feita de modo seguro.

Referências

APPLETON, Brad. Patterns **and Software: Essential Concepts and Terminology**. Disponível em http://www.cmcrossroads.com/bradapp/docs/patternsintro.html (1997). último acesso em: 05/08/2007.

ARAUJO, João. **Padrões de Requisitos e Análise, e Anti-padrões.** Material de Apoio - Requisito vs Análise (crítica). disponível em: http://www-ctp.di.fct.unl.pt/~ja/ces.htm. último acesso em 10/08/2005.

ALEXANDER, Christopher et al. **A Pattern Language**. Oxford University Press, 1979.

ASTELS, David; MILLER, Granville; NOVAK, Miroslav. **Extreme Programming: Guia Prático**. Rio de Janeiro: Campus, 2002.

AMBLER, Scott W. **Building Object Applications That Work**. Cambridge University Press : SIGS Books, 1998.

BECK, Kent. **"Extreme Programming Explained: Embrace Change".** Addison Wesley Professional, 2000.

BERRY, D.M.; LAWRENCE, B. **Requirements Engineering**.1ed. USA : IEEE Software, p 26-29, april / 1998.

BEYER, H.; HOLTZBLATT, K. **Contextual Design: Defining Customer- Centered Systems**. Morgan Kaufmann Publishers, Inc. 1998.

BROOKS, Frederick. P. Jr. **The Mythical Man–Month: Essays on Software Engineering**. 2ª Edition. Addison–Wesley, 1995.

BRYMAN, Alan. **Research Methods and Organizational Studies**. Inglaterra: Urwin Hyman Ltd., 1989.

BUSCHMANN, Frank et al. **Pattern Oriented Software Architecture: A System of Patterns**. John Wiley & Sons, 1996.

BUSCHMANN, F.; JOHNSON, R.; COPLIEN, J.; RISING, L.; DELANO, D.; GAMMA E.; SCHMIDT, D. **How to Hold a Writer's Workshop**. Ultimo acesso em 09/03/2008. http://www.cs.wustl.edu/~schmidt/writersworkshop.html. 1996a

CHIAVENATO, I. **Introdução à teoria geral da administração**: **uma visão abrangente da moderna administração nas organizações**. 7ª ed. Rio de Janeiro: Elsevier, 2003.

CLAVER, E.; GONZÁLEZ, R.; LLOPIS, J. **An analysis of research in information systems (1981-1997)**. USA: Information & Management 37, 2000.

COPLIEN, J. O.; SCHMIDT, D**. Pattern Languages of Program Design**, Reading-MA, Addison-Wesley. 1995.

COPLIEN, James O. **Software Design Patterns: Common Questions and Answers.** in: Rising L., (Ed.), The Patterns Handbook: Techniques, Strategies, and Applications, Cambridge University Press, New York, pp. 311-320, January 1998. Disponível em http://st.cs.uiuc.edu/pub/patterns/papers/PatQandA.ps

CROSBY, Philip B. **Quality Is Free: The Art of Making Quality Certain**. Mentor Books, Denver Colo, 1992.

CROOM, SIMON. **Methodology Editorial**. IJOPM – Special Issue on Research Methodology in Operations Management, International Journal of Operations and Production Management Vol. 22, No. 2, pp. 148-151. 2002.

CROZATTI, Jaime. **Modelo de Gestão e Cultura Organizacional – Conceitos e Interações**. Cadernos de Estudos, São Paulo , FIPECAFI, V.10, n.18, p. – maio/agosto, disponível em http://www.eac.fea.usp.br/cadernos/completos/cad18/_modelogestao.pdf . 1998.

CUNNINGHAM, W. **The CHECKS Pattern Language of Information Integrity**, in Coplien & Schmidt, p. 145-155. 1996.

DAMIAN, Adrian, et al. **"Joint Application Development and Participatory Design"**. http://www.cpsc.ucalgary.ca/~pand/seng/613/report.html . 1997

DAVIS, Alan M. **201 Principles of Software Development**. 2ª edition. McGraw-Hill. 1993.

DAVIS, A., HICKEY A., **Requirements Researchers: Do We Practice What We Preach?** , Requirements Engineering Journal, 7 vol , pp107-111, 2002.

DEUTSCHES Elektronen-Synchrotron: **DESY'06**. Disponível em: <http://www.desy.de>. Acesso em: 01 ago. 2006.

EASTERBROOK, Steve. **Empirical Research Methods in Requirements Engineering - Tutorial T1**. 15th IEEE International Requirements Engineering Conference (RE'07). India Habitat Center, New Delhi. October 15-19th. 2007.

ECO, Humberto. **Como se faz uma tese**. São Paulo : Perspectiva, 1977.

ERICSSON, K. A., SIMON, H. A **Protocol Analysis: Verbal Reports as Data**. MIT Press, Cambridge, MA. 1993.

ESPITI - European Software Process Improvement Training Initiative: **Personal Software Processes.** Workshop Berlin 7 May 1996. Report in September 1996.

FAULK, Stuart R. **"Software Requirements: A Tutorial" in Software Requirements Engineering,** IEEE-CS Press, Second Edition, p.p. 128-149, 1997.

FINKELSTEIN, A., KRAMER, J., NUSEIBEH, B., GOEDICKE, M. **Viewpoints: a framework for integrating multiple perspectives in system development.** International Journal of Software Engineering and Knowledge Engineering, 1992, pp.31-58.

FOWLER, M. **Analysis Patterns: Reusable Object Models**, Addison Wesley, 1996.

GABRIEL, Richard P. **Patterns of Software: Tales from the software community.** Oxford: Oxford University Press, 1996.

GAMMA, E. et al. **Design Patterns: Elements of Reusable Object-Oriented Software**. Boston: Addison-Wesley, 1995.

GAUSE, Donald C. **Seeing Customer Requirements: Defining Quality Before Design, Assuring Quality During Design, Improving Quality After Design**. ICRE'98 Third International Conference on Requirements Engineering (Colorado Springs, Colorado, USA) 1ed. USA : IEEE CSP, Los Alamitos, CA. Tutorial, 1998, abril, 22 p.

GAUSE, D. C., WEINBERG, G. M. **Are Your Lights On? How to Figure Out What the Problem Really Is**. 1ed. USA : Dorset House Publishing Co. Inc., 1990.

GIL, Antonio C. **Como Elaborar Projetos de Pesquisa**. 4ª ed. São Paulo: Atlas, 2002.

GOGUEN, J., C. LINDE, **Software Requirements Analysis and Specification in Europe: An Overview**, First International Symposium on Requirements Engineering, IEEE Computer Society Press, pp. 152-164, 1993.

GOGUEN, Joseph A. **Techniques for Requirements Elicitation**. In Software Requirements Engineering, IEEECS Press, Second Edition, p.p.110 –122. 1997

GOTEL, Orlena; FINKELSTEIN, Anthony. **Extended Requirements Traceability: Results of na Industrial Case Study**. ISRE'97 Third International Symposium on Requirements Engineering. 1ed. USA:IEEE CSP, Los Alamitos, CA. Proceedings, 1997.

GREENFIELD, J.; SHORT, K. **Software Factories - Assembling Applications with Patterns, Models, Frameworks, and Tools**. Wiley Publishing, Inc, 2004.

GRUNDY, S. J. **Three modes of action research. Curriculum Perspectives**, Geelong, v. 2, n. 3, p. 23-34, 1983.

HAGGE, L., LAPPE K. **Using Requirements Engineering (RE) Patterns for Organizational Learning**. Journal of Universal Knowledge Management. Version 1. 137-148. 2006.

HAMMER, T. et al. **Requirement Metrics – Value Added**. ISRE'97 Third International Symposium on Requirements Engineering. 1ed. USA:IEEE CSP, Los Alamitos, CA. Proceedings, 1997.

HARRISON, N., FOOTE, B., ROHNERT, H. **Pattern Languages of Program Design 4**. Addison-Wesley, 1999.

