UM MODELO FUZZY PARA AVALIAÇÃO DA QUALIDADE DE SOFTWARE

Arnaldo Dias Belchior

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Aprovada por:

_____

Prof[a]. Ana Regina Cavalcanti da Rocha, D.Sc.

_____

Prof. Geraldo Bonorino Xexéo, D.Sc.

_____

Prof. Arndt von Staa, Ph.D.

_____

Prof. Carlos Alberto Nunes Cosenza, D.Sc.

_____

Prof. José Carlos Maldonado, D.Sc.

_____

Prof. Jano Moreira de Souza, Ph.D.

Rio de Janeiro, RJ - Brasil

Maio / 1997

BELCHIOR, ARNALDO DIAS

Um Modelo Fuzzy para Avaliação da Qualidade de Software [Rio de Janeiro] 1997

xii, 185 p. 29,7 cm (COPPE/UFRJ, D.Sc., Engenharia de Sistemas e Computação, 1997)

Tese - Universidade Federal do Rio de Janeiro, COPPE

1. Engenharia de Software
2. Avaliação da Qualidade de Software
3. Teoria dos Conjuntos *Fuzzy*

I. COPPE/UFRJ II. TÍTULO (série)

SALMO 22 (de *Davi*)

O Senhor é meu pastor, nada me faltará.
Em verdes prados ele me faz repousar,
Conduz-me junto às águas refrescantes,
Restaura as forças de minha alma.
Pelos caminhos retos ele me leva,
Por amor do seu nome.
Ainda que eu atravesse o vale escuro,
Nada temerei, pois estais comigo.
Vosso bordão e vosso báculo
São o meu amparo.
Preparais para mim a mesa
A vista de meus inimigos.
Derramais o perfume sobre minha cabeça,
Transborda a minha taça.
A vossa bondade e misericórdia hão de seguir-me
Por todos os dias da minha vida,
E habitarei na casa do Senhor
Por longos dias.

AGRADECIMENTOS

Ao Banco do Nordeste do Brasil (BNB) por ter tornado possível este trabalho.

À Professora Ana Regina por sua orientação segura e sua objetividade, um especial agradecimento por minha formação a nível de pós-graduação.

Ao Professor Geraldo Xexéo por sua co-orientação, incentivo e entusiasmo contagiante de pesquisador.

Aos Professores Arndt, Cosenza, Maldonado e Jano pela honra de tê-los em minha banca examinadora.

Aos professores Guilherme, Cláudia Werner, Fátima Gaio e Vera pelo apoio.

Aos colegas e funcionários da COPPE/Sistemas pela amizade, especialmente, Morganna, Gisela, Karina, Kátia, Cristina, Rosa, Renata, Fernanda, José Xexéo, Arlindo, Claudinha, Ana Paula e Mercedes.

A Clifton Clunie por ter me cedido, gentilmente, os dados de sua pesquisa de campo, que me foi de grande utilidade neste trabalho.

Aos colegas da Agência BNB-Rio, Alberto, Márcio, Flávio, Ana Paula e Isabel.

A amizade de Ferreirinha, Porfírio, Bezerra, Julinha, Alba, Célia, Manoel Otávio, Jacob, Elvira, Maria Helena, Avany, José Antônio, Luzia, Myrian, Maria de Lourdes, Maria Ivany, Ana Maria, Daisy, Júlio, Norma, Renata, Pe. Marques, Pe. Freitas, João Fonseca, Maria Lúcia, Lúcia Barros, Lúcia Lessa, Debra, e tantos outros ...

A meu Pai, a minha Mãe, às minhas irmãs (Neyla, Dayne, Suely e Luciana), avós e tios, pelo apoio e incentivo recebidos.

Ao CNPQ pelo auxílio.

À minha mulher, **Fátima**, por seu suporte indispensável ao longo desta caminhada.

A meus filhos, **Arnaldo** e **Mairon**, dom precioso recebido de Deus, pelo carinho.

A **Deus** Uno e Trino, minha fortaleza, meu porto seguro, a que não tenho palavras cabíveis de gratidão, confiança e esperança. Obrigado, Senhor, por ter muito mais a agradecer do que a pedir!

Obrigado, **Maria**, Nossa Senhora Rainha da Paz, a quem aprendi a amar e confiar, pelas abundantes graças e bênçãos recebidas de mãe. Obrigado **Jesus**, **Maria** e **José**!

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

UM MODELO FUZZY PARA AVALIAÇÃO DA QUALIDADE DE SOFTWARE

Arnaldo Dias Belchior

Maio / 1997

Orientador: Prof[a]. Ana Regina Cavalcanti da Rocha
Co-Orientador: Prof. Geraldo Bonorino Xexéo

Programa: Engenharia de Sistemas e Computação

Este trabalho desenvolve um modelo *fuzzy* para a avaliação da qualidade de software, a partir da extensão de um modelo para avaliação da qualidade, anteriormente proposto, e utiliza, para isto, conceitos e propriedades da teoria dos conjuntos *fuzzy*. O modelo proposto possui cinco etapas para a execução de seus objetivos, podendo envolver três situações distintas. A primeira, objetiva obter um padrão de qualidade para o produto de software ou para o domínio de aplicação considerado. A segunda, avalia a qualidade de um produto de software, apoiando-se em um padrão de qualidade já definido, anteriormente, para esse produto ou seu domínio de aplicação. A terceira, estima a qualidade de um produto de software, quando não há um padrão de qualidade disponível. Uma experiência com a avaliação da qualidade de especificações de software foi realizada através deste modelo *fuzzy* com o objetivo de validá-lo e exemplificar seu uso. Os resultados experimentais obtidos corroboram com os resultados previstos axiomaticamente.

Abstract of Thesis presented to COPPE/UFRJ as partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

A FUZZY MODEL TO SOFTWARE QUALITY EVALUATION

Arnaldo Dias Belchior

May / 1997

Advisor: Prof[a]. Ana Regina Cavalcanti da Rocha, D. Sc.
Co-advisor: Prof. Geraldo Bonorino Xexéo, D. Sc.

Department: Computing and Systems Engineering

This work presents a fuzzy model to software quality evaluate, extending a software quality model previously proposed, and using to do this concepts and properties of fuzzy sets theory. The proposed model has five steps to do its objectives, embracing three different situations. The first aims to get a quality standard to a software product or application domain considered. The second evaluates the quality of a software product, supported by a quality standard previously defined to this product or its application domain. The third assesses the quality of a software product, when there is no quality standard available. An experience with evaluation of software quality specification was realized through this fuzzy model intending to validate them and to exemplify its use. The experimental results agree with those predicted axiomatically.

Índice

Índice de Figuras

Índice de Tabelas

Capítulo I

INTRODUÇÃO

No processo de avaliação da qualidade de software, não basta apenas identificar que atributos determinam essa qualidade, mas também que procedimentos adotar, para controlar seu processo de desenvolvimento, de forma a atingir o nível de qualidade desejado. Isto é realizado através da aplicação de métricas de forma organizada e bem projetada, tornando os desenvolvedores mais conscientizados da relevância do gerenciamento e dos compromissos para com a qualidade.

Uma métrica válida deve obedecer a condições de representação da teoria de medidas, tal que o entendimento intuitivo de alguns atributos seja preservado, quando é mapeado para um sistema de relação numérica (FENTON, 1994). Portanto, o uso de métricas de qualidade de software deve estar apoiado em um método para o controle da qualidade, para que se alcance, convenientemente, os objetivos almejados.

Uma proposta para avaliação da qualidade de software é o modelo proposto por Rocha (ROCHA, 1983) que, neste trabalho, é estendido através do uso de conceitos e propriedades da teoria dos conjuntos *fuzzy*, por não resolver, satisfatoriamente, o problema de agregação de medidas, utilizando-a de forma bastante *ad-hoc*.

Com o advento da teoria dos conjuntos *fuzzy*, obteve-se uma importante ferramenta para a representação de várias facetas do conhecimento humano (YAGER, 1991), capturando suas imprecisões, através de uma estrutura formal quantitativa (DUBOIS, 1991).

A mente humana opera com conceitos subjetivos tais como *alto, baixo*, *velho* e *novo*, que são incorporados em classes de objetos nessa teoria, onde a transição de pertinência ou não de um elemento a um conjunto, realiza-se de maneira gradual, e não abrupta (ZADEH, 1990).

A teoria *fuzzy* é usada, essencialmente, para mapear modelos qualitativos de tomada de decisão, e para métodos de representação imprecisa (KLIR, 1995b). Neste contexto, é que se pode utilizá-la no processo de avaliação da qualidade de software, que envolve uma enorme gama de conceitos subjetivos e não precisos.

A princípio, pode parecer incoerente a utilização da teoria *fuzzy* (nebulosa) para a avaliação da qualidade de software. Para justificar esta escolha, cita-se a argumentação de ZIMMERMANN (1991): "*A teoria dos conjuntos fuzzy fornece uma estrutura matemática estrita* (*não há nada de nebuloso na teoria dos conjuntos fuzzy*!), *na qual um fenômeno conceitual vago pode ser precisa e rigorosamente estudado*".

Uma estrutura de avaliação de software bem definida tem por objetivo estimar a sua qualidade, através de um conjunto básico de atributos, evidenciando seus aspectos relevantes. Para isto, as informações sobre o objeto de avaliação devem estar dispostas de forma organizada, onde características específicas do software possam ser identificadas para otimizar o processo de tomada de decisão (BOLOIX, 1995).

Uma vez que o processo de tomada de decisão é centrado em pessoas humanas, como também o é o processo de avaliação de software, com suas inerentes subjetividades e inconsistências na definição do problema, os conjuntos *fuzzy* são adequados nesta área, pois (IBRAHIM *et al.*, 1992):

- *possuem a habilidade para a representação de atributos*;
- *detêm formas convenientes e avaliáveis para a agregação de atributos, que podem estar vaga ou precisamente definidos*; e
- *manuseiam diferentes graus de importância para cada atributo considerado*.

Neste contexto, a utilização do *Modelo Rocha Estendido*, suportado pela robustez da teoria *fuzzy*, pode atuar como "*um mecanismo capaz de interpretar resultados e sintetizar informações*" (BOLOIX, 1995), através de "*procedimentos normatizados para a avaliação da qualidade*" (BASILI *et al*., 1995). Com a extensão desse *Modelo*, foi proposto o *Modelo Fuzzy para Avaliação da Qualidade de Software*, que envolve três situações distintas, desenvolvidas ao longo deste trabalho:

- *Determinação do padrão de qualidade para um produto de software ou para um domínio de aplicação*;

- *Avaliação da qualidade de um produto de software específico, apoiando-se em um padrão de qualidade previamente definido*;
- *Avaliação de um produto de software, mesmo que não haja um padrão de qualidade já estabelecido*.

Esta tese constitui-se de cinco capítulos, além desta introdução, e de quatro apêndices, descritos, resumidamente, a seguir:

No **Capítulo II**, *Qualidade de Software*, discorre-se sobre qualidade de processos e produtos de software, evidenciando-se a sua relevância, segundo o padrão ISO/IEC. Outras questões tratadas são o emprego de métricas, técnicas para controle da qualidade, modelos de avaliação e de gestão da qualidade.

No **Capítulo III**, *Enfoques sobre a Teoria Fuzzy*, procura-se fornecer uma visão ampla sobre os aspectos estruturais da teoria *fuzzy*, para melhor fundamentar o modelo proposto, baseado nessa teoria.

No **Capítulo IV**, *Modelo Fuzzy para Avaliação da Qualidade de Software*, são apresentadas as etapas do *modelo fuzzy para avaliação da qualidade de software*, através da extensão do *Modelo Rocha*.

No **Capítulo V**, *Uma Experiência de Utilização do Modelo Fuzzy*, efetua-se uma experiência de uso do *modelo fuzzy* proposto, com o intuito de validá-lo e de exemplificar a sua utilização.

No **Capitulo VI**, *Conclusão*, são apresentadas as conclusões deste trabalho, suas possíveis implementações e perspectivas futuras.

O **Apêndice I** contém o *Questionário de Identificação do Perfil do Especialista*. O **Apêndice II** apresenta o conjunto dos atributos de qualidade de especificações de requisitos de software. O **Apêndice III** mostra o instrumento para hierarquizar critérios de qualidade de especificações de requisitos de software. Finalmente, o **Apêndice IV** possui o formulário utilizado para a avaliação da qualidade de especificações de requisitos de software para sistemas reais.

Capítulo II

QUALIDADE DE SOFTWARE

A globalização da economia, a evolução célere da tecnologia de informação e o movimento irreversível da qualidade estão alavancando o processo de restruturação de conceitos, princípios e crenças, onde organizações bem sucedidas fundamentam suas estratégias e seus planejamentos, para assegurar a vantagem competitiva no mercado (ALMEIDA *et al.*, 1995). Esta é a chamada *era das organizações baseadas em conhecimento* ou *era do capitalismo intelectual* (BELASCO *et al*., 1994, PETERS, 1993).

A introdução do Brasil, na conjuntura das economias mais desenvolvidas, está subordinada à modernização de sua indústria, alinhada a grandes reformas estruturais. Nesta direção, surgiu o Programa Brasileiro da Qualidade e Produtividade (PBQP), com o intuito de estabelecer um conjunto ordenado de ações indutoras à modernização industrial e tecnológica brasileira, contribuindo para a retomada do desenvolvimento econômico e social. Posteriormente, foi criado o Subprograma Setorial da Qualidade e Produtividade em Software (SSQP/SW) que, entre outras funções, fornece o diagnóstico do setor em relação à qualidade e produtividade, analisa tendências mundiais, e traça ações estratégicas, que influenciam na aquisição de padrões internacionais de qualidade e produtividade (WEBER *et al.*, 1997).

Neste contexto, o engenheiro de software é solicitado a atuar como um agente para promover a qualidade de produtos e serviços. No entanto, a consolidação das práticas de engenharia de software ainda depende (LUCENA, 1991):

- *do entendimento do processo de desenvolvimento de software*;
- *do desenvolvimento de uma tecnologia de reuso robusta* (ROCHA *et al.*, 1996, WERNER *et al.*, 1996, CALDIERA, 1991);
- *da larga utilização de ferramentas de software*;

- *da estabilidade no gerenciamento de software (métricas)*;
- *da transferência conjunta de tecnologia (treinamento) e de pesquisa*.

O desenvolvimento de produtos de software tem-se mantido uma atividade complexa, devido à necessidade de serem construídos projetos, que combinem múltiplos requisitos entre si, envolvam várias equipes de trabalho e produtos não materiais, associados a seus programas e suas documentações (HAASE *et al.*, 1994, SCHWARE, 1992).

Um software perfeito, para um sistema complexo, não pode ser garantido na prática (COBB *et al.*, 1990). Um software, que atenda a suas especificações satisfatoriamente, pode conter erros, uma vez que essas especificações podem estar incorretas. Um software que tenha uma especificação isenta de erros e atenda a ela, pode ser usado indevidamente por seus usuários. A questão é, portanto, quando tornar o software operacional e como salvaguardar seus usuários de erros, que possam ser evitados. Neste contexto, o software difere de outros produtos, em vários aspectos críticos (COLLINS *et al.*, 1994):

- *erros graves podem permanecer no produto de software, mesmo depois de testes rigorosos, em virtude de sua complexidade lógica*;
- *não é trivial estabelecer padrões uniformes de software, que possam ser matéria de regulamentação e inspeção*;
- *o software afeta um grande e crescente número de indivíduos, devido a sua fácil reprodução e maleabilidade em computadores* (MOOR, 1985), *largamente propalados*;
- *uma pequena equipe e a baixo custo, pode produzir software, levando não somente a proliferação de fornecedores de software, como também a produtos de qualidade inferior*.

Atualmente, a tendência das instituições é reduzir o número de seus fornecedores de software, excluindo de seus negócios os que não estão habilitados a proverem software de boa qualidade, uma vez que a qualidade passou a ser um fator crítico para a sobrevivência e o sucesso das organizações (MILLER *et al*., 1993, SANDERS *et al*., 1994).

II.1. Conceitos de Qualidade

Não se pode pensar em qualidade como sinônimo de perfeição. Trata-se de algo factível, relativo, substancialmente dinâmico e evolutivo, amoldando-se à granularidade dos objetivos a serem atingidos. Considerá-la como algo absoluto e definitivo seria transportar-se para o inatingível e, com base neste sofisma, propiciar entraves a qualquer esforço de produzi-la. Para se ter uma noção mais abrangente sobre qualidade, descreve-se, a seguir, a visão conceitual de vários pesquisadores neste assunto.

Qualidade é uma característica intrínseca e multifacetada de um produto (BASILI, *et al*., 1991, TAUSWORTHE, 1995). A relevância de cada faceta pode variar com o contexto e ao longo do tempo, pois as pessoas podem mudar seus posicionamentos e atualizar seus referenciais, com relação a um objeto ou a uma questão. Portanto, a qualidade não é absoluta, mas depende da perspectiva do avaliador. Como tal, qualquer medida de qualidade deve ser subjetiva, sumarizando as impressões de alguma classe particular de indivíduos, que interajam com o produto (GENTLEMAN, 1996).

Qualidade é um conceito complexo, porque possui significados diversos para diferentes pessoas, em um contexto altamente dependente. Portanto, não é trivial haver medidas simples de qualidade aceitáveis para todos. Para estimar ou melhorar a qualidade de software numa organização, deve-se definir as características de qualidade, que se está interessado e, então, decidir como serão medidas (KITCHENHAM *et al*., 1996).

Qualidade é um conceito multidimensional, realizando-se por intermédio de um conjunto de atributos ou características. As empresas responsáveis pelo desenvolvimento de software devem assumir a responsabilidade de estabelecer este nível aceitável de qualidade e meios para verificar se ele foi alcançado. Qualidade de software é, portanto, um conjunto de propriedades a serem satisfeitas, em determinado grau, de modo que o software satisfaça as necessidades de seus usuários (ROCHA, 1987).

Qualidade de software é o grau para que o software processe uma combinação desejável de atributos (IEEE, 1988). Para realizar uma alta qualidade de software, essa combinação desejada de atributos deve ser claramente definida, caso contrário a qualidade passa a ser intuição (SCHNEIDEWIND, 1993, BELCHIOR, 1992b).

Quando se deseja um produto de software de alta qualidade, deve-se assegurar que cada uma de suas partes constituintes possua alta qualidade (HUMPHREY, 1995).

Portanto, os resultados intermediários, no processo de produção, devem ser imediatamente examinados após sua conclusão, procurando garantir que erros e inadequações no produto sejam detectados o mais cedo possível, pois a qualidade final do produto é uma função de todas as fases anteriores de seu ciclo de desenvolvimento.

Em IEEE (1990), a qualidade já é definida como o grau pelo qual um sistema, um componente ou um processo satisfazem seus requisitos especificados, e as necessidades ou expectativas de clientes ou usuários.

A ISO 8402 (ISO, 1994) enuncia qualidade como a *totalidade de características de uma entidade, que lhe confere a capacidade de satisfazer suas necessidades explícitas e implícitas*. A ISO/IEC 9126 (ISO, 1991) fornece o significado de *características de qualidade* de software, como sendo uma referência básica à qualidade de um produto de software, utilizada em uma avaliação.

PFLEEGER (1991) propôs que um software de alta qualidade deve possuir características, que atendam às necessidades de usuários, desenvolvedores e manutenedores. Seguindo o mesmo enfoque, PRESSMAN (1992) define qualidade de software como o conjunto das adequações aos requisitos funcionais e de desempenho, documentação inteligível dos padrões de desenvolvimento e características implícitas esperadas por todos os seus profissionais.

Para CROSBY (1990), qualidade é a conformidade do produto com os requisitos ou especificações estabelecidos. Vários pesquisadores conceituam qualidade de software apenas como uma implementação correta de sua especificação (TINNIRELLO, 1995, JOYCE, 1994, HAILSTONE, 1991, SCHACH, 1990). Esta definição pode ser usada durante o desenvolvimento de produtos, mas é inadequada para se realizar comparações entre produtos (SIMMONS, 1996).

Na visão de CAMPOS (1990), o conceito de qualidade acomoda-se no equilíbrio dos seguintes fatores:

- *qualidade intrínseca do produto ou serviço*: pode ser atestada por estar em conformidade com as normas, ou avaliada pela ausência ou presença de certos critérios;
- *custo*: corresponde ao preço, pelo qual o usuário se dispõe a pagar;

- *atendimento*: pode ser entendido como a satisfação do usuário quanto a tempo, espaço e quantidade.

Qualidade, portanto, não significa somente excelência ou um outro atributo de um certo produto final. A qualidade dever ser perseguida dentro da organização, pois, certamente, é isto o que os usuários esperam de um produto.

II.2 Em Busca da Qualidade

Há muitas maneiras de se selecionar uma estratégia de qualidade: (*i*) buscar a melhoria de processos, pois muitos erros no produto são conseqüências do processo utilizado; (*ii*) esforçar-se em melhorar a forma de encontrar e corrigir defeitos; (*iii*) identificar a causa dos erros, que produziram defeitos (GALE *et al*., 1990, MAYS *et al*., 1990).

DEMING (1986) sugere o ciclo *Planejar-Executar-Verificar-Agir* (*Plan-Do-Check-Act*), para processos orientados à qualidade, na construção de um produto. Na fase *Planejar*, são estabelecidos os alvos a serem atingidos, na medição dos atributos de qualidade do produto. Na fase *Executar*, o produto é elaborado em concordância com os padrões de desenvolvimento e guias de qualidade. Na fase *Verificar*, o produto é confrontado com seus objetivos de qualidade. Na fase *Agir*, são gerados relatórios de possíveis problemas, que se tornam base para ações corretivas.

A garantia da qualidade do software está diretamente relacionada com as características de qualidade dos produtos intermediários e de seu processo de desenvolvimento (TROSTER *et al*., 1995, BAZZANA *et al*., 1993). Os padrões ISO 9000 enfatizam que a melhoria contínua de processos resultará na melhoria de produtos e serviços (SCHMAUCH, 1994).

Um padrão é efetivo se, quando, usado apropriadamente, melhora a qualidade dos resultados e reduz os custos dos produtos de software (FENTON, 1996). Há fortes evidências de que padrões e seus guias relacionados contribuam para a melhoria da qualidade do produto (SCHNEIDEWIND, 1996).

A ISO 9000 é um padrão internacional de qualidade, que aplica o gerenciamento da qualidade do processo para gerar produtos, que atentam às expectativas de seus usuários. Esses padrões foram criados sob a premissa de que, se o desenvolvimento e o gerenciamento do sistema são de boa qualidade, então, o produto ou o serviço resultante

também será de boa qualidade. Um sistema de qualidade em conformidade com a ISO 9000 assegurará que seu processo de desenvolvimento possui um nível de controle, disciplina e repetibilidade, garantindo a qualidade de seus produtos (SCHMAUCH, 1994).

A busca de qualidade e produtividade no desenvolvimento tem sido intensa. No entanto, ainda não está bem definido como se desenvolver software de qualidade. A avaliação da qualidade e a tentativa de corrigir erros no produto, por si só, mostrou-se insuficiente e limitada para garantir a qualidade. Atualmente, tem-se evidenciado que a qualidade do produto depende, fortemente, da qualidade e adequação de seu processo de desenvolvimento. Desta forma, a qualidade do processo passou a ser vista como um pré-requisito e condicionante da qualidade do produto (ROCHA, 1997, ROCHA *et al*., 1994).

II.2.1 Qualidade de Processos de Software

O processo de software é uma seqüência de estágios para desenvolver ou manter o software, apresentando estruturas técnicas e de gerenciamento para o uso de métodos e ferramentas, e incluindo pessoas para as tarefas do sistema (HUMPHREY, 1995).

A avaliação de processos de software objetiva (OLIVEIRA *et al*., 1995c):

- *a compreensão do estado dos processos de uma organização, para a melhoria dos mesmos*;
- *estabelecer a conformidade dos processos de uma organização para com um requisito em particular ou uma classe de requisitos*;
- *determinar a adequação dos processos de uma outra organização com um contrato ou uma classe de contratos*.

A ISO/IEC 12207 define os processos do ciclo de vida do software, como sendo constituídos por um conjunto de atividades, formadas também por um conjunto de tarefas. As atividades, que devem ser realizadas durante o ciclo de vida do software, estão divididas em cinco *processos primários* (aquisição, fornecimento, desenvolvimento, operação e manutenção), em oito *processos de suporte* (documentação, gerência de configuração, garantia da qualidade, verificação, validação, revisão conjunta, auditoria e resolução do problema) e em quatro *processos organizacionais* (gerenciamento, infra-estrutura, melhoria e treinamento) (ISO, 1995).

Os padrões ISO 9000-3 enunciam procedimentos para a garantia da qualidade de software em relação a seu processo de desenvolvimento, presumindo que o produto de software é o resultado de um acordo contratual entre um cliente e um fornecedor, sendo este último uma organização com um sistema de qualidade suportado pela ISO 9000. A ISO 9000-3 está constituída de três partes (ISO, 1990):

- *Estrutura*: envolve aspectos organizacionais da produção de software.
- *Atividades do ciclo de vida*: define as ações necessárias para as fases ao longo do processo de desenvolvimento.
- *Atividades de apoio*: estabelece atividades de suporte à produção, operação e manutenção de software.

A aceitação do padrão internacional de qualidade ISO 9000 tem despertado um grande interesse das organizações, pois, através dele, podem conquistar sua certificação de qualidade. Essa certificação, significa alcançar um padrão internacional em seus processos. Da mesma forma, os clientes, que adquirem produtos e serviços, vêem, nessa certificação, um indicador que assegura a qualidade de seus fornecedores (WEBER *et al*., 1997, DAVIS *et al*., 1993, SANDERS *et al*., 1994).

Muitas organizações buscam novos paradigmas, que conduzam a uma melhoria contínua e progressiva da qualidade de seus processos, atenuando os problemas com o desenvolvimento de seus produtos de software. Assim, surgiram alguns modelos, como o Modelo de Maturidade e Capacidade do Software (*Capability Maturity Model* - CMM) (SEI, 1995).

II.2.1.1 O CMM

Desde o início desta década, o CMM vem sendo implantado em muitas organizações *urbe et orbi*, objetivando a padronização e a melhoria de processos de desenvolvimento de software. Esse objetivo segue a tendência da globalização da economia mundial, onde a padronização da produção de software surge como um elemento estratégico, para a obtenção de novos mercados e seu posterior controle (BASTOS, 1996). O princípio fundamental desse modelo é que a qualidade do produto de software é alcançada através da melhoria da qualidade de seus processos (SANDERS *et al*., 1994).

O CMM é estruturado em cinco níveis em ordem crescente de maturidade. Quando a organização se encontra em um certo nível, deve seguir atividades determinadas pelo modelo, para atingir o nível seguinte (HUMPHREY, 1991):

- *Nível 1 ou inicial*: as organizações não possuem um ambiente estável para o desenvolvimento e a manutenção de software, ficando na dependência exclusiva da habilidade e eficácia de seu pessoal técnico.
- *Nível 2 ou repetitivo*: a instituição já possui projetos, cujos processos são gerenciados, medidos, documentados, tendo sua equipe devidamente treinada.
- *Nível 3 ou definido*: há uma retroalimentação contínua do aprendizado dos processos utilizados na empresa, havendo, para isto, uma biblioteca de processos padrões, que podem ser escolhidos durante a fase de planejamento.
- *Nível 4 ou gerenciado*: são estabelecidas métricas mais estritas, para a identificação de pontos críticos e oportunidades de melhoria.
- *Nível 5 ou otimizado*: a organização se encontra em uma contínua melhoria de seus processos.

A partir do *nível 2* do CMM, são incluídos grupos de atividades chaves, KPAs (*key process areas*), que têm por objetivo segmentar e facilitar o trabalho de melhoria dos processos de software, em cada nível considerado. As KPAs são conjuntos de atividades a serem seguidas, visando a obtenção de um novo nível de maturidade, e estão subdivididas em: (*i*) alvos, (*ii*) desempenho de execução, (*iii*) habilidades para execução, (*iv*) atividades de execução, (*v*) medidas e análise, e (*vi*) verificação da implementação (SAIEDIAN *et al*., 1995).

Conforme estudos do SEI/CMU, uma empresa de software, que alcance certificação ISO 9001, atende a todos os requisitos para ser classificada no nível 2 do CMM, mesmo que tenha alcançado alguns dos requisitos de níveis superiores (WEBER, 1997).

O CMM possui quatro critérios para determinar se há compromissos e habilidade para a garantia da qualidade (JONES, 1995):

- *Existe pessoal designado para uma unidade organizacional, responsável pela garantia da qualidade*?
- *Há recursos financeiros (e outros) suficientes e disponíveis para a realização do trabalho desse pessoal*?

- *O pessoal responsável pela qualidade tem sido treinado, de tal forma que estejam conscientes do que é esperado deles e de como farão o seu trabalho*?
- *O grupo de garantia da qualidade é tido como pessoal de alto nível, que está satisfeito com o que faz*?

Entretanto, o CMM tem sido criticado por alguns pesquisadores por não ter uma base teórica formal, sendo fundamentado na experiência de um grupo de pessoas. BACH (1994), por exemplo, argumenta que o CMM é uma mistificação do processo evolutivo e não uma representação legítima do processo de software, podendo levar a organização a um colapso em seu potencial competitivo.

II.2.1.2 *Bootstrap* e *Trillium*

O *Bootstrap* (KUVAJA *et al*., 1994), derivado do modelo CMM, é um projeto que faz parte do Projeto Esprit, desenvolvido na Comunidade Européia, para avaliação de organizações.

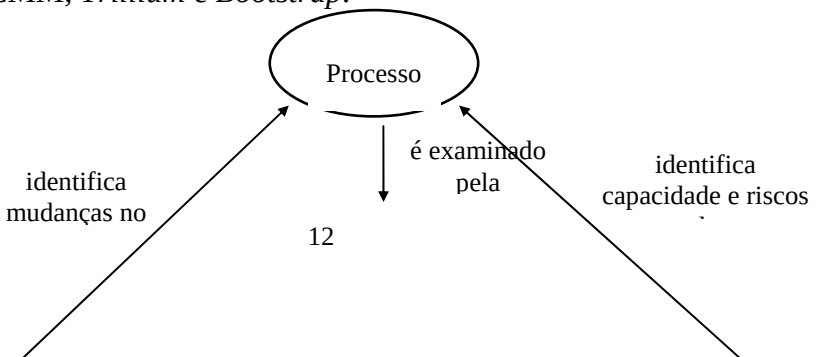
O *Bootstrap* desenvolve um método de avaliação, medição quantitativa e melhoria do processo de software. Avalia, também, as unidades produtoras de software e seus projetos. Além disso, determina o nível de maturidade da organização, identificando méritos e deficiências, fornecendo guias para aperfeiçoamento da mesma.

O *Trillium* (BELL, 1994), desenvolvido no Canadá, é derivado, também, do modelo CMM e de outros modelos para a avaliação de organizações.

O objetivo do *Trillium* é prover um método para o início e a condução de um programa de melhoramento contínuo da qualidade de processos. O projeto Trillium é orientado às telecomunicações, fornecendo uma perspectiva do produto, segundo os padrões ISO, sendo desenvolvido sob a perspectiva do cliente (TRILLIUM, 1997).

II.2.1.3 O Projeto SPICE

O Projeto SPICE (*Software Process Improvement and Capability dEtermination*), *figura II.1*, objetiva a criação de normas para avaliação de processos e a contínua melhoria desses processos, baseando-se nas melhores características de modelos de avaliação como o CMM, *Trillium* e *Bootstrap*.



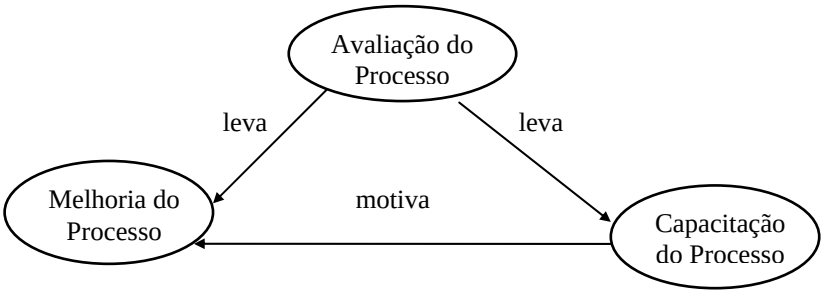


Figura II.1: Avaliação de Processos de Software: Projeto SPICE (OLIVEIRA *et al*., 1995c)

A melhoria de processos é realizada através de avaliações, que descrevam práticas usuais da organização, de uma unidade organizacional ou de um projeto. A análise dos resultados é feita em relação às necessidades do negócio da organização, levantando aspectos negativos e positivos, como também os riscos envolvidos no processo.