HOLLENBACH, C., FRAKES, W. **Software Process Reuse in an Industrial Setting.** Fourth international Conference on Software Reuse, Orlando, Florida, IEEE Computer Society Press, Los Alamitos, CA, pp 22-30, 1996.

HOLTZBLATT, K., JONES, S. **Contextual Inquiry: a participatory technique for systems design**. in Participatory Design: principles and practice. Editors: Schuler, D. and Namioka, A., Lawrence Erlbaum, New Jersey, pp 177-210, 1993.

IVERSEN, Jakob H.; MATHIASSEN, Lars ; NIELSEN, Peter Axel, **Managing Risk in Software Process Improvement: An Action Research Approach**, MIS Quarterly, Vol. 28, No. 3, September 2004.

JACKSON, Michael. **Software Requirements and Specifications: A Lexicon of Pratice, Principles and Prejudices**. Addison-Wesley, 1ed. USA, Massachustes, Reading. 228p. 1995.

JACKSON, Michael. **Conectando Viewpoint by Shared Phenomena**. ISAW-2 International Workshop on Multiple Perspectives in Software Development. (San Francisco, CA, USA)1ed. USA : ACM. Joint Proceedings SIGSOFT'96, p 180-183. 1996.

KARLSSON, C. **Guest Editorial**. IJOPM – Special Issue on Research Methodology in Operations Management, International Journal of Operations and Production Management Vol. 22, No. 2, p 141-147. 2002.

KAUPPINEN, Marjo; SULONEN, Reijo. **A Practical Framework for Requirements Engineering**. ISRE'97 Third International Symposium on Requirements Engineering. (Annapolis, Maryland, USA) 1ed.USA : IEEE CSP, Los Alamitos,CA. Proceedings, january, p59-64. 1997.

KIRAWOWSKI, Jane; **Requirements Engineering and Specification in Telematics: Methods for User-Orientated Requirements Specification**. Human Factors Research Group, Cork, Ireland, disponível em http://www.ejeisa.com. 1997

KOSCIANSKI, A.; SOARES, M. S. **Qualidade de Software**. 1ª Ed. São Paulo: Novatec, 395 p. 2006.

KOTONYA, Gerald; SOMMERVILLE, Ian. **Requirements Engineering with Viewpoints**. Software Engineering Journal. Vol. 11. No 1. Pages 5−11. 1996.

KOTONYA, Gerald; SOMMERVILLE, Ian. **Requirements Engeneering (Processes and Techniques)**. 1ed. England : John Wiley & Sons Ltd, 1998.

KOTONYA, Gerald. **Practical Experience with Viewpoint-Oriented Requirements Specification**. Requirements Engineering Journal. Vol. 4. No. 3. Pages 115−133. 1999.

KRUCHTEN, Philippe. **The Rational Unified Process: An Introduction**. Object Technology Series. 2 ed. Addison-Wesley, 2000.

LAKATOS, E ; V., MARCONI, M. A.**Fundamentos da metodologia científica**. São Paulo: Atlas,6ª Edição, 2005.

LAMSWEERDE, Axel van, **Requirements Engineering in the Year 00: A Research Perspective**. 22nd International Conference on Software Engineering (ICSE"2000), 2000.

LAPPE K.,CZIHARZ T.,DREIER H.,FAHNEY R.,GUMM D.,HAGGE L.,HÖHNE G.,HOUDEK F.,ITTNER J.,JANZEN D.,PAECH B. **Requirements Engineering Patterns: An Approach to Capturing and Exchanging Requirements Engineering Experience**. Working Group on Requirements Engineering Patterns (WGREP) of Section 2.1.6 “Requirements Engineering” in the German Informatics Society (GI). DESY 04-233. 2004.

LARMAN, Craig. **Utilizando UML e Padrões: uma introdução à análise e ao projeto orientados ao objetos e ao Processo Unificado**. Trad. Luiz A. M. Salgado e João Tortello. Porto Alegre: Bookman, 2004.

LEITE, Júlio C.S.P. **Viewpoint Analysis: A case Study**. IWSSD'89 Fifth International Workshop on Software Specification and Design. (Pittsburg, Pensylvania, USA) 1ed.USA : ACM Sigsoft Engineering. Proceedings, p111-119. may / 1989.

LEITE, Júlio C.S.P; FRANCO, Ana P.M. **A Strategy for Conceptual Model Acquisition**. ISRE'93 First International Symposium on Requirements Engineering. (San Diego, CA, USA) 1ed.USA : IEEE CSP,Los Alamitos, CA. Proceedings, january,p243-246. 1993.

LEITE, J.C.S.P; OLIVEIRA, A.P.A. **A Client Oriented Requirements Baseline**. ISRE'95 Second International Symposium on Requirements Engineering. (York, England, UK) 1ed. USA : IEEE CSP,Los Alamitos, CA. Proceedings, p108-115. march / 1995.

LEITE, J.C.S.P et al. **Enhancing a Requirements Baseline with Scenarios.** ISRE'97 Third International Symposium on Requirements Engineering. 1ed.USA : IEEE CSP, Los Alamitos, CA.Proceedings, p 44-53, January, 1997.

LEITE, Júlio C.S.P; LEONARDI, Maria C. **Business Rules as Organizational Policies**. IWSSD'98 Ninth International Workshop on Software Specification Design. (ISE-Shima, Japan) 1ed.USA : IEEE CSP, Los Alamitos, CA. Proceedings, p 68-76. april / 1998.

LEITE, Júlio C.S.P. **Como Registrar Requisitos de Software**. Livro Qualidade 2002, diponível em www-di.inf.puc-rio.br/~julio/Livro-qualidade-2002.pdf, último acesso em 19/10/2006. 17 pags.

LEFFINGWELL, Dean; WIDRIG, Don. **Managing Software Requirements– a unified approach**. 1ed. USA : Indianápolis, IN ; Addison-Wesley, 2001, 491 p.

MACEDO, Nestor A. M. **Integrando Requisitos Não Funcionais aos Requisitos Baseados em Ações Concertas.** Dissertação de mestrado, Departamento de informática, PUC-RIO, Maio, 1999.

MACAULAY, Linda A. **Requirements Engineering**. 1ed. Great Britain : Springer-Verlag London Limited, 1996, 202 p.

MARCA, D.A.; MCGOWAN, C. L. **SADT: Structured Analysis and Design Techniques**. McGraw-Hill. 1988.

MARTIN, R.C.; RIEHLE, D.; BUSCHMANN, F. **Pattern Languages of Program Design 3**, Reading-MA, Addison-Wesley. 1998.

MARTINS, Luiz E. G. **Utilização dos Preceitos da Teoria da Atividade na Elicitação de Requisitos do Software** , XII Simpósio Brasileiro de Engenharia de Software, Out. 1999.

MESZAROS Gerard; DOBLE, Jim. **A Pattern Language for Pattern Writing**. Pattern Language of Program Design 3, edited by Robert C. Martin, Dirk Riehle, and Frank Buschmann, Addison-Wesley (Software Patterns Series), 1997.

MILLARD, Nicola; LYNCH, Paula; TRACEY, Karina. **Child's Play: using Techniques Developed to Elicit Requirements from Children with Adults**. ICRE'98 Third International Conference on Requirements Engineering (Colorado Springs, Colorado, USA) 1ed.USA : IEEE CSP, Los Alamitos,CA. Proceedings, p 66-73, april / 1998.

MILLEN, David R. **Rapid Ethnography: Time Deepening Strategies for HCI Field Research**, Brooklyn, New York, 2000.

MULLERY, G.P. **CORE: . A Method for Controlled Requirement Specification**. Proceedings of IEEE Fourth International Conference on Software Engineering. 1979.

NAUR, P.; RANDELL , B. **Software Engineering**: **Proceedings of the NATO Conference on Software Engineering.** Editors Report of a conference sponsored by the NATO Science Committee, Garmisch, Germany, 7-11 October 1968, Brussels, Scientific Affairs Division, NATO (1969) 231pp.