O Projeto SPICE pode ser usado por organizações com atividades de planejamento, gerenciamento, monitoração, controle, fornecimento, desenvolvimento, operação e suporte de software.

Esse projeto é interessante por seu direcionamento e sua flexibilidade, para que as organizações que o utilizem, determinem a capacitação de cada um de seus processos com o intuito de promover melhorias contínuas nos mesmos. Desta forma, obtém-se uma avaliação mais detalhada do estado da organização (TSUKUMO *et al*., 1996b).

A estratégia de melhoria de processos deve ser dinâmica, pois, para assegurar a qualidade de produtos de software, as habilidades se multiplicam, a tecnologia é modificada, e surgem novos ambientes de trabalho (HUMPHREY, 1995).

II.2.2 Qualidade de Produtos de Software

Para a avaliação da qualidade de produtos de software, as organizações internacionais de normalização ISO/IEC definiram as seguintes normas (SCALET, 1995):

- *Características de qualidade e métricas*: compreendendo (*i*) características e subcaracterísticas da qualidade, (*ii*) métricas externas e (*iii*) métricas internas.
- *Avaliação dos produtos de software*: envolvendo (*i*) a visão geral do produto, (*ii*) o planejamento e o gerenciamento, (*iii*) o processo para equipe de desenvolvimento, (*iv*) o processo para adquirentes, (*v*) o processo para avaliadores, e (*vi*) os módulos de avaliação.
- *Requisitos de qualidade e teste de pacote de software*.

A estrutura da ISO/IEC 9126 possui, também, um conjunto de documentos técnicos, que definem características de qualidade de software e seus indicadores, orientando o planejamento e a execução da avaliação (SCALET, 1995):

- ISO/IEC 9126-1: fornece características e subcaracterísticas de qualidade, sendo uma norma essencialmente de definições (WEBER, 1997, TSUKUMO *et al*., 1995, HAUSEN *et al*., 1993), com é representado na *tabela II.1*.
- ISO/IEC 9126-2: define métricas externas para a medição das características e subcaracterísticas de qualidade da ISO/IEC 9126-1. Essas métricas referem-se a medições indiretas de um produto de software, a partir da medição do comportamento do sistema computacional, do qual o produto faz parte.
- ISO/IEC 9126-3: estabelece métricas internas para a avaliação de um produto de software. Essas métricas referem-se a medições diretas de um produto, a partir de sua características internas, sem que seja necessária a execução do programa.

QUALIDADE DE PRODUTOS DE SOFTWARE - ISO/IEC 9126	
CARACTERÍSTICAS	**Subcaracterísticas**
FUNCIONALIDADE: as funções e propriedades específicas do produto, satisfazem as necessidades do usuário	**Adequação**: existência de um conjunto de funções apropriadas para as tarefas requeridas
	Acurácia: produção de resultados ou efeitos corretos
	Interoperabilidade: habilidade de interação do produto de software com outros produtos
	Conformidade: o produto está de acordo com as convenções, as normas ou os regulamentos estabelecidos
	Segurança: aptidão para evitar acessos não autorizados a programas e dados.
CONFIABILIDADE: o produto de software é capaz de manter seu nível de desempenho, ao longo do tempo, nas condições estabelecidas	**Maturidade**: estado de maturação do software, detectada por sua baixa freqüência de falhas
	Tolerância a falhas: o nível de desempenho é mantido, quando ocorrem falhas
	Recuperabilidade: existem mecanismos que restabelecem e restauram os dados após a ocorrência de falhas
USABILIDADE: esforço necessário para a utilização do sistema, baseado em um conjunto de implicações e de condições do usuário	**Inteligibilidade**: facilidade de entendimento dos conceitos utilizados no produto de software
	Apreensibilidade: facilidade de aprendizado do software
	Operacionalidade: faculdade de operar e controlar operações pertinentes ao software
EFICIÊNCIA: os recursos e os tempos envolvidos são compatíveis com o nível de desempenho requerido pelo software	**Comportamento no tempo**: refere-se ao tempo de resposta de processamento
	Comportamento dos recursos: relaciona-se com a quantidade dos recursos empregados

MANUTENIBILIDADE: refere-se ao esforço necessário para a realização de alterações específicas, no produto de software	**Analisabilidade**: característica de ser possível diagnosticar deficiências e causas de falhas
	Modificabilidade: característica que o produto deve ter de forma a facilitar modificações e remoções de defeitos.
	Estabilidade: ausência de riscos ou ocorrências de defeitos inesperados no software
	Testabilidade: facilidade de o produto ser testado
PORTATILIDADE: facilidade de o software pode ser transferido de um ambiente para outro	**Adaptabilidade**: faculdade de o produto poder ser adaptado a novos ambientes
	Instalabilidade: facilidade de instalação do produto de software
	Conformidade com padrões de portatilidade: o produto está segundo os padrões ou convenções de portatilidade
	Substituibilidade: o produto de software pode ser substituído por outro, sem grandes esforços

Tabela II.1: Características e subcaracterísticas de software - ISO/IEC 9126

Além das características de qualidade do produto e processo, SIMMONS (1996) aponta o resultado do desenvolvimento do software como o terceiro nível de qualidade, considerado como de grande interesse no gerenciamento dos negócios.

BAZZANA *et al*. (1993) considera a norma ISO/IEC 9126 (ISO, 1991) como o padrão para a modelagem da qualidade de software, pois, em uma pesquisa realizada em países da Comunidade Européia, 70% dos entrevistados já haviam, pelo menos, ouvido falar dela.

Apesar da grande relevância da ISO/IEC 9126, há dificuldades em adequar sua aplicabilidade na avaliação prática de produtos de software (TSUKUMO *et al*., 1996a), pois as características de qualidade, por ela determinadas, não são diretamente mensuráveis (KITCHENHAM *et al*., 1996).

Portanto, para que se obtenha a qualidade desejada de produtos de software, fazem-se necessários modelos que viabilizem a avaliação da qualidade desses produtos. Segundo STAA (1987), não é mais tolerado o uso de modelos artesanais na elaboração de programas, pois é necessário assegurar um nível elevado de qualidade de produtos de software.

II.2.2.1 Modelos para Avaliação da Qualidade de Software

O principal propósito da avaliação da qualidade de software é fornecer resultados quantitativos referentes aos produtos de software, que sejam compreensíveis, aceitáveis

e confiáveis por qualquer parte interessada (ISO, 1987). A satisfação dos usuários e o retorno econômico são, também, importantes considerações que devem conduzir efetivamente a avaliação da qualidade de um produto de software (KUMAR *et al*., 1992).

A avaliação da qualidade de software pode ser feita em várias etapas: (*i*) definição dos requisitos de qualidade, (*ii*) seleção de métricas e definição de níveis de graduação, (*iii*) medição e graduação, (*iv*) estimativa (DROMEY, 1996, BOEGH *et al*., 1992, BOEGH *et al*., 1993, MIYOSHI *et al*., 1993).

Sem um conjunto bem definido de requisitos de qualidade, os projetos de software tornam-se vulneráveis a falhas (BOEHM *et al*., 1996). Para a análise e a robustez de novos projetos de software, podem ser usados requisitos de qualidade de projetos já concluídos, com características semelhantes (BOLOIX *et al*., 1995).

A quantificação dos níveis de requisitos de qualidade implica na escolha de métricas apropriadas e na definição de modelos para medir essas métricas (MAININI *et al*., 1990).

Os modelos devem mapear a realidade, e/ou os requisitos pretendidos pelo usuário, enfocando as múltiplas questões referentes à construção do produto, e monitorando possíveis desvios. Neste contexto, os modelos de avaliação da qualidade estão diretamente associados com o processo de medição, determinando como as medidas serão executadas e planejadas (OLIVÉ *et al*., 1996).

Um bom método de desenvolvimento não garante, necessariamente, um produto de qualidade. Deve-se considerar, também, fatores de suporte à decisão, como a qualidade da equipe de desenvolvimento, e a pressão do tempo sob a qual os elementos da equipe devem trabalhar (STRIGINI, 1996).

Ganhos significativos em qualidade de software não serão alcançados até que haja um modelo compreensivo e disponível de qualidade de produto. O maior desafio na proposição de um modelo de qualidade de software é encontrar uma estrutura que possa acomodar a riqueza do conhecimento disponível sobre qualidade, de forma que seja construtivo, refinável e intelectualmente gerenciável. Neste contexto, um modelo deve fornecer (DROMEY, 1995):

- *orientação sistemática para o desenvolvimento de produtos de software de qualidade*;
- *recursos para identificar e classificar características de qualidade e defeitos em software*;
- *uma estrutura que seja inteligível, refinável e adaptável em uma camada de níveis.*

Têm surgido vários modelos para avaliação da qualidade de software, como, por exemplo, o Paradigma GQM (BASILI *et al*., 1987), o Projeto SCOPE (BOEGH *et al*., 1993), e o Modelo Rocha (ROCHA, 1983).

II.2.2.1.1 O Paradigma GQM

O paradigma GQM (*Goal/Question/Metric*) é uma estrutura para o desenvolvimento de um programa de métricas: definição, planejamento, construção, análise e *feedback*. Foi desenvolvido para várias áreas de estudo, especialmente aquelas concernentes a questões de melhoramento, e consiste nas três etapas seguintes (BASILI *et al*., 1994, BASILI *et al*., 1987):

- *Gerar um conjunto de alvos*: baseando-se nas necessidades da organização, determina-se o que se quer melhorar e, então, definem-se alvos em termos de propósitos, perspectivas e ambientes, usando-se modelos genéricos.

i. *Propósito*: manuseia (caracteriza, avalia, prediz, etc.) o objeto (processo, produto, modelo, métrica, etc.), com o propósito de entender (estimar, gerenciar, melhorar, etc.) esse objeto.

ii. *Perspectiva*: examina custo, correção, defeitos, mudanças, métricas de produto, confiabilidade, etc. sob o ponto de vista do desenvolvedor, gerente, cliente, etc.

iii. *Ambiente*: consiste em fatores (de processo, pessoas, problemas, etc.) e métodos, ferramentas, restrições, etc.

- *Derivar um conjunto de questões*: as questões quantificam os alvos como sendo possíveis, requerendo a interpretação de termos nebulosos, dentro do contexto do ambiente de desenvolvimento. As questões são classificadas como sendo relacionadas a produtos ou processos e fornecem *feedback* da perspectiva de qualidade.

- *Desenvolver um conjunto de métricas*: essas métricas fornecem as informações necessárias para responder a cada questão. As métricas podem ser objetivas e subjetivas e possuem guias de interpretação, isto é, um valor que indique se o produto é de alta qualidade. Geralmente, uma questão não é respondida simplesmente por uma métrica, mas por uma combinação de métricas. Uma vez definidos os alvos, derivadas as questões, e desenvolvidas as métricas, são criadas matrizes para relacionar *alvos*/*questões*/*métricas*.

II.2.2.1.2 O Projeto SCOPE

Na estrutura de pesquisa e desenvolvimento do Projeto Esprit, o projeto que trata das questões de certificação da qualidade de produtos de software é chamado SCOPE (*Software CertificatiOn Programme in Europe*).

Um dos mais importantes resultados desse projeto é a definição de uma estrutura de avaliação, que tem sido experimentada em muitos estudos de caso. A avaliação é realizada para vários ciclos de vida, através do uso de diversas classes de métricas como o tamanho, a estrutura de fluxo de controle, a modularidade e fluxo de informação, a estrutura de dados, a eficiência e complexidade de algoritmos, e medidas gerais de complexidade.

Os principais objetivos do Projeto SCOPE são (BAZZANA *et al*., 1993):

- *elucidar o relacionamento entre fornecedores e clientes, através de procedimentos de definição, permitindo concessões de um selo de qualidade, quando um produto possui um determinado conjunto de atributos de qualidade*;
- *desenvolver tecnologias de avaliação eficientes e efetivas, para a concessão do selo de certificação*;
- *promover a divulgação de modernas tecnologias de engenharia de software, para que sejam usadas durante o desenvolvimento de produtos de software*.

II.2.2.1.3 O Modelo Rocha

O *Modelo Rocha* (ROCHA, 1983) para qualidade de produtos de software, define qualidade a partir dos seguintes conceitos:

1. *Objetivos da qualidade*: são as propriedades gerais, que o produto deve possuir.

2. *Fatores de qualidade*: determinam a qualidade na visão dos diferentes usuários do produto. Podem ser compostos por *subfatores*, quando estes não definem completamente, por si só, um objetivo.

3. *Critérios*: são atributos primitivos, possíveis de serem avaliados.

4. *Processos de avaliação*: determinam o processo e os instrumentos a serem utilizados, de forma a se medir o grau de presença, no produto, de um determinado critério.

5. *Medidas*: são o resultado da avaliação do produto, segundo os critérios.

6. *Medidas agregadas*: são o resultado da agregação das medidas obtidas ao se avaliar de acordo com os critérios, e quantificam os fatores.

Os objetivos de qualidade são atingidos através dos fatores de qualidade, que podem ser compostos por subfatores. Objetivos, fatores e subfatores não são, diretamente, mensuráveis e só podem ser avaliados através de critérios. Um critério é um atributo primitivo. Nenhum critério isolado é uma descrição completa de um determinado fator ou subfator. Da mesma maneira, nenhum fator define completamente um objetivo.

O Modelo Rocha foi utilizado para definir qualidade em especificação de requisitos (CLUNIE, 1987), especificações de requisitos e projetos orientados a objetos (CLUNIE, 1997), especificações de projeto (PASSOS, 1991) e código (BELCHIOR, 1992a, ANDRADE, 1991).

Segundo Chung (CHUNG *et al*., 1995), os modelos para avaliação da qualidade de produtos de software devem considerar, também, as características peculiares dos domínios de aplicação (TERVONEN, 1996, BAZZANA *et al*., 1993), definindo um conjunto de requisitos da qualidade e seus processos de avaliação.

Neste contexto, o Modelo Rocha foi utilizado, também, para definir vários domínios de aplicação, juntamente com seus processos de avaliação: software científico (COMMERLATO, 1994, BAHIA, 1992, PALERMO *et al*., 1989), software financeiro (BELCHIOR, 1992b), software educacional (CAMPOS, 1994a, CAMPOS, 1994b), sistemas especialistas (OLIVEIRA, 1995b), software médico (CARVALHO, 1994), sistemas de informação (BLASCHEK, 1995), sistemas de acesso público (VALLE, 1997) e software orientado a objetos (CLUNIE, 1997, COELHO, 1997). No *capítulo IV*,

o Modelo Rocha é estendido, para suportar a utilização da teoria *fuzzy*, na avaliação de produtos de software.

A medição é a pedra angular de modelos industriais, que retratam um processo estável e divisível em um número de etapas, onde o processo é monitorado e medições são realizadas e comparadas, para a padronização de suas características (DUNN, 1990). As métricas são, por conseguinte, inseparáveis do modelo de desenvolvimento de produto de software (ROMBACH, 1990).

II.3 Métricas de Qualidade de Software

Muitos trabalhos têm usado a teoria das medições para métricas de software (FENTON *et al*., 1997, KITCHENHAM *et al*., 1996, PRATHER, 1996, ROSSI *et al*, 1996, ZAGE *et al*., 1995, FENTON, 1994, AMBRIOLA *et al*., 1994, MÖLLER, 1993, SCHNEIDEWIND, 1992, FENTON *et al*., 1991, BAKER, 1990, ZUSE, 1990, INCE, 1990, SHEPPERD, 1989, FINKELSTEIN, 1984), e aplicado métricas para controlar e estimar desenvolvimento formal (RAFFY, 1995, WHITTY, 1990).

Um projeto de software é um processo de tomada de decisão, onde métricas podem ser usadas para fornecer uma base de identificação de procedimentos, que não estejam em conformidade com os alvos pretendidos, e medidas de atributos de projeto, além de auxiliar na elaboração de novas soluções, que levem à melhoria da qualidade (YU, 1995).

Já não é mais aceitável conceber projetos de engenharia de software, que lidem com alvos firmados em ambigüidades e abstrações (GILB, 1996). "*Menos que o perfeito é bom o bastante*" (YOURDON, 1995). Somente os alvos e as prioridades podem determinar quanto '*menos que o perfeito*' pode ser aceitável.

O primeiro passo de qualquer pesquisa, baseada em métricas, é formular os alvos a serem alcançados: *Qual é o propósito da pesquisa*? *Quem usará as medidas e para que fim*? Assim, os alvos podem ser refinados em questões mais específicas, identificando-se métricas, que forneçam respostas quantitativas a estas questões (SHEPPERD, 1990).

Métricas podem ser definidas como um processo pelo qual números ou símbolos são atribuídos a requisitos de entidades do mundo real, descrevendo-as segundo regras claramente definidas (MELTON, 1996, HABRIAS, 1995, SHEPPERD, 1992, FENTON *et al*., 1991, INCE, 1990).

CARD *et al*., (1990) define métricas como uma escala de valores possíveis, que corresponde às variações observadas, em uma determinada característica. KARISSON (1995) afirma que as métricas são valores, que avaliam uma ou mais características de um produto. INCE (1990) conceitua métricas de qualidade de software como medidas numéricas, empregadas para quantificar vários aspectos de um produto de software. MARIANO (1996) considera as métricas de software como números que, de alguma forma, caracterizam o software produzido ou o processo, que é utilizado para produzi-lo. Para MOONEY (1993), as métricas definem uma base para medições quantitativas de propriedades relevantes e atividades de desenvolvimento e manutenção de software.

Embora não haja concordância universal na teoria de medição, os enfoques são voltados para as seguintes considerações (FENTON, 1994):

- *o que é e o que não é medição*;
- *que tipos de atributos podem ou não podem ser medidos e em que tipo de escala*;
- *como se sabe se um atributo foi realmente medido*;
- *como definir escalas de medidas*;
- *quando uma margem de erro é aceitável ou não*;
- *que declarações sobre medição são significativas*.

Um modelo estrutural para medição de software permite descrever os objetos envolvidos na medição e seus relacionamentos como (KITCHENHAM *et al*., 1995):

- *Entidades*: são os objetos observados no mundo real, que são capturados em suas características e manipulados formalmente. Por exemplo, produtos e processos.
- *Atributos*: são propriedades que as entidades possuem. Para um dado atributo, há um relacionamento de interesse no mundo empírico, que, formalmente, se quer apreender no mundo matemático.
- *Relacionamento entre entidades e atributos*: uma entidade pode ter vários atributos, enquanto que um atributo pode qualificar diferentes entidades.

Quando se mede um atributo, aplica-se uma unidade de medida específica a uma entidade, para se obter um valor correspondente. Quando se considera unidades de medidas, é necessário conhecer os tipos de escala mais comuns: nominal, ordinal, intervalar e proporcional (FENTON *et al*., 1991).

A estrutura das métricas de qualidade introduz categorias, que se estendem através das fases do ciclo de vida do produto, sendo independente do método de desenvolvimento (MÖLLER, 1993).

II.3.1 Categorias de métricas

As principais categorias de métricas de qualidade são: objetivas/subjetivas, absolutas/relativas, explícitas/derivadas, dinâmicas/estáticas e preditivas/explanatórias.

As *métricas objetivas* são facilmente quantificadas e medidas através de expressões numéricas ou representações gráficas dessas expressões, e calculadas de documentos de software. As *métricas subjetivas* são medidas relativas baseadas em estimativas pessoais ou de grupo, sendo obtidas por termos lingüísticos como *alto*, *médio*, *baixo* (MÖLLER, 1993).

As *métricas absolutas* são tipicamente invariantes para a medição de novos itens, enquanto que as *métricas relativas* não o são.

As *métricas explícitas* ou *diretas* não dependem da medida de outro atributo, quantificando um fator observado no produto. As *métricas derivadas* ou *indiretas* de um atributo envolvem medidas de um ou mais atributos a ele relacionados (FENTON 1994, PRESSMAN, 1992).

As *métricas dinâmicas* possuem uma dimensão temporal. As *métricas estáticas* permanecem invariáveis a despeito do tempo.

As *métricas preditivas* ou *métricas a priori* (KARISSON, 1995) podem ser obtidas ou geradas previamente, para realizar prognósticos do valor de uma propriedade do sistema, que somente se tornará diretamente observável em um estágio posterior a seu desenvolvimento (PONNAMBALAM, 1996, FENTON, 1994, SCHNEIDEWIND, 1992, FARBEY, 1990, ZUSE, 1990, KITCHENHAM *et al*., 1990c, BROCKLEHURST *et al*., 1990). Para as métricas preditivas, também, necessita-se definir procedimentos (estatísticos ou probabilísticos, por exemplo), para determinação de parâmetros de seu modelo de medição e interpretação de resultados (FENTON, 1994).

As *métricas explanatórias*, também conhecidas como *métricas de resultado* (INCE 1990), métricas *descritivas* (PONNAMBALAM, 1996) ou *métricas a posteriori* (KARISSON, 1995) são geradas depois do fato ocorrido, baseadas em dados coletados,

indicando, simplesmente, o estado atual do produto. Por exemplo, a medição do número de falhas de testes (SCHNEIDEWIND, 1995), em um período de tempo, pode indicar a confiabilidade do software.

Existem, também, as métricas de produto (PONNAMBALAM, 1996, BERTOLINO *et al*., 1996, LIGGESMEYER, 1995, BROEK *et al*., 1995, BIEMAN, 1994, BACHE *et al*., 1994, INCE, 1990, McCABE, 1976) e as métricas de processo (MÖLLER, 1993, MOONEY, 1993, INCE, 1990).

As *métricas de produto* indicam, objetivamente, características mensuráveis do produto de software tal como tamanho, complexidade, acoplamento (PONNAMBALAM, 1996).

As *métricas de processo* medem aspectos de desenvolvimento e manutenção e são, geralmente, usadas para caracterizar os custos destas atividades, como, por exemplo, quão efetivo é o processo de remoção de defeitos (MOONEY, 1993).

PONNAMBALAM (1996) considera, também, métricas de projeto, como sendo da mesma importância que as métricas de produto e de processos. As *métricas de projeto* mensuram características de projeto e sua execução como, por exemplo, o custo, o cronograma e o número de desenvolvedores de software.

MUNSON (1995) sugere uma taxonomia para atributos de software, envolvendo além das métricas de processo e de produto, também, métricas de pessoal e de ambiente. As *métricas de pessoal* consideram os recursos humanos, ao longo do processo de desenvolvimento. As *métricas de ambiente* quantificam as circunstâncias físicas, que se referem à organização.

Na concepção de MÖLLER (1993), as métricas podem ser classificadas, ainda, em métricas globais e métricas de fase. *Métricas globais* são indicadores de alto nível, que se disseminam ao longo do processo de desenvolvimento do software, fornecendo direcionamentos para os gerentes. *Métricas de fase* são indicadores somente para fases específicas do processo de desenvolvimento do produto de software.

A qualidade de software é geralmente expressa em função de diversas métricas de software, onde diferentes índices (COLEMAN *et al*., 1994) medem diferentes aspectos de qualidade (PONNAMBALAM, 1996, OMAN *et al*., 1994). Na prática, com o processo de medição, deseja-se saber se um bom software é um bom negócio

(KITCHENHAM *et al*., 1996). A seguir, apresentar-se-á as principais características das métricas de qualidade.

II.3.2 Características das Métricas

Uma métrica pode ser identificada em termos de suas características. A definição de uma métrica deve conter (MARIANO, 1996):

- *Nome*: identifica a métrica para o seu uso na organização.
- *Procedimento de obtenção*: definição da metodologia de extração da métrica (JONES, 1991), pela descrição de seu processo operacional e de seus cálculos de construção.
- *Critérios*: subsídios para a avaliação dos dados obtidos pelo uso da métrica.
- *Objetivos*: definem a métrica, favorecendo a sua conveniente interpretação.
- *Localização*: descrição de onde a métrica deve ser usada, durante o processo de desenvolvimento.

As métricas podem ser classificadas, segundo suas características gerais, em organizacionais e técnicas. As *características organizacionais* são usadas no ambiente organizacional. São elas (MÖLLER, 1993):

- *Processo de software e gerenciamento de projeto*: métricas globais são usadas durante todo o ciclo de vida do software e não somente na codificação.
- *Alta visibilidade*: uso de um número limitado de métricas, tornando-as mais visíveis interna e externamente na organização.
- *Consistência na aplicação*: aplicação de métricas de forma consistente em projetos, fornecendo indicações de sua qualidade relativa, para uma melhoria contínua.
- *Interesse do gerenciamento e suporte*: o programa de métricas deve ter o suporte da gerência da corporação, para que seja bem sucedido.
- *Aceitação organizacional*: o sucesso da aplicação de métricas depende de sua aceitação pela organização e, por isso, deve haver uma revisão extensiva das mesmas antes de serem introduzidas como procedimento, para congregar a cultura da instituição.

- *Política*: o programa de métricas deve ser bem estruturado, e imune a mudanças infundadas, procurando evitar ofensas a sentimentos pessoais ou causar impactos moralmente negativos (KITCHENHAM *et al*., 1986).
- *Disponibilidade de dados históricos*: é altamente desejável que as métricas possam utilizar dados, previamente coletados pela organização, para identificar com destreza o estado corrente de prática dessa organização, e seus alvos.
- *Conformidade com o processo de desenvolvimento do produto*: as métricas selecionadas devem estar de acordo com o processo de desenvolvimento do produto de software, apontando áreas desse processo, que possam ser melhoradas.
- *Alvos de melhoria do processo*: eleger um conjunto de métricas, que dêem suporte aos alvos de melhoria de processo, através de um plano coerente.
- *Paciência*: paciência e persistência devem ser características de desenvolvedores de um programa de métricas, pois muitos dados de métricas, altamente valorosos para a melhoria de muitos processos, somente podem ser coletados ao longo dos anos.

As *características técnicas* das métricas aplicam-se à definição da própria métrica (MÖLLER, 1993):

- *Número limitado*: é recomendado um número limitado de métricas (cinco ou menos), para facilitar o seu uso e visibilidade, pela equipe de desenvolvimento.
- *Facilidade de cálculo*: as métricas devem ser prontamente calculadas, para poderem ser identificadas ou preditas com rapidez, quando um processo de desenvolvimento de software exibe uma tendência negativa.
- *Disponibilidade de dados*: os dados usados nos cálculos das métricas devem estar acessíveis.
- *Precisão na definição*: as métricas selecionadas devem ser precisas e suas definições inteligíveis.
- *Apoio de ferramentas*: as métricas devem estar apoiadas por ferramentas acessíveis, resultando num menor esforço manual na coleta e publicação de dados.

- *Experimentação*: é de bom alvitre que os desenvolvedores experimentem cuidadosamente as métricas, que utilizam, podendo substituí-las por outras mais consistentes, quando necessário.
- *Padronização*: a identificação de padrões é, extremamente, difícil devido à larga variação encontrada nos diferentes tipo de desenvolvimento de aplicações de software, e à ausência de uniformização de seus processos.

Muitas organizações usam métricas, mas falta-lhes um enfoque sistemático para a coleta e análise de dados, além de não darem importância suficiente e necessária à interpretação do uso dessas métricas, nem à questão da padronização das mesmas (KITCHENHAM, 1990a).

Na avaliação do uso de métricas de software, é importante considerar a qualidade das próprias métricas (SHAW, 1981). Segundo WATTS (1987), as características que tornam uma métrica de software de qualidade são:

- *Objetividade*: os resultados são independentes de seu medidor;
- *Confiabilidade*: os resultados são repetíveis e precisos;
- *Validabilidade*: os resultados medem as características pretendidas;
- *Padronização*: a métrica não possui ambigüidades, seguindo um mesmo padrão;
- *Comparabilidade*: pode ser comparada com outras medidas para os mesmos critérios;
- *Economia*: a métrica é parcimoniosa e simples em sua utilização;
- *Utilidade*: a métrica deve comunicar uma necessidade e não simplesmente uma medida para seu próprio fim;
- *Consistência*: a métrica não deve combinar fatores conflitantes entre si;
- *Automação*: a métrica deve ser mensurável, através de ferramentas apropriadas.

Há relatos na literatura, que mostram aplicações de métricas em ambientes industriais, que tiveram êxito na melhoria de processos de desenvolvimento e de produtos de software em KHOSHGOFTAAR *et al*. (1996), STARK *et al*. (1994) e GRADY (1994), aumentando a credibilidade na ciência das métricas. No entanto, ainda existem muitas questões não resolvidas no uso de métricas de software em aplicações do mundo real (PONNAMBALAM, 1996).

II.3.3 A Aplicação de Métricas

A aplicação de métricas, de uma maneira organizada e projetada, apoiada por uma metodologia, possui efeito benéfico, tornando os desenvolvedores conscientes da real importância do gerenciamento e dos compromissos para com a qualidade do produto, oferecendo as seguintes vantagens (MARIANO, 1996, KARISSON, 1995, IEEE, 1987, CONTE, 1986):

- *estabelecer requisitos de qualidade para um sistema, desde o princípio de seu desenvolvimento*;
- *definir critérios de aceitação, padronização e classificação*;
- *desenvolver um plano de medidas, baseado nos requisitos estabelecidos*;
- *avaliar o nível de qualidade realizado, confrontando-o com os requisitos estabelecidos*;
- *controle do processo de desenvolvimento*;
- *melhorar o gerenciamento do produto, oferecendo meios de serem detectadas anomalias ou pontos potenciais de problemas no sistema, ao longo do desenvolvimento*;
- *predizer o nível de qualidade, que será realizado no futuro*;
- *comparar os atributos de qualidade de um sistema com outro*;
- *quantificar as mudanças, que podem ser feitas por gerentes na alocação de recursos*;
- *monitorar a degradação da qualidade, durante a fase de manutenção*;
- *calcular o custo do produto ao longo de seu ciclo de vida*.

Um dos métodos para a aplicação de métricas é usar valores de forma similar ao controle estatístico convencional de qualidade, isto é, identificar o intervalo numérico que seja '*aceitável*' ou '*não aceitável*' para itens, e '*rejeitar'* ou '*reparar'* itens com valores de métricas não aceitáveis. Este procedimento para produtos de software é mais difícil porque (KITCHENHAM *et al*., 1990b):

- Os valores de métricas '*aceitáveis'* para componentes de software diferirão de produto para produto. Itens podem ter valores de métricas em uma região '*não aceitável*' por diferentes razões, algumas das quais poderia significar que o item

é de alta qualidade. Portanto, a decisão de '*rejeitar'* um item e o método apropriado de '*reparar'* são problemas distintos.

- A indústria de software não pode se basear apenas em distribuições estatísticas convencionais (média e desvio padrão), para identificar intervalos de medidas, porque as métricas de software são, invariavelmente, dependentes do produto desenvolvido. Além disso, os itens de software, que estão sendo mensurados, usualmente não são obtidos de um experimento aleatório de itens equivalentes.
- Um problema adicional é que, normalmente, o controle da qualidade está baseado em medidas simples. Entretanto, muitos componentes de software somente são de fato avaliados, quando vários valores de métricas são agregados.

Diferentes observadores de um mesmo produto de software podem obter diferentes medidas, ainda quando a mesma propriedade é medida. Os usuários, por exemplo, estimam a qualidade do produto em termos de sua interação com o produto final, isto é, estão interessados na confiabilidade e na usabilidade. Já a qualidade, segundo a visão da produção, sugere duas características de medida: contagem de defeitos e custo de retrabalho (KITCHENHAM *et al*., 1996).

A utilização de métricas de qualidade, também, oferece algumas limitações (CONTE, 1986):

- *a medição deve ser consistente, com o mínimo de subjetividade, e apoiada em definições precisas, de modo que a análise dos dados não seja prejudicada*;
- *alguns ambientes requerem um ajuste das métricas utilizadas, para que se reduzam as possibilidades de falhas no processo de avaliação*;
- *as métricas auxiliam no processo de tomada de decisão, todavia não substituem o gerente*;
- *as métricas avaliam o desempenho do produto e não da equipe técnica*.

Não obstante as métricas tenham-se mostrado eficientes para auxiliar o desenvolvimento de software, muitos engenheiros de software ainda relutam em utilizá-las, pois temem que sejam aplicadas para avaliar o seu próprio desempenho. Isto gera um sentimento de rejeição ao uso de métricas ou promove a manipulação de números ou do próprio produto de software, para que os alvos da medição sejam alcançados (ZAGE *et al*., 1995). Neste sentido, GRADY (1992) alerta que as métricas não serão

consistentes e suficientemente definidas, quando alguém considera que o uso delas é para medir e avaliar pessoas.

Um dos propósitos do uso de métricas é, também, identificar os aspectos fortes e frágeis da organização e prover recomendações para melhorias, baseadas em laudos de estimativa, que forneçam à organização uma linha básica, confrontada por práticas aceitas como adequadas pela indústria de software (MILLER *et al*., 1993).

II.3.3.1 As Medições por Estimativas

Na indústria de software, mais do que em outras áreas, decisões importantes podem depender de julgamentos subjetivos. É difícil obter critérios objetivos para essas decisões, porque faltam modelos matemáticos do comportamento do produto ou dados estatísticos sobre experiências passadas. O argumento para a aceitação de um produto de software deve fundamentar-se, sempre que possível, em resultados de testes, na solidez das práticas usadas na engenharia de software, principalmente, no rigor da documentação, e no controle da configuração (STRIGINI, 1996).

Se os julgamentos subjetivos forem reforçados com análises científicas poderão ter sua confiabilidade incrementada. Entretanto, todo método científico, no auxílio a tomada de decisão, também, possui suas limitações, pois as informações empíricas não podem predizer o futuro com absoluto grau de certeza. Alinhando-se os julgamentos subjetivos à disciplina científica, é possível concluir que deve-se (STRIGINI, 1996):

- *usar métodos quantitativos, sempre que possível*;
- *projetar experimentos cujos resultados dependam, mais de fatos concretos do que de influências individuais*;
- *explorar como o uso da teoria pode ser refutado por experimentos*;
- *verificar a consistência das conjecturas sobre si mesmas e com fatos conhecidos*.

Psicólogos, reportando-se a julgamentos intuitivos, têm observado que (AYTON, 1993):

- *não se pode esperar que pessoas sem a devida preparação possam resolver corretamente problemas de intuição em domínios estatísticos e probabilísticos, especialmente, quando as questões são formuladas indiretamente*;

- *um especialista geralmente fará melhores predições, em sua área de atuação do que um não especialista*.

Uma das vantagens de estimativas obtidas de especialistas é que estes conhecem a força potencial e as deficiências do objeto julgado, e como estudá-las. A desvantagem é que, por conhecer muito, o especialista pode não reconhecer obstáculos que impediriam o uso do software por usuários novos ou advindos de outras áreas.

Em geral, especialistas, que realizam boas estimativas, possuem um razoável conhecimento do ambiente, além de: (*i*) fazerem prognósticos freqüentemente, (*ii*) receberem rápidos '*feedback'* sobre seus sucessos ou erros, (*iii*) serem treinados em questões específicas, e (*iv*) executarem medições.

Na realização de estimativas subjetivas, deve-se tomar alguns cuidados com relação ao questionário a ser aplicado (STRIGINI, 1996):

- *as questões devem estar expostas claramente e sem ambigüidades*;
- *as frases das questões formuladas devem ser variadas para ser possível a verificação de interpretações intuitivas errôneas*;
- *desenvolver implicações das hipótese formuladas, procurando evidências de refutação*.

O uso de padrões largamente aceitos e reconhecidos pode auxiliar de sobremaneira nas medições por estimativas. Neste sentido, o padrão IEEE Std 1061-1992 fornece uma metodologia para o estabelecimento de requisitos de qualidade e identificação, implementação, análise e validação de métricas de qualidade de software, aplicada às fases de seu ciclo de vida. Esse padrão de métricas pode ser empregado por uma organização, em suas aplicações, independentemente de suas métricas particulares, mas é essencial que essas medidas sejam validadas (SCHNEIDEWIND, 1993).

II.3.4 Validação de Métricas de Software

Pesquisadores e a indústria em geral preocupam-se com a falta de validação empírica para métricas de software, mormente para projetos e especificações, e a ausência de ferramentas de suporte (MUNSON, 1995, ANGER *et al.*, 1994, SCHNEIDEWIND, 1992, INCE, 1990).

Na opinião de GRADY (1992), métricas que não estejam mensuradas e provadas não deveriam ser aceitas como *engenharia de software*, devendo haver uma análise rigorosa que as validasse.

O propósito da validação é identificar métricas de produto e processo, que possam predizer fatores de qualidade especificados, que sejam representações quantitativas de requisitos de qualidade. As métricas devem indicar, acuradamente, se os requisitos de qualidade foram alcançados ou se, provavelmente, o serão (SCHNEIDEWIND, 1993).

Segundo a teoria das medidas, pode-se eleger dois níveis de validação: interna e externa. Uma medida de software é internamente válida, se fornece uma caracterização numérica de algum atributo intuitivamente entendido, e que não seja dependente do ambiente. Em todos os casos, deve-se saber em que aspecto, o produto (ou processo) de uma dada medida, foi definido e se há um modelo formal para tal definição (garantindo a não ambigüidade). Uma medida de software é externamente válida, se pode ser vista como sendo um importante componente ou preditor de qualquer atributo de software de interesse, isto é, de um atributo dependente de um ou mais aspectos do ambiente, mesmo que não possa ser diretamente mensurável (FENTON, 1990).

Para decidir se uma medição é válida, necessita-se assegurar a (KITCHENHAM *et al*., 1995):

- *validade do atributo*: investiga-se se o atributo de interesse é exibido pela entidade, que se está medindo, considerando-se medidas diretas ou indiretas.
- *validade da unidade*: averigua-se se a unidade de medição usada, tem um significado condizente com a medição do atributo;
- *validade do instrumento*: certifica-se a validade do modelo de medição, e se o instrumento de medição está calibrado apropriadamente;
- *validade do protocolo*: verifica-se se o protocolo de medição adotado é aceitável.

Uma métrica de software, para ser considerada válida, deve demonstrar um alto grau de correlação (KHOSHGOFTAAR *et al*., 1996) com os fatores de qualidade que representa. Uma métrica pode ser válida com relação a certos critérios de validade e inválida com relação a outros critérios, como os que seguem, para produtos ou processos (SCHNEIDEWIND, 1993):

- *correlação*: a variação dos valores de um fator de qualidade deve estar em um intervalo especificado;
- *rastreabilidade*: a alteração do valor de um fator de qualidade deve ser acompanhada por uma alteração correspondente no valor de uma métrica;
- *consistência*: se há uma ordenação de valores dos fatores de qualidade, então os valores das métricas correspondentes devem ter a mesma ordenação;
- *prognosticação*: se a métrica é usada para predizer um fator de qualidade, seu prognóstico deve ter uma acurácia específica;
- *força discriminativa*: uma métrica deve ser capaz de discriminar produtos ou processos de alta ou baixa qualidade;
- *confiabilidade*: uma métrica deve demonstrar as propriedades da correlação, rastreabilidade, consistência, prognosticação e força discriminativa, descritas acima, para uma percentagem especificada de suas aplicações.

As métricas de qualidade de software são dotadas de potencial para auxiliar na garantia da qualidade de grandes projetos. A validação dessas métricas tem o propósito de controlar e avaliar a qualidade de software, durante o projeto (SCHNEIDEWIND, 1993).

II.4 Técnicas para Controle da Qualidade

O controle da qualidade é o conjunto planejado e sistematizado de todas as ações necessárias, para produzir a confiança suficiente de que o componente ou produto estão em conformidade com os requisitos técnicos estabelecidos (ANSI, 1993).

O planejamento do controle da qualidade envolve a identificação de características de qualidade do produto, o levantamento do grau de importância de cada uma dessas características, para que as necessidades do usuário sejam satisfeitas, e a identificação do relacionamento entre essas características.

Para que o controle da qualidade tenha sucesso, são necessários alguns procedimentos (WEINGBERG, 1993, CARD *et al.*, 1990):

- *construir um banco de dados comum, que armazene os resultados das avaliações realizadas, formando assim, um histórico para possíveis comparações e referências*;

- *proporcionar um conjunto básico de ferramentas para facilitar as atividades de coleta de dados*;
- *definir um conjunto de requisitos, documentando os conceitos de qualidade para o projeto em desenvolvimento, no contexto da organização*;
- *elaborar um sistema consistente de revisões*.

As revisões são uma importante técnica de controle, para a garantia da qualidade de um produto, evitando que engenheiros de software gastem tempo e recursos, projetando e construindo um produto errado. As revisões podem identificar defeitos e promover sua reparação a baixo custo, desde as primeiras fases do processo de desenvolvimento (WEINGBERG, 1993). A seguir, são apresentadas algumas técnicas de controle de qualidade.

II.4.1 *Walkthrough* e Inspeções

O *walkthrough* é uma revisão minuciosa de um produto feita por uma equipe, com o objetivo de melhorar a qualidade desse produto, através de refinamentos sucessivos. É um processo pouco formal, envolvendo atividades de planejamento, preparação e uma reunião, onde participam de três a cinco pessoas, com duração em torno de duas horas. O planejamento envolve a marcação da reunião e a seleção dos participantes. Todo o material a ser avaliado é distribuído previamente para os avaliadores, que farão suas observações dos problemas detectados (YOURDON, 1989, PAGE-JONES, 1988, FREEDMAN *et al*., 1984).

O grupo de pessoas, que se reúne no *walkthrough*, deve ser de mesmo nível técnico, para a observação e a detecção de erros, no produto em questão. Conta-se com a presença de um moderador, usuários, e membros da equipe de desenvolvimento, além de avaliadores externos ao projeto. O narrador e o secretário são escolhidos da equipe de desenvolvedores. Para (WEINGBERG, 1993), quando ambos usuários e desenvolvedores estão presentes, o *walkthrough* pode ser mais efetivo, descobrindo-se e resolvendo-se omissões e mal entendidos.

No término da reunião, os participantes decidem pela aceitação do produto sem modificações, com modificações parciais ou rejeitam-no. Sempre que houver qualquer correção no produto, este será submetido a um novo *walkthrough*.

Assemelhando-se ao *walkthrough*, tem-se a técnica de inspeção (FAGAN, 1976), embora seja mais formal por estar embasada em critérios para avaliação de características de qualidade, definidos previamente.

As inspeções são parte de um processo de engenharia, isto é, o produto é repetidamente examinado por outros engenheiros, para descobrirem erros ou defeitos remanescentes no projeto (VOTTA *et al*. 1993). As inspeções suportam a análise de dados, durante o ciclo de desenvolvimento de um produto de software (KIRKHAM, 1996), desempenhando um importante papel na redução de falhas estruturais não esperadas (IBRAHIM, 1992).

A inspeção é um procedimento de revisão estruturado, cobrindo, por vez, uma pequena porção do produto, por uma equipe de revisores, que varia de quatro a seis participantes (MASHAYEKHI, 1993, FAGAN, 1986). Pode ser realizada em várias etapas: *planejamento*, *preparação*, *reunião* e *apresentação*.

Na etapa de apresentação, os desenvolvedores exibem todo o material a ser inspecionado, destacando o que deve ser analisado, em torno de trinta minutos. Os inspetores devem analisar o material de acordo com a lista de critérios definida previamente. O narrador sumariza cada seção do material, para assegurar que todos os critérios definidos foram considerados. No final da reunião, a lista de todos os critérios deve ter sido analisada, trazendo respostas consensuais a todas as questões, sendo decidido a aceitação ou não do produto, podendo haver outras inspeções.

YOURDON (1989) e HUMPHREY (1989) desenvolveram alguns enfoques, onde os membros da equipe, com o conhecimento técnico requerido para detalhar a inspeção, preparam-se individualmente, assistem à reunião de inspeção, e descobrem defeitos que resultam em itens de ação.

Na Técnica de Yourdon, os revisores lêem o material em questão e descrevem, informalmente, defeitos e referências. Os revisores são, também, encorajados a anotar aspectos positivos do material apreciado.

Na Técnica de Humphrey, cada revisor cria uma lista de defeitos e a entrega para o produtor do material antes do encontro, que relaciona os defeitos e os apresenta na reunião de inspeção. Há, também, uma reunião introdutória opcional, durante a qual os participantes revisam os materiais anteriores e os critérios da inspeção.

As reuniões são consideradas como um ponto central nas inspeções. VOTTA *et al*. (1993) cita algumas razões importantes para as reuniões de inspeção, dispostas em ordem decrescente de freqüência:

i. *Sinergia*: a interação entre os revisores, conduz à identificação de muitos defeitos durante a reunião, que não foram identificados pelos revisores individualmente, ao se prepararem para um reunião.

ii. *Formação*: os revisores de menor experiência aprendem com os revisores de maior experiência, ao longo do processo de inspeção.

iii. *Prazo final de planejamento*: a inspeção cria um evento planejado, para que os participantes possam trabalhar convenientemente.

iv. *Incentivo*: os participantes da revisão publicam suas contribuições e ganham a estima de seus examinadores e, portanto, esforçam-se por melhorar a si mesmos.

v. *Requisitos*: o documento do processo de inspeção especifica que a reunião deve trazer coletados os comentários dos revisores.

KNIGHT *et al*. (1993) sugeriu a *inspeção por fases*, que é uma técnica mais disciplinada e rigorosa do que a inspeção comum, podendo ser avaliados todos os produtos gerados ao longo do processo de desenvolvimento. Cada fase da avaliação objetiva verificar se o produto possui uma ou mais propriedades, detectando possíveis erros. As propriedades, já avaliadas, podem ser utilizadas em fases posteriores, servindo assim para assegurar a qualidade do produto. As avaliações, em cada fase, podem ser feitas por inspetores individuais ou múltiplos inspetores. Com inspetores individuais, a avaliação parcial é mais minuciosa, sendo controlada por uma lista de questões a serem respondidas pelo avaliador. Com múltiplos inspetores, a avaliação parcial é executada para a análise de questões subjetivas, com o objetivo de obter um consenso entre os avaliadores.

As inspeções têm um custo relevante e obrigam que os seus participantes se desloquem para o local da reunião. Esses custos poderiam ser reduzidos, em ambientes de reuniões distribuídas e colaborativas, que eliminem a necessidade de reuniões face-a-face. Assim, as reuniões poderiam ser feitas em ambientes geograficamente distribuídos, onde os participantes se reunissem através de suas estações de trabalho, e onde a versão corrente de todo o material fosse acessível via *on-line* (MASHAYEKHI *et al*., 1993).

II.4.2 Testes de Software

Os teste são, também, técnicas de controle de qualidade, que avaliam diretamente o produto, que está sendo construído, atuando, basicamente, na identificação e remoção de erros, e devem ser conduzidos de forma sistemática, para que sejam bem sucedidos (PRESSMAN, 1992).

Consoante LEVENDEL (1990), as mudanças no produto de software são, provavelmente, as atividades com maior tendência a erros, pois enquanto se remove um erro, outros podem ser introduzidos. O problema se amplia, porque o processo de correção de erros (GALE *et al*., 1990) é altamente propenso a defeitos, os testes são caros, e a demora na liberação do produto pode impactar em sua rentabilidade.

Uma das atividades principais em testes de software é o projeto e a avaliação de casos de teste, onde se utilizam técnicas, métodos e critérios, teoricamente embasados, que sistematizam essa atividade. Em geral, as técnicas podem ser classificadas em: funcional, estrutural, baseada em erros ou uma combinação delas (VILLAS *et al*., 1995).

A *técnica funcional* orienta a seleção dos casos de testes, baseada na especificação do software. Essa técnica, no entanto, não garante que todos os requisitos do programa foram satisfeitos. A *técnica estrutural* apoia-se na implementação, principalmente no fluxo de controle de dados (MALDONADO *et al*., 1995, SRI, 1994, DELAMARO, 1993, CHAIM, 1991). A *técnica baseada em erros* emprega informações de erros padrões cometidos, no processo de desenvolvimento de software, para derivar requisitos de teste.

COWARD (1988) sugere, também, a *técnica não funcional*, que é usada para identificar se o software atende às obrigações legais, possui um desempenho em conformidade com o que foi especificado e se foi codificado segundo normas e padrões estabelecidos.

Tradicionalmente, os defeitos são tidos como inevitáveis, usando-se técnicas de remoção de defeitos, como parte integrante do processo de desenvolvimento. No entanto, reconhece-se que a remoção de defeitos é uma atividade ineficiente e propensa a erros, consumindo recursos, que poderiam ser alocados na elaboração correta do código

fonte desde o princípio. Com este intuito, foi criado o *método cleanroom* (LINGER, 1994).

II.4.3 Método *Cleanroom*

O método *cleanroom* tem demonstrado que pode melhorar tanto a produtividade de desenvolvedores, que o utilizam, quanto a qualidade do software que estes produzem. A engenharia de software *cleanroom* é um processo orientado à equipe, que torna o desenvolvimento gerenciável e preditível, porque é feito sob o controle estatístico da qualidade (LINGER, 1994).

O processo *cleanroom* é baseado no desenvolvimento e na certificação de um fluxo incremental de informações de software, elaborado por pequenas equipes independentes. Nesse processo, a correção é obtida pela equipe de desenvolvimento (geralmente próxima de zero defeitos), através de especificação, projeto e verificação formais. A equipe de verificação da correção substitui os teste de unidade e a conseqüente depuração, passando o software diretamente para a fase de testes do sistema, sem que seja executado previamente pela equipe de desenvolvimento.

II.4.4 Modelos de Confiabilidade

Modelos estatísticos vêm sendo usados amplamente nas indústrias manufatureiras, como uma técnica de controle de qualidade. Os modelos estatísticos, para o cálculo da confiabilidade do hardware, baseiam-se no envelhecimento de seus componentes e, para a confiabilidade do software, apoia-se em seu perfil operacional (DYER, 1992).

Segundo MUSA *et al*., (1987), um modelo de confiabilidade de software, geralmente, tem a forma de um processo aleatório, descrevendo o comportamento de falhas no tempo. Esses modelos são usados para caracterizar e predizer possíveis comportamentos de um produto de software.

Os modelos de confiabilidade de software conhecidos podem ser classificados em (ASANOME, 1995):

- *Modelos de confiabilidade baseados em injeção de falhas*: utilizado nas fases de teste de depuração, requerendo que faltas sejam introduzidas num determinado programa ou módulo, assumindo-se que a distribuição das mesmas é igual a das faltas intrínsecas do programa ou módulo.

- *Modelos baseados no domínio do tempo*: faz algumas suposições básicas como: (*i*) o tempo entre falhas é independente, (*ii*) os testes são representativos do perfil operacional utilizado, (*iii*) novas faltas ou falhas não são introduzidas durante a correção do programa, (*iv*) faltas achadas durante os teste são corrigidas logo que encontradas.
- *Modelos baseados no domínio da entrada de dados*: calculam a probabilidade de um software falhar, obtida ao executá-lo com muitos casos de testes selecionados através de amostras do domínio de entradas do software.
- *Modelos baseados no domínio de cobertura*: concebem testes funcionais sem considerar o perfil operacional, sendo que a taxa de detecção de faltas é alcançada através das faltas remanescentes e da cobertura obtida.

Notoriamente, para se obter uma qualidade consistente de software, necessita-se controlar os processos de produtos e serviços, através de técnicas para este fim. No entanto, não se pode controlar acuradamente um processo, sem que haja informações confiáveis. Para que isto seja possível, deve-se investir na gestão da qualidade, onde se define, planeja e controla todas as ações a serem desenvolvidas, para manter a qualidade do produto (ARTHUR, 1993).

II.5 A Gestão da Qualidade de Software

Entre outras atividades, a gerência da qualidade dissemina na organização uma cultura voltada para a qualidade, evidenciando a sua relevância (SEFCIK, 1994, JACK, 1993, GARVIN, 1988). Além disso, a gerência controla todos os procedimentos, garantindo a qualidade através de auditoria e da certificação do produto.

O gerenciamento da qualidade envolve os seguintes aspectos (CROSBY, 1990, BERGAMO, 1991):

- *a gerência entende a função de qualidade*;
- *a qualidade é definida em conformidade com os requisitos*;
- *a qualidade é conseguida através de prevenção*;
- *os custos da qualidade estão implantados e são gerenciados*;
- *existe um sistema formal da qualidade*;
- *o processo é o principal enfoque*.

Muitas falhas, no gerenciamento da engenharia de software, são derivadas de deficiências na observação, pois, em qualidade, nada é tão pequeno que não mereça ser observado (WEINGBERG, 1993).

DEMING (1986) notou que indivíduos, que não conseguem visualizar em que trabalham, certamente terão dificuldades em dar uma real contribuição à qualidade. A motivação para a melhoria da qualidade sempre começa com o estudo do valor da qualidade (CROSBY, 1990), embora muitos gerentes confundam, conceitualmente, custo e valor.

DeMARCO (1987) afirma que '*o esforço move o que é contado*'. A contagem do custo leva à redução do custo; a contagem do valor conduz a um aumento do valor. Questões relacionadas ao custo são limitadas pelo orçamento da organização; o incremento do valor de um produto não tem limites. Neste contexto, tem-se utilizado várias técnicas de *benchmarking* (GILB, 1996, DARKIN, 1996), para que seja alcançado um desempenho superior em áreas críticas de trabalho, auxiliando no aprendizado de inovações e melhorando as práticas já existentes (DALE *et al*., 1992).

A qualidade deve ser perseguida dentro da organização, pois, com certeza, é isto o que os usuários esperam de um produto. Neste contexto, para que a gestão da qualidade seja eficaz e efetiva, é necessária a instalação de um programa de qualidade, dentro da organização. No entanto, a implementação desse programa de qualidade requer, sobretudo, vontade política de seus dirigentes e, para que isto aconteça, é necessário que estes estejam despertados para este propósito.

II.5.1 A Instalação de um Programa de Qualidade

Na instalação de um programa de qualidade, é necessário que se estabeleça uma anistia geral, ou seja, o que aconteceu até aquele momento, não é culpa de ninguém. A principal meta é enfrentar os problemas existentes, em benefício de todos. Contudo, os seguintes aspectos devem ser contínua e dinamicamente avaliados e melhorados: a estratégia geral da empresa, o planejamento, e os meios disponíveis com vistas ao estabelecimento de condições favoráveis, para se alcançar os objetivos almejados.

Um programa de melhoria da qualidade leva ao estabelecimento de um sistema de qualidade, que deve envolver aspectos técnicos e culturais, que são igualmente importantes. O aspecto técnico envolve o desenvolvimento de padrões e procedimentos

para implementar a qualidade em todas as atividades. O aspecto cultural está diretamente relacionado com a aceitação da qualidade por todos os indivíduos da organização. Para se iniciar um programa de qualidade pode-se seguir as seguintes etapas (SANDERS *et al*., 1994):

- *Prepararão de uma política de qualidade*: a iniciativa da elaboração de um programa de qualidade deve advir do topo da gerência, formulando uma política de qualidade, a ser publicada e comunicada de tal forma que seja entendida e implementada em todos os níveis da organização.
- *Estabelecimento de um suporte à qualidade*: criação de um *comitê de condução da qualidade* e uma *equipe de melhoria da qualidade*. O comitê direciona as estratégias, estabelece a equipe de melhoria da qualidade, autoriza e aprova o orçamento, e fornece suporte de alto nível para o programa de qualidade. A equipe de melhoria da qualidade estima as necessidades da organização, e projeta, planeja e monitora o sistema de qualidade.

Ainda segundo o mesmo autor, o planejamento de um programa de qualidade tem as seguintes etapas:

- *Estimativa da organização*: medição da prática corrente, através de um padrão apropriado, onde a organização seleciona o índice que melhor lhe satisfaça.
- *Projeto do sistema de qualidade*: os resultados da estimativa da organização são analisados, os objetivos de seu sistema de qualidade são definidos e, então, a equipe de melhoria da qualidade projeta-o, gerando a primeira versão do *manual de qualidade*.
- *Plano de implementação do programa de qualidade*: com a primeira versão do manual de qualidade, a equipe de melhoria da qualidade pode determinar o montante de trabalho necessário para a implementação do sistema de qualidade e, convenientemente, elaborar cronogramas e atividades, estabelecer marcos e alocar recursos para este fim.
- *Implementação de um programa cultural*: para um programa de qualidade ter sucesso é necessário o suporte de toda a organização e, para isto, um programa cultural deve ser cuidadosamente planejado e implementado desde cedo, para a formação de consciências voltadas para a qualidade, e encorajar a participação de todos.

- *Implementação do programa técnico*: envolve (*i*) adoção de um ciclo de vida, (*ii*) desenvolvimento de procedimentos e padrões, (*iii*) seleção e implementação de métodos e ferramentas, (*iv*) definição e implementação de um programa de medição, e (*v*) treinamento.
- *Revisão e avaliação*: componentes do sistema de qualidade, procedimentos e padrões, métodos, ferramentas e recursos devem ser revistos regularmente e devem ser tomadas ações para assegurar sua contínua efetividade.

Um plano de qualidade, como o descrito pela ISO 9000-3 (ISO, 1990), auxilia as organizações a medir e controlar a qualidade de seus produtos de software e, para isto, deve-se considerar quatro fatores críticos (SEFCIK, 1994):

- *Ambiente de desenvolvimento*: deve facilmente incorporar mudanças, onde novas ferramentas e novos procedimentos possam ser implementados para a coleta de métricas, documentação adicional e revisões.
- *Cronograma*: deve ajustar o tempo, para a melhoria de novos processos.
- *Utilidade*: uso de análises custo/benefício, no gerenciamento de processos.
- *Organização*: deve dar suporte às necessidades de melhoria de processos dos produtos de software em desenvolvimento.

As organizações, buscando a melhoria de seus processos e produtos, que têm investido em métricas, tomam decisões mais fundamentadas e em tempo hábil, e estão obtendo maior vantagem competitiva sobre outras que não o fazem (GRADY, 1992). Mas, para que este investimento seja profícuo, é necessária, também, a criação de um programa de métricas de qualidade na organização, cujos benefícios se estenderão não somente às aplicações existentes, mas também a projetos futuros.

II.5.2. Programa de Métricas

Quando se decide implementar um programa de métricas de qualidade, o passo seguinte é como fazê-lo. A maneira como um programa de métricas é desenvolvido pode determinar seu sucesso ou fracasso. Inicialmente, deve-se investigar as ferramentas disponíveis e adquirir novas ferramentas, se for o caso, para que a coleta de dados de métricas seja feita de forma automatizada, preferencialmente. A coleta de dados deve ser monitorada, para que não se incorra na tentativa de colher dados em demasia,

dificultando o seu uso, e tornando a própria coleta dispendiosa (ROSENBERG *et al*., 1996).

Os dados obtidos através de um programa de métricas são úteis para: (*i*) estabelecer objetivos, (*ii*) melhorar a qualidade, (*iii*) aumentar a produtividade, (*iv*) planejar, gerenciar e monitorar projetos, e (*v*) aumentar a confiança do usuário. Um programa de métricas deverá viabilizar estimativas, o monitoramento e a identificação de ações de melhoria, para que os objetivos de qualidade e produtividade da organização sejam alcançados. No entanto, muitas organizações não aceitam de imediato os conceitos para a implementação de um programa de métricas, porque (MÖLLER, 1993):

- *consideram que as métricas podem restringir o processo criativo*;
- *geram trabalho adicional*;
- *os benefícios a serem alcançados não são bem elucidados*;
- *existem receios de que seja um meio para medir a própria organização*;
- *há dificuldades em admitir que a melhoria é necessária*.

Há dois grandes componentes de um programa de métricas de qualidade: *medida da satisfação do usuário* e *medida de defeitos*. Por razões sociológicas, os programas de métricas são geralmente estabelecidos em seqüência, como uma tentativa de medir todos os fatores simultaneamente. A seqüência de medição abaixo tem sido utilizada com sucesso (JONES, 1991):

i. *Medidas operacionais*: envolvem principalmente as medidas de uso do computador, tempo ocioso de máquina e tempo de resposta.

ii. *Medidas do projeto atual*: acompanham o projeto em desenvolvimento.

iii. *Medidas de biblioteca*: referem-se à produção de relatórios de ocorrência.

iv. *Medidas da satisfação do usuário*: reportam-se a entrevistas com os usuários.

v. *Medidas do projeto completo*: medem o projeto final.

vi. *Medidas de fatores imprecisos*: referem-se a dados sem alto grau de precisão.

vii. *Medidas de defeitos*: mensuram os defeitos do software.

viii. *Medidas demográficas*: arrolam os recursos humanos, enquadrando-os em classes de habilidades relevantes para a organização.

A organização precisa certificar-se das vantagens geradas pela instalação e manutenção de um programa de qualidade, e de um programa de métricas de qualidade. É essencial, portanto, que ela saiba o quanto custa melhorar e manter a qualidade desejada de seus produtos e serviços. Além disso, é importante, também, que estejam bem identificadas todas as áreas, que devam ser atacadas neste processo, para que os custos sejam reduzidos.

II.5.3 Custos com a Qualidade

Um agente fomentador da política de qualidade é o reconhecimento de que os desenvolvedores têm alcançado um incremento considerável na produtividade e conseqüente redução de custos de seus produtos (ACKERMAN *et al*., 1989).

JURAN *et al*. (1988) descreve o custo da medição da qualidade como uma forma de quantificar o tamanho do problema da qualidade em uma linguagem, que terá impacto na administração superior.

Os custos com a qualidade dividem-se nas seguintes categorias, de acordo com (BERGAMO, 1991, MANDEVILLE, 1990):

- *custos de prevenção*: relacionam-se, diretamente, com as equipes de desenvolvimento, implementação e manutenção do sistema de qualidade, incluindo, também, os custos das ações preventivas, efetuadas durante o processo produtivo;
- *custos de avaliação*: associados às atividades de controle, avaliação ou auditoria de produtos ou serviços, para assegurarem se estão em conformidade com as especificações ou são adequados ao uso;
- *custos de falhas internas*: ligados aos custos com produtos ou com serviços, que não estão em conformidade com as especificações, ou não são adequados ao uso, assim como os custos de análise de falhas;
- *custos de falhas externas*: gerados por produtos, que não atendem às especificações da qualidade, após sua entrega ao usuário.

O custo da correção de defeitos de produtos de software cresce quase que em ordem geométrica, quando se percorre de forma ascendente as etapas de seu ciclo de vida (SANDERS *et al*., 1994, MÖLLER, 1993, PRESSMAN, 1992, BOEHM, 1987). Na

tentativa de alcançar produtos de alto valor qualitativo e a um custo aceitável, entre outras alternativas, surgiu o gerenciamento da qualidade total (*Total Quality Management* - TQM) (DEMING, 1989).

II.5.4 Gerenciamento da Qualidade Total

O TQM pode ser compreendido como um conjunto de princípios e métodos, que visam a mobilização e a cooperação de todos os participantes de uma organização (FIORINI, 1995), para melhorar a qualidade de seus produtos e serviços, suas atividade e seus objetivos, para obter a satisfação dos usuários e um incremento do bem estar de seus membros (XAVIER, 1992), a um custo compatível. Isto exige um gerenciamento de liderança e um esforço de todos os membros da instituição, apoiado por uma cultura direcionada para a qualidade (SOARES *et al*., 1994).

O TQM possui três componentes chaves: o planejamento, a solução do problema e o gerenciamento do processo. A qualidade começa com a definição dos requisitos dos usuários e tem seu ápice na satisfação desses usuários (ARTHUR, 1993).

O *planejamento da qualidade* deve ser conduzido pela alta gerência da organização:

- *na condução de esforços de melhorias da qualidade*;
- *no estabelecimento de uma política de qualidade e produtividade de software*;
- *na criação de estruturas de apoio (reconhecimento, recompensa, treinamento, entre outras), requeridas para alcançar os objetivos de qualidade*.

A *solução do problema* deverá ser obtida através de uma equipe, que busque a satisfação do usuário, eliminando raízes das causas dos problemas como, por exemplo, a falta de correção de requisitos ou a dificuldade de interpretar especificações.