NGUYEN, L., SWATMAN, P. **Managing the Requirements Engineering Process**. Journal of Requirements Engineering, Springer-Verlag London Ltd, England, pp 55 – 68.2003

NETO, J.S.M. **Integrando Requisitos Não Funcionais ao Modelo de Objetos**. Dissertação de Mestrado da PUC-Rio, Mar/2000

OLIVEIRA, K.R.; SPINOLA, M.M.. **Patterns-Oriented Requirements Elicitation Integrated – Proposal of a Metamodel Patterns-Oriented for Integration of the Requirement Elicitation Process** In: EURO AMERICAN ASSOCIATION ON TELEMATICS AND INFORMATION SYSTEMS - EATIS 2007, v. 1, p. 43, 2007. Faro/Algarve - Portugal, 14-17 May 2007.

OLIVEIRA, K.R.; SPINOLA, M.M. **POREI: Patterns-Oriented Requirements Elicitation Integrated: Proposta de um Metamodelo Orientado à Padrão para Integração do Processo de Eliciação de Requisitos**. In: 6th LATIN AMERICAN CONFERENCE ON PATTERN LANGUAGES OF PROGRAMMING - SugarLoafPLoP 2007, Porto de Galinhas/PE. 23-30 Maio, 2007.

OLIVEIRA, K.R.; SPINOLA, M.M. **PORE Patterns-Oriented Requirements Engineering - Proposta de um Meta-Modelo Orientado ao Processo de Especificação de Requisitos de Software**. In: XIII SIMPÓSIO DE ENGENHARIA DE PRODUÇÃO - SIMPEP, Bauru/SP. 2006.

PINHEIRO, Francisco A.C.; GOGUEN, Joseph A. **An Object Oriented Tool for Tracing Requirements.** 1ed. USA:IEEE Software, 1996.

PORTLAND Patterns Repository. Disponível em: http://c2.com/ppr/index.html. último acesso em: 02/09/2007.

PRESSMAN, Roger S. **Engenharia de software**, 5a. edição. McGraw Hill, 843 p., 2002.

PRIETO-DIAZ, R.; ARANGO, G. **Domain Analysis and Software Systems Modeling.** IEEE Computer Society Press Tutorial. IEEE Computer Society Press, Los Alamitos CA, 1991.

RANDELL ,B.; BUXTON, J.N. **Software Engineering Techniques**: (Eds.) Report of a conference sponsored by the NATO Science Committee, Rome, Italy, 27-31 Oct. 1969, Brussels, Scientific Affairs Division, NATO (1970) 164pp.

ROBERTSON, Suzanne. **Mastering the Requirements Process 2 Book Cased (Hardback)**, Addison-Wesley, 2a ed, 2006.

RYAN, Kevin. **Requirements Engineering - getting value for money**. SBES´98, XII Simpósio Brasileiro de Engenharia de Software 1.ed. Brasil : SBC Maringá, Paraná, 55 p. , outubro /1998.

SAWYER, Pete; SOMMERVILLE, Ian; VILLER, S.**. Requirements Process Improvement Through The Phased Introduction of Good Practice**. *Software Process – Improvement and Practice*, 1997. disponível em www.comp.lancs.ac.uk/computing/research/cseg/reaims/publications.html, último acesso 20/09/2003.

SHALLOWAY, A.; TROTT, J. **Explicando Padrões de Projeto – Uma Nova Perspectiva em Projeto Orientado a Objetos,** Bookman, 2004.

SHELDON, Frederick**. Reliability Measurement from Theory do Practice**. 1ed. USA : IEEE Software, p 10. july / 1992.

SIDDIQI, J.; SHEKARAN, M.C. **Requirements Engineering: the Emerging Wisdom**. 1ed. USA : IEEE Software, p 15-19. march / 1996.

SIDDIQI, Jawed. **"Requirements Engineering: the emerging window"** in Software Requirements Engineering, IEEE-CS Press, Second Edition, p.p. 36-40, 1997.

SCHMIDT, D; STAL, M; ROHNERT, H.; BUSCHMANN, F.**Pattern-Oriented Software Architecture, Patterns for Concurrent and Networked Objects**, John Wiley & Sons. Vol 2. 2000.

SOMMERVILLE, Ian. **Engenharia de Software.** Tradução A.M. Andrade. Revisão Kechi Hirama. Addison Wesley, 6ed. São Paulo, 2003.

SOMMERVILLE, Ian; SAWYER, Pete. **Requirements Engineering (A Good Practice Guide).** 1ed. England : John Wiley & Sons Ltd, 1997, 391p.

SOMMERVILLE, Ian; SAWYER, Pete; VILLER, S. **Viewpoint for Requirements Elicitation: a practical approach.** ICRE'98 Third International Conference on Requirements Engineering (Colorado Springs, Colorado, USA) 1ed.USA : IEEE CSP, Los Alamitos, CA. Proceedings, april, p 74-81 1998.

SOMMERVILLE, I.; RODDEN, T.; SAWYER, P.; BENTLEY, R.; TWIDALE, M.**Integrating ethnography into the requirements engineering process**. Proceedings of the IEEE International Symposium on Requirements Engineering. Pages 165-173.1993.

STRINGER, Ernest T. **Action Research: a Handbook for Practitioners**. Sage, 1996.

SUTCLIFFE, A. **Scenario-Based Requirement Analysis. Requirements Engineering Journal**. Vol. 3. No 1. Pages 48-65. 1998.

SUTCLIFFE, A.; RYAN, M. **Experience with SCRAM, a Scenario Requirements Analysis Method**. Proceedings of the Third International Conference on Requirements Engineering. 1998.

TELES, Vinícius Manhães. **Extreme Programming: Aprenda como encantar seus usuários desenvolvendo software com agilidade e alta qualidade**. São Paulo: Novatec, 2004. 316 p.

THAYER, R.H.; DORFMAN, M. **Software Requirements Engineering**. Segunda edição, IEEE Computer Society, 1997.

THIOLLENT, M. **Metodologia da pesquisa-ação**. 14ª ed. São Paulo: Editora Cortez. 132p. 2005.

TRIPP, David**. Pesquisa-ação: uma introdução metodológica**. Revista Educação e Pesquisa, São Paulo, v. 31, n. 3, p. 443-466, set./dez. 2005

VALENTE, Luis. **Design Patterns – Padrões Para Projetos**. Dissertação Mestrado. Universidade Federal Fluminense, 2003.

VLISSIDES, J. **Pattern hatching – design patterns applied**. Addison-Wesley, 1998.

VLISSIDES, J.; COPLIEN, J.; KERTH, N. **Pattern Languages of Program Design 2**. Reading-MA; Addison-Wesley. 1996

VOSS, C.; TSIKRIKTSIS, N.; FROHLICH, M. **Case research in Operations Management**. International Journal of Operations and Production Management, Vol. 22, No. 2, pp. 195-219. 2002.

WEINBERG, Gerald M., **Quality Software Management.** Volume 3: Congruent Action, ISBN 0-932633-28-5, Dorset House, New York, NY, 1994.

WESTBROOK, Roy. **Action research: a new paradigm for research in production and operations management**. International Journal of Operations and Production Management, Vol. 15, No. 12, pp. 46-58.1994.

WIERINGA, Roel. **Advanced Structured and Object-Oriented Requirements Specification Methods.** ICRE'98 Third International Conference on Requirements Engineering (Colorado Springs, Colorado, USA) 1ed. USA : IEEE CSP, Los Alamitos, CA. Tutorial, 50 p. april / 1998.

WILSON, B.; DUFFY, M. **Soft Systems Methodology: Conceptual Model Building and Its Contribution**. John Wiley & Sons. 2001

WITHALL, Stephen J**. Software Requirement Patterns (Best Practices)**. Microsoft Press. 2007.

WRIGHT, Richard**. Game design: theory and practice.** Entrevista in: ROUSE III, Plano, Texas: Wordware Publishing, 2001.

YIN, Robert K. **Estudo de Caso – Planejamento e Métodos**. tradução de Daniel Grassi. Brasil: Bookman Companhia Editora, 1994.

YU, E., **Modelling Strategic Relationships for Process Reengineering**, Phd Thesis, University of Toronto, 1995.

ZANLORENCI, Edna P.; BURNETT, Robert C. **Modelo para Qualificação da Fonte de Informação Cliente e de Requisito Funcional.** SBES98, XII Simpósio Brasileiro de Engenharia de Software WER'98, I Workshop de Engenharia de Requisitos. 1.ed. Brasil : SBC, Maringá,Paraná. Anais 1998, outubro, vol.1, n1, p39-48.

ZANLORENCI, Edna. **Descrição e Qualificação de Requisitos: um modelo aplicável à análise e validação da informação**. 245f. Dissertação (Mestrado em Informática Aplicada) - Centro de Ciências Exatas e de Tecnologia, Pontifícia Universidade Católica do Paraná, Curitiba, 1999.

ZAVE, P.; JACKSON, M. **Four Dark Corners of Requirements Engineering**. 1ª ed. USA: ACM, Inc - Association for Computing Proceedings 1997, january, vol.6, n.1, p 1-30.

APÊNDICE A – Extração dos Padrões Candidatos – Requisitos Funcionais

A.1 - Padrão Candidato I

Identificação [P]	"Usuário".
Problema [R]	"Acessar o Sistema".

Parte do Requisito (Projeto A)	"... os usuários devem utilizar apenas partes do sistema que lhe são atribuídos de acordo com suas funções dentro da sua unidade, com objetivo de padronizar os processos. Mas o gerenciamento dos usuários deve ser flexível, pois um funcionário em determinadas ocasiões poderá substituir outro colaborador em horários diferentes..."
Elicitação Aplicada [A]	Caso de Uso, Observação e Análise de Protocolo.
Solução [S]	Criar um controle de acesso aos usuários com senhas e controle de prioridades de acessos que se relacionem as unidades e funcionalidades do sistema de informação.
Forças	$F^{\leftarrow}$ = "padronizar os processos". $F^{\rightarrow}$ = "o gerenciamento deve ser flexível".
Vetor	(padronizar processos, o gerenciamento deve ser flexível, criar um controle de acesso aos usuários com senhas e prioridades, [caso de uso, observação e análise de protocolo])

Parte do Requisito (Projeto B)	"... Dever haver um controle de usuários no acesso ao sistema, pois estes terão suas produtividades gerenciadas. Nenhum usuário poderá ter acesso à senha alheia, mas poderá utilizar terminais diferentes (aleatoriamente) de acordo com a ordem de chegada na empresa...".
Elicitação Aplicada [A]	Caso de Uso, Observação e Análise de Protocolo.
Solução [S]	Criar um controle de acesso aos usuários com senhas e prioridades.
Forças	$F^{\leftarrow}$ = "controle de usuários". $F^{\rightarrow}$ = "utilizar terminais diferentes".
Vetor	(controle de usuários, utilizar terminais diferentes, criar um controle de acesso aos usuários com senhas e prioridades, [caso de uso, observação e análise de protocolo])

Parte do Requisito (Projeto C)	"... cada usuário terá um perfil para utilização do sistema, onde o software terá que ser customizado de acordo com a função de cada colaborador. Será gerado um controle (*log*) de acesso, e este será independente do computador utilizado pelos usuários. Usuários do mesmo nível funcional terão os privilégios compartilhados...".
Elicitação	Caso de Uso, Observação e Análise de Protocolo.

Aplicada [A]	
Solução [S]	Criar um controle de acesso aos usuários com senhas e prioridades agrupadas por funcionalidades.
Forças	$F^{\leftarrow}$ = "utilizar o sistema adequadamente". $F^{\rightarrow}$ = "mesmo nível funcional terão os privilégios compartilhados".
Vetor	(utilizar o sistema adequadamente, mesmo nível funcional terão os privilégios compartilhados, criar um controle de acesso aos usuários com senhas e prioridades agrupadas por funcionalidades, [caso de uso, observação e análise de protocolo])

A.1.1 – Extraindo Padrão Candidato I

É feita a organização das observações comuns:

$$A = (F^{\leftarrow 1}, F^{\rightarrow 1}, S1, A1) \;;\; B = (F^{\leftarrow 2}, F^{\rightarrow 2}, S1, A1) \text{ e } C = (F^{\leftarrow 3}, F^{\rightarrow 3}, S2, A1)$$

Onde,

$[F^{\leftarrow 1,2,3}]$ = "padronizar os processos"; "controle de usuários"; "utilizar o sistema adequadamente".

$[F^{\rightarrow 1,2,3}]$ = "o gerenciamento deve ser flexível" ; "utilizar terminais diferentes"; "mesmo nível funcional terão os privilégios compartilhados".

S1 = "criar um controle de acesso aos usuários com senhas e prioridades".
A1 = [caso de uso; observação; análise de protocolo].

Identificado os padrões descobertos em diferentes padrões pelo processo de elicitação, é feita a filtragem dos elementos comuns e a adição das forças que influenciam o padrão candidato.

$$\mathbf{P = (R, F^{\leftarrow 1,2,3}, F^{\rightarrow 1,2,3}, S1, A1)}$$

O resultado do padrão candidato **"usuário"** diz que o requisito ou problema **"acessar ao sistema"** recebe as forças que direcionam a ação no fortalecimento da sua implementação: **"padronizar os processos"; "controle de usuários"; "utilizar o sistema adequadamente",** mas sofre influência de forças opostas e que podem causar riscos a solução oferecida: **"o gerenciamento deve ser flexível"; "utilizar terminais diferentes"; "mesmo nível funcional terão os privilégios compartilhados"**. Para o problema em questão é oferecido solução genérica **"criar um controle de acesso aos**

usuários com senhas e prioridades", elicitada através dos métodos "**caso de uso; observação; análise de protocolo**".

***Observação**: Nos ensaios abaixo, serão omitidas as narrações relativas á extração dos padrões, entretanto continuarão com mesmo grau de detalhamento.*

A.2 - Padrão Candidato II

Identificação [P]	"Processo".
Problema [R]	"Identificar Processos".

Parte do Requisito (A)	"... O processo de agendamento é feito pela identificação do médico e a busca de seu período de atendimento, em seguida, é escolhido o horário em que o paciente será marcado..."
Elicitação Aplicada [A]	Caso de Uso, User Stories e QFD.
Solução [S]	Identificar as principais etapas dos processos.
Forças	$F^{\leftarrow}$ = "Buscar a identificação de cada processo". $F^{\rightarrow}$ = "Os processos podem se repetir em momentos distintos".
Vetor	(buscar a identificação de cada processo, os processos podem se repetir em momentos distintos, identificar as principais etapas dos processos, [caso de uso, user stories e QFD])

Parte do Requisito (B)	"... O processo de agendamento de captação de pedidos pode se feitos via internet ou fone, sendo que ambos alimentam a mesma base, e no caso da internet através de um EDI..."
Elicitação Aplicada [A]	Caso de Uso, User Stories e QFD.
Solução [S]	Identificar as principais etapas dos processos.
Forças	$F^{\leftarrow}$ = "Buscar a identificação de cada processo" $F^{\rightarrow}$ = "Processos duplicados pode ser reflexos de canais diferentes".
Vetor	(buscar a identificação de cada processo, processos duplicados pode ser reflexos de canais diferentes, identificar as principais etapas dos processos, [caso de uso, user stories e QFD])

Parte do Requisito (C)	"... O processo de lançamentos de viagens parte da identificação do cliente e do destinatário. Posteriormente, é feita a busca da taxa definida conforme a rota..."
Elicitação Aplicada [A]	Caso de Uso, User Stories e QFD.
Solução [S]	Identificar as principais etapas dos processos

Forças	$F^{\leftarrow}$ = “Buscar a identificação de cada processo” $F^{\rightarrow}$ = “Informações dos processos depende de etapas ou processos anteriores”.
Vetor	(buscar a identificação de cada processo, informações dos processos depende de etapas ou processos anteriores, identificar as principais etapas dos processos, [caso de uso, user stories e QFD])