O *gerenciamento do processo* envolve muitas atividades repetitivas ao longo do desenvolvimento do produto de software, possuindo requisitos próprios, que serão medidos convenientemente, para assegurar a qualidade.

MILLET *et al*. (1993) relata alguns princípios da qualidade total aplicados à área de informática: (*i*) a satisfação total do cliente, (*ii*) a gerência participativa, (*iii*) a delegação de poderes, (*iv*) o desenvolvimento de recursos humanos, (*v*) a constância de propósitos, (*vi*) a gerência de processos, e (*vii*) o aperfeiçoamento contínuo.

Muitos gerentes, apesar de seus méritos pessoais e técnicos, não utilizam todo o potencial, que a informática lhes pode oferecer, nem dispõem das habilidades administrativas necessárias, para gerirem adequadamente os recursos humanos de que dispõem. Além destes agravantes, não possuem uma visão gerencial macroscópica dos processos organizacionais da empresa, seja por falta de iniciativa própria, pelas dificuldades internas de obter tais informações, ou até pela ausência desses dados (BELCHIOR, 1995a).

II.5.5 Recursos Humanos

Administrar recursos humanos talvez se tenha tornado o desafio mais importante e mais difícil de vencer dos próximos dez ou quinze anos, pelo menos entre os países em desenvolvimento (DRUCKER, 1990).

Empresários e pesquisadores das áreas de gestão da qualidade, produtividade e do comportamento organizacional afirmam que a filosofia de administração das empresas, que almejam o sucesso, deve estar direcionada para o princípio da valorização do ser humano, como razão maior de sua existência, e como agente motivador de seu contínuo crescimento (PFEFFER, 1994, ISHIKAWA, 1993).

Portanto, tem-se buscado a *qualidade de vida no trabalho* como uma estratégia na administração de recursos humanos, promovendo a harmonização do indivíduo com o ambiente de trabalho e sua realização pessoal e profissional (ALMEIDA *et al.*, 1995).

COELHO (1997) propõe que os aspectos sociais, que permeiam a cultura da empresa, sejam analisados periodicamente (LÉVY, 1993), a fim de assegurar a seus funcionários a qualidade de vida no trabalho e o sucesso na implantação do software. Isto porque o padrão comportamental do empregado influencia, significativamente, na produtividade da organização e na qualidade dos produtos ou serviços gerados.

Portanto, a tendência dos programas de melhoria da qualidade e produtividade em empresas é não investir somente em processos e tecnologia de software, mas dar relevo ao treinamento melhorado e sistemático de seus recursos humanos (HEFLEY *et al.*, 1995, HAMMER, 1990, DAVENPORT *et al.*, 1990). Na área de software, o treinamento é uma atividade primordial devido à rápida evolução de seus produtos e a seus ciclos de tecnologia, relativamente curtos (WEBER *et al.*, 1997).

II.6 Conclusão

No desenvolvimento de produtos de software, muitas decisões importantes ainda dependem de julgamentos subjetivos. Faltam modelos matemáticos do comportamento do produto e não se tem dados disponíveis sobre experiências passadas.

Se os julgamentos subjetivos fossem reforçados com análises científicas poder-se-ia aumentar a confiabilidade do software (STRIGINI, 1996) e, conseqüentemente, garantir a qualidade do produto.

No capítulo seguinte, apresentar-se-á alguns enfoques sobre a teoria *fuzzy*, na tentativa de fornecer um suporte matemático à subjetividade das estimativas, realizadas na avaliação da qualidade de um produto de software.

Capítulo III

ENFOQUES SOBRE A TEORIA *FUZZY*

A teoria dos conjuntos *fuzzy* (nebulosos) é usada para representar modelos de raciocínio impreciso, que possuem um papel essencial na notável habilidade humana de tomar decisões racionais, em ambientes de incertezas e imprecisões (ZADEH, 1988).

A principal motivação da teoria dos conjuntos *fuzzy* é o desejo de construir uma estrutura formal quantitativa, capaz de capturar as imprecisões do conhecimento humano, isto é, como esse conhecimento é formulado na linguagem natural. Essa teoria visa ser a ponte, que une modelos matemáticos tradicionais, precisos, de sistemas físicos, e a representação mental, geralmente imprecisa, desses sistemas (DUBOIS, 1991).

A mente humana opera com conceitos subjetivos tais como *alto*, *baixo*, *velho* e *novo*, que são incorporados em classes de objetos na teoria *fuzzy*, onde a pertinência ou não de um elemento a um conjunto dá-se de forma gradual e não abrupta (ZADEH, 1990). Com o advento dessa teoria, obteve-se substanciais instrumentos, para a representação de várias facetas cognitivas humanas (YAGER, 1991). Ela provê ferramentas robustas para a aplicação do conhecimento, da experiência e do pensamento humano em muitos sistemas industriais, de tráfego, ciência médica, entre outros (SUZUKI, 1993).

GILES (1988) levanta dois enfoques, igualmente importantes, para a formulação da teoria dos conjuntos *fuzzy*: o *axiomático* e o *semântico*. No axiomático, uma função numérica é usada para modelar o conjunto *fuzzy*, fornecendo uma interpretação consistente para o mesmo, embora não evidencie, diretamente, a estrutura sob esse conjunto (FRENCH, 1989, FRENCH, 1986). Para a adoção de certas definições de operações de conjuntos *fuzzy*, é condição necessária e suficiente que essas definições

sejam avaliadas por vários pesquisadores (BELLMANN *et al*., 1973) e que se utilizem do enfoque axiomático.

No enfoque semântico, é analisado, empiricamente, o significado físico dos conceitos do sistema envolvido, modelando-os através de conjuntos *fuzzy*. Esses conceitos são precisamente formulados como axiomas quantitativos, produzindo não somente uma teoria consistente, mas, também, uma interpretação específica dos mesmos. Esse enfoque pode, ainda, ser dividido em métodos *normativos* e *descritivos* (FRENCH, 1986). O método normativo conjectura como as pessoas poderiam organizar seus julgamentos em uma situação particular, através de conjuntos *fuzzy*. O descritivo (mais usado) investiga como de fato as pessoas realizam seus julgamentos (SDORRA *et al*., 1993).

Ambos os enfoques, *axiomático* e *semântico*, têm sido reportados na literatura atual em medidas de funções *fuzzy* ou simplesmente *funções de pertinência*. Enfoques *híbridos* têm sido mencionados, onde métodos experimentais foram usados em conjunção com os formais, testando e validando suas hipóteses e conseqüências (TURKSEN, 1991).

A seguir, serão abordados alguns conceitos da teoria dos conjuntos *fuzzy*, que darão suporte ao propósito deste trabalho, isto é, a aplicação dessa teoria na avaliação da qualidade de software.

III.1 Conceitos Básicos de Conjuntos *Fuzzy*

Um *conjunto nítido* (ou ordinário) é definido como uma coleção de elementos $x \in X$, onde X é o conjunto universo. Cada elemento pode pertencer ou não a um conjunto A, onde $A \subseteq X$. Em um conjunto *fuzzy*, representado por $\tilde{A}$ em X, cada elemento x pode ter um grau de pertinência, usualmente no intervalo real [0, 1], em decorrência de sua *função de pertinência* característica.

Portanto, cada função de pertinência mapeia elementos de um dado conjunto universo X, que é sempre um conjunto nítido, para um número real em [0, 1]. Em casos limites, se o grau de pertinência é 0, o elemento não pertence ao conjunto e, se é 1, o elemento pertence 100% ao conjunto (TURKSEN, 1991, ZIMMERMANN, 1991).

Qualquer representação adequada de um conjunto *fuzzy* envolve o entendimento básico de cinco símbolos conceituais diferentes, relacionados entre si (TURKSEN, 1991):

- *Conjunto de elementos* $\theta \in \Theta$: por exemplo, "*item*" em "*estoque*".
- *Variável lingüística V*: rótulo para um atributo dos elementos $\theta \in \Theta$, como o "*nível de estoque*" de uma empresa.
- *Termo lingüístico T*: referente a uma *variável lingüística*, correspondendo a um adjetivo ou a um advérbio, como "*estoque baixo*", relacionado com possíveis "*níveis de estoque*" de uma empresa.
- *Conjunto referencial* $X \in [-\infty, \infty]$: um atributo particular de *V*, num conjunto de elementos $\theta \in \Theta$, como, por exemplo, "[250, 750] *unidades*" para "*nível de estoque*".
- *Grau de pertinência* $\mu_{\tilde{A}}(\theta)$: valor de pertinência de um elemento θ em relação ao conjunto de elementos, rotulado por uma *variável lingüística V*, e identificado pelo *termo lingüístico T*. Por exemplo, seja o valor de pertinência dado por um gerente a "*estoque*", através do adjetivo "*baixo*", envolvendo os níveis de estoque sob seu gerenciamento.

Os conceitos básicos de conjuntos *fuzzy*, apresentados a seguir, estão baseados em (KLIR *et al*., 1995a, ZIMMERMANN, 1991, FUHRMANN, 1990, KLIR *et al*., 1988, DUBOIS, 1980, ZADEH, 1965).

Função de pertinência

A *função de pertinência* é o componente crucial de um conjunto *fuzzy*, e muitas operações são definidas em conformidade com ela (ZADEH, 1965). Muitas vezes, a estimação da função de pertinência $\mu_{\tilde{A}}(x)$, em situações do mundo real, vem de informações imprecisas e incompletas. Dado um conjunto *fuzzy* $\tilde{A}$, duas notações são comumente empregadas, para caracterizar essas funções:

$$\mu_{\tilde{A}}(x) : X \rightarrow [0, 1]$$

ou

$$\tilde{A} : X \rightarrow [0, 1]$$

De acordo com a primeira notação, o identificador do conjunto *fuzzy* $\tilde{A}$ é distinguido do símbolo de sua função de pertinência $\mu_{\tilde{A}}(x)$. Na segunda, esta distinção não é feita. No entanto, não há ambigüidades nos resultados desta dupla utilização, pois cada conjunto *fuzzy* é completa e unicamente definido por uma função de pertinência particular e, conseqüentemente, identificadores das funções de pertinência podem ser também usados como símbolos de seus conjuntos *fuzzy* associados. Neste trabalho, utilizar-se-á a primeira notação.

Exemplo III.1:

Seja *X* o conjunto universo de idades, e os subconjuntos *fuzzy*, contidos em *X*, classificados como *infantil*, *adulto*, *jovem*, e *velho*, envolvendo todas as possibilidades de subconjuntos *fuzzy* de *X*, que é denotado por $\tilde{N}(X)$, como mostra a *Tabela III.1*.

$X = \{5, 10, 20, 30, 40, 50, 60, 70, 80\}$

IDADES	Infantil	Adulto	Jovem	Velho
5	0	0	1	0
10	0	0	1	0
20	0	0,8	0,8	0,1
30	0	1	0,5	0,2
40	0	1	0,2	0,4
50	0	1	0,1	0,6
60	0	1	0	0,8
70	0	1	0	1
80	0	1	0	1

Tabela III.1: Exemplo de conjuntos *fuzzy* (KLIR *et al.*, 1988)

Representação de um conjunto *fuzzy*

Os conjuntos *fuzzy* prestam-se às representações de conceitos vagos, expressados na linguagem natural, dependendo do contexto em que são usados. Usar-se-á, a seguir, a formalização de conjuntos *fuzzy* com suporte finito (uma das várias existentes na literatura).

Um conjunto *fuzzy* é denotado por um conjunto de pares ordenados, em que o primeiro elemento é $x \in X$, e o segundo, $\mu_{\tilde{A}}(x)$, é o grau de pertinência ou a função de pertinência de *x* em $\tilde{A}$, que mapeia *X* para o espaço de pertinência *M*. Quando *M* contem somente os pontos 0 e 1, $\tilde{A}$ é *não-fuzzy* (ZIMMERMANN, 1991):

$$\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) \mid x \in X\}$$

Exemplo III.2:

O conjunto *fuzzy* $\tilde{A}$, *jovens*, da *Tabela III.1*, pode ser descrito como:

$$\tilde{A} = \{(5; 1), (10; 1), (20; 0{,}8), (30; 0{,}5), (40; 0{,}2), (50; 0{,}1)\}$$

Suporte

O suporte de um conjunto *fuzzy* $\tilde{A}$, em um conjunto universo X, é o conjunto nítido, que contém todos os elementos de X com graus de pertinência diferentes de zero em $\tilde{A}$.

O suporte de um conjunto *fuzzy* $\tilde{A}$ em X, denotado por supp($\tilde{A}$) ou $S(\tilde{A})$, onde $\tilde{\mathsf{P}}(X)$ contem todos os subconjuntos *fuzzy* possíveis, é obtido pela função:

$$S: \tilde{\mathsf{P}}(X) \rightarrow \mathsf{P}(X)$$

onde,

$$S(\tilde{A}) = \{x \in X \mid \mu_{\tilde{A}}(x) > 0\}$$

Exemplo III.3:

O suporte do conjunto *fuzzy* $\tilde{A}$, jovem, da *Tabela III.1* é o conjunto nítido

$$S(\tilde{A}) = \{5, 10, 20, 30, 40, 50\}$$

O conjunto *fuzzy infantil* é um conjunto vazio, no conjunto universo escolhido. Neste caso, o suporte também é vazio, isto é, sua função de pertinência assinala 0 a todo elemento do conjunto universo.

Supremo

O supremo, $\sup_{x \in X} \mu_{\tilde{A}}(x)$, de um conjunto *fuzzy* $\tilde{A}$ é o maior grau de pertinência obtido nesse conjunto por um desses elementos, isto é, sua altura, $h(\tilde{A})$. O contradomínio de uma função de pertinência é um subconjunto de números reais não negativos, cujo supremo é finito. Então,

$$h(\tilde{A}) = \sup_{x \in X} \mu_{\tilde{A}}(x)$$

Normalização

Embora uma função de pertinência não esteja limitada ao contradomínio [0, 1], por conveniência, isto é usualmente considerado como verdade, ou seja, assume-se que um

conjunto *fuzzy* $\tilde{A}$ é *normal* ou *normalizado*. Portanto, um conjunto *fuzzy* $\tilde{A}$ é chamado normal, quando $\sup_{x \in X} \mu_{\tilde{A}}(x) = 1$. Se $\sup_{x \in X} \mu_{\tilde{A}}(x) < 1$, ele passa a se denominar *subnormal*.

A normalização de um conjunto *fuzzy* $\tilde{A}$, não vazio, é efetuada por:

$$\mu'_{\tilde{A}}(x) = \mu_{\tilde{A}}(x) / \sup_{x \in X} \mu_{\tilde{A}}(x)$$

Conjuntos de corte-α

Dado um conjunto *fuzzy* $\tilde{A}$, definido em X, a partir do grau de pertinência $\alpha \in [0, 1]$, o *conjunto de corte-α* (*α-cut*) é o conjunto nítido A_α, contendo todos os elementos de X, que possuem graus de pertinência em $\tilde{A}$ maiores ou iguais do que o valor especificado em α. Então,

$$A_\alpha = \{x \in X \mid \mu_{\tilde{A}}(x) \geq \alpha\}$$

O *conjunto de corte-α robusto* (*strong α-cut*), A'_α, inclui apenas os elementos de graus de pertinência maiores que α. Então,

$$A'_\alpha = \{x \in X \mid \mu_{\tilde{A}}(x) > \alpha\}$$

Neste caso, percebe-se que o suporte de $\tilde{A}$ corresponde, exatamente, ao conjunto de corte-α robusto de $\tilde{A}$ para $\alpha = 0$.

Exemplo III.4:

Ainda em referência à *Tabela III.1*, os conjuntos de corte-α possíveis, para o conjunto *fuzzy* $\tilde{A}$, *jovem*, são:

$A_{0,1} = \{5, 10, 20, 30, 40, 50\}$

$A_{0,2} = \{5, 10, 20, 30, 40\}$

$A_{0,5} = \{5, 10, 20, 30\}$

$A_{0,8} = \{5, 10, 20\}$

$A_{1,0} = \{5, 10\}$

Neste caso, o conjunto de corte-α robusto para $\alpha = 0{,}8$ é $A'_\alpha = \{5, 10\}$.

O conjunto de todos os níveis $\alpha \in [0, 1]$, que representa distintos conjuntos de corte-α de um dado conjunto *fuzzy* $\tilde{A}$, definido em X, é denominado *conjunto de nível de* $\tilde{A}$, $\Lambda_{\tilde{A}}$, dado por:

$$\Lambda_{\tilde{A}} = \{\alpha \mid \mu_{\tilde{A}}(x) = \alpha \text{ para todo } x \in X\}$$

A seguinte propriedade pode ser deduzida dos conjuntos de corte-α e corte-α robusto:

- Qualquer conjunto *fuzzy* $\tilde{A}$, com $\alpha_1, \alpha_2 \in [0, 1]$ e $\alpha_1 \neq \alpha_2$, para $\alpha_1 < \alpha_2$, tem-se:

$$A_{\alpha_1} \supseteq A_{\alpha_2} \qquad \text{e} \qquad A'_{\alpha_1} \supseteq A'_{\alpha_2}$$

Em conseqüência dessa propriedade, todos os conjuntos de corte-α de um conjunto *fuzzy* $\tilde{A}$, em X, formam uma família de subconjuntos nítidos, aninhados em X.

Convexidade

A *convexidade* de conjuntos *fuzzy* é definida em $\mathbb{R}^n$, para todo $n \in \mathbb{N}$, como sendo uma generalização do conceito de convexidade de conjuntos nítidos. Para isto é requerido que os conjuntos de corte-α de um conjunto *fuzzy* convexo sejam convexos para todo $\alpha \in (0, 1]$. Portanto, um conjunto *fuzzy* é convexo, se todos os seus conjuntos de corte-α são convexos.

De forma simplificada, trabalhando-se em $\mathbb{R}$, e para todo $x_1, x_2 \in X$ e $\lambda \in [0, 1]$, onde *min* caracteriza o *operador mínimo* (trabalha com o menor valor em um conjunto de valores), um conjunto *fuzzy* é convexo se:

$$\mu_{\tilde{A}}(\lambda x_1 + (1 - \lambda)x_2) \geq \min (\mu_{\tilde{A}}(x_1), \mu_{\tilde{A}}(x_2))$$

Cardinalidade

A *cardinalidade escalar* ou, simplesmente, *cardinalidade*, $| \tilde{A} |$, de um conjunto *fuzzy* $\tilde{A}$, definido em X, é o somatório dos graus de pertinência de todos os elementos de X em $\tilde{A}$. Formalmente,

$$| \tilde{A} | = \sum_{x \in X} \mu_{\tilde{A}}(x)$$

Para um conjunto universo infinito X, a cardinalidade, que nem sempre existe (é necessário que $\mu_{\tilde{A}}(x)$ seja integrável), é dada por:

$$| \tilde{A} | = \int_x \mu_{\tilde{A}}(x) dx$$

A *cardinalidade relativa*, $\| \tilde{A} \|$, de um conjunto *fuzzy* $\tilde{A}$ depende da cardinalidade do conjunto universo considerado. Assim, deve-se escolher o mesmo conjunto universo X, caso se queira comparar conjuntos *fuzzy* através de sua cardinalidade relativa. Pode

ser interpretada como a fração dos elementos de X, presentes em $\tilde{A}$, medidos por seus graus de pertinência:

$$|| \tilde{A} || = | \tilde{A} | / | X |$$

Exemplo III.5:

A cardinalidade escalar do conjunto *fuzzy* $\tilde{A}$, *velho*, da *Tabela III.1* é:

$$| \tilde{A} | = 0 + 0 + 0,1 + 0,2 + 0,4 + 0,6 + 0,8 + 1 + 1 = 4,1$$

A cardinalidade escalar do conjunto *fuzzy* $\tilde{A}$, *infantil*, é 0.

A cardinalidade relativa do conjunto *fuzzy* $\tilde{A}$, *velho*, é:

$$|| \tilde{A} || = 4,1 / 9 = 0,456$$

Fuzificação

A *fuzificação* ("*fuzzification*") acontece, quando um conjunto *fuzzy* $\tilde{A}$ é obtido pelo "*alargamento*" *fuzzy* de um conjunto nítido, isto é, um conjunto nítido é convertido em um conjunto *fuzzy* apropriado, para expressar medidas de incertezas.

Exemplo III.6:

Seja o conjunto nítido $A = \{x \mid 7 < x < 10\}$, então, pelo processo da *fuzzificação*, ter-se-ia o seguinte conjunto *fuzzy* $\tilde{A} = \{x \mid 7 \lessapprox x \lessapprox 10\}$, onde o símbolo $\approx$ é denominado um *fuzzificador*, e significa *aproximadamente*.

Defuzificação

A *defuzificação* é a conversão de um conjunto *fuzzy* em um valor nítido (ou um vetor de valores) (OLIVEIRA, 1995a, SONG, 1994, FILEV, 1993, MABUCHI, 1993, YAGER *et al.*, 1993a). Na literatura de controle *fuzzy*, predominam alguns métodos de *defuzificação* (LEE, 1990), entre eles, o *método do centro de gravidade* ou *método centróide*.

Princípio da extensão

O *princípio da extensão* é utilizado para generalizar conceitos da matemática clássica para a teoria *fuzzy* (ZADEH, 1973a, DUBOIS, 1980). Sejam X o produto cartesiano dos conjuntos universos $X = X_1, X_2, ..., X_r$ e r conjuntos *fuzzy* $\tilde{A}_1, \tilde{A}_2, ..., \tilde{A}_r$ em $X_1, X_2, ..., X_r$, respectivamente. Se f é um mapeamento de X para um conjunto universo Y, $y = f(x_1, x_2, ..., x_r)$, então, pelo princípio da extensão, pode-se definir um conjunto *fuzzy* $\tilde{B}$ em Y, onde f^{-1} é a função inversa de f.

$$\tilde{B} = \{(y, \mu_{\tilde{B}}(x)) \mid y = f(x_1, x_2, ..., x_r), (x_1, x_2, ..., x_r) \in X\}$$

onde

$$\mu_{\tilde{B}}(y) = \begin{cases} \sup_{(x_1, x_2, ..., x_r) \in f^{-1}(y)} \min\{\mu_{\tilde{A}_1}(x_1), \mu_{\tilde{A}_2}(x_2), ..., \mu_{\tilde{A}_r}(x_r)\} & \text{se } f^{-1}(y) \neq \phi \\ 0 & \text{outros casos} \end{cases}$$

Para $r = 1$, o princípio da extensão, reduz-se a:

$$\tilde{B} = f(\tilde{A}) = \{(y, \mu_{\tilde{B}}(x)) \mid y = f(x), x \in X\}$$

onde

$$\mu_{\tilde{B}}(y) = \begin{cases} \sup_{x \in f^{-1}(y)} \mu_{\tilde{A}}(x), & \text{se } f^{-1}(y) \neq \phi \\ 0 & \text{outros casos} \end{cases}$$

Exemplo III.7:

Sejam $\tilde{A}$ = {(-1; 0,5), (0; 0,8), (1; 1), (2; 0,4)} e

$f(x) = x^2$

Aplicando-se o princípio da extensão, ilustrado na *Figura III.1*, obtém-se:

$\tilde{B} = f(\tilde{A})$ = {(0; 0,8), (1; 1), (4; 0,4)}

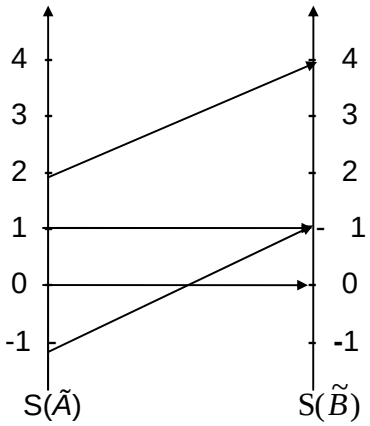


Figura III.1: O Princípio da Extensão (ZIMMERMANN, 1991)

Funções *fuzzy*

Uma *função fuzzy* é uma extensão do conceito de uma função clássica *f*, podendo ser obtida através de diferentes "*graus*" de *fuzificação* (SASAKI, 1993, TURKSEN, 1991, ZIMMERMANN, 1991):

i. Mapeamento clássico de um conjunto *fuzzy*, que se realiza ao longo do processo de *fuzificação* do domínio da função, sendo que seu contradomínio seria nítido.

ii. Mapeamento *fuzzy* de um conjunto *fuzzy*, tornando seu contradomínio também *fuzzy*. Este processo é conhecido como "*fuzzifying* funções".

iii. Funções nítidas podem ter propriedades *fuzzy* ou estarem sujeitas a restrições *fuzzy*.

Dada uma função clássica f: $X \rightarrow Y$ e um domínio *fuzzy* $\tilde{A}$, em X, pelo princípio da extensão é gerada uma imagem *fuzzy* de $\tilde{B}$ com a função de pertinência (DUBOIS, 1980):

$$\mu_{\tilde{B}}(y) = \sup_{x \in f^{-1}(y)} \mu_{\tilde{A}}(x)$$

Em função dos conceitos básicos dos conjuntos *fuzzy* apresentados, serão mostradas algumas de suas operações mais importantes, no contexto deste trabalho.

III.2 Operações com Conjuntos *Fuzzy*

As operações *fuzzy* - *complemento*, *interseção* e *união* - constituem uma estrutura consistente da teoria dos conjuntos *fuzzy*, para a extensão de conjuntos nítidos (ZADEH, 1965). Nessas operações padrões, são utilizados os operadores *min* (mínimo) e *max* (máximo) para a interseção e a união de conjuntos *fuzzy*, respectivamente. Outros operadores têm sido sugeridos (YAGER *et al*., 1993a, DUBOIS *et al*., 1982, YAGER, 1980), variando em generalidade e adaptabilidade, satisfazendo a certas propriedades, dependendo do contexto empregado.

III.2.1 Complemento *Fuzzy*

A função de pertinência padrão do *complemento* de um conjunto *fuzzy* $\tilde{A}$, $\mu_{c\tilde{A}}(x)$, para todo $x \in X$, é definida por:

$$\mu_{c\tilde{A}}(x) = 1 - \mu_{\tilde{A}}(x)$$

Exemplo III.8:

O complemento do conjunto *fuzzy* $\tilde{A}$, *velho*, da *Tabela III.1*, é $c\tilde{A}$, *não-velho*, que é representado por:

$\mu_{c\tilde{A}}(x)$ = {(5; 1), (10; 1), (20; 0,9), (30; 0,8), (40; 0,6), (50; 0,4), (60; 0,2)}

O *complemento fuzzy* genérico de dois conjuntos $\tilde{A}$ e $\tilde{B}$ pode ser denotado pela operação binária, no intervalo unitário por:

$$c : [0, 1] \times [0, 1] \rightarrow [0, 1]$$

O complemento *fuzzy* de $\tilde{A}$ de tipo *c*, $\mu_{c\tilde{A}}(x)$, pode ser interpretado não somente como o grau de pertinência pelo qual *x* pertence à $c\tilde{A}$, mas também como o grau pelo qual *x* não pertence ao conjunto *fuzzy* $\tilde{A}$. Assim, $\mu_{\tilde{A}}(x)$ pode ser entendido como o grau pelo qual *x* não pertence à $c\tilde{A}$. Dado um conjunto *fuzzy* $\tilde{A}$, obtemos $c\tilde{A}$ aplicando-se a função *c* aos valores $\mu_{\tilde{A}}(x)$ para todo *x*:

$$\mu_{c\tilde{A}}(x) = c(\mu_{\tilde{A}}(x))$$

Para produzir complementos *fuzzy* convenientes, a função *c* deve satisfazer, no mínimo, os dois axiomas seguintes (KLIR, 1995a):

- *Axioma* c_1: $c(0) = 1$; e $c(1) = 0$ (condições de contorno), isto é, a função *c* produz o mesmo complemento para conjuntos nítidos.
- *Axioma* c_2: para todo $a, b \in [0, 1]$, se $a \leq b$, então $c(a) \geq c(b)$ (monotonicidade).

Pode considerar, também, os seguintes requisitos para complemento *fuzzy*. Cada um deles reduz a generalização do complemento *fuzzy* para subclasses especiais.

- *Axioma* c_3: *c* é uma função contínua.
- *Axioma* c_4: *c* é *involutiva*, isto é, $c(\mu_{c\tilde{A}}(x)) = x$, para todo $x \in [0, 1]$.

III.2.2 Interseção *Fuzzy*

A *interseção fuzzy* padrão dos conjuntos *fuzzy* $\tilde{A}$ e $\tilde{B}$ é o conjunto *fuzzy* $\tilde{A} \cap \tilde{B}$, para todo $x \in X$, de modo que:

$$\mu_{\tilde{A}\cap\tilde{B}}(x) = \min [\mu_{\tilde{A}}(x),\ \mu_{\tilde{B}}(x)]$$

Exemplo III.9:

A interseção dos conjuntos *fuzzy* $\tilde{A}$, *velho*, e de $\tilde{B}$, *jovem*,da *Tabela III.1*, é dada por:

$$\mu_{\tilde{A}\cap\tilde{B}}(x) = \{(20; 0,1), (30; 0,2), (40; 0,2), (50; 0,1)\}$$

A *interseção fuzzy* genérica de dois conjuntos $\tilde{A}$ e $\tilde{B}$, geralmente, é especificada pela operação binária no intervalo unitário, isto é, pela função na forma:

$$i : [0, 1] \times [0, 1] \rightarrow [0, 1]$$

Os argumentos para essa função são o par constituído pelo grau de pertinência de cada elemento *x*, em $\tilde{A}$, e o grau de pertinência de seu correspondente em $\tilde{B}$. A função retorna, então, com o grau de pertinência do elemento em $\tilde{A} \cap \tilde{B}$. Portanto,

$$\mu_{\tilde{A} \cap \tilde{B}}(x) = i\,[\mu_{\tilde{A}}(x),\ \mu_{\tilde{B}}(x)\,]$$

A função *i*, para poder ser considerada como interseção *fuzzy*, deve satisfazer os quatro axiomas abaixo (Klir 95a]:

- *Axioma* i_1: $i(1, 1) = 1$; $i(0, 1) = i(1, 0)$; $i(0, 0) = 0$ (condições de contorno). A função *i* comporta-se como a interseção clássica de conjuntos nítidos.
- *Axioma* i_2: $i(a, b) = i(b, a)$; isto é, *i* é *comutativa*. Assim sendo, não importa a ordem, na qual os conjuntos são combinados.
- *Axioma* i_3: Se $a \leq a'$ e $b \leq b'$, então, $i(a, b) \leq i(a', b')$ (monotonicidade). Portanto, um decréscimo no grau de pertinência de $\tilde{A}$ ou de $\tilde{B}$ não pode produzir um acréscimo no grau de pertinência de $\tilde{A} \cap \tilde{B}$.
- *Axioma* i_4: $i(i(a, b), c) = i(a, i(b, c))$ (associatividade). É possível fazer a interseção de qualquer quantidade de conjuntos, em qualquer ordem de agrupamentos desejada.

Os requisitos adicionais para interseção de conjuntos *fuzzy*, que são desejáveis em certas aplicações, são expressados pelos seguintes axiomas:

- *Axioma* i_5: *i* é uma função *contínua*. Neste caso, um pequeno incremento no grau de pertinência de $\tilde{A}$ ou de $\tilde{B}$, não produzirá uma variação substancial no grau de pertinência de $\tilde{A} \cap \tilde{B}$.
- *Axioma* i_6: $i(a, a) = a$ (idempotência). A interseção de um conjunto *fuzzy* consigo mesmo produz o mesmo conjunto *fuzzy*.