A.2.1 – Extraindo Padrão Candidato II

$$A = (F^{\leftarrow 1}, F^{\rightarrow 1}, S1, A1) \; ; B = (F^{\leftarrow 1}, F^{\rightarrow 2}, S1, A1) ; C = (F^{\leftarrow 1}, F^{\rightarrow 3}, S1, A1)$$

Portanto, P = (R, $F^{\leftarrow 1}$**,** $F^{\rightarrow 1,2,3}$**,S1,A1)**

O resultado do padrão candidato **“processo”** diz que o requisito **“identificar processos”** recebe as forças que direcionam a ação no fortalecimento da sua implementação: **“buscar a identificação de cada processo”,** mas sofrem influência de forças opostas e que podem causar riscos a solução oferecida: **“os processos podem se repetir em momentos distintos”; “processos duplicados pode ser reflexos de canais diferentes”; “informações dos processos depende de etapas ou processos anteriores”**. Para o problema em questão é oferecido solução genérica **“identificar as principais etapas dos processos”,** elicitada através dos métodos “**caso de uso; user stories; QFD**”.

A.3 - Padrão Candidato III

Identificação [P]	“Contexto”.
Problema [R]	“Descrever Contextos”.

Parte do Requisito (A)	“... O processo de agendamento ocorre em contextos variados, sendo feito pela coordenação de horário, pelos próprios médicos, ou pela equipe de planejamento, e cada um determina a seqüência de atendimento dos pacientes...”
Elicitação Aplicada [A]	Caso de Uso, Prototipação e User Stories, SCRAM.
Solução [S]	Descrever as variações do contexto, suas idéias, ligações e formalidades.
Descrições	$F^{\leftarrow}$ = “Descrever as variações de contexto”. $F^{\rightarrow}$ = “Identificar suas variações e ligações”.

Vetor	(descrever as variações de contexto; identificar suas variações e ligações; descrever as variações do contexto, suas idéias, ligações e formalidades, [caso de uso; prototipação ; user stories; SCRAM])

Parte do Requisito (B)	"... O processo de captação de pedidos ocorre em dois contextos, podendo ser por pedidos diários via fone e pedidos programados via WEB..."
Elicitação Aplicada [A]	Caso de Uso, Prototipação e User Stories.
Solução [S]	Descrever as variações do contexto, suas idéias, ligações e formalidades.
Forças	$F^{\leftarrow}$ = "Descrever as variações de contexto". $F^{\rightarrow}$ = "Identificar suas variações e ligações".
Vetor	(descrever as variações de contexto, identificar suas variações e ligações, esboçar o contexto e os meios de executar os processos [caso de uso; prototipação ; user stories])

Parte do Requisito (C)	"... A solicitação de carregamento de combustível pode ser feita em contextos diferentes, que são determinados pelos tipos de contratos e recursos disponibilizados pelos fornecedores..."
Elicitação Aplicada [A]	Caso de Uso, Prototipação e User Stories.
Solução [S]	Descrever as variações do contexto, ligações e formalidades.
Forças	$F^{\leftarrow}$ = "Descrever as variações de contexto". $F^{\rightarrow}$ = "Identificar suas variações e ligações".
Vetor	(descrever as variações de contexto; identificar suas variações e ligações; descrever as variações do contexto, suas idéias, ligações e formalidades, [caso de uso; prototipação; user stories])

A.3.1 – Extraindo Padrão Candidato III

$$A = (F^{\leftarrow 1}, F^{\rightarrow 1}, S1, A1) \; ; B = (F^{\leftarrow 1}, F^{\rightarrow 1}, S1, A2) \; ; C = (F^{\leftarrow 1}, F^{\rightarrow 1}, S2, A2)$$

Portanto, P = (R, $F^{\leftarrow 1}$, $F^{\rightarrow 1}$, S1, A2)

O resultado do padrão candidato **"contexto"** diz que o requisito **"descrever contexto"** recebe as forças que direcionam a ação no fortalecimento da sua implementação: **"descrever as variações de contexto",** mas sofre influência de forças opostas e que podem causar riscos à solução oferecida: **"identificar suas**

variações e ligações”. Para o problema em questão é oferecido a solução genérica **“descrever as variações do contexto, suas idéias, ligações e formalidades”,** elicitada através dos métodos “**caso de uso; prototipação ; user stories**”.

A.4 - Padrão Candidato IV

Identificação [P]	“Módulo”.
Problema [R]	“Distinguir Módulos”.

Parte do Requisito (A)	“... O módulo financeiro estará relacionando ao módulo de produção e de atendimento aos pacientes, e este gerará um laudo para envio à Secretaria Estadual da Saúde...”
Elicitação Aplicada [A]	Caso de Uso, Prototipação, User Stories e SSM.
Solução [S]	Identificar e distinguir os módulos do sistema.
Descrições	$F^{\leftarrow}$ = “Representar os possíveis módulos do sistema”. $F^{\rightarrow}$ = “Os módulos podem ser intersecionais”.
Forças	(representar os possíveis módulos do sistema, os módulos podem ser intersecionais, identificar e distinguir os módulos do sistema [caso de uso; prototipação ; user stories; SSM])

Parte do Requisito (B)	“... O módulo financeiro está diretamente relacionado ao registro dos pedidos desde que estes estejam confirmados pelos atendentes...”
Elicitação Aplicada [A]	Caso de Uso, Prototipação e User Stories, SSM.
Solução [S]	Identificar e distinguir os módulos do sistema.
Descrições	$F^{\leftarrow}$ = “Representar os possíveis módulos do sistema”. $F^{\rightarrow}$ = “Mostrar as ligações dos módulos e o tipo de relacionamento”.
Forças	(contextos, descrever contextos, identificar suas variações e ligações, descrever as variações de contexto, descrever as variações do contexto, suas idéias, ligações e formalidades, [caso de uso; prototipação ; user stories])

Parte do Requisito (C)	“... Haverá módulos distintos para solicitação de carregamento de combustíveis, sendo que um será proprietário (cliente) e outro independente e implementado pela própria organização...”
Elicitação Aplicada [A]	Caso de Uso, Prototipação e User Stories.
Solução [S]	Identificar e distinguir os módulos do sistema.
Forças	$F^{\leftarrow}$ = “Representar os possíveis módulos do sistema e seus relacionamentos”. $F^{\rightarrow}$ = “Identificar suas variações e ligações”.

Vetor	(contextos, descrever contextos, identificar suas variações e ligações, descrever as variações de contexto, descrever as variações do contexto, suas idéias, ligações e formalidades, [caso de uso; prototipação ; user stories; SSM])

A.4.1 – Extraindo Padrão Candidato IV

$$A = (F^{\leftarrow 1}, F^{\rightarrow 1}, S1, A1);\ B = (F^{\leftarrow 1}, F^{\rightarrow 2}, S1, A1);\ C = (F^{\leftarrow 1}, F^{\rightarrow 3}, S2, A2)$$

Portanto, P = (R, $F^{\leftarrow 1}$**,** $F^{\rightarrow 1,2,3}$**,S1,A1)**

O resultado do padrão candidato **"módulo"** diz que o requisito **"distinguir módulos"** recebe as forças que direcionam a ação no fortalecimento da sua implementação: **"representar os possíveis módulos do sistema",** mas sofrem influência de forças opostas e que podem causar riscos a solução oferecida: **"os módulos podem ser intersecionais";" Mostrar as ligações dos módulos e o tipo de relacionamento"; "identificar suas variações e ligações"**. Para o problema em questão é oferecido solução genérica **"identificar e distinguir os módulos do sistema",** elicitada através dos métodos "**caso de uso; prototipação ; user stories ; SSM**".

A.5 - Padrão Candidato V

Identificação [P]	"Inverso".
Problema [R]	"Identificar Exceções".

Parte do Requisito (Projeto A)	"... Não será possível agendar um paciente sem que este não tenha passado pela Triagem médica e com a condição de sua aprovação ao tratamento, seja feito seu prontuário..."
Elicitação Aplicada [A]	Caso de Uso, Prototipação, User Stories.
Solução [S]	Identifica os processos que possuem restrições.
Forças	$F^{\leftarrow}$ = "Mapear todas as restrições dos processos". $F^{\rightarrow}$ = "Identificar se há pontos condicionais".
Vetor	(mapear todas as restrições dos processos, identificar se há pontos condicionais, identifica os processos que possuem restrições, [caso de uso, prototipação, user stories])

Parte do Requisito (Projeto B)	"... Os pedidos feitos via WEB, não poderão gerar um registro oficial sem antes passar pelo crivo do usuário que atende o cliente em questão...".
Elicitação Aplicada [A]	Caso de Uso, Prototipação, User Stories.
Solução [S]	Identifica os processos que possuem restrições, separados por módulos.
Forças	$F^{\leftarrow}$ = "Mapear todas as restrições dos processos". $F^{\rightarrow}$ = "Relacionar as restrições a cada módulo".
Vetor	(mapear todas as restrições dos processos, relacionar as restrições a cada módulo, identifica os processos que possuem restrições, separados por módulos [caso de uso, prototipação, user stories])

Parte do Requisito (Projeto C)	"... Os lançamentos não podem permitir a utilização de um caminhão que esteja com a calibração vencida ou um motorista com a habilitação também vencida..."
Elicitação Aplicada [A]	Caso de Uso, Prototipação, User Stories.
Solução [S]	Identificar os processos que possuem restrições.
Forças	$F^{\leftarrow}$ = "Mapear todas as restrições dos processos". $F^{\rightarrow}$ = "Identificar os pontos em cada processo com restrição".
Vetor	(mapear todas as restrições dos processos, Identificar os pontos em cada processo com restrição, Identificar os processos que possuem restrições [caso de uso, prototipação, user stories])

A.5.1 – Extraindo Padrão Candidato V

$$A = (F^{\leftarrow 1}, F^{\rightarrow 1}, S1, A1) \;;\; B = (F^{\leftarrow 1}, F^{\rightarrow 2}, S2, A1) \text{ e } C = (F^{\leftarrow 1}, F^{\rightarrow 3}, S1, A1)$$

$$\textbf{Portanto, } P = (R, F^{\leftarrow 1}, F^{\rightarrow 1,2,3}, S1, A1)$$

O resultado do padrão candidato **"inverso"** diz que o requisito **"identificar exceções"** recebe as forças que direcionam a ação no fortalecimento da sua implementação: **"mapear todas as restrições dos processos",** mas sofrem influência de forças opostas e que podem causar riscos a solução oferecida: **"identificar se há pontos condicionais"; "relacionar as restrições a cada módulo"; "Identificar os pontos em cada processo com restrição"**. Para o problema em questão é oferecido solução genérica **"identificar os cenários e**

representar os departamentos e suas responsabilidades”, elicitada através dos métodos “**caso de uso, prototipação, user stories**”.

APÊNDICE B – Extração dos Padrões Candidatos – Requisitos Não-Funcionais

B.1 - Padrão Candidato I

Identificação [P]	“Interface”.
Problema [R]	“Tratar as Conexões dos Dispositivos, Físicos ou Lógicos”.

Parte do Requisito (A)	“... O controle de agendamento, especificamente as funcionalidades que ficarão nos consultórios dos profissionais, deverá ter o máximo de funções via teclado, com o objetivo de acelerar o processo de inclusão de pacientes na agenda...”
Elicitação Aplicada [A]	Entrevista e Brainstorming.
Solução [S]	Estabelecer meios para redução dos esforços na operação do sistema.
Forças	$F^{\leftarrow}$ = “Identificar os métodos de entradas de dados”. $F^{\rightarrow}$ = “Mostrar aos usuários todos os atalhos relativos a entradas de dados”.
Vetor	(identificar os métodos de entradas de dados, mostrar aos usuários todos os atalhos relativos a entradas de dados, estabelecer meios para redução dos esforços na operação do sistema, [entrevista e brainstorming])

Parte do Requisito (B)	“... A organização de janelas visuais e cores distintas, entre os grupos de produtos, serão imprescindíveis para a busca feita pelos usuários no momento da captação dos pedidos...”.
Elicitação Aplicada [A]	Entrevista e Brainstorming
Solução [S]	Estabelecer meios para redução dos esforços na operação do sistema.
Forças	$F^{\leftarrow}$ = “Identificar os meios de entradas de dados feitos por atalhos”. $F^{\rightarrow}$ = “Padronizar as ações de teclas de acordo com as funcionalidades do sistema”.
Vetor	(identificar os meios de entradas de dados feitos por atalhos, padronizar as ações de teclas de acordo com as funcionalidades do sistema, estabelecer meios para redução dos esforços na operação do sistema, [entrevista e brainstorming])

Parte do Requisito (C)	"... Será acionada no visor, na tela de solicitação/permissão de abastecimento, a execução completa do processo, no instante que o motorista através do dispositivo móvel enviar o sinal de finalização da operação..."
Elicitação Aplicada [A]	Entrevista, Brainstorming e Observação
Solução [S]	Estabelecer meios para redução dos esforços na operação do sistema e facilidades visuais para melhorar desempenho do usuário.
Forças	$F^{\leftarrow}$ = "Criar controles visuais e operacionais na entrada de dados". $F^{\rightarrow}$ = "Escolher cores e teclas de funções dentro das limitações do equipamento e abrangências dos programas".
Vetor	(Criar controles visuais e operacionais na entrada de dados, Escolher cores e teclas de funções dentro das limitações do equipamento e abrangências dos programas, Estabelecer meios para redução dos esforços na operação do sistema e facilidades visuais para melhorar desempenho do usuário, [entrevista, brainstorming e observação])

B.1.1 – Extraindo Padrão Candidato I

$$A = (F^{\leftarrow 1}, F^{\rightarrow 1}, S1, A1) ; B = (F^{\leftarrow 2}, F^{\rightarrow 2}, S1, A1) ; C = (F^{\leftarrow 3}, F^{\rightarrow 3}, S2, A2)$$

$$\textbf{Portanto, } P = (R, F^{\leftarrow 1,2,3}, F^{\rightarrow 1,2,3}, S1, A1)$$

O resultado do padrão candidato **"interface"** diz que o requisito **"tratar as conexões dos dispositivos físicos ou lógicos"** recebe as forças que direcionam a ação no fortalecimento da sua implementação: **"identificar os métodos de entradas de dados"; "identificar os meios de entradas de dados feitos por atalhos"; "criar controles visuais e operacionais na entrada de dados",** mas sofre influência de forças opostas e que podem causar riscos a solução oferecida: **"padronizar as ações de teclas de acordo com as funcionalidades do sistema"; "mostrar aos usuários todos os atalhos relativos a entradas de dados"; "escolher cores e teclas de funções dentro das limitações do equipamento e abrangências dos programas"**. Para o problema em questão é oferecida a solução genérica **"estabelecer meios para redução dos esforços na operação do sistema",** elicitada através dos métodos "e**ntrevista; brainstorming**".

B.2 - Padrão Candidato II

Identificação [P]	“Segurança”.
Problema [R]	“Ilustrar Sobre a Segurança dos Dispositivos”.