III.2.3 União *Fuzzy*

A *união fuzzy* padrão dos conjuntos $\tilde{A}$ e $\tilde{B}$ é o conjunto *fuzzy* $\tilde{A} \cup \tilde{B}$, para todo $x \in X$, é formalizada por:

$$\mu_{\tilde{A}\cup\tilde{B}}(x) = \max\,[\mu_{\tilde{A}}(x),\ \mu_{\tilde{B}}(x)\,]$$

Exemplo III.10:

A união dos conjuntos *fuzzy* $\tilde{A}$, *velho*, e de $\tilde{B}$, *jovem*, da *Tabela III.1*, é dada por:

$$\mu_{\tilde{A}\cup\tilde{B}}(x) = \{(5;\ 1),\ (10;\ 1),\ (20;\ 0{,}8),\ (30;\ 0{,}5),\ (40;\ 0{,}4),\ (50;\ 0{,}6),$$
$$(60;\ 0{,}8),\ (70;\ 1),\ (80;\ 1)\}$$

A *união fuzzy* genérica dos conjuntos $\tilde{A}$ e $\tilde{B}$, de maneira semelhante à interseção, é representada, geralmente, pela função:

$$u : [0,\ 1] \times [0,\ 1] \rightarrow [0,\ 1]$$

Para cada elemento $x \in X$, essa função possui, como argumentos, o par constituído dos graus de pertinência dos elementos de $\tilde{A}$ e de $\tilde{B}$, produzindo o grau de pertinência do conjunto união $\tilde{A} \cup \tilde{B}$. Formalmente, temos:

$$\mu_{\tilde{A}\cup\tilde{B}}(x) = u\,[\mu_{\tilde{A}}(x),\ \mu_{\tilde{B}}(x)\,]$$

Qualquer que seja a forma da função *u*, para que seja qualificada como união *fuzzy*, deve atender, no mínimo, os quatro axiomas seguintes (KLIR, 1995a):

- *Axioma* u_1: $u(0,\ 0) = 0$; $u(0,\ 1) = u(1,\ 0)$; $u(1,\ 1) = 1$ (condições de contorno). A função *u* comporta-se como a união clássica de conjuntos nítidos.

- *Axioma* u_2: $u(a,\ b) = u(b,\ a)$ (comutatividade). Portanto, a ordem, pela qual os conjuntos são combinados, não é relevante.

- *Axioma* u_3: Se $a \leq a'$ e $b \leq b'$, então, $u(a,\ b) \leq u(a',\ b')$ (*monotonicidade*). Um decréscimo no grau de pertinência de $\tilde{A}$ ou de $\tilde{B}$ não pode produzir um incremento no grau de pertinência de $\tilde{A} \cup \tilde{B}$.

- *Axioma* u_4: $u(u(a,\ b),\ c) = u(a,\ u(b,\ c))$ (associatividade). Pode-se efetuar a união de qualquer número de conjuntos e em qualquer ordem de agrupamentos desejados.

Outros requisitos, igualmente importantes, para a união de conjuntos *fuzzy* são mostrados abaixo:

- *Axioma* u_5: *u* é uma função *contínua*.

- *Axioma* u_6: $u(a, a) = a$ (idempotência). Semelhantemente à interseção, a união de um conjunto *fuzzy* consigo mesmo gera, precisamente, o mesmo conjunto.

Em BELLMANN (1973), a interpretação da interseção como "*e lógico*", e da união como "*ou lógico*" é fundamentada axiomaticamente. Entretanto, os operadores *min* e *max* não são os únicos, que podem ser escolhidos, respectivamente, para as operações de interseção e de união de conjuntos *fuzzy*. A seguir, serão mostradas algumas operações algébricas com conjuntos *fuzzy*.

III.2.4 Operações Algébricas com Conjuntos *Fuzzy*

As operações algébricas são, também, extensões dos conceitos básicos dos conjuntos *fuzzy*, envolvendo suas definições e operações (ZIMMERMANN, 1991).

Produto cartesiano

Sejam os conjuntos *fuzzy*, $\tilde{A}_1$, $\tilde{A}_2$, ..., $\tilde{A}_n$, em X_1, X_2, ..., X_n. Então, o *produto cartesiano* é o conjunto *fuzzy* no espaço do produto $X_1 \times X_2 \times ... \times X_n$, com a função de pertinência:

$$\mu_{(\tilde{A}_1 \times \tilde{A}_2 \times ... \times \tilde{A}_n)}(x) = \min_i \{\mu_{\tilde{A}i}(x_i) \mid x = (x_1, x_2, ..., x_n), x_i \in X_i\}$$

Soma algébrica

A *soma algébrica* $\tilde{C} = \tilde{A} + \tilde{B}$ é definida como

$$\tilde{C} = \{(x, \mu_{\tilde{A}+\tilde{B}}(x)) \mid x \in X\}$$

onde

$$\mu_{\tilde{A}+\tilde{B}}(x) = \mu_{\tilde{A}}(x) + \mu_{\tilde{B}}(x) - \mu_{\tilde{A}}(x) \cdot \mu_{\tilde{B}}(x)$$

Soma de contorno

A *soma de contorno* $\tilde{C} = \tilde{A} \oplus \tilde{B}$ é definida como

$$\tilde{C} = \{(x, \mu_{\tilde{A}\oplus\tilde{B}}(x)) \mid x \in X\}$$

onde

$$\mu_{\tilde{A}\oplus\tilde{B}}(x) = \min \{1, \mu_{\tilde{A}}(x) + \mu_{\tilde{B}}(x)\}$$

Diferença de contorno

A *diferença de contorno* $\tilde{C} = \tilde{A}\,\theta\,\tilde{B}$ é definida como

$$\tilde{C} = \{(x, \mu_{\tilde{A}\theta\tilde{B}}(x)) \mid x \in X\}$$

onde

$$\mu_{\tilde{A}\theta\tilde{B}}(x) = \max\{0, \mu_{\tilde{A}}(x) + \mu_{\tilde{B}}(x) - 1\}$$

Produto algébrico

O *produto algébrico* de dois conjuntos *fuzzy* $\tilde{C} = \tilde{A} \cdot \tilde{B}$ é definido como

$$\tilde{C} = \{(x, \mu_{\tilde{A}}(x) \cdot \mu_{\tilde{B}}(x)) \mid x \in X\}$$

Exemplo III.11:

Sejam os conjuntos *fuzzy* $\tilde{A}(x) = \{(3; 0{,}5), (5; 1), (7; 0{,}6)\}$ e

$$\tilde{B}(x) = \{(3; 1), (5; 0{,}6)\}$$

Aplicando-se as definições acima, tem-se os seguintes resultados:

$$\tilde{A} \times \tilde{B} = \{[(3; 3); 0{,}5], [(5; 3); 1], [(7; 3); 0{,}6],$$
$$[(3; 5); 0{,}5], [(5; 5); 0{,}6], [(7;5); 0{,}6]\}$$

$$\tilde{A}^2 = \{(3; 0{,}25), (5; 1), (7; 0{,}36)\}$$

$$\tilde{A} + \tilde{B} = \{(3; 1), (5; 1), (7; 0{,}6)\}$$

$$\tilde{A} \oplus \tilde{B} = \{(3; 1), (5; 1), (7; 0{,}6)\}$$

$$\tilde{A}\,\theta\,\tilde{B} = \{(3; 0{,}5), (5; 0{,}6)\}$$

$$\tilde{A} \cdot \tilde{B} = \{(3; 0{,}5), (5; 0{,}6)\}$$

Muitas operações com conjuntos *fuzzy* estão direcionadas para a agregação de suas funções de pertinência. Esta é uma questão de particular interesse em qualidade de software, isto é, a agregação de informações pertinentes a atributos de qualidade do produto ou do processo considerado.

III.3 Agregação de Conjuntos *Fuzzy*

A agregação é um processo utilizado em muitas tecnologias (YAGER, 1994), especialmente na tomada de decisão multicriterial (ZIMMERMANN, 1997). Têm sido propostas muitas alternativas para este fim, como o processo de dedução e inferência lógica e outros tipos de conhecimento indutivo. As *redes neurais artificiais* (ZOVÁCS, 1996, KASABOV *et al*., 1996, KOSKO, 1995, BUCKLEY *et al*., 1994, GUPTA, 1992),

por exemplo, estão baseadas em agregações de taxas de sinais para cada neurônio, gerando um sinal de saída (YAGER, 1993b).

A idéia principal do processo de agregação é obter-se um grau de consenso entre as informações disponíveis, calculando-se um valor final. Se estes dados forem extraídos de especialistas, então ter-se-á a taxa de aceitação ou rejeição entre eles, isto é, o grau pelo qual especialistas concordam em suas estimativas, tornando possível a elaboração de classificações das avaliações realizadas (KUNCHEVA *et al*., 1996).

DAY (1988) argumenta que os modelos de consenso são potencialmente férteis e podem ser usados em vários domínios de aplicação. Os métodos de consenso referem-se, principalmente, a esquemas de graduação, expandindo-se para relações de preferência (FEDRIZZI *et al*., 1993, GILARDONI *et al*., 1993, KACPRZYK *et al*., 1992), decisões de grupo (CARLSSON *et al*., 1996, GRABISH, 1995, SAKAWA *et al*., 1994) e estimativas lingüisticamente definidas (MICH *et al*., 1993). O grau de consenso tem sido estimado, usando-se alguns operadores de agregação *fuzzy*, através das preferências individuais de *n* especialistas, para cada atributo considerado, e gerando-se uma matriz de decisão, chamada em (KUNCHEVA, 1995) de '*perfil de decisão*'.

Segundo BARDOSSY *et al* (1993), as seguintes propriedade podem ser desejáveis, quando se utiliza funções de agregação:

i. *Preservação da concordância*: se todas as estimativas são idênticas, o resultado da combinação deverá ser a própria estimativa (requisito de consistência).

ii. *Independência de ordem*: o resultado não deve depender da ordem com que opiniões ou estimativas individuais são combinadas (requisito de consistência).

iii. *Invariância da transformação*: transformações ocorridas no ambiente, onde se processa a agregação, não deverão afetar seu resultado.

iv. *Conservação da possibilidade*: se um dado valor for considerado possível para uma única estimativa, então, ele permanece possível para toda a combinação.

v. *Conservação do intervalo de possibilidade*: um certo valor localizado entre dois valores, considerados como estimativas possíveis, é tido também como possível.

vi. *Incertezas individuais versus globais*: a medida de incerteza, $H(\tilde{R}_i)$, de uma estimativa individual, $\tilde{R}_i$, é definida como a área sob sua função de pertinência:

$$H(\tilde{R}_i) = \int_{-\infty}^{\infty} \mu_{\tilde{R}_i}(x)dx \qquad \text{A}$$

medida de incerteza global, $H(\tilde{R})$, isto é, a combinação de todas as estimativas individuais, $\tilde{R}$, também pode ser medida analogamente.

vii.Conveniência da estimativa dos resultados: combinação de várias características de técnicas de agregação, que são avaliadas pelos critérios expostos nos itens acima, através de uma escala subjetiva, variando de A (excelente) a E (péssimo).

Conceitualmente, operações de *agregação fuzzy* são combinações de vários conjuntos *fuzzy* ($n \geq 2$), de uma forma desejável, para produzirem um único conjunto. Em geral, uma operação de agregação é representada pela função:

$$h : [0, 1]^n \rightarrow [0, 1]$$

Quando a função h é aplicada para n conjuntos *fuzzy*, $\tilde{A}_1, \tilde{A}_2, ..., \tilde{A}_n$, definidos em X e operando com os graus de pertinência de cada $x \in X$, é produzido um *conjunto fuzzy agregado* $\tilde{A}$. Assim, para cada $x \in X$:

$$\mu_{\tilde{A}}(x) = h(\mu_{\tilde{A}_1}(x), \mu_{\tilde{A}_2}(x), ..., \mu_{\tilde{A}_n}(x))$$

Para que h seja qualificada como uma função de agregação, deve satisfazer, no mínimo, os três axiomas a seguir (KLIR, 1995a):

- *Axioma* h_1: $h(0, 0, ..., 0) = 0$ e $h(1, 1, ..., 1) = 1$ (condições de contorno).
- *Axioma* h_2: para qualquer par $\langle a_1, a_2, ..., a_n\rangle$ e $\langle b_1, b_2, ..., b_n\rangle$ de n-tuplas tal que $a_i, b_i \in [0, 1]$ para todo $i \in \mathbb{N}_n$, h é *monotônica* incremental em todos os seus argumentos, se para $a_i \leq b_i$, $i \in \mathbb{N}_n$, então,

  $$h(a_1, a_2, ..., a_n) \leq h(b_1, b_2, ..., b_n)$$

- *Axioma* h_3: h é uma *função contínua*, isto é, garante-se que uma variação infinitesimal em qualquer argumento de h não produz mudanças consideráveis na agregação.

Dois axiomas adicionais são também empregados para operações de agregação, embora não sejam essenciais:

- *Axioma* h_4: h é uma função *simétrica* em todos os seus argumentos, para qualquer permutação p em $\mathbb{N}_n$, isto é,

$$h(a_1, a_2, ..., a_n) = h(a_{p(1)}, a_{p(2)}, ..., a_{p(n)})$$

- *Axioma* h_5: h é uma função *idempotente*, para todo $a \in [0, 1]$, isto é:

$$h(a, a, ..., a) = a$$

A união e a interseção *fuzzy* qualificam-se, normalmente, como operações de agregação. Embora sejam definidas primariamente para dois argumentos, suas propriedades de associatividade, garantidas pelos *axiomas* i_4 e u_4, fornecem um mecanismo para extensão de suas definições para qualquer número de argumentos.

Qualquer operação de agregação h que satisfaça os *axiomas* h_2 e h_5, satisfaz, também, todas as n-tuplas $\langle a_1, a_2, ..., a_n\rangle \in [0, 1]^n$, para as seguintes inequações:

$$\min(a_1, a_2, ..., a_n) \leq h(a_1, a_2, ..., a_n) \leq \max(a_1, a_2, ..., a_n)$$

Neste caso, a operação de agregação h é chamada *operação de média*. Portanto, os operadores padrões *max* e *min* representam os limites entre as operações de média e a união e a intercessão *fuzzy*, respectivamente.

YAGER (1988) desenvolveu a classe de *operações de média de pesos ordenados* ou *OWA* (*Ordered Weighted Averaging*), que satisfazem os *axiomas* de h_1 a h_5. Dado um vetor de pesos, **w**, tal que $w_i \in [0, 1]$ para todo $i \in \mathbb{N}_n$, então,

$$\mathbf{w} = \langle w_1, w_2, ..., w_n\rangle, \text{ e}$$

$$\sum_{i=1}^{n} w_i = 1$$

Em uma operação *OWA* associada ao vetor **w**, onde b_i é o maior i-ésimo elemento em $a_1, a_2,...,a_n$, para qualquer $i \in \mathbb{N}_n$, isto é, $\langle b_1, b_2, ..., b_n\rangle$ é uma permutação do vetor $\langle a_1, a_2, ..., a_n\rangle$, no qual os elementos estão ordenados: $b_i \geq b_j$, se $i < j$ para qualquer par $i, j \in \mathbb{N}_n$, então,

$$h_w(a_1, a_2, ..., a_n) = w_1b_1 + w_2b_2 + ... + w_nb_n$$

Exemplo III.12:

Dado $\mathbf{w} = \langle 0{,}3; 0{,}1; 0{,}2; 0{,}4\rangle$, tem-se que

$$h_{\mathbf{w}}(0{,}6; 0{,}9; 0{,}2; 0{,}7) = 0{,}3 \times 0{,}9 + 0{,}1 \times 0{,}7 + 0{,}2 \times 0{,}6 + 0{,}4 \times 0{,}2 = 0{,}54.$$

Em toda operação de agregação, os operadores *fuzzy* exercem uma grande influência no resultado final. Um número significativo desses operadores foram

propostos (CHEN *et al*., 1992, KRISHNAPURAM *et al*., 1992, ZIMMERMANN, 1991, DUBOIS, 1984, DUBOIS, 1985), e a escolha apropriada dos mesmos para uma determinada situação, é de grande relevância.

III.3.1 Operadores *Fuzzy*

Embora haja muitos operadores de agregação desenvolvidos, não há enfoques semânticos para a escolha apropriada destes conectivos, para uma situação em particular, uma vez que não existem métodos formais de análise independentes da situação-problema (FELIX, 1994).

Quando se usa a teoria *fuzzy* para modelar fenômenos reais, apenas justificativas puramente matemáticas dos operadores não são suficientes. Além da adequação da modelagem do problema é necessário que esta tenha sido provada empiricamente (THOLE *et al*., 1979).

Com a variedade de operadores para a agregação de conjuntos *fuzzy*, pode ser difícil decidir qual deles usar para um modelo específico ou em uma determinada situação. Há alguns critérios que podem ser úteis na seleção apropriada desses conectivos (ZIMMERMANN, 1991):

- *Força axiomática*: devem satisfazer a seus axiomas, em seus mínimos detalhes, tendo certas qualidades formais (tais como comutatividade, associatividade).
- *Idoneidade empírica*: devem modelar convenientemente o comportamento de sistemas reais, podendo ser obtidos através de testes empíricos.
- *Adaptabilidade*: devem ser adaptáveis a uma semântica ou a um contexto específico (através de parametrização, por exemplo).
- *Eficiência numérica*: requerem um certo esforço computacional, que deve ser considerado, principalmente, em problemas de grande porte.
- *Compensação*: o grau de pertinência de um *conjunto fuzzy agregado* é dado por:

$$\mu_{Ag}(x_k) = f(\mu_{\tilde{A}}(x_k),\ \mu_{\tilde{B}}(x_k)) = k$$

onde f é uma *função compensatória*, se $\mu_{Ag}(x_k) = k$ for obtida para diferentes $\mu_{\tilde{A}}(x_k)$, alterando-se $\mu_{\tilde{B}}(x_k)$. O operador-min, por exemplo, não é compensatório, enquanto que o *operador-produto* o é.

- *Intervalo de Compensação*: em geral, quanto maior for o intervalo de compensação, melhor é o operador compensatório.

- *Ambiente de agregação*: para conjuntos *fuzzy* normais ou subnormais, o grau de pertinência do conjunto agregado depende, freqüentemente, da quantidade dos conjuntos combinados e geralmente não é incremental.

- *Nível de escala requerido das funções de pertinência*: o nível de escala (*nominal*, *intervalar*, *taxação* e *absoluta*) (TURKSEN, 1991), no qual as informações de pertinência podem ser obtidas, depende de vários fatores.

Em seções anteriores, a interseção de conjuntos *fuzzy*, interpretada pelo operador lógico "*e*", foi modelada pelo *operador-min*, e a união, interpretada por "*ou*", pelo *operador-max*. Outros operadores têm sido sugeridos de acordo com a sua generalidade ou adaptabilidade, bem como pela justificativa empírica ou axiomática de sua escolha.

A seguir, serão investigadas as duas classes de operadores mais utilizadas, no processo de agregação: operadores para a interseção e a união de conjuntos *fuzzy*, referenciados como *classe padrão triangular* (*t-norm*) e *classe de co-padrão triangular* (*t-conorm*). Uma outra classe, a dos operadores de média, modela conectivos para conjuntos *fuzzy* entre *t-norms* e *t-conorms* (DUBOIS, 1985, ALSINA, 1985, KLEMENT *et al*., 1982], possuindo operadores parametrizados ou não.

III.3.1.1 Classes *t-norms* e *t-conorms*

As classes *t-norms* englobam os operadores: *min*, *produto* e *diferença de contorno*. A interseção de conjuntos *fuzzy* (ZADEH, 1965) foi delineada pelo *operador-min* e o *produto algébrico*. A "*interseção-robusta*" (GILES, 1976) foi modelada pela *diferença de contorno*. As classes *t-norms* são funções *t* bivaloradas de [0, 1] × [0 ,1], que satisfazem as seguintes condições (DUBOIS, 1980):

i. $t(0, 0) = 0;\ t(\mu_{\tilde{A}}(x), 1) = t(1, \mu_{\tilde{A}}(x)) = \mu_{\tilde{A}}(x), \qquad x \in X$

ii. $t(\mu_{\tilde{A}}(x),\ \mu_{\tilde{B}}(x)) \leq t(\mu_{\tilde{C}}(x), \mu_{\tilde{D}}(x))$

se $\mu_{\tilde{A}}(x) \leq \mu_{\tilde{C}}(x)$ e $\mu_{\tilde{B}}(x) \leq \mu_{\tilde{D}}(x)$ (monotonicidade)

iii. $t(\mu_{\tilde{A}}(x),\ \mu_{\tilde{B}}(x)) = t(\mu_{\tilde{B}}(x), \mu_{\tilde{A}}(x))$ (comutatividade)

iv. $t(\mu_{\tilde{A}}(x), t(\mu_{\tilde{B}}(x), \mu_{\tilde{C}}(x))) = t(t(\mu_{\tilde{A}}(x),\ \mu_{\tilde{B}}(x)),\ \mu_{\tilde{C}}(x))$ (associatividade)

As classes *t-conorms* englobam os operadores: *max*, *soma algébrica* e *soma de contorno*. A união de conjuntos *fuzzy* (ZADEH, 1965) foi delineada pelo *operador-max* e a *soma algébrica*. A "*união-robusta*" (GILES, 1976) foi modelada pela *soma de*

contorno. As classes *t-conorms* são funções *s* bivaloradas de [0, 1] × [0 ,1], que atendem às seguintes propriedades (DUBOIS, 1985):

i. $s(1, 1) = 1$; $s(\mu_{\tilde{A}}(x), 0) = s(0, \mu_{\tilde{A}}(x)) = \mu_{\tilde{A}}(x)$, $x \in X$

ii. $s(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)) \leq s(\mu_{\tilde{C}}(x), \mu_{\tilde{D}}(x))$

se $\mu_{\tilde{A}}(x) \leq \mu_{\tilde{C}}(x)$ e $\mu_{\tilde{B}}(x) \leq \mu_{\tilde{D}}(x)$ (monotonicidade)

iii. $s(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)) = s(\mu_{\tilde{B}}(x), \mu_{\tilde{A}}(x))$ (comutatividade)

iv. $s(\mu_{\tilde{A}}(x), s(\mu_{\tilde{B}}(x), \mu_{\tilde{C}}(x))) = s(s(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)), \mu_{\tilde{C}}(x))$ (associatividade)

ALSINA (1985) definiu *t-conorm*, *s*, em função de *t-norm*, *t*:

$$t(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)) = 1 - s(1 - \mu_{\tilde{A}}(x), 1 - \mu_{\tilde{B}}(x))$$

III.3.1.2 Outras classes de operadores *fuzzy*

YAGER *et al*. (1993a) criou os operadores MICA (*Monotonic Identify Commutative Aggregation*), possuindo um elemento identidade, que pode ser adicionado ao argumento de uma agregação sem mudar o valor agregado. Neste caso, *t-norm* e *t-conorm* são casos especiais desses operadores.

YAGER *et al*. (1996) unificou e generalizou, também, os operadores *t-norm* e *t-conorm*, nas classes *uni-norms*. Um operador *uni-norm R*, mapeado de [0, 1] × [0, 1] em [0, 1], tem as propriedades abaixo:

i. $R(a, b) = R(b, a)$ (comutatividade)

ii. $R(a, b) \geq R(c, d)$ se $a \geq c$ e $b \geq c$ (monotonicidade)

iii. $R(a, R(b, c)) = R(R(a, b), c)$ (associatividade)

iv. Há elementos (identidade) $e \in [0, 1]$, tal que para todo $a \in [0,1]$, $R(a, e) = a$.

Portanto, *t-norm* é um caso especial de *uni-norm* com $e = 1$, enquanto que *t-conorm* é um caso especial com $e = 0$.

Existem outras formas de combinação de conjuntos *fuzzy*. O uso dos operadores "*e*" e "*ou*" são casos limites especiais, no processo de agregação. Foram sugeridos, então, modelos generalizados para os operadores lógicos "*e*" e "*ou*" representados, respectivamente, por "*e fuzzy*" e "*ou fuzzy*" (WERNERS, 1984). A combinação desses operadores tem levado a resultados satisfatórios, em dados empíricos, permitindo compensações entre os graus de pertinência dos conjuntos agregados.

É usual, em processos de agregação, a combinação de números *fuzzy*, representando o julgamento de um especialista para uma *variável lingüística*, na avaliação de um determinado produto ou processo (HSU *et al*., 1996, HAPKE *et al*., 1994, RÖMER *et al*., 1995, BARDOSSY *et al*., 1993, LASEK, 1992, RUONING *et al*., 1992, KAUFMANN *et al*., 1991, GENEST, 1984).

III.4 Números *Fuzzy*

Em um processo de avaliação de resultados, os dados obtidos dos especialistas são geralmente imprecisos e contêm muitas ambigüidades, principalmente, em virtude de como foram capturados. A origem dessas ambigüidades pode ser (RÖMER *et al*., 1995):

- *a não acurácia dos dispositivos utilizados, envolvendo erros de medição de natureza fuzzy* (BANDEMER *et al*., 1990);
- *a natureza lingüística dos dados observados*;
- *a natureza subjetiva dos dados obtidos*.

Muitas informações vagas podem ser convenientemente modeladas por números *fuzzy* (HSU *et al*., 1996, HAPKE *et al*., 1994, RUONING *et al*., 1992, KAUFMANN, 1991, DUBOIS, 1985. O conceito de números *fuzzy* em incertezas *fuzzy* desempenha um papel semelhante ao de uma variável aleatória, nas relações de incertezas probabilísticas (SAADE, 1996).

Um *número fuzzy* $\tilde{N}$ (ou um *intervalo fuzzy*) é um conjunto *fuzzy* convexo e normalizado definido no conjunto dos números reais $\mathbb{R}$, tal que sua função de pertinência tem a forma (ZIMMERMANN, 1991, KLIR *et al*., 1995a):

$$\mu_{\tilde{A}} : \mathbb{R} \rightarrow [0, 1]$$

Um número *fuzzy* deve capturar a concepção intuitiva de números ou intervalos aproximados, tal como "*valores que estão próximos de um certo número real*", ou "*valores que estão em torno de um dado intervalo de números reais*". Tais conceitos são essenciais para a caracterização dos estados das variáveis *fuzzy* e, conseqüentemente, são importantes para aplicações tais como controle *fuzzy*, tomada de decisão, raciocínio aproximado e estatística.

Para qualificar um número *fuzzy*, um conjunto *fuzzy* $\tilde{A}$ em $\mathbb{R}$ deve possuir, no mínimo, as seguintes propriedades (KLIR *et al*., 1995a, YU, 1993, ZIMMERMANN, 1991):

i) $\tilde{A}$ deve ser um conjunto *fuzzy* normalizado;

ii) $\tilde{A}_\alpha$ deve ser um intervalo fechado para todo $\alpha \in (0, 1]$, isto é, todo número *fuzzy* é convexo;

iii) o *suporte* de $\tilde{A}$ deve ser limitado.

Casos especiais de números *fuzzy* incluem números e intervalos reais ordinários, como mostra a *Figura III.2*: (a) um número real 3; (b) um intervalo nítido [3, 4]; (c) um número *fuzzy* dado pela proposição "*próximo a 3*" (número *fuzzy* triangular); (d) um número *fuzzy* com uma região plana (um intervalo *fuzzy* ou número *fuzzy* trapezoidal).

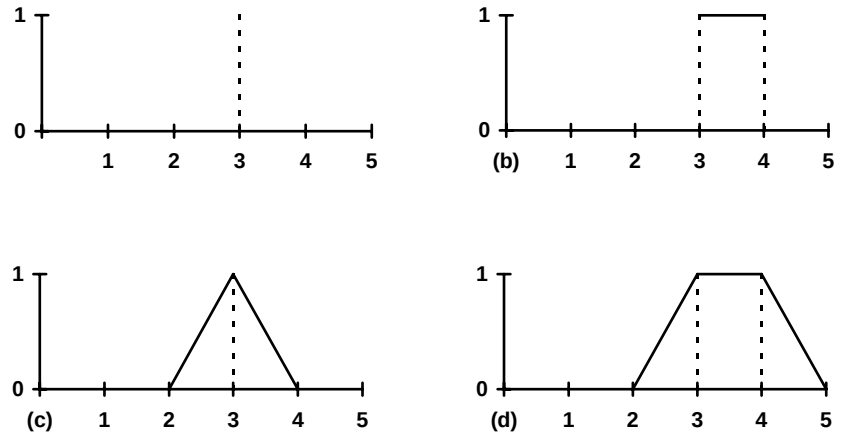


Figura III.2: Comparação de um número real e um intervalo nítido com um número *fuzzy* e um intervalo *fuzzy* respectivamente (KLIR *et al*., 1995a)

Embora as funções de pertinência de números *fuzzy* tenham, usualmente, as formas triangular ou trapezoidal, existem outras formas para representá-las, que nem sempre são simétricas, dependendo do contexto da aplicação, como funções com "*forma de sino*", funções estritamente crescentes ou decrescentes.

Dados imprecisos podem ser modelados pelo significado de números *fuzzy L-R* (HAPKE *et al*., 1994, ROUBENS, 1990, NAKAMURA, 1986,), que é uma outra importante forma de representação de funções de pertinência desses números (DUBOIS, 1980).

III.4.1 Conjuntos *fuzzy* tipo-*LR*

Um número *fuzzy* $\tilde{N}$ é do tipo-*LR*, se houver as funções de referência *L* (para a esquerda (*left*)), *R* (para a direita (*right*)), os números escalares $\alpha > 0$ e $\beta > 0$ (extensões da esquerda e da direita, respectivamente), e o número real *m*, chamado valor médio de $\tilde{N}$. Neste caso, $\tilde{N}$ é um número *fuzzy triangular*. Simbolicamente, $\tilde{N}$ é denotado por $(m, \alpha, \beta)_{LR}$ (ZIMMERMANN, 1991, DUBOIS *et al*., 1980):

$$\mu_{\tilde{N}}(x) = \begin{cases} L\left(\dfrac{m-x}{\alpha}\right) & \text{para } x \le m \\ R\left(\dfrac{x-m}{\beta}\right) & \text{para } x \ge m \end{cases}$$

Baseando-se em (LEE, 1996a, HSU, 1996, KAUFMANN, 1991), um número *fuzzy* triangular normal do tipo-*LR* pode ser representado por, $\tilde{N} = (a; m; b)$, onde $a = m - |\alpha|$ e $b = m + |\beta|$.

Se *m* não é um número real, mas um intervalo $[\underline{m}, \overline{m}]$, o número *fuzzy* $\tilde{N}$ se transforma em um *intervalo fuzzy* (número *fuzzy* trapezoidal). Um *intervalo fuzzy* $\tilde{N}$ é do tipo-*LR*, se houver funções na forma *L* e *R* e, também, os parâmetros $(\underline{m}, \overline{m}) \in \mathbb{R}^2 \cup \{-\infty, +\infty\}$, *a*, *b*. O intervalo *fuzzy* é denotado, então, por $\tilde{N} = (\underline{m}, \overline{m}, a, b)_{LR}$ (DUBOIS *et al*. 1988):

$$\mu_{\tilde{N}}(x) = \begin{cases} L\left(\dfrac{\underline{m}-x}{\alpha}\right) & \text{para } x \le \underline{m} \\ 1 & \text{para } \underline{m} \le x \le \overline{m} \\ R\left(\dfrac{x-\overline{m}}{\beta}\right) & \text{para } x \ge \overline{m} \end{cases}$$

Um intervalo *fuzzy* normal do tipo-*LR* pode ser representado por $\tilde{N} = (a, \underline{m}, \overline{m}, b)_{LR}$, onde $a = \underline{m} - |\alpha|$ e $b = \overline{m} + |\beta|$.

A teoria dos conjuntos *fuzzy* (notadamente os números *fuzzy*) é utilizada no raciocínio aproximado, para efetivamente manusear a ambigüidade envolvida na avaliação de dados e em propriedades imprecisas de variáveis lingüísticas (LEE, 1996b).

III.5 Variáveis Lingüísticas

Uma *variável lingüística* é totalmente caracterizada por uma quíntupla (x, $T(x)$, U, G, $\tilde{M}$). O *nome* da variável é x. O conjunto dos *termos lingüísticos* de x é $T(x)$, ou simplesmente T, que se refere a uma *variável base u*, cujos valores estão no conjunto universo U. G é uma regra sintática, para a geração dos termos lingüísticos. M é uma *regra semântica*, que associa a cada termo lingüístico $t \in T$ o seu significado, $\tilde{M}(t)$, que é um conjunto *fuzzy* em U (ZIMMERMANN, 1991).

Exemplo III.13 (ZADEH, 1973a):

Seja X uma variável lingüística identificada por "*Idade*" com U = [0, 100] e seus termos lingüísticos, que também são conjuntos *fuzzy*, "*velho*", "*jovem*", "*muito velho*", etc. A variável base u é a idade em anos de vida. $\tilde{M}(t)$ é a regra que atribui um significado, isto é, um conjunto *fuzzy*, para estes termos.

$\tilde{M}$ (velho) = {(u, $\mu_{velho}(u)$) | $u \in$ [0, 100]}, onde,

$$\mu_{velho}(u) = \begin{cases} 0 & \text{para } u \in [0,\ 50] \\ \left(1 + \left(\dfrac{u-50}{5}\right)^{-2}\right)^{-1} & \text{para } u \in [50,\ 100] \end{cases}$$

$T(x)$ define o conjunto de termos da variável x, que, neste caso, é:

T(Idade) = {*velho*, *muito velho*, *não tão velho*, *mais ou menos jovem*, *inteiramente jovem*, *muito jovem*}

onde $G(x)$ é uma regra que gera os rótulos dos termos no conjunto de termos, conforme a *Figura III.3* abaixo.

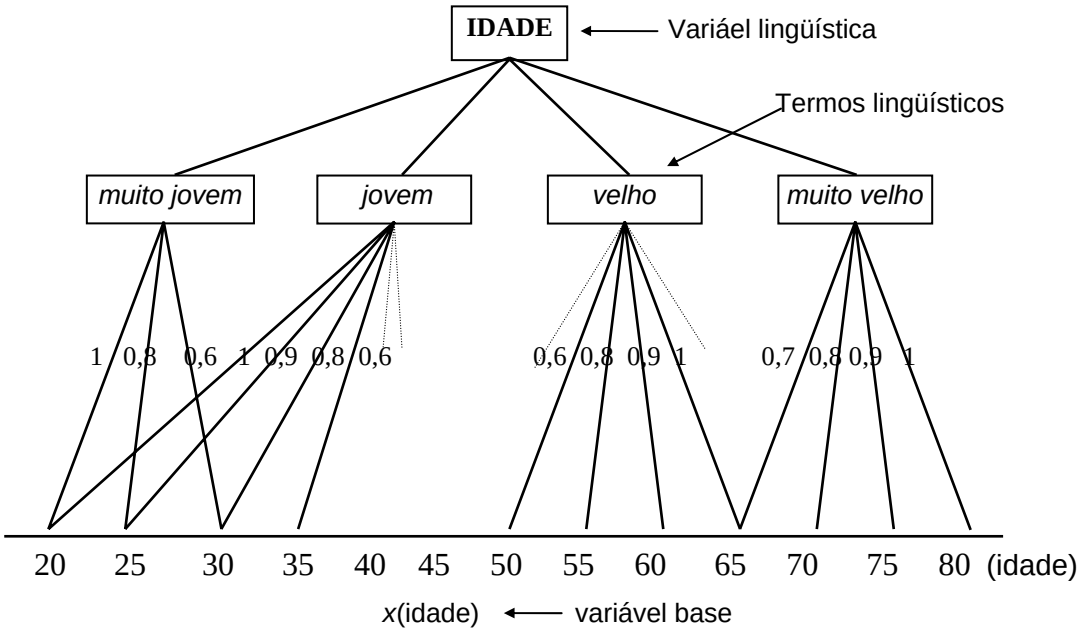


Figura III.3: Variável lingüística "*Idade*" (ZADEH, 1973b)

Portanto, cada variável lingüística, definida em termos de uma *variável base*, tem seu estado denotado por *termos lingüísticos*, que são interpretados como números *fuzzy* específicos. Os termos lingüísticos representam valores aproximados de uma variável lingüística, relacionados a uma aplicação particular. Uma variável base é uma variável no sentido clássico, exemplificada por uma variável física (temperatura, pressão, velocidade) ou por uma variável numérica (desempenho, confiabilidade, probabilidade) (KLIR *et al*., 1995a).

ZADEH (1977) observou que os *termos lingüísticos* podem ser modelados através de funções, cujos valores são graus no domínio de uma *função de pertinência*, e que cada representação é fundamental para a modelagem do raciocínio aproximado.

Limitadores lingüísticos são termos lingüísticos especiais, que modificam outros termos lingüísticos, como por exemplo, *menos*, *muito*, *razoavelmente*, *fracamente* e *extremamente*. Qualquer limitador lingüístico pode ser interpretado como uma operação unária (ou modificadora), *h*, em um intervalo unitário [0, 1]. Por exemplo, o limitador *muito* é freqüentemente interpretado como uma operação unária $h(a) = a^2$, enquanto que *razoavelmente* é interpretado como $h(a) = \sqrt{a}$ ($a \in [0, 1]$). Conhecendo-se o significado de um termo lingüístico e de sua operação modificadora, pode-se estabelecer *regras semânticas*, que traduzam o significado desse termo.

BALDWIN (1979), baseado também nos conceitos acima, delineou um conjunto de termos lingüísticos da variável lingüística "*Verdade*", mostrado na *Figura III.4*.

$$\text{verdadeiro} = \{(v, \mu_{\text{verdadeiro}}(v) = v) \mid v \in [0, 1]\}$$

$$\text{falso} = \{(v, \mu_{\text{falso}}(v) = 1 - \mu_{\text{verdadeiro}}(v)) \mid v \in [0, 1]\}$$

$$\text{muito verdadeiro} = \{(v, (\mu_{\text{verdadeiro}}(v))^2) \mid v \in [0, 1]\}$$

$$\text{razoavelmente verdadeiro} = \{(v, (\mu_{\text{verdadeiro}}(v))^{1/2}) \mid v \in [0, 1]\}$$

$$\text{indeciso} = \{(v, 1) \mid v \in [0, 1]\}$$

Muito falso e razoavelmente falso são definidos semelhantemente, e

$$\text{absolutamente verdadeiro} = \{(v, \mu_{av}(v)) \mid v \in [0, 1]\},$$

$$\text{onde } \mu_{av}(v) = \begin{cases} 1 & \text{para } v = 1 \\ 0 & \text{outros} \end{cases}$$

absolutamente falso = $\{(v, \mu_{af}(v)) \mid v \in [0, 1]\}$,

onde, $\mu_{af}(v) = \mu_{af}(v) \begin{cases} 1 & \text{para } v = 0 \\ 0 & \text{outros} \end{cases}$

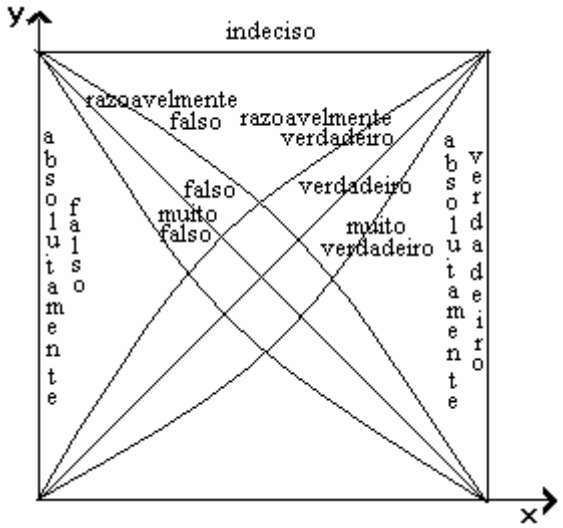


Figura III.4: Variável lingüística "*Verdade*" (BALDWIN, 1979)

Os termos lingüísticos podem ser, também, caracterizados por *quantificadores fuzzy* que são, em geral, números *fuzzy*, estendidos do escopo da lógica *fuzzy* (ZADEH, 1973a, VIOT, 1993), sendo usados para representar uma certa maioria, aproximando-se da percepção humana real (KACPRZYK *et al*., 1992, NOVÁK, 1992).

III.6 A Lógica *Fuzzy*

A lógica *fuzzy* (difusa) é uma extensão da lógica (*boleana*) convencional, que lida com conceitos de verdade parcial - valores verdade entre '*completamente verdade*' e '*completamente falso*' - modelando as incertezas da linguagem natural (KANTROWITZ *et al*., 1996). Neste contexto, uma decisão, tida como correta, poderá ser alterada posteriormente, quando novas informações adicionais estiverem disponíveis (CHIUEH *et al*., 1992).

A lógica *fuzzy* utiliza *predicados fuzzy* (*velho*, *raro*, *perigoso*, etc.), *quantificadores fuzzy* (muito, pouco, quase tudo, usualmente, e semelhante), *valores-verdade fuzzy* (totalmente verdadeiro, mais ou menos verdadeiro, muito falso, etc.) (ZIMMERMAN, 1991).

Os quantificadores *fuzzy*, de particular interesse para os termos lingüísticos *fuzzy*, podem ser (BOSC *et al*., 1995):

i. *absolutos*: definidos em $\mathbb{R}$ e denotados por um número, como, por exemplo, "*no mínimo 7*";

ii. *relativos*: definidos no intervalo [0,1], referindo-se a uma proposição quantificada lingüisticamente (YAGER, 1983a, YAGER, 1983b, ZADEH, 1983) como, por exemplo, "*muitos especialistas são convincentes*".

Uma proposição quantificada lingüisticamente pode ser apresentada como (KACPRZYK *et al*., 1992):

Q y's são F,

Q é um *quantificador lingüístico* (por exemplo, *muitos*), $Y = \{y\}$ é um *conjunto de objetos* de um conjunto universo (por exemplo, *especialistas*), e *F* é uma *propriedade*, isto é, um predicado *fuzzy* (por exemplo, *convincentes*).

Pode-se atribuir a *y* particulares (objetos) uma importância diferente (relevância, competência, ...), *B*, e adicioná-la à equação anterior, gerando:

Qby's são F.

No caso da importância (um predicado *fuzzy*) ser adicionada, *B* é definido como um conjunto *fuzzy* em *Y* e $\mu_{\tilde{B}}(y_i) \in [0, 1]$ é um grau de pertinência de y_i. Assim sendo, o valor de pertinência $\mu_{\tilde{B}}(y_i)$ também é definido como a possibilidade de *B* assumir o valor y_i. Por este motivo, a lógica *fuzzy* é chamada também de teoria da possibilidade.

A *teoria da possibilidade* (ZADEH, 1978) está, intrinsecamente, ligada à linguagem natural, onde a "*possibilidade*" é melhor interpretada do que a "*probabilidade*". No entanto, a possibilidade não substitui a probabilidade - ambas lidam com incertezas e se complementam entre si. Observa-se que um alto grau de possibilidade não implica em um alto grau de probabilidade, todavia, se um evento não é possível, também é improvável, isto é, a possibilidade é o limite superior da probabilidade (ZIMMERMANN, 1991).

A lógica *fuzzy* possui, também, alguns operadores básicos *fuzzy*, definidos no intervalo de 0 (falso) a 1 (verdadeiro), pelas operações binárias: *fuzzy* AND (f-AND),

fuzzy OR (f-OR), *probabilidade* AND (p-AND) e *probabilidade* OR (p-OR), e uma operação unária, NOT (RICHARDS, 1988):

i. a f-AND b = min (a, b);

ii. a f-OR b = max (a, b);

iii. a p-AND $b = a \times b$;

iv. a p-OR $b = a + b - (a \times b)$;

v. NOT $a = 1 - a$.

Várias aplicações da teoria *fuzzy* envolvem o uso de uma base de regras *fuzzy* para modelos complexos, como os sistemas de controle de lógica *fuzzy*, os sistemas de controle *fuzzy* multivariável (GEGOV, 1995) e os sistemas adaptativos (GÜRMAN, 1995).

Todos os sistemas de controle *fuzzy* (HERTZ *et al*., 1991, KAUFMANN *et al*., 1991, LEE, 1990, NEGOITA, 1985, ZADEH, 1973b) são baseados em regras, e se aplicam quase que, exclusivamente, a sistemas de produção e controle (DUBOIS, 1989, GRAHAM *et al*., 1988). Geralmente, essas regras não são extraídas de especialistas humanos através do sistema, mas produzidas, explicitamente, por seus projetistas.

A idéia básica é incorporar a "*experiência*" de um processo executado por um operador humano, no projeto de um controlador. De um conjunto de regras lingüísticas, que descrevem a estratégia de controle dos operadores, um algoritmo de controle é construído, onde seus termos são definidos como conjuntos *fuzzy* (KICKERT *et al*., 1978).

Portanto, a lógica *fuzzy* simula o pensamento humano, incorporando sua inerente falta de precisão a um sistema físico. A incerteza é o resultado da imprecisão não aleatória de medições ou da impossibilidade de se obter uma descrição numérica exata para quantidades observadas (NICULESCU *et al*., 1992).

III.7 A Teoria das Incertezas e Aplicações

A incerteza está relacionada com o ambiente e aumenta com a inerente imprecisão de uma classificação ou descrição específica. Muitas entidades possuem medidas com acurácia limitada, imprecisões, falta de informação, de observação e de sensibilidade

das condições iniciais. O termo "*incerteza*" pode ser comparado ao termo "*aleatório*", sendo que, nos conjuntos *fuzzy*, a incerteza refere-se ao grau de pertinência de um elemento, enquanto que, estatisticamente, a incerteza refere-se à probabilidade do resultado de um experimento aleatório (ROTHMAN, 1989).

Existem três princípios de incertezas, aplicáveis somente nas situações, onde emergem deficiências de informações, dependendo da teoria pela qual a incerteza é conceitualizada (KLIR, 1995b):

i. princípio de incerteza mínima: seleção das alternativas mais importantes do conjunto solução, reduzindo-se, tanto quanto possível, o número das informações iniciais.

ii. princípio da incerteza máxima: procura-se usar todos os dados disponíveis, evitando-se, porém, que informações adicionais sejam inadvertidamente adicionadas.

iii. princípio da invariância da incerteza: requer que uma quantidade de incertezas (e de informações) sejam preservadas, independentemente da representação teórica usada, isto é, garante que informações não sejam inapropriadamente adicionadas ou eliminadas, somente pela troca da estrutura matemática, na qual um fenômeno particular foi formalizado.

Devido à conexão íntima entre incerteza e informação, esses princípios podem ser concebidos, também, como *princípios de informação*. A quantidade de incertezas pode ser reduzida pela obtenção de novas informações relevantes, como o resultado de algumas ações (observação de um fato novo, execução de um experimento, obtenção de um registro histórico) (KLIR, 1995b).

O "*gerenciamento das incertezas*" desempenha um importante papel nos *sistemas baseados em conhecimento* (SBCs) *fuzzy*, e nos *sistemas de tomada de decisão em ambientes fuzzy*, apresentados a seguir.

III.7.1 Sistemas Baseados em Conhecimento Fuzzy

Várias propostas têm sido publicadas sobre como se usar a teoria *fuzzy* em SBCs (SIMONELLI, 1996, FICKAS *et al*., 1992, CHAN, 1991, HULL *et al*., 1991, DUBOIS *et al*., 1988, ZIMMERMANN, 1988, ZADEH, 1973a). Essa propostas têm, em comum,

a preocupação de que esses sistemas sejam providos com mecanismos, que manuseiem incertezas, otimizando o seu desempenho (LEE, 1994).

A incerteza das informações na base do conhecimento, por exemplo, induz incertezas nas conclusões. Assim sendo, a máquina de inferência (LESMO *et al*., 1982) deve ser equipada com capacidade computacional, para analisar a transmissão de incertezas das premissas para as conclusões, associando-as a algumas medidas de incertezas, que sejam inteligíveis e interpretadas convenientemente pelo usuário (ZIMMERMANN, 1991).

Um dos principais benefícios de um SBC *fuzzy* é sua habilidade de usar e assimilar o conhecimento de múltiplos especialistas, outorgando-lhes uma expressividade não existente em sistemas convencionais de suporte a decisão (COX, 1995). Tanto os SBCs *fuzzy*, quanto os sistemas de controle *fuzzy*, objetivam modelar a experiência e o ambiente humano de tomada de decisões (CARCHIOLO *et al*., 1995, SUGENO, 1985, ZADEH, 1973a).

III.7.2 Tomada de Decisão em Ambientes *Fuzzy*

Um dos problema de tomada de decisão consiste na escolha da melhor alternativa de acordo com critérios estabelecidos, a partir de uma certa quantidade de informações, com o propósito de atingir um objetivo estabelecido (GRABISCH, 1995, SLOWINSKI *et al*., 1990, TURBAN, 1988). Atualmente, a multidimensionalidade é a principal característica dos problemas de tomada de decisão do mundo real, tendo objetivos econômicos, ambientais, sociais e técnicos (SAKAWA, 1994).

Desde que a teoria dos conjuntos *fuzzy* foi sugerida como uma estrutura conceitual apropriada de tomada de decisão (BELLMANN, 1970), duas direções relevantes podem ser observadas. A primeira reflete a *fuzificação* de enfoques convencionais. A segunda está baseada na assunção de que o processo de tomada de decisão pode ser modelado por operadores de agregação especificados automaticamente (DUBOIS, 1984).

Tipicamente, três formas de imprecisão podem ser identificadas em tomada de decisão em ambientes *fuzzy* (RIBEIRO *et al*., 1995):

i. Não completitude: quando não há dados suficientes como, por exemplo, ausência de alguns atributos ou alternativas;

ii. Imprecisão: quando há dificuldades na obtenção de conceitos precisos para melhor caracterizar atributos ou critérios;

iii. Ilusão da validade (TVERSKY *et al*., 1990): detecção de saídas errôneas, tais como a seleção de alternativas, que não cumpram os critérios impostos.

Na tomada de decisão em ambientes *fuzzy*, termos como multiobjetivos, multiatributos e multicritérios são, geralmente, usados indistintamente, embora haja diferenças entre eles (RIBEIRO, 1996, KOSKO, 1992 CHEN *et al.* 1992):

i. Tomada de decisão multiobjetivos (MODM) (ZIMMERMANN, 1991): consiste de um conjunto de objetivos conflitantes, que não podem ser alcançados simultaneamente.

ii. Tomada de decisão multiatributos (MADM) (YAGER, 1978, HWANG, 1981): escolha de uma alternativa em um conjunto de alternativas, caracterizada por seus atributos.

iii. Tomada de decisão multicriterial (MCDM): aplicada tanto à tomada de decisão, envolvendo multiobjetivos, quanto multiatributos (CARLSSON *et al*., 1996, FODOR *et al*., 1994, CHEN *et al*., 1993, LUHANDJULA, 1989,). Neste caso, algumas considerações importantes foram feitas (ZELENY, 1992):

- *a pressão do tempo reduz o número de critérios a serem considerados*;
- *quanto mais completa e precisa for a definição do problema, menos critérios são necessários*;
- *indivíduos que tomam decisão em sistemas estritamente hierárquicos, geralmente utilizam menos critérios do que indivíduos, que lidam com outros tipos de sistemas*;
- *o isolamento de perturbações no ambiente, reduz a necessidade de múltiplos critérios*;
- *o conhecimento maior (ou completo) e integrado do problema leva à utilização de mais critérios, enquanto que o conhecimento parcial (ou limitado) e não integrado restringe o número de critérios*;

- *organizações com cultura voltada para o planejamento central e tomadas de decisões coletivas, apoiam-se na agregação e na redução de critérios, para alcançar um consenso.*

Um dos elementos básicos na tomada de decisão de grupo é o conceito de *maioria*, isto é, a solução encontrada destaca a opinião mais aceitável pela maioria dos membros do grupo. Uma maioria menos rígida (uma concordância geral, sem a necessidade de uma inferência individual) pode auxiliar, certamente, a formação de modelos de decisão de grupo mais consistentes e humanizados (KACPRZYK, 1992).

Um outro elemento empregado, habitualmente, nas ciências de decisão é a *média de pesos*, através de parâmetros quantificáveis. Na coleta de informações, busca-se a estimativa do avaliador, que esteja mais próxima do modelo de requisitos. Sendo assim, as estimativas coletadas e a apuração de seus resultados são essenciais neste processo e, sem isto, a avaliação poderia tornar-se irrealista (ROBINS, 1995).

Um modelo *fuzzy* de decisão adequado deve incluir processos de identificação, medição e combinação de critérios e alternativas, promovendo a modelagem conceitual, da decisão e a avaliação em ambientes nebulosos (RIBEIRO, 1996). A seguir, serão apresentados alguns modelos *fuzzy* de decisão, encontrados na literatura.

III.7.3 Modelos *Fuzzy* de Decisão

Tem sido, freqüentemente, evidenciado que a relevância (ou a qualidade) de um modelo influencia fortemente a qualidade da solução. Se o método não é relevante, pode ser inapropriado continuar a busca de soluções do problema em questão, ou ser necessário reformulá-lo com algumas extensões. As três etapas, que se sucedem, podem auxiliar na tarefa de identificação da relevância de um modelo *fuzzy* (PEDRYCZ, 1990):

i. aquisição e determinação de dados requeridos pela estrutura do modelo;

ii. estimação de parâmetros;

iii. validação do modelo *fuzzy*.

No entanto, embora aferida a qualidade do método escolhido, há outros fatores que podem conduzir a inconsistências de seus resultados (DYER, 1992):

i. a ausência de informações ou de experiências sobre o objeto avaliado;

ii. o desinteresse ou a falta de concentração dos avaliadores, no processo de julgamento.

Portanto, é de grande importância selecionar com acuidade os avaliadores, conduzir com destreza todo o processo de aquisição das opiniões dos especialistas, seja individualmente ou em reuniões próprias para a tomada de decisão.

Muitos modelos de estimativas, fundamentados na teoria *fuzzy*, efetuam alguns procedimentos básicos, que podem ser dispostos em três estágios (SCHMUCKER, 1984):

i. Primeiro estágio: tradução de expressões da linguagem natural para conjuntos *fuzzy*, através de operações *fuzzy* correspondentes.

ii. Segundo estágio: agregação desses conjuntos *fuzzy* em um conjunto resultante, que reproduza o sistema, que está sendo medido, como um todo.

iii. Terceiro estágio: associação do conjunto *fuzzy* resultante a expressões naturais convenientes, que definam o sistema apropriadamente.

Vários modelos *fuzzy* de estimativas, apoiaram-se de algum modo na estrutura do *método de análise hierárquica* (SAATY, 1980). Portanto, embora esse método não utilize a teoria *fuzzy*, será apresentado sucintamente, juntamente com outros, que se servem dessa teoria.

III.7.3.1 Método de Análise Hierárquica

O *Método de Análise Hierárquica* (SAATY, 1980) parte de uma situação não estruturada e complexa, decompondo-a em partes ordenadas hierarquicamente e, através de julgamentos sucessivos, obtém a melhor solução para o problema. Esse método combina o uso de duas abordagens: (*i*) a *abordagem dedutiva*, que decompõe o sistema em partes, para a melhor compreensão do comportamento e funcionamento das mesmas; (*ii*) a *abordagem de sistema*, que busca analisar o comportamento do sistema, não importando suas partes constituintes.

Esse método utiliza uma escala de proporções, através de comparações de pares de elementos em relação a um dado critério imediatamente superior, partindo do topo da hierarquia. Para m critérios, C_1, C_2, ..., C_m, é construída uma escala de proporção, que evidencia a importância relativa de cada critério em relação a cada um dos outros. Isto é feito através do julgamento de especialistas, que se baseiam em uma escala de números,

para representar a importância relativa de um elemento sobre outro. Os julgamentos quantificados em pares de critérios C_i e C_j são reproduzidos em uma matriz $m \times m$:

$$M = \{a_{ij}: i = 1, 2, ..., m; j = 1, 2, ..., m\}$$

As entradas dessa matriz são definidas pelas seguintes regras:

- Regra 1: se $a_{ij} = \beta$ então $a_{ji} = 1/\beta$, onde $\beta \neq 0$.
- Regra 2: se C_i é julgado ser da mesma importância relativa de C_j, então $a_{ij} = a_{ji}$.

Todos os elementos diagonais $a_{ii} = 1$ para 1, 2, ..., m.

A prioridade obtida da comparação de cada par de elementos em relação a um critério ascendente é denominada *prioridade local*. A prioridade de um elemento em relação ao objetivo geral é chamada de *prioridade global*.

Em síntese, pode-se afirmar que uma das principais vantagens desse método é que busca um resultado final, através da síntese de julgamentos de vários participantes, não insistindo no consenso entre eles.

III.7.3.2 Método *Fuzzy* Hierárquico de Avaliação

LASEK (1992) propôs um método *fuzzy* hierárquico para a avaliação e a classificação de problemas com múltiplos atributos, isto é, estruturas hierárquicas de avaliação *fuzzy* para a análise de alvos estratégicos de empresas. No topo da hierarquia desse método, estão os objetivos mais gerais, que são decompostos em subobjetivos e estratégias, nos níveis mais inferiores.

Nesta metodologia, a imprecisão de dados é representada por conjuntos *fuzzy*, onde os números reais são tratados como um caso especial de números *fuzzy*, com grau de pertinência igual a um e exprimem prioridade, classificação ou o resultado de uma avaliação. Dependendo dos relacionamentos existentes entre objetivos, subobjetivos e estratégias é usado o enfoque da *dependência funcional* ou da *relação de preferência*.

Na dependência funcional, os julgamentos feitos pelos especialistas são realizados nos níveis mais baixos da hierarquia. As avaliações dos outros níveis da hierarquia processam-se através da execução de operações algébricas *fuzzy*. Nas relações de preferências, são indicadas somente as prioridades (pesos) dos objetivos, subobjetivos e estratégias e, para isto, é empregado o Método de Análise Hierárquica (SAATY, 1980).

III.7.3.3 Método de Agregação de Similaridades

HSU *et al*. (1996) combina opiniões individuais para formar uma opinião consensual de um grupo de especialistas, através de um método de agregação de similaridade. As opiniões dos especialistas são representadas por números *fuzzy* trapezoidais, assumido que estes têm uma interseção comum, em um conjunto de corte de nível-α, onde $\alpha \in (0, 1]$. Esta é uma condição imposta por esse método, para que a agregação dos resultados das opiniões dos especialistas seja aceitável.

Se não houver interseção entre as estimativas iniciais do *k*-ésimo e do *l*-ésimo especialista é usado, então, o Método Delphi, para auxiliar na obtenção de mais informações, objetivando ajustar os dados fornecidos por cada especialista, para que haja essa interseção.

Posteriormente, é introduzida uma função de medida de similaridade, para medir o grau de concordância entre as opiniões dos especialistas, e estas informações são postas em uma matriz de concordância. Finalmente, as opiniões dos especialistas são combinadas, podendo-se levar em consideração, também, a importância de cada especialista participante do processo de avaliação.

III.7.3.4 Modelo Hierárquico para Estruturas de Risco

LEE (1996a) desenvolveu uma estrutura de tomada de decisão de grupo, considerando os riscos no desenvolvimento de software. Partiu do princípio de que a avaliação de custos de software é caracterizada p ela alta incerteza, existindo muitos problemas ao longo do processo de desenvolvimento, como a postergação de cronograma, o aumento de despesas, a ineficiência e o abandono de projeto.

Nessa estrutura, a teoria *fuzzy* é utilizada para manusear, efetivamente, as ambigüidades envolvidas na avaliação de taxas de fatores de risco e para lidar com propriedades vagas, através de expressões lingüísticas. Para isto, os números *fuzzy* triangulares foram empregados, para representar a imprecisão dos dados quantitativos do modelo.

Um grupo de *n* especialistas é selecionado, para a avaliação da taxa de agressividade do risco de um projeto de software em desenvolvimento. O processo de agregação dos julgamentos desses especialistas é promovido de duas maneiras: (*i*)

através da média desses julgamentos, para produzir o resultado final; (*ii*) a agregação dos julgamentos individuais de cada especialista, para a obter a média dos resultados.

Esse modelo compõe-se de um conjunto de atributos que, por sua vez, são compostos por itens de risco. Cada atributo, juntamente com seus itens de risco, possui um grau de risco e o grau de importância (peso), que podem ser ajustados até que o resultado esperado seja alcançado.

III.7.3.5 Modelo para Localização Industrial

O *modelo para localização industrial* objetiva facilitar a estruturação de um sistema de informações, que possibilite ao Governo a formulação conveniente de políticas de desenvolvimento industrial. Esse modelo está baseado na teoria *fuzzy*, onde a análise de alternativas se apoia na construção de hierarquia ponderada para cada proposição (COSENZA, 1981).

Esse modelo foi adaptado por CAMPOS (1994b) para a área de qualidade de software educacional, visando confrontar, sistematicamente, a demanda de fatores de qualidade do produto de software (*matriz de demanda*), através de critérios selecionados e validados por professores, para a avaliação e a oferta desses produtos (*matriz de oferta*).

Através dos dados das matrizes acima, foram geradas: (*i*) a *matriz de possibilidade de alocação*, descrevendo, em valores absolutos, o grau de satisfação dos usuários por produto de software, e (*ii*) a *matriz de resultados*, contendo os índices de alocação, que representam o resultado final do experimento, evidenciando o grau de satisfação dos usuários por cada produto analisado.

III.7.3.6 Outros Modelos

ENGEMANN *et al.* (1997) propuseram um método geral para seleção de uma alternativa debaixo de incertezas para casos do mundo real, envolvendo riscos de gerenciamento.

PRESEDO (1996) apresenta um modelo *fuzzy* de um sistema especialista, na detecção de isquemia baseada em eletrocardiograma, para resolver o problema de transformações do conhecimento impreciso, fornecido pelos especialistas, em estruturas computáveis.

COMMERLATO (1994) desenvolveu um modelo e uma ferramenta de software, para a identificação de componentes FORTRAN, pretensos à reutilização, segundo medidas obtidas através de métricas de software e seleção através de variáveis lingüísticas e termos lingüísticos *fuzzy* predeterminados. Definiu, também, um processo de avaliação de candidatos à reutilização em três etapas: (*i*) medição, (*ii*) definição de variáveis e termos lingüísticos (representam critérios de qualidade), e (*iii*) seleção de candidatos, através dessas variáveis.

GARCIA *et al*. (1992) formaliza uma *metodologia de avaliação lingüística em situações de risco*, usando técnicas *fuzzy*. Neste método, o cálculo do risco global de um sistema, estruturado hierarquicamente, é obtido através da avaliação do risco de cada um de seus componentes, em termos de seus descendentes.

RUONING *et al*. (1992) propõe um método de tomada de decisão de grupo, em sistemas complexos, envolvendo uma grande variedade de fatores, e constrói uma matriz de julgamento *fuzzy*, através de um método estatístico, provando que cada elemento dessa matriz pode ser representado por um número *fuzzy*. Segundo ele, teóricos de sistemas argumentam que relacionamentos complexos podem sempre ser analisados, tomando-se pares de atributos e relacionando-os entre si.

Em KACPRZYK *et al*. (1992) e TANINO (1984) foi proposta uma relação de preferência *fuzzy* para cada especialista, da qual é derivada a relação *fuzzy* de preferência para o grupo de especialistas, no sentido de determinar a melhor alternativa para a solução do problema.

ISHIKAWA *et al*. (1993) e XU *et al*., (1992) propuseram que cada especialista representasse o julgamento de critérios avaliados por eles, através de um grau de importância obtido de um intervalo numérico *fuzzy*. A partir deste resultado, é gerada uma distribuição de freqüência cumulativa, para derivar o julgamento consensual do grupo.

BARDOSSY *et al* (1993) sugeriu que as estimativas de critérios, avaliadas por especialista, fossem representadas por números *fuzzy* e combinadas através de operações *fuzzy*.

A teoria *fuzzy* tem sido aplicada com sucesso em muitas áreas de pesquisas de modelos e em diversos estágios de processos de pesquisas, como, por exemplo,

reconhecimento de padrões (KUMAR *et al*. 1996, SEONG, 1995, PAL *et al*. 1992), surgindo os chamados *sistemas fuzzy*.

III.8 Os Sistemas *Fuzzy*

Com o emprego de sistemas *fuzzy* (COX, 1995, DRIANKOV *et al*. 1993, KANDEL *et al*. 1993) de forma apropriada, julga-se ter produzido respostas mais rápidas e "*suaves*" que os sistemas convencionais. A maneabilidade, a robustez e, sobretudo, o baixo custo são fatores de qualidades característicos dos sistemas *fuzzy*, contribuindo para um melhor desempenho dos mesmos. São úteis para problemas ou aplicações complexas, que envolvam descrições humanas ou pensamento indutivo. A *Tabela III.2* mostra suas principais áreas de aplicação.