Parte do Requisito (A)	“... A confirmação de operações complexas e executadas pelos médicos será feita por leitores biométricos...”
Elicitação Aplicada [A]	Caso de Uso e User Stories, Etnografia.
Solução [S]	Entender as formas de entradas de dados e sobre os equipamentos
Forças	$F^{\leftarrow}$ = “Identificar do usuário em cada processo”. $F^{\rightarrow}$ = “Os processos podem se repetir em momentos distintos”.
Vetor	(identificar do usuário em cada processo, os processos podem se repetir em momentos distintos, entender as formas de entradas de dados e sobre os equipamentos [caso de uso, user stories e Etnografia])

Parte do Requisito (B)	“... O pedido poderá ser feito por no máximo dois usuários diferentes e obrigatoriamente o sistema deverá manter estes nas bases destinadas ao controle de pedidos...”
Elicitação Aplicada [A]	Caso de Uso, User Stories, Etnografia.
Solução [S]	Entender as formas de entradas de dados e sobre os equipamentos.
Forças	$F^{\leftarrow}$ = “Identificar do usuário em cada processo” $F^{\rightarrow}$ = “Registrar a primeira identificação”.
Vetor	(identificar do usuário em cada processo, registrar a primeira identificação, entender Como os usuários se identificam no sistema e como o sistema reage [caso de uso; user stories e Etnografia])

Parte do Requisito (C)	“... Cada registro de solicitação de abastecimento deverá ser feito somente com o pedido de identificação digital do usuário, essa identificação deverá ser feita via biometria ou assinatura digital (e-card)...”
Elicitação Aplicada [A]	Caso de Uso, User Stories Etnografia.
Solução [S]	Entender as formas de entradas de dados e sobre os equipamentos
Forças	$F^{\rightarrow}$ = “Identificar do usuário em cada processo” $F^{\leftarrow}$ = “Informações dos processos depende de etapas ou processos anteriores”.

Vetor	(identificar do usuário, em cada processo, informações dos processos depende de etapas ou processos anteriores, entender as formas de entradas de dados e sobre os equipamentos [caso de uso; user stories e etnografia])

B.2.1 – Extraindo Padrão Candidato II

$A = (F^{\leftarrow 1}, F^{\rightarrow 1}, S1, A1)$; $B = (F^{\leftarrow 1}, F^{\rightarrow 2}, S1, A1)$; $C = (F^{\leftarrow 1}, F^{\rightarrow 3}, S1, A1)$

Portanto, P = (R, $F^{\leftarrow 1}$**,** $F^{\rightarrow 1,2,3}$**,S1,A1)**

O resultado do padrão candidato **"segurança"** diz que o requisito **"ilustrar sobre a segurança dos dispositivos"** recebe as forças que direcionam a ação no fortalecimento da sua implementação: **"identificar do usuário em cada processo**"**,** mas sofrem influência de forças opostas e que podem causar riscos a solução oferecida: **"os processos podem se repetir em momentos distintos"; "registrar a primeira identificação"; "informações dos processos depende de etapas ou processos anteriores"**. Para o problema em questão é oferecido a solução genérica **"entender as formas de entradas de dados e sobre os equipamentos",** elicitada através dos métodos "**caso de uso ; user stories ; etnografia**".

B.3 - Padrão Candidato III

Identificação [P]	"Componente".
Problema [R]	"Propor Uso de Componentes de Implementação".

Parte do Requisito (A)	"... O sistema fará a integração com um componente de terceiros (Prodesp) para gerar e validar o envio dos laudos..."
Elicitação Aplicada [A]	Caso de Uso, Prototipação e User Stories.
Solução [S]	Indentificar as necessidades e obrigações de uso de componentes
Forças	$F^{\leftarrow}$ = "Elencar os componentes". $F^{\rightarrow}$ = "Aprender sobre cada um deles".

Vetor	(elencar os componentes, aprender sobre cada um deles, indentificar as necessidades e obrigações de uso de componentes [caso de uso; prototipação ; user stories])

Parte do Requisito (B)	"... O módulo de pedidos fará uma integração com a plataforma WEB através de um componente SOAP para busca dos pedidos feitos online pelos clientes..."
Elicitação Aplicada [A]	Caso de Uso, Prototipação e User Stories.
Solução [S]	Identificar os pontos de uso de componentes.
Forças	$F^{\leftarrow}$ = "Definir quais componentes serão usados". $F^{\rightarrow}$ = "Adquirir componentes proprietários".
Vetor	(definir quais componentes serão usados, adquirir componentes proprietários, identificar os pontos de uso de componentes [caso de uso; prototipação ; user stories])

Parte do Requisito (C)	"... O acompanhamento das viagens será integrada com o sistema JABURSAT de rastreamento através de uma DLL proprietária fornecida pela empresa parceira (JABUR)..."
Elicitação Aplicada [A]	Caso de Uso, Prototipação e User Stories.
Solução [S]	Identificar os pontos de uso de componentes.
Forças	$F^{\leftarrow}$ = "Definir quais componentes serão usados". $F^{\rightarrow}$ = "Fazer adaptações no software para uso".
Vetor	(definir quais componentes serão usados, fazer adaptações no software para uso, identificar os pontos de uso de componentes [caso de uso; prototipação ; user stories])

B.3.1 – Extraindo Padrão Candidato III

$$A = (F^{\leftarrow 1}, F^{\rightarrow 1}, S1, A1) ; B = (F^{\leftarrow 2}, F^{\rightarrow 2}, S1, A1) ; C = (F^{\leftarrow 3}, F^{\rightarrow 3}, S2, A1)$$

O resultado do padrão candidato **"componente"** diz que o requisito **"propor uso de componentes de implementação"** recebe as forças que direcionam a ação no fortalecimento da sua implementação: **"elencar os componentes"; "definir quais componentes serão usados",** mas sofrem influência de forças opostas e que podem causar riscos a solução oferecida: **"aprender sobre cada um deles"; "adquirir componentes proprietários"; "fazer adaptações no software para uso"**. Para o problema em questão é oferecido a solução genérica **"identificar os pontos de uso de componentes",** elicitada através dos métodos "**caso de uso, prototipação e user stories**".

APÊNDICE C – Extração dos Padrões Candidatos – Requisitos Técnicos

C.1 - Padrão Candidato I

Identificação [P]	"Dicionário".
Problema [R]	"Definir Glossário de Termos".

Parte do Requisito (A)	"... Laudo APAC é um documento oficial exigido pela Secretaria Estadual da Saúde para que se possa fazer o repasse de verba de acordo com os atendimentos realizados..."
Elicitação Aplicada [A]	Entrevista, User Stories, Participatory Design
Solução [A]	Criar um glossário com os termos do contexto.
Forças	$F^{\leftarrow}$ = "entender os termos do ambiente" $F^{\rightarrow}$ = "mapear todas as siglas e termos"
Vetor	(entender os termos do ambiente, mapear todas as siglas e termos, criar um glossário com os termos do contexto [entrevista, user stories, participatory design])

Parte do Requisito (B)	"... Roteiro é o itinerário metódico que cada motorista deverá fazer como caminho para entrega das mercadorias...".
Elicitação Aplicada [A]	Entrevista, User Stories, Participatory Design
Solução [S]	Criar um glossário com os termos do contexto.
Forças	$F^{\leftarrow}$ = "entender os termos do ambiente" $F^{\rightarrow}$ = "captar os significados com quem conhece e aplica"
Vetor	(entender os termos do ambiente, captar os significados com quem conhece e aplica, criar um glossário com os termos do contexto [entrevista, user stories, participatory design])

Parte do Requisito (C)	"...Romaneio é a relação das notas constando as mercadorias que constarão no baú do veículo de entrega, com informações de quantidade, peso, valores monetários..."
Elicitação Aplicada [A]	Entrevista, User Stories, Participatory Design
Solução [S]	Criar um glossário com os termos do contexto.
Forças	$F^{\leftarrow}$ = "entender os termos do ambiente" $F^{\rightarrow}$ = "deixar a disposição para atualizações"
Vetor	(entender os termos do ambiente, deixar a disposição para atualizações, criar um glossário com os termos do contexto, [entrevista, user stories, participatory design])

C.1.1 – Extraindo Padrão Candidato I

É feita a organização das observações comuns:

$$A = (F^{\leftarrow 1}, F^{\rightarrow 1}, S1, A1) \ ; \ B = (F^{\leftarrow 1}, F^{\rightarrow 2}, S1, A1) \ ; \ C = (F^{\leftarrow 1}, F^{\rightarrow 3}, S1, A1)$$

O resultado do padrão candidato **"dicionário"** diz que o requisito **"definir glossário de termos"** recebe as forças que direcionam a ação no fortalecimento da sua implementação: **"entender os termos do ambiente",** mas sofrem influência de forças opostas e que podem causar riscos a solução oferecida: **"mapear todas as siglas e termos"; "captar os significados com quem conhece e aplica"; "deixar a disposição para atualizações"**. Para o problema em questão é oferecido a solução genérica **"criar um glossário com os termos do contexto",** elicitada através dos métodos "**entrevista; user stories; participatory design**". Resumidamente:

C.2 - Padrão Candidato II

Identificação [P]	"Implementação".
Problema [R]	"Determinar Tecnologias de Implementação".