Categorias	**Aplicações**
Controle	utilizado em larga escala em aplicações industriais
Reconhecimento de Padrões	processamento de imagem, áudio e sinal
Análise Quantitativa	pesquisa operacional, estatística e gerenciamento
Inferência	sistemas especialistas para diagnóstico, planejamento e predição
	processamento de linguagem natural
	interfaces inteligentes
	robôs inteligentes
	engenharia de software
Recuperação de Informações	base de dados

Tabela III.2: Categorias genéricas de aplicações de sistemas *fuzzy* (MUNAKATA *et al*., 1994)

Os principais passos para o desenvolvimento de um sistema *fuzzy* são (MUNAKATA *et al*., 1994):

i. Constatação de que o conhecimento sobre o ambiente da aplicação em questão é descrito de forma aproximada ou com regras heurísticas.

ii. Identificação das entradas e saídas do sistema e seus respectivos intervalos de valores.

iii. Definição de uma função para cada parâmetro de entrada ou saída. A quantidade de funções requeridas depende do desenvolvedor e do ambiente do sistema.

iv. Construção de uma base de regras pelo projetista, que determine quantas regras serão necessárias e quando parar de adicionar novas regras.

v. Verificação das saídas da base de regras e se seus intervalos de valores estão corretos e em conformidade com o conjunto de entradas usado, permitindo uma posterior validação.

Vários métodos têm sido utilizados para o desenvolvimento e a implementação de sistemas *fuzzy*. Os mais comuns são: o *método de desenvolvimento baseado em especialistas humanos*, e o *método de desenvolvimento baseado em tentativa e erro*.

Os principais problemas e limitações de sistemas *fuzzy* são (MUNAKATA *et al*., 1994):

- *Estabilidade*: não há garantias teóricas de que um sistema *fuzzy* não venha a atingir um estado caótico e que seja estável. No entanto, a experiência tem mostrado que uma expressiva maioria dos sistemas *fuzzy* são estáveis.

- *Capacidade de aprendizagem*: falta-lhes a capacidade de aprendizagem e não possuem memória do que foi especificado previamente. Atualmente, sistemas híbridos, particularmente os sistemas *neurofuzzy* vêm suprindo esta limitação.

- *Determinação ou refinamento de funções e regras fuzzy* apropriadas: mesmo depois de muitos testes, pode ser ainda difícil o estabelecimento conveniente das funções *fuzzy* requeridas.

- *Conceituação*: há ainda um entendimento errôneo do termo *fuzzy*, como significando imprecisão ou imperfeição, e como se não tivesse uma fundamentação matemática sedimentada.

- *Verificação e validação*: geralmente requerem testes extensivos.

No entanto, os benefícios intrínsecos na utilização de sistemas *fuzzy* excedem a suas limitações e seus problemas e podem ser sumariados na redução do custo do desenvolvimento, da execução e da manutenção da aplicação (COX, 1995).

III.9 Conclusão

Neste capítulo, apresentou-se a teoria *fuzzy* em seus aspectos mais gerais, sendo que alguns temas foram mais aprofundados, objetivando-se uma melhor fundamentação para o desenvolvimento de um modelo *fuzzy* para a avaliação da qualidade de software, o que será feito no próximo capítulo.

Capítulo IV

MODELO *FUZZY* PARA AVALIAÇÃO DA QUALIDADE DE SOFTWARE

Uma estrutura de avaliação de software tem por objetivo estimar a sua qualidade, através de um conjunto básico de atributos, evidenciando seus aspectos mais relevantes. Para isto, as informações sobre o objeto da avaliação devem estar dispostas de forma organizada, onde características específicas do software podem ser identificadas acurada e tempestivamente, para otimizar o processo de tomada de decisão (BOLOIX *et al*., 1995).

A tomada de decisão pode ser vista como um processo de seleção de alternativas "*suficientemente boas*" ou da escolha de cursos de ações, para a obtenção de um objetivo. Esse processo envolve incertezas e, portanto, é necessário que se tenha a habilidade de manusear informações imprecisas e vagas, levando-se em consideração diferentes visões, atitudes e crenças das partes envolvidas (RIBEIRO, 1996). Portanto, é importante fazer a conexão entre a informação e a decisão de um indivíduo, que deve escolher uma ação entre muitas possíveis, em áreas distintas (SIMONELLI, 1996).

Uma vez que o processo de tomada de decisão é centrado em pessoas humanas, como também o é o processo de avaliação de software, com suas inerentes subjetividades e inconsistências na definição do problema, os conjuntos *fuzzy* são potencialmente adequados nesta área, pois (IBRAHIM *et al*., 1992):

i. possuem a habilidade de representar atributos;

ii. detêm formas convenientes e avaliáveis para a combinação de atributos, que podem estar vaga ou precisamente definidos;

iii. manuseiam diferentes graus de importância para cada atributo considerado.

Como em muitas outras áreas do conhecimento humano, a avaliação da qualidade de software envolve a apreciação de múltiplos atributos, através do julgamento de um grupo de especialistas (SANDRI, 1997). Cada especialista tem a sua própria opinião e estima um grau de importância para cada atributo julgado, segundo sua percepção ou o seu nível de entendimento da questão proposta. Daí o interesse de se conseguir um processo de agregação, que consolide o consenso dos especialistas envolvidos, na avaliação.

No processo de avaliação da qualidade de software não basta, apenas, identificar que atributos determinam essa qualidade, mas também que procedimentos adotar, para controlar seu processo de desenvolvimento de forma a atingir o nível de qualidade desejado. Isto é realizado através da aplicação de métricas de forma organizada e projetada, tornando os desenvolvedores mais conscientizados da relevância do gerenciamento e dos compromissos para com a qualidade (BELCHIOR, 1992b).

A medição de atributos de software tem estabelecido, por si mesma, um importante e incremental papel no entendimento e controle da prática do desenvolvimento de software. Portanto, é essencial usar métricas válidas e acuradas, que representem os atributos, que se pretende observar e quantificar (FENTON *et al*., 1997).

O uso de métricas de qualidade de software deve estar apoiado em um método de controle de qualidade, para que se alcance, convenientemente, os objetivos almejados (IEEE, 1987). Neste trabalho, o método escolhido para a avaliação da qualidade de software é o *Método Rocha* (ROCHA, 1983), por já ter sido utilizado satisfatória e eficazmente em uma grande quantidade de domínios de aplicação. Para alinhar a confiabilidade desse método às propriedades da teoria *fuzzy*, propõe-se uma extensão do mesmo, para suportar o Modelo *fuzzy* para Avaliação da Qualidade de Software.

IV.1. Extensão do Modelo Rocha

Como apresentado na *Seção II.2.2.1.3*, o *Modelo Rocha* para avaliação da qualidade de produtos de software, *Figura IV.1*, originalmente, define qualidade de software a partir dos seguintes conceitos:

1. *Objetivos da qualidade*: são as propriedades gerais, que o produto deve possuir.

2. *Fatores de qualidade*: determinam a qualidade na visão dos diferentes usuários do produto. Podem ser compostos por *subfatores*, quando estes não definem completamente, por si só, um objetivo.
3. *Critérios*: são atributos primitivos, possíveis de serem avaliados.
4. *Processos de avaliação*: determinam o processo e os instrumentos a serem utilizados, de forma a se medir o grau de presença, no produto, de um determinado critério.
5. *Medidas*: são o resultado da avaliação do produto, segundo os critérios.
6. *Medidas agregadas*: são o resultado da agregação das medidas, obtidas ao se avaliar de acordo com os critérios, e quantificam os fatores.

Os objetivos de qualidade são atingidos através dos fatores de qualidade, que podem ser compostos por subfatores. Objetivos, fatores e subfatores não são, diretamente, mensuráveis e só podem ser avaliados através de critérios. Um critério é um atributo primitivo, e pode qualificar diferentes subfatores ou fatores. Nenhum critério isolado é uma descrição completa de um determinado subfator ou fator. Da mesma maneira, nenhum fator define completamente um objetivo.

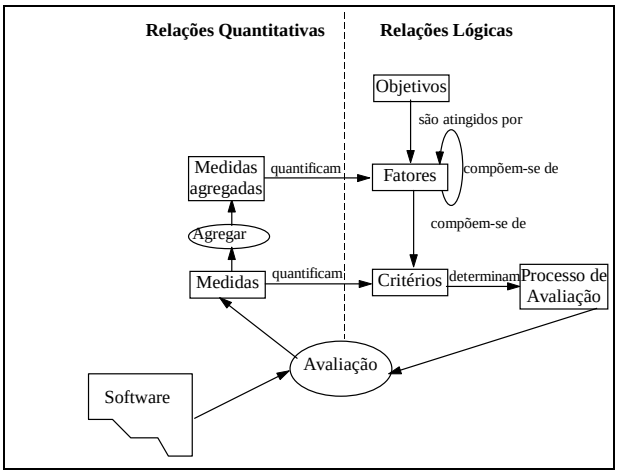


Figura IV.1: Modelo Rocha (ROCHA, 1983)

Este modelo não fornecia formas adequadas para realizar o processo de agregação, por ele apresentado. A proposta, que é descrita a seguir, vem resolver este problema.

Definiu-se, então, a extensão do *Modelo Rocha* através da utilização de conceitos e propriedades da teoria dos conjuntos *fuzzy* e, assim, o *Modelo Rocha Estendido* foi dotado da potencialidade dessa teoria em mapear modelos qualitativos de tomada de

decisão e da consistência dessa teoria no tratamento de incertezas e na agregação de informações. Neste contexto, alguns conceitos do Modelo Rocha foram estendidos (➢), outros foram acrescidos (✠) (*Figura IV.2*):

➢ *Medidas*: são o resultado da avaliação do produto, segundo os critérios, através de termos lingüísticos *fuzzy*, mapeados por números *fuzzy*.

➢ *Medidas agregadas*: são o resultado da agregação das medidas obtidas ao se avaliar de acordo com os critérios. São, também, o resultado da agregação de critérios em subfatores, fatores, objetivos, e no valor final do produto de software.

✠ *Funções fuzzy*: mapeiam os atributos de qualidade primitivos ou agregados, através do conjunto de termos lingüísticos estabelecido, quantificando-os.

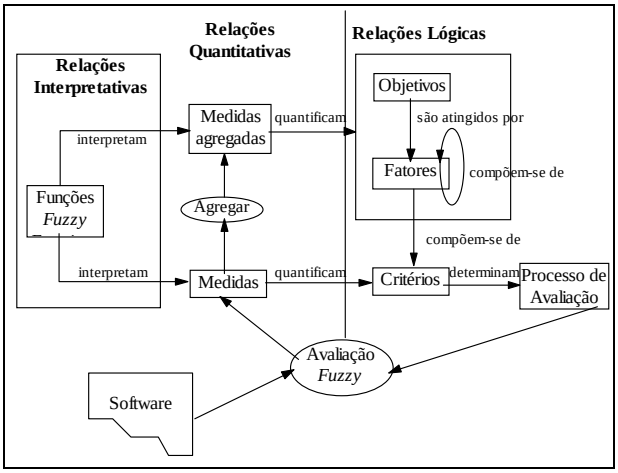


Figura IV.2: Modelo Rocha Estendido

A 'avaliação *fuzzy*' segundo o Modelo Rocha estendido será mostrada, a seguir, através de um *modelo fuzzy para avaliação da qualidade de software*.

IV.2 Uso da Teoria *Fuzzy* em Qualidade de Software

Na ótica da teoria *fuzzy*, cada atributo de qualidade de software pode ser visto como uma *variável lingüística*, relacionada a um conjunto de *termos lingüísticos*, associados a *funções de pertinência*, em um *conjunto referencial* estabelecido previamente. Cada critério de qualidade será uma composição de termos lingüísticos, obtidos em um processo de avaliação, feito através do julgamento de especialistas selecionados. Assim sendo, também serão números *fuzzy*. Por sua vez, os *atributos agregados*, constituídos por um subconjunto de critérios, serão também números *fuzzy*.

Os termos lingüísticos T_i, para $i = 1, 2, ..., n$, serão representados por *números fuzzy normais triangulares positivos* $\tilde{N}_i(a_i, m_i, b_i)$ do tipo-*LR* (LEE, 1996b, HSU, 1996, RÖMER *et al*., 1995, HAPKE *et al*., 1994, BARDOSSY *et al*., 1993, LASEK, 1992, RUONING *et al*., 1992, KAUFMANN *et al*., 1991, GENEST, 1984, DUBOIS *et al*., 1980), que denotarão o grau de importância de cada atributo considerado, onde $a_i < b_i$, embora possa se ter $a_i \leq m_i$ ou $m_i \leq b_i$.

O valor de *n* pode ser estabelecido de acordo com as conveniências do projeto, possíveis peculiaridades do domínio de aplicação ou determinação da equipe gestora da qualidade. Isto porque a quantidade de termos lingüísticos e eles próprios, a serem usados em uma aplicação específica, podem ser escolhidos, dependendo da situação em questão. Na medição do nível de pressão sangüínea, por exemplo, pode ser suficiente apenas os três termos lingüísticos: *alta*, *normal* e *baixa*, para que se consiga diagnosticar um paciente apropriadamente. Já na aplicação de conceitos a alunos universitários, por exemplo, muitas vezes são utilizados quatro termos lingüísticos: *excelente* (A), *bom* (B), *regular* (C) e *deficiente* (D).

Entretanto, uma vez estabelecido o valor de *n* e o conjunto de termos lingüísticos para o domínio de aplicação, ou para o objeto a ser apreciado, isto deve permanecer constante ao longo de todo o processo de avaliação. Nas situações apresentadas acima, não se poderia usar, por exemplo, as expressões: *pressão deficiente* (a pressão estaria *alta* ou *baixa*?), nem *aluno normal* (o conceito desse aluno seria A, B ou C?).

Sendo assim, baseado em HSU *et al*. (1996), LEE (1996a), KACPRZYK *et al*. (1992), BALDWIN (1979), ZADEH (1977), são sugeridos alguns conjuntos de termos lingüísticos, que poderão ser utilizados, na avaliação da qualidade de um produto de software.

A *Tabela IV.1* é mais abrangente, possuindo graus de pertinência no intervalo [0, 10], isto é, $\forall\ a_i, m_i, b_i \in [0, 10]$ e $n = 11$. Como 10 é um fator de 100, a utilização dessa tabela pode facilitar o processo de avaliação, isto é, tem-se uma visão percentual do objeto avaliado. Seja, por exemplo, o termo lingüístico *médio*, de grau de importância 5,0 que corresponderia a 50%.

A *Tabela IV.2* é uma simplificação da anterior, possuindo graus de pertinência no intervalo [0, 5], isto é, $\forall\ a_i, m_i, b_i \in [0, 5]$ e $n = 6$. Pode ser usada quando não houver necessidade de maiores detalhamentos dos termos lingüísticos básicos.

A *Tabela IV.3* possui graus de pertinência no intervalo [0, 4], isto é, $\forall\ a_i, m_i, b_i \in [0, 4]$ com $n = 5$. Os termos lingüísticos dessa tabela têm sido largamente utilizados em muitos trabalhos de pesquisa, especialmente na área de qualidade de software, utilizando-se o Modelo Rocha (CLUNIE, 1997, OLIVEIRA, 1995b). Por este motivo, é que essa tabela é empregada, no experimento realizado no próximo capítulo, para a validação do método *fuzzy*, que definimos na seção a seguir.

Grau de importância	**Simbologia**	**Termo Lingüístico**	**Número *fuzzy* normal**
0,0	N	Nenhum ou ausente	$\tilde{N}_1$= (0,0; 0,0; 1,0)
1,0	EB	Extremamente baixo	$\tilde{N}_2$= (0,0; 1,0; 2,0)
2,0	MB	Muito Baixo	$\tilde{N}_3$= (1,0; 2,0; 3,0)
3,0	B	Baixo	$\tilde{N}_4$ = (2,0; 3,0; 4,0)
4,0	LB	Ligeiramente Baixo	$\tilde{N}_5$ = (3,0; 4,0; 5,0)
5,0	M	Médio	$\tilde{N}_6$ = (4,0; 5,0; 6,0)
6,0	LA	Ligeiramente Alto	$\tilde{N}_7$ = (5,0; 6,0; 7,0)
7,0	A	Alto	$\tilde{N}_8$ = (6,0; 7,0; 8,0)
8,0	MA	Muito Alto	$\tilde{N}_9$ = (7,0; 8,0; 9,0)
9,0	EA	Extremamente Alto	$\tilde{N}_{10}$ = (8,0; 9,0; 10,0)
10,0	I	Imprescindível	$\tilde{N}_{11}$ = (9,0; 10,0;10,0)

Tabela IV.1: Números *fuzzy* normais para termos lingüísticos (abrangente)

Grau de importância	**Simbologia**	**Termo Lingüístico**	**Número fuzzy normal**
0,0	N	Nenhum ou ausente	$\tilde{N}_1$= (0,0; 0,0; 1,0)
1,0	MB	Muito Baixo	$\tilde{N}_2$= (0,0; 1,0; 2,0)
2,0	B	Baixo	$\tilde{N}_3$ = (1,0; 2,0; 3,0)
3,0	M	Médio	$\tilde{N}_4$ = (2,0; 3,0; 4,0)
4,0	A	Alto	$\tilde{N}_5$ = (3,0; 4,0; 5,0)

5,0	MA	Muito Alto	$\tilde{N}_6$ = (4,0; 5,0; 5,0)

Tabela IV.2: Números *fuzzy* normais para termos lingüísticos (simplificado)

Grau de importância	**Simbologia**	**Termo Lingüístico**	**Número *fuzzy* normal**
0,0	NR	Nenhuma Relevância	$\tilde{N}_1$= (0,0; 0,0; 1,0)
1,0	PR	Pouca Relevância	$\tilde{N}_2$= (0,0; 1,0; 2,0)
2,0	R	Relevante	$\tilde{N}_3$ = (1,0; 2,0; 3,0)
3,0	MR	Muito Relevante	$\tilde{N}_4$ = (2,0; 3,0; 4,0)
4,0	I	Imprescindível	$\tilde{N}_5$ = (3,0; 4,0; 4,0)

Tabela IV.3: Números *fuzzy* normais para termos lingüísticos (por relevância)

O conjunto dos termos lingüísticos das três tabelas propostas acima possuem as seguintes funções de pertinência, adaptadas de LEE (1996b):

$$\tilde{N}_1 = (0{,}0;\ 0{,}0;\ 1{,}0) \qquad \mu_{N_1}(x) = \begin{cases} 1 - x, & 0 \le x \le 1 \\ 0, & 1 \le x \le n \end{cases}$$

$$\tilde{N}_k = (k - 2;\ k - 1;\ k) \qquad \mu_{N_k}(x) = \begin{cases} 0, & 0 \le x \le k - 2 \\ x - (k-2), & k - 2 \le x \le k - 1 \\ k - x, & k - 1 \le x \le k \\ 0, & k \le x \le n \end{cases} \qquad \text{para } k = 2, \ldots, (n-1)$$

$$\tilde{N}_n = (n - 2;\ n-1;\ n-1) \quad \mu_{N_n}(x) = \begin{cases} 0, & 0 \le x \le n - 2 \\ x - (n-2), & n - 2 \le x \le n - 1 \end{cases}$$

Como neste trabalho será utilizado o conjunto de termos lingüísticos da *Tabela IV.3*, apresenta-se abaixo o gráfico de suas funções de pertinência, na *Figura IV.3*.

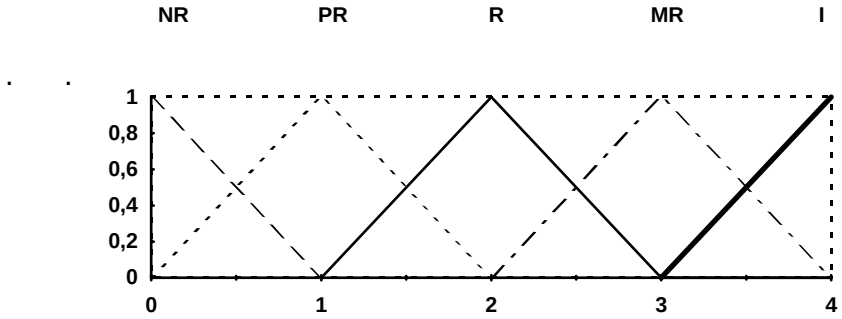


Figura VI.3: Funções de pertinência de números fuzzy, para termos lingüísticos

Depois de estabelecido o conjunto de termos lingüísticos e suas funções características, apresenta-se, a seguir, a etapas do **Modelo *Fuzzy* para Avaliação da Qualidade de Software**.

IV.3 Etapas do Modelo *Fuzzy* para Avaliação da Qualidade de Software

O método *fuzzy* para avaliação da qualidade de software possui cinco etapas, para a consecução de seus objetivos, que podem envolver três situações distintas:

❶ **Determinação do Padrão de Qualidade (*PQ*) de um produto de software ou de um domínio de aplicação**. Procura-se obter de especialistas do produto (ou do domínio de aplicação) o grau de importância para cada atributo, de forma que o produto seja considerado de qualidade, tomando-se, como base, o objeto da avaliação. Isto significa dizer que o peso atribuído a cada atributo, por um especialista, deve retratar como o produto deveria estar. Portanto, neste caso, não se está avaliando o estado de um certo produto, mas o *padrão ideal de qualidade* que ele deveria apresentar.

❷ **Avaliação da qualidade de um produto de software, apoiando-se em um *PQ* já previamente definido**. Cada especialista julga o conjunto de atributos de qualidade, considerando o estado em que o software se encontra. Os resultados deste julgamento serão confrontados com o *PQ*, já estabelecido para o produto ou para o domínio de aplicação. São gerados, então, *índices de qualidade*, para cada atributo considerado, chegando-se à medição da qualidade do produto final. Esses índices medem o quanto o produto avaliado atinge, percentualmente, do padrão ideal de qualidade estabelecido, que tem índice igual a 1.

❸ **Estimativa da qualidade de um produto de software, sem que haja um *PQ* já estabelecido**. Neste caso, como não há um *PQ* definido, os resultados serão apurados levando-se em conta apenas o julgamento dos especialistas para o produto em questão. Com este procedimento, a equipe de desenvolvimento ou a equipe gestora da qualidade do produto terá um conjunto de dados úteis, que poderão auxiliar na continuidade do desenvolvimento do produto ou servir de parâmetros para possíveis melhorias em um produto já desenvolvido.

A seguir, as **etapas do modelo *fuzzy*** para avaliação da qualidade de software são apresentadas e, após cada etapa, é fornecido um exemplo de utilização deste modelo. Esse exemplo está baseado em uma pesquisa de campo realizada por CLUNIE (1997), para avaliação do padrão de qualidade para especificações de requisitos de software.

1. PRIMEIRA ETAPA: identificação do objeto a ser avaliado, do conjunto de atributos de qualidade de software a ser considerado na avaliação e das instituições pesquisadas.

1.1. ***Estabelecer o objeto da avaliação***: nesta etapa, estabelece-se qual é o produto de software que será objeto da avaliação. Podem ser avaliados subprodutos gerados durante as diversas fases do ciclo de vida de um produto de software ou o próprio produto final.

1.2. ***Definir o conjunto de atributos de qualidade***: o conjunto de atributos é definido considerando-se o objeto a ser avaliado, o domínio da aplicação e a tecnologia de desenvolvimento. Podem acontecer duas situações. Na primeira, os atributos já estarem definidos, fruto de trabalhos anteriores. Para o Modelo Rocha e, portanto, passíveis de utilização com o Modelo Rocha Estendido, proposto neste trabalho, tem-se conjuntos de atributos definidos para diversos domínios de aplicação, como mostra a *Seção 2.2.1.3*. Na segunda situação, os atributos ainda não estão definidos e deve-se, portanto, proceder à sua identificação, podendo-se para isto contar com a ferramenta proposta por (PASSOS, 1996).

1.3. ***Selecionar a(s) instituição(ões) onde a pesquisa será realizada***:

- *Em uma única instituição*: quando esta processa a avaliação de atributos de qualidade de um produto de software, que lhe é próprio.

- *Em várias instituições*: quando dados são coletados com o objetivo de ser definido o padrão de qualidade de um determinado produto de software, em uma área de aplicação específica.

EXEMPLO Etapa 1:

1.1. ***Objeto da avaliação***: especificações de requisitos de software (ERS).

1.2. ***Conjunto de atributos***: o conjunto de atributos de qualidade considerados para ERS, foram definidos por CLUNIE (1997), segundo o Modelo Rocha (ROCHA, 1983), *Apêndice II*, e estão distribuídos da seguinte forma:

- *Objetivo Confiabilidade da Representação*: 19 critérios, 8 subfatores e 2 fatores.
- *Objetivo Confiabilidade Conceitual*: 9 critérios, 5 subfatores e 2 fatores.
- *Objetivo Utilizabilidade*: 13 critérios (que lhe são próprios), 28 critérios (não pertencentes diretamente a esse *objetivo*, mas a ele relacionados), 12 subfatores e 4 fatores.

1.3. ***Instituições onde a pesquisa foi realizada***: foram selecionadas 3 instituições, com larga experiência na elaboração de ERS: uma empresa estatal e uma multinacional, ambas de grande porte, e a COPPE/ Sistemas, da UFRJ.

2. SEGUNDA ETAPA: **escolha dos especialistas**.

De uma maneira geral, todos os indivíduos, que estão ou já estiveram, direta ou indiretamente, envolvidos com produtos similares ao objeto de avaliação poderão participar do processo de avaliação.

2.1. ***Obter o perfil dos especialistas***: obtenção do perfil dos especialistas (ISHIKAWA *et al*., 1993), E_i (i = 1, 2, ..., n), que participarão do processo de investigação, através do *Questionário de Identificação do Perfil Especialista* (*QIPE*), *Apêndice I*, para se ter a indicação da importância relativa de cada um deles. (Tem-se encontrado, na literatura, a atribuição de pesos aos especialistas, sem uma estrutura formal de cálculo).

2.2. ***Calcular o peso do especialista***: calcula-se o grau de importância relativa de cada especialista, isto é, o peso p_{Ei}, através dos dados obtidos pelo preenchimento do *QIPE*, levando-se em consideração os seguintes critérios:

- cada *QIPE* contém as informações de um único especialista;
- O total de escores de cada especialista, t$QIPE_i$, é calculado, segundo indicações contidas na apuração dos resultados do *QIPE*.
- o peso de cada especialista p_{Ei}, que é seu peso relativo em relação aos outros especialistas (média ponderada), é definido por:

$$p_{Ei} = \frac{\mathrm{t}QIPE_i}{\sum_{i=1}^{n} \mathrm{t}QIPE_i}$$

EXEMPLO Etapa 2:

Foram selecionados 16 especialistas, que estão, ou já estiveram, diretamente envolvidos com a elaboração de especificações de requisitos de software.

2.1. ***Perfil dos especialistas***: o perfil de cada especialista, E_i, que participou desse processo de investigação, foi obtido através do *Questionário de Identificação do Perfil Especialista* (*QIPE*), *Apêndice I*, para se ter a indicação da importância relativa de cada um deles.

2.2. ***Cálculo do peso de cada especialista***: para cada um dos 16 especialista E_i, foram preenchidos os 7 *itens* do *QIPE* (i_1, i_2, ..., i_7). Os escores de cada *item* foram obtidos, segundo o *Apêndice I*, chegando-se, finalmente, ao o peso p_{Ei} relativo de cada especialista, como exibe a *Tabela IV.4*.

Apuração dos Resultados do *QIPE*									
E_i / item	i_1	i_2	i_3	i_4	i_5	i_6	i_7	*tQIPE*	p_{Ei}
1	2,60	2,30	3,60	0,30	0,90	0,70	1,40	4,149	0,059
2	1,70	1,70	2,40	0,30	0,90	0,60	4,00	3,578	0,051
3	1,90	2,00	3,90	0,70	0,70	0,60	2,00	4,134	0,059
4	1,70	3,60	3,60	1,00	0,90	0,80	4,90	5,384	0,077
5	2,10	2,70	3,10	0,00	0,90	0,80	7,80	4,321	0,062
6	1,70	1,90	1,80	0,00	1,00	1,00	10,80	4,211	0,060
7	4,50	2,60	3,60	1,00	1,00	1,00	8,10	6,278	0,090
8	2,60	1,30	2,40	0,30	0,70	0,80	4,00	3,667	0,052
9	1,40	1,70	3,60	0,00	0,70	0,70	2,70	3,317	0,047
10	3,10	3,00	3,90	0,70	0,90	0,80	9,20	5,641	0,080
11	0,80	1,30	3,60	0,00	0,90	0,70	7,40	3,640	0,052
12	2,30	2,00	3,60	0,00	0,70	0,40	4,60	3,449	0,049
13	3,60	3,30	3,60	0,70	0,90	1,00	12,80	6,240	0,089
14	2,60	1,70	2,40	0,30	0,70	0,70	4,00	3,678	0,052
15	1,70	1,70	2,40	0,30	0,70	0,70	4,00	3,478	0,050
16	2,60	3,30	3,60	0,00	1,00	0,80	9,30	4,944	0,071
Total								**Total**	**Total**
16								70,108	1,000

Tabela IV.4.: Resultados do *QIPE*

3. TERCEIRA ETAPA: **determinação do grau de importância de cada atributo de qualidade, identificado na *Etapa 1***.

O processo de investigação consiste em se obter dos especialistas, a serem selecionados, graus de importância, para obter o julgamento destes em relação a cada um dos atributos de qualidade mensuráveis (critérios), através do uso do conjunto de *termos lingüísticos*, caracterizados por números *fuzzy* triangulares normais do tipo-*LR*, $\tilde{N}_i(a_i, m_i, b_i)$, previamente delineados (por exemplo, a *Tabela IV.3*).

Caso o especialista avaliador deseje utilizar um outro número *fuzzy* $\tilde{N}_k(a_k, m_k, b_k)$, que não pertença ao conjunto previamente estabelecido, poderá fazê-lo, desde que $k \in [0, n-1]$ e $\tilde{N}_1 < \tilde{N}_k < \tilde{N}_n$. Por exemplo, se a *Tabela IV.2* for usada, um especialista, em sua avaliação de um determinado atributo, poderia optar pelo número *fuzzy* $\tilde{N}_k = (2{,}5;3{,}7;4{,}8)$, com significado semântico entre os termos lingüísticos *médio* e *alto*, uma vez que $n = 6$ e, $\forall\ a_k, m_k, b_k \in [0, 5]$.

Nesta etapa, os especialistas deverão ser comunicados de que o processo de investigação, que está sendo aplicado, é para o levantamento de um determinado *Padrão de Qualidade* (*PQ*), ou apenas é uma avaliação do estado de um produto de software real, para que possam realizar seus julgamentos convenientemente.

3.1. ***Definir o procedimento de investigação***: esse procedimento poderá consistir na elaboração de um *questionário* ou um outro dispositivo de investigação, e na definição de técnicas de aplicação próprias para o mesmo, usando-se *graus de importância* (dos termos lingüísticos) estabelecidos.

3.2. ***Aplicar o dispositivo de investigação aos especialistas***: o dispositivo de investigação é aplicado aos especialistas selecionados na etapa anterior.

EXEMPLO Etapa 3:

3.1. ***Procedimento de investigação***: consistiu na elaboração de um *questionário* e na definição de técnicas para o manuseio do mesmo (*Apêndice III*).