Parte do Requisito (A)	"... As informações geradas pelos sistemas serão armazenadas num banco de dados classificado como open source, de gerenciamento relacional, de nome Firebird versão 2.1 e replicado para outra base externa do fornecedor Oracle versão 8i..."
Elicitação Aplicada [A]	SCRAM, Prototipação.
Solução [S]	Criar um documento de cenário e um protótipo de operações.
Forças	$F^{\rightarrow}$ = "Implementar para as bases especificadas". $F^{\leftarrow}$ = "O custo da tecnologia".
Vetor	(implementar para as bases especificadas, o custo da tecnologia, criar um documento de cenário e um protótipo de operações, [SCRAM e Prototipação])

Parte do Requisito (B)	"... Para implementação do sistema local, será utilizada linguagem Cliente/Servidor e para os pedidos via internet uma linguagem com suporte especifico pra WEB, embora as bases (Banco de Dados) sejam compartilhas por ambos os sistemas..."
Elicitação Aplicada [A]	SCRAM, Prototipação.

Solução [S]	Criar um documento de cenário e um protótipo de operações.
Forças	$F^{\rightarrow}$ = "Criar mecanismo de implementação por componentes" $F^{\leftarrow}$ = "Encontrar profissionais que conheçam a tecnologia aplicada".
Vetor	(criar mecanismo de implementação por componentes, encontrar profissionais que conheçam a tecnologia aplicada, criar um documento de cenário e um protótipo de operações [SCRAM e Prototipação])

Parte do Requisito (C)	"... Na implementação será aplicado linguagem Cliente/Servidor para uso local e nas filiais a implementação será feita com linguagem WEB... "
Elicitação Aplicada [A]	SCRAM, Prototipação
Solução [S]	Criar um documento de cenário e um protótipo de operações.
Forças	$F^{\rightarrow}$ = "Criar mecanismo de implementação por componentes" $F^{\leftarrow}$ = "O custo da tecnologia".
Vetor	(criar mecanismo de implementação por componentes, o custo da tecnologia, criar um documento de cenário e um protótipo de operações, [SCRAM, Prototipação])

C.2.1 – Extraindo Padrão Candidato II

$$A = (F^{\leftarrow 1}, F^{\rightarrow 1}, S1, A1) \; ; \; B = (F^{\leftarrow 2}, F^{\rightarrow 2}, S1, A1) \; ; \; C = (F^{\leftarrow 2}, F^{\rightarrow 1}, S1, A1)$$

O resultado do padrão candidato **"implementação"** diz que o requisito **"determinar tecnologias de implementação"** recebe as forças que direcionam a ação no fortalecimento da sua implementação: **"implementar para as bases especificadas"; "criar mecanismo de implementação por componentes",** mas sofrem influência de forças opostas e que podem causar riscos a solução oferecida: **"O custo da tecnologia"; "Encontrar profissionais que conheçam a tecnologia aplicada"**. Para o problema em questão é oferecido a solução genérica **"criar um documento de cenário e um protótipo de operações.",** elicitada através dos métodos "**SCRAM, Prototipação**". Resumidamente:

ANEXO A – ATA DA SESSÃO TÉCNICA

ATA DA REUNIÃO REALIZADA NO DIA 05 DE FEVEREIRO DE 2009, RELATIVO A CRÍTICAS SOBRE OS PADRÕES ESCRITOS PELO DOUTORANDO KLEBER ROCHA DE OLIVEIRA.

No dia cinco de fevereiro de dois mil e nove, às vintes horas, após o convite formal aos participantes, estiveram reunidos na sala B14 da Escola Politécnica da USP, Departamento de Engenharia de Produção: Profº Drº Mauro de Mesquita Spínola – orientador e professor do departamento, o doutorando Kleber Rocha de Oliveria, os srs Jair Rillo Junior da IBM, Rodrigo Aparecido Marques e Luís Fernando Gaido ambos do Correios. Foram tratados os seguintes assuntos:

1. O doutorando fez a abertura com agradecimentos aos presentes e explanou sobre o objetivo da sessão.
2. Embora tenha havido a distribuição do material previamente por email, o doutorando entregou aos participantes cópias do material relativo a descrição dos padrões.
3. O coordenador da sessão, profº Mauro, abriu espaço para os demais participantes fazerem suas criticas, sugestões e comentários em geral sobre os padrões.
4. Jair questionou sobre a identificação do problema em cada padrão, dizendo que não tinha ficado claro na descrição do padrão qual exatamente o problema que cada padrão estava resolvendo. Kleber mostrou no texto onde situava o problema que os padrões se propõem a resolver. Junior sugeriu que este item, pela importância, tivesse maior destaque no quadro de apresentação.
5. Os participantes discutiram sobre a amplitude da aplicação dos padrões, citando a dificuldade desta aplicação em sistema de Inteligência Artificial, Sistemas Críticos e Embarcados entre outros. Sugeriram um foco em Gestão de Sistemas Comerciais devido às características dos padrões apresentados.
6. Luis Fernando comentou sobre a dificuldade do processo de elicitação de requisitos e que via benefícios nos padrões apresentados, pois de certa forma orientam no processo de levantamento de requisitos.
7. Junior comentou que no caso do Design Patterns, o usuário conhece o problema e com isso busca a compreensão e a sua solução, enquanto que nos requisitos o problema, concretamente, não existe, mas é sabido que está no ambiente que se pretende estudar. Luis Fernando complementou dizendo que muitas vezes o problema ou a falha neste processo é percebido apenas na implementação, devido a sua característica abstrata. Ambos se colocaram a favor desta proposta, pois seria uma forma de reduzir a complexidade desta atividade.
8. Rodrigo analisou os documentos e equiparou com outro documento utilizado no Correios chamado de Modelo de Extração de Conhecimento (MEC), mas com menos elementos do que o proposto pelo doutorando.
9. Os participantes sugeriram algumas mudanças relevantes na descrição dos padrões, na qual trariam mais clareza para quem estivesse lendo e buscando a compreensão da sua utilizando, citando a dificuldade descrita no item 7.
10. O doutorando solicitou algumas sugestões sobre os requisitos inversos, e os participantes opinaram pela redução dos padrões, pois havia grandes possibilidades de os usuários dos padrões ficarem em dúvida sobre a

aplicabilidade destes. O doutorando Kleber fez as anotações dos padrões sugeridos para permanecer.

11. Próximo ao fim da sessão, foram feitas mais algumas sugestões, principalmente sobre possível aplicabilidade destes padrões em um software de apoio, uma vez que o doutorando havia mostrado interesse em produzir uma ferramenta que suportasse os padrões sugeridos nesta sessão.
12. Chegando ao limite estabelecido e sem mais colocações por parte dos participantes, o doutorando Kleber e o Profº Mauro agradeceram a presença de todos e colocou o departamento à disposição dos colegas, e os convidou para as reuniões do Grupo de Pesquisa GTI do departamento.

Terminada a reunião às 23h15min, eu Kleber Rocha de Oliveira redigi o presente documento, que depois de lido e aprovado, é assinado pelos presentes.

Profº. Drº. Mauro de Mesquita Spínola .....

Doutorando Kleber Rocha de Oliveira .....

Srº Jair Rillo Junior .....

Srº Rodrigo Marques da Silva .....

Srº Luís Fernando Gaido .....