3.2. ***Aplicação do questionário***: o questionário foi aplicado a uma pesquisa de campo (CLUNIE, 1997), sendo obtido de cada especialista avaliador, graus de importância para cada um dos atributos de qualidade mensuráveis (critérios), selecionados na *primeira etapa*, através da utilização somente

do conjunto de *termos lingüísticos*, caracterizados por números *fuzzy* normais triangulares do tipo-*LR*, delineados na *Tabela IV.3*

4. QUARTA ETAPA: **tratamento dos dados coletados dos especialistas, na avaliação de cada atributo de qualidade mensurável considerado (critério).**

Nesta etapa, os prognósticos individuais, obtidos de cada especialista (números *fuzzy* normais triangulares), são combinados, para cada um dos atributos de qualidade diretamente mensuráveis (critérios), gerando-se uma opinião consensual entre esses especialistas, para cada critério avaliado, sendo formalizada por uma função de pertinência *fuzzy* característica $\tilde{N}$, onde:

$$\tilde{N} = f(\tilde{N}_1, \tilde{N}_2, ..., \tilde{N}_n)$$

4.1. ***Calcular o grau de concordância*** (HSU *et al*., 1996, CHEN *et al*., 1993, ZWICK *et al*., 1987): calcula-se o grau de concordância, $C(\tilde{N}_i, \tilde{N}_j)$, combinando-se os julgamentos dos especialistas E_i e E_j, através da razão entre a área de interseção entre eles e a área total, de suas funções de pertinência, isto é:

$$C(\tilde{N}_i, \tilde{N}_j) = \frac{\int_x (min\{\mu_{\tilde{N}_i}(x), \mu_{\tilde{N}_j}(x)\})dx}{\int_x (max\{\mu_{\tilde{N}_i}(x), \mu_{\tilde{N}_j}(x)\})dx}$$

4.2. ***Construir a matriz de concordância***: calculados todos os graus de concordância entre cada par de especialistas E_i e E_j, constrói-se uma *matriz de concordância*, *MC*, que fornece as indicações da aquiescência entre eles, onde $C_{ij} = C(\tilde{N}_i, \tilde{N}_j)$, se $i \neq j$ e $C_{ij} = 1$, se $i = j$.

$$MC = \begin{bmatrix} 1 & C_{12} & \cdots & C_{1j} & \cdots & C_{1n} \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ C_{i1} & C_{i2} & \cdots & C_{ij} & \cdots & C_{in} \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ C_{n1} & C_{n2} & \cdots & C_{nj} & \cdots & 1 \end{bmatrix}$$

Após a construção da matriz MC, deve-se observar os seguintes itens:

i. Caso $C_{ij} = 0$, isto é, não há interseção entre o *i*-enésimo e o *j*-ésimo especialista, obtém-se mais informações desses especialistas (segundo a conveniência da avaliação), com o objetivo de ajustar (convergir) suas opiniões, isto é, chegar a uma interseção entre eles.

ii. Depois de obtidas as novas informações, conforme o item (*i*), se persistir algum grau de concordância nulo, considerá-lo na matriz *MC*, pois, no processo de agregação (*item 4.6*), os valores nulos (isto é, os que destoaram do consenso geral entre os especialistas) terão peso zero no resultado final da agregação.

iii. Atentar para o fato de que se houver uma grande disparidade de respostas (isto é, um baixo consenso entre os especialistas), isto pode significar que estes não entenderam convenientemente a definição do objeto de investigação (DYER, 1992). Neste caso, o item (*i*) deve ser executado, tanto quanto possível, no sentido de se chegar ao maior consenso entre esses especialistas.

4.3. ***Calcular a concordância relativa***: através dos dados obtidos de *MC*, calcula-se a *concordância relativa* de cada especialista (CR_i) envolvido neste processo, pela *média quadrática* do grau de concordância entre eles:

$$CR_i = \sqrt{\frac{1}{n-1}\sum_{\substack{j=1 \\ j\neq i}}^{n} C_{ij}^{\,2}}$$

Com este procedimento, o cálculo de CR_i tenderá para os maiores índices de consenso entre os especialistas avaliadores.

4.4. ***Calcular o grau de concordância relativa***: o *grau da concordância relativa*, GCR_i, de um especialista, em relação a todos os outros especialistas, é obtido pela *média ponderada* de CR_i de cada especialista:

$$GCR_i = \frac{CR_i}{\sum_{i=1}^{n} CR_i}$$

4.5. ***Calcular o coeficiente de consenso dos especialistas***: o *coeficiente de consenso*, obtido para cada especialista (CCE_i) considerará tanto o GCR_i, quanto o peso, p_{Ei} (item 2.2), de cada um deles.

$$CCE_i = \frac{GCR_i \cdot p_{Ei}}{\sum_{i=1}^{n}(GCR_i \cdot p_{Ei})}$$

4.6. ***Avaliar o critério de qualidade***: o resultado da avaliação de cada critério de qualidade será dado por $\tilde{N}$, que também é um número *fuzzy* normal triangular, onde • é o *produto algébrico fuzzy* (HSU *et al*., 1996, KAUFMANN *et al*., 1991):

$$\tilde{N} = \sum_{i=1}^{n}(CCE_i \bullet \tilde{N}_i)$$

EXEMPLO Etapa 4:

Neste exemplo, serão apresentados os cálculos da avaliação do critério, *Correção da Notação*, que é um dos atributos de qualidade pertencentes ao objetivo *Confiabilidade da Representação*.

4.1. ***Cálculo do grau de concordância***: o cálculo do grau de concordância, $C(\tilde{N}_i, \tilde{N}_j)$, entre os especialistas E_i e E_j, foi obtido através da razão entre a *área de interseção* das funções de pertinência, $\tilde{N}_i$ e $\tilde{N}_j$, *Tabela IV.6*, correspondentes aos *termos lingüísticos*, utilizados no julgamento do critério, por esses especialistas, *Tabela IV.5*, e a *área de união* dessas mesmas funções de pertinência.

Especialistas avaliadores	**Números *fuzzy* $\tilde{N}$**			**Área de $\tilde{N}$**
	a	***m***	*b*	
1	2,00	**3,00**	4,00	1,00
2	2,00	**3,00**	4,00	1,00
3	3,00	**4,00**	4,00	0,50
4	3,00	**4,00**	4,00	0,50
5	3,00	**4,00**	4,00	0,50
6	2,00	**3,00**	4,00	1,00
7	1,00	**2,00**	3,00	1,00
8	3,00	**4,00**	4,00	0,50
9	2,00	**3,00**	4,00	1,00
10	2,00	**3,00**	4,00	1,00
11	3,00	**4,00**	4,00	0,50
12	2,00	**3,00**	4,00	1,00
13	3,00	**4,00**	4,00	0,50
14	2,00	**3,00**	4,00	1,00
15	3,00	**4,00**	4,00	0,50
16	3,00	**4,00**	4,00	0,50

Tabela IV.5: Termos lingüísticos (números *fuzzy*), usados pelos especialistas na avaliação do critério, *Correção da Notação*.

Área de Interseção entre E_i e E_j																
E_i/E_j	E_1/E_j	E_2/E_j	E_3/E_j	E_4/E_j	E_5/E_j	E_6/E_j	E_7/E_j	E_8/E_j	E_9/E_j	E_{10}/E_j	E_{11}/E_j	E_{12}/E_j	E_{13}/E_j	E_{14}/E_j	E_{15}/E_j	E_{16}/E_j
E_i/E_1	1,00	1,00	0,25	0,25	0,25	1,00	0,25	0,25	1,00	1,00	0,25	1,00	0,25	1,00	0,25	0,25
E_i/E_2	1,00	1,00	0,25	0,25	0,25	1,00	0,25	0,25	1,00	1,00	0,25	1,00	0,25	1,00	0,25	0,25
E_i/E_3	0,25	0,25	0,50	0,50	0,50	0,25	0,00	0,50	0,25	0,25	0,50	0,25	0,50	0,25	0,50	0,50
E_i/E_4	0,25	0,25	0,50	0,50	0,50	0,25	0,00	0,50	0,25	0,25	0,50	0,25	0,50	0,25	0,50	0,50
E_i/E_5	0,25	0,25	0,50	0,50	0,50	0,25	0,00	0,50	0,25	0,25	0,50	0,25	0,50	0,25	0,50	0,50
E_i/E_6	1,00	1,00	0,25	0,25	0,25	1,00	0,25	0,25	1,00	1,00	0,25	1,00	0,25	1,00	0,25	0,25
E_i/E_7	0,25	0,25	0,00	0,00	0,00	0,25	1,00	0,00	0,25	0,25	0,00	0,25	0,00	0,25	0,00	0,00
E_i/E_8	0,25	0,25	0,50	0,50	0,50	0,25	0,00	0,50	0,25	0,25	0,50	0,25	0,50	0,25	0,50	0,50
E_i/E_9	1,00	1,00	0,25	0,25	0,25	1,00	0,25	0,25	1,00	1,00	0,25	1,00	0,25	1,00	0,25	0,25
E_i/E_{10}	1,00	1,00	0,25	0,25	0,25	1,00	0,25	0,25	1,00	1,00	0,25	1,00	0,25	1,00	0,25	0,25
E_i/E_{11}	0,25	0,25	0,50	0,50	0,50	0,25	0,00	0,50	0,25	0,25	0,50	0,25	0,50	0,50	0,50	0,50
E_i/E_{12}	1,00	1,00	0,25	0,25	0,25	1,00	0,25	0,25	1,00	1,00	0,25	1,00	0,25	1,00	0,25	0,25
E_i/E_{13}	0,25	0,25	0,50	0,50	0,50	0,25	0,00	0,50	0,25	0,25	0,50	0,25	0,50	0,25	0,50	0,50
E_i/E_{14}	1,00	1,00	0,25	0,25	0,20	1,00	0,25	0,25	1,00	1,00	0,25	1,00	0,25	1,00	0,25	0,25
E_i/E_{15}	0,25	0,25	0,50	0,50	0,50	0,25	0,00	0,50	0,25	0,25	0,50	0,25	0,50	0,25	0,50	0,50
E_i/E_{16}	0,25	0,25	0,50	0,50	0,50	0,25	0,00	0,50	0,25	0,25	0,50	0,25	0,50	0,25	0,50	0,50

Tabela IV.6: Área de interseção entre os especialistas E_i e E_j na avaliação do critério, *Correção da Notação*.

4.2. ***Construção da matriz de concordância***: calculados todos os graus de concordância entre cada par de especialistas E_i e E_j, foi construída a *matriz de concordância*, *MC*, *Tabela IV.7*.

Matriz de Concordância (MC)																
E_i/E_j	E_1/E_j	E_2/E_j	E_3/E_j	E_4/E_j	E_5/E_j	E_6/E_j	E_7/E_j	E_8/E_j	E_9/E_j	E_{10}/E_j	E_{11}/E_j	E_{12}/E_j	E_{13}/E_j	E_{14}/E_j	E_{15}/E_j	E_{16}/E_j
E_i/E_1	1,00	1,00	0,20	0,20	0,20	1,00	0,14	0,20	1,00	1,00	0,20	1,00	0,20	1,00	0,20	0,20
E_i/E_2	1,00	1,00	0,20	0,20	0,20	1,00	0,14	0,20	1,00	1,00	0,20	1,00	0,20	1,00	0,20	0,20
E_i/E_3	0,20	0,20	1,00	1,00	1,00	0,20	0,00	1,00	0,20	0,20	1,00	0,20	1,00	0,20	1,00	1,00
E_i/E_4	0,20	0,20	1,00	1,00	1,00	0,20	0,00	1,00	0,20	0,20	1,00	0,20	1,00	0,20	1,00	1,00
E_i/E_5	0,20	0,20	1,00	1,00	1,00	0,20	0,00	1,00	0,20	0,20	1,00	0,20	1,00	0,20	1,00	1,00
E_i/E_6	1,00	1,00	0,20	0,20	0,20	1,00	0,14	0,20	1,00	1,00	0,20	1,00	0,20	1,00	0,20	0,20
E_i/E_7	0,14	0,14	0,00	0,00	0,00	0,14	1,00	0,00	0,14	0,14	0,00	0,14	0,00	0,14	0,00	0,00
E_i/E_8	0,20	0,20	1,00	1,00	1,00	0,20	0,00	1,00	0,20	0,20	1,00	0,20	1,00	0,20	1,00	1,00
E_i/E_9	1,00	1,00	0,20	0,20	0,20	1,00	0,14	0,20	1,00	1,00	0,20	1,00	0,20	1,00	0,20	0,20
E_i/E_{10}	1,00	1,00	0,20	0,20	0,20	1,00	0,14	0,20	1,00	1,00	0,20	1,00	0,20	1,00	0,20	0,20
E_i/E_{11}	0,20	0,20	1,00	1,00	1,00	0,20	0,00	1,00	0,20	0,20	1,00	0,20	1,00	0,20	1,00	1,00
E_i/E_{12}	1,00	1,00	0,20	0,20	0,20	1,00	0,14	0,20	1,00	1,00	0,20	1,00	0,20	1,00	0,20	0,20
E_i/E_{13}	0,20	0,20	1,00	1,00	1,00	0,20	0,00	1,00	0,20	0,20	1,00	0,20	1,00	0,20	1,00	1,00
E_i/E_{14}	1,00	1,00	0,20	0,20	0,20	1,00	0,14	0,20	1,00	1,00	0,20	1,00	0,20	1,00	0,20	0,20
E_i/E_{15}	0,20	0,20	1,00	1,00	1,00	0,20	0,00	1,00	0,20	0,20	1,00	0,20	1,00	0,20	1,00	1,00
E_i/E_{16}	0,20	0,20	1,00	1,00	1,00	0,20	0,00	1,00	0,20	0,20	1,00	0,20	1,00	0,20	1,00	1,00

Tabela IV.7: Matriz de Concordância entre os especialistas E_i e E_j na avaliação do critério, *Correção da Notação*.

Após a construção da matriz *MC* acima, observou-se que apenas o especialista E_7 possui graus de concordância nulos, isto é, não há concordância de seu julgamento com os julgamentos dos especialistas E_3, E_4, E_5, E_8, E_3, E_{11}, E_{13}, E_{15} e E_{16}.

Como este trabalho utiliza os dados da pesquisa de campo realizada por CLUNIE (1997), o acesso posterior a todos os especialistas, que participaram desta pesquisa, tornou-se muito difícil. Como o ajuste para a convergência das opiniões entre todos os especialistas, isto é, $C_{ij} \neq 0$, para cada um dos 41 critérios avaliados, demandaria várias entrevistas com esses especialistas, tendo em conta a dificuldade de acesso a vários deles, decidiu-se pela utilização apenas das informações originais da pesquisa de campo, uma vez que isto é possível no MFAQS. Com este procedimento, os graus de concordância nulos, de um dado especialista, agirão no sentido de reduzir o peso desse especialista no julgamento final do atributo avaliado.

4.3. ***Cálculo da concordância relativa***: através dos dados obtidos da matriz *MC*, calculou-se a *concordância relativa* (CR_i) de cada especialista envolvido neste processo, pela *média quadrática* do grau de concordância entre eles, *Tabela IV.8*. Para o especialista E_1, por exemplo, tem-se:

$$CR_1 = \sqrt{\frac{1}{16-1}(1{,}00^2 + 0{,}20^2 + 0{,}20^2 + 0{,}20^2 + 1{,}00^2 + 0{,}14^2 + \ldots + 0{,}20^2)} = 0{,}6501$$

4.4. ***Cálculo do grau de concordância relativa***: o grau da concordância relativa, GCR_i, de cada especialista, em relação aos demais especialistas, foi obtido pela *média ponderada* de CR_i de cada especialista, mostrado na *Tabela IV.8*. Para o especialista E_1, por exemplo, tem-se:

$$GCR_1 = 0{,}6501 / 10{,}1723 = 0{,}06391$$

4.5. ***Cálculo do coeficiente de consenso dos especialistas***: o coeficiente de consenso foi obtido para cada especialista (CCE_i), sendo considerado tanto o GCR_i, quanto o peso, p_{Ei}, *Tabela IV.8*, de cada um deles. Para o especialista E_1, por exemplo, tem-se:

$$CCE_1 = \frac{0{,}06391 \cdot 0{,}0592}{0{,}0610} = 0{,}0621$$

Especialistas	CR_i	GCR_i	CCE_i
1	0,6501	0,0639	0,0621

2	0,6501	0,0639	0,0535
3	0,6967	0,0685	0,0663
4	0,6967	0,0685	0,0863
5	0,6967	0,0685	0,0693
6	0,6501	0,0639	0,0630
7	0,0976	0,0096	0,0141
8	0,6967	0,0685	0,0588
9	0,6501	0,0639	0,0496
10	0,6501	0,0639	0,0844
11	0,6967	0,0685	0,0583
12	0,6501	0,0639	0,0516
13	0,6470	0,0636	0,0929
14	0,6501	0,0639	0,0550
15	0,6967	0,0685	0,0557
16	0,6967	0,0685	0,0792
Total	**Total**	**Produto**	**Total**
16	10,1723	0,0610	1,0000

Tabela IV.8: Concordância relativa, grau de concordância relativa, e coeficiente de concordância entre os E_i e E_j na avaliação do critério, *Correção da Notação*.

4.6. ***Avaliação do critério de qualidade***: o resultado da avaliação do Critério, *Correção da Notação*, é dado por $\tilde{N}$, que também é um número *fuzzy* normal triangular:

$\tilde{N}$ = { [0,0621 • $\tilde{N}_1$] + . . . + [0,0792 • $\tilde{N}_{16}$] }

= { [(0,0621 · 2,00) + . . . + (0,0792 · 3,00)];

[(0,0621 · 3,00) + . . . + (0,0792 · 4,00)];

[(0,0621 · 4,00) + . . . + (0,0792 · 4,00)] }

⇛ $\tilde{N}$ = (2,55; 3,55; 3,99)

5. QUINTA ETAPA: agregação dos atributos de qualidade de software, em cada nível hierárquico do modelo de qualidade.

Nesta etapa, faz-se a agregação dos atributos de qualidade (BELCHIOR *et al*., 1995c, BELCHIOR *et al*., 1996a, BELCHIOR *et al*., 1996b) do tipo $\tilde{N}$, de acordo com o *Modelo Rocha Estendido*, gerando-se uma função de pertinência *fuzzy* característica, para cada subconjunto de atributos de qualidade em questão, isto é, os *atributos agregados*:

$$Software\begin{cases} Objetivos \\ (1,o) \end{cases}\begin{cases} Fatores \\ (1,f) \end{cases}\begin{cases} Subfatores \\ (0,s) \end{cases}\begin{cases} \text{Crité rios} \\ (1,c) \end{cases}$$

Assim sendo, tem-se de 1 a *o objetivos* de qualidade para um produto de software. Para cada objetivo, são definidos de 1 a *f fatores* de qualidade a ele relacionados. Cada *fator* contém de 0 a *s subfatores*. Cada *fator*/*subfator* possui de 1 a *c critérios* (BELCHIOR, 1992b).

- Cada *atributo agregado* avaliado, $\tilde{N}$, composto por seus atributos constituintes, $\tilde{N}_1, \tilde{N}_2, \ldots, \tilde{N}_n$, também será formalizado por:

$$\tilde{N} = f(\tilde{N}_1, \tilde{N}_2, \ldots, \tilde{N}_n)$$

5.1 ***Estabelecer o padrão de qualidade***: quando se tratar do estabelecimento do padrão de qualidade (*PQ*) para um determinado produto de software ou para um domínio de aplicação específico, calcula-se o peso W_i (BELCHIOR *et al*., 1995b, BELCHIOR *et al*., 1996c, CHEN *et al*., 1995, LIOU *et al*., 1992), isto é, o grau de contribuição de cada atributo, que compõe o *atributo agregado* avaliado. O peso de cada atributo W_i será obtido pela média ponderada dos graus de importância de cada um de seus atributos constituintes, isto é, o valor w_i, que será calculado através da *defuzificação* de seu número *fuzzy* $\tilde{N}_i(a_i, m_i, b_i)$ correspondente. Portanto:

i. O valor w_i será dado por:

$w_i = m_i$, que corresponde ao valor com grau de pertinência igual a 1, isto é, este é o valor nítido (clássico) do atributo de qualidade.

ii. O *peso* W_i, será dado por:

$W_i = w_i / \sum w_i$

5.2 ***Calcular o grau de concordância agregado***: calcula-se o grau de concordância, $C(\tilde{N}_i, \tilde{N}_j)$, entre os atributos de qualidade, que estão sendo agregados (que são números *fuzzy* $\tilde{N}$) da mesma maneira que no *item 4.1*.

5.3 ***Construir a matriz de concordância de agregação***: após o cálculo de todos os C_{ij} dos atributos pertencentes ao subconjunto, que está sendo agregado, gera-se a *matriz de concordância de agregação* (*MCA*) similarmente ao *item 4.2*. Se $C_{ij} = 0$, isto é, não há interseção entre os atributos *i* e *j*, procede-se ao cálculo do *grau da não concordância*, $\overline{C}_{ij}$, entre esses atributos, que deverá ser um valor no intervalo [-1, 0]:

$\overline{C_{ij}} = -\frac{d}{D} \cdot r$, onde:

i. *d* é a menor *distância* entre os dois números *fuzzy* considerados (atributos) , ou seja, $d = a_j - b_i$ ou $d = a_i - b_j$ (utiliza-se o menor *valor absoluto* de *d*).

ii. *D* é a maior *distância* entre o maior e o menor número *fuzzy* do conjunto de termos lingüísticos considerados, isto é, $\tilde{N}_n$ e $\tilde{N}_1$ respectivamente. Portanto, $D = a_n - b_1$.

iii. *r* é a *razão* entre as áreas dos números *fuzzy* $\tilde{N}_i$ e $\tilde{N}_j$, sendo que $0 < r \leq 1$, daí:

$$r = \frac{\int_x (\mu_{\tilde{N}_i}(x))dx}{\int_x (\mu_{\tilde{N}_j}(x))dx} \quad \text{ou} \quad r = \frac{\int_x (\mu_{\tilde{N}_j}(x))dx}{\int_x (\mu_{\tilde{N}_i}(x))dx}$$

5.4 ***Construir a matriz dos estados de agregação***: constrói-se a nova *matriz do estado de agregação*, *MEA*, que conterá os valores de C_{ij} e $\overline{C_{ij}}$, isto é, os *grau dos estados de agregação*, E_{ij}, substituindo a matriz *MCA*.

5.5 ***Calcular o estado relativo de agregação***: obtém-se o *estado relativo de agregação*, *ERA* (*era*), através da concordância ou não concordância de cada um dos atributos, que estão sendo agregados, através de dois procedimentos:

i. Quando existe algum *grau de não concordância*, $\overline{C_{ij}}$, na matriz *MA*, calcula-se, inicialmente, o '*era*' de cada atributo, pela *média aritmética* do *grau de concordância* e do *grau de não concordância* do atributo em questão com os outros que estão sendo agregados, quantificando-se, assim, o quanto o atributo concorda ou diverge em relação aos demais, no processo de agregação.

$$era_i = \frac{1}{n-1} \sum_{\substack{j=1 \\ j \neq i}}^{n} E_{ij}$$

Com este procedimento, visualiza-se o estado de cada atributo no processo de agregação, isto é, o quanto cada atributo realmente contribui na composição do novo *atributo agregado*. Quando algum *atributo agregante* tiver seu '*era*' não positivo (condição de alerta), deve-se inquirir se, de fato, esse atributo deveria pertencer ou não àquele ramo da

hierarquia de composição dos atributos do produto de software, que está sendo avaliado. Esta pode ser uma informação útil, para a validação da árvore hierárquica dos atributos de qualidade de software.

ii. Uma vez que os atributos, que estão sendo agregados, realmente pertencem ao ramo hierárquico da árvore dos atributos de qualidade em questão, procede-se ao cálculo do '*ERA*' pela média quadrática de todos os graus dos estados de agregação, E_{ij}, de seus atributos agregantes.

$$ERA_i = \sqrt{\frac{1}{n}\sum_{j=1}^{n} E_{ij}^{\,2}}$$

- Neste caso, os graus negativos (de não concordância) são tratados e exercem a mesma influência como se fossem positivos. Com este procedimento, objetiva-se chegar a um *valor compensatório* da contribuição de cada atributo no processo de agregação, através do modelo *fuzzy* proposto. Desta forma, preservando-se as características particulares de composição dos atributos, isto é, sua localização na árvore hierárquica de atributos de qualidade de software, estabelecida pela equipe gestora da qualidade na organização, executa-se o processo de agregação. Se for detectado algum erro de localização nessa árvore, este deve ser tratado pela equipe responsável pela qualidade do produto avaliado, com é sugerido no *item* (*i*) anterior, ou através de outro procedimento estabelecido ou utilizado por essa equipe, sendo seus resultados de inteira responsabilidade da mesma.
- Exemplificando: se um determinado *atributo* de qualidade de software for composto por dois outros *atributos* com graus de importância diferentes entre si, para um determinado produto de software, e se um desses atributos tiver um grau de importância *baixo* e o outro *alto*, isto significaria que eles contribuiriam em uma percentagem baixa e alta, respectivamente, para a composição do atributo agregado considerado. Através do procedimento (*ii*) acima, obter-se-á para o atributo agregado um grau de importância *médio*. Portanto, o atributo agregado avaliado tem grau de importância *médio*, sendo composto por dois atributos, que possuem graus de importância *baixo* e *alto*.

- No entanto, se na análise dos resultados, este processo de agregação não for conveniente, para o produto que está sendo avaliado, deve-se justificar o motivo da agregação não poder ser realizada com tal, relacionando-se apenas os *graus de importância* de cada atributo, que seria agregado.

5.6 ***Calcular o grau do estado relativo de agregação***: o *grau do estado relativo de agregação*, *GERA*, de cada atributo agregado é obtido pela *média ponderada* entre seus atributos constituintes:

$$GERA_i = \frac{ERA_i}{\sum_{i=1}^{n} ERA_i}$$

5.7 ***Calcular o coeficiente de consenso do atributo***: o *coeficiente de consenso do atributo* (CCA_i), obtido para cada atributo, que compõe o atributo que está sendo agregado, considerará tanto o $GERA_i$, como também o peso W_i (*item 5.1* (*ii*)) de cada um desses atributos. Caso não haja ainda um padrão de qualidade (*PQ*) estabelecido, ou se esteja determinado *PQ* para o produto de software, que está sendo avaliado, ou para o seu domínio de aplicação, deve-se considerar $W_i = 1$, isto é, $CCA_i = GERA_i$ (*item 5.6*).

$$CCA_i = \frac{GERA_i \cdot W_i}{\sum_{i=1}^{n} (GERA_i \cdot W_i)}$$

5.8 ***Avaliar o atributo agregado***: o resultado da avaliação de cada atributo de qualidade agregado será dado, também por $\tilde{N}$, que também é um número *fuzzy*, onde $\bullet$ é o *produto algébrico fuzzy* (KAUFMANN *et al*., 1991), formalizado por:

$$\tilde{N} = \sum_{i=1}^{n} (CCA_i \bullet \tilde{N}_i)$$

EXEMPLO Etapa 5:

Neste exemplo, serão apresentados os cálculos da avaliação do subfator, *Correção do Uso do Modelo*, que é um atributo agregado pertencente ao objetivo *Confiabilidade da Representação*, *Apêndice II*.

5.1 ***Estabelecimento do padrão de qualidade***: por se tratar do estabelecimento do padrão de qualidade (*PQ*) para ERS, foram calculados

os pesos W_i, isto é, o grau de contribuição (média ponderada) de cada atributo, que compõe o *atributo agregado* avaliado, conforme a *Tabela V.6*.

Critérios	Números *Fuzzy Ñ*			Área de *Ñ*	Cálculo do peso *W*
	a	***m***	*b*		
1	2,55	3,55	3,99	0,717	0,2560
2	2,40	3,40	3,95	0,775	0,2449
3	2,86	3,86	4,00	0,569	0,2784
4	2,06	3,06	3,82	0,880	0,2207
Total					**Total**
4					1,0000

Tabela IV.9: Critérios que compõem o subfator *Correção no Uso do Método*, com seus respectivos pesos *W*.

5.2 ***Cálculo do grau de concordância agregado***: o cálculo do grau de concordância, $C(\tilde{N}_i, \tilde{N}_j)$, entre os atributos de qualidade, que estão sendo agregados, foi feito da mesma forma que no *item 4.1*. Neste caso, foi gerada uma única matriz, por objetivo de qualidade, contendo as área de interseção entre todos os seus critérios constituintes, *Tabela IV.10*, que servirá como base na agregação dos subfatores, fatores e do próprio objetivo de qualidade. Para o subfator, dado como exemplo, sua área total está na *Tabela IV.9*.

Área de Interseção entre os critérios C*i* e C*j*																				
C_i/C_j	C_1/C_j	C_2/C_j	C_3/C_j	C_4/C_j	C_5/C_j	C_6/C_j	C_7/C_j	C_8/C_j	C_9/C_j	C_{10}/C_j	C_{11}/C_j	C_{12}/C_j	C_{13}/C_j	C_{14}/C_j	C_{15}/C_j	C_{16}/C_j	C_{17}/C_j	C_{18}/C_j	C_{19}/C_j
C_i/C_1	0,72	0,63	0,44	0,46	0,54	0,70	0,31	0,15	0,63	0,51	0,63	0,65	0,56	0,66	0,41	0,34	0,42	0,71	0,53
C_i/C_2	0,63	0,77	0,38	0,58	0,47	0,61	0,41	0,22	0,76	0,63	0,55	0,62	0,69	0,58	0,35	0,29	0,37	0,62	0,66
C_i/C_3	0,44	0,38	0,57	0,26	0,51	0,45	0,16	0,05	0,39	0,30	0,47	0,38	0,34	0,46	0,53	0,44	0,55	0,45	0,31
C_i/C_4	0,46	0,58	0,26	0,88	0,33	0,44	0,65	0,40	0,62	0,85	0,40	0,46	0,73	0,42	0,24	0,19	0,25	0,45	0,72
C_i/C_5	0,54	0,47	0,51	0,33	0,63	0,56	0,22	0,09	0,48	0,38	0,58	0,48	0,42	0,57	0,48	0,40	0,49	0,55	0,39
C_i/C_6	0,70	0,61	0,45	0,44	0,56	0,71	0,30	0,14	0,62	0,49	0,65	0,63	0,54	0,67	0,42	0,35	0,44	0,71	0,51
C_i/C_7	0,31	0,41	0,16	0,65	0,22	0,30	0,97	0,66	0,44	0,63	0,27	0,32	0,53	0,28	0,14	0,11	0,15	0,31	0,52
C_i/C_8	0,15	0,22	0,05	0,40	0,09	0,14	0,66	0,97	0,24	0,38	0,12	0,15	0,31	0,13	0,04	0,03	0,05	0,15	0,30
C_i/C_9	0,63	0,76	0,39	0,62	0,48	0,62	0,44	0,24	0,83	0,67	0,56	0,61	0,74	0,58	0,36	0,30	0,38	0,63	0,70
C_i/C_{10}	0,51	0,63	0,30	0,85	0,38	0,49	0,63	0,38	0,67	0,90	0,44	0,51	0,79	0,46	0,28	0,23	0,29	0,50	0,77
C_i/C_{11}	0,63	0,55	0,47	0,40	0,58	0,65	0,27	0,12	0,56	0,44	0,68	0,57	0,49	0,66	0,44	0,36	0,45	0,64	0,46
C_i/C_{12}	0,65	0,62	0,38	0,46	0,48	0,63	0,32	0,15	0,61	0,51	0,57	0,66	0,56	0,59	0,35	0,28	0,36	0,64	0,53
C_i/C_{13}	0,56	0,69	0,34	0,73	0,42	0,54	0,53	0,31	0,74	0,79	0,49	0,56	0,87	0,51	0,31	0,26	0,33	0,55	0,82
C_i/C_{14}	0,66	0,58	0,46	0,42	0,57	0,67	0,28	0,13	0,58	0,46	0,66	0,59	0,51	0,69	0,43	0,36	0,44	0,67	0,48
C_i/C_{15}	0,41	0,35	0,53	0,24	0,48	0,42	0,14	0,04	0,36	0,28	0,44	0,35	0,31	0,43	0,55	0,46	0,54	0,42	0,28
C_i/C_{16}	0,34	0,29	0,44	0,19	0,40	0,35	0,11	0,03	0,30	0,23	0,36	0,28	0,26	0,36	0,46	0,50	0,45	0,35	0,23
C_i/C_{17}	0,42	0,37	0,55	0,25	0,49	0,44	0,15	0,05	0,38	0,29	0,45	0,36	0,33	0,44	0,54	0,45	0,56	0,43	0,30
C_i/C_{18}	0,71	0,62	0,45	0,45	0,55	0,71	0,31	0,15	0,63	0,50	0,64	0,64	0,55	0,67	0,42	0,35	0,43	0,72	0,52
C_i/C_{19}	0,53	0,66	0,31	0,72	0,39	0,51	0,52	0,30	0,70	0,77	0,46	0,53	0,82	0,48	0,28	0,23	0,30	0,52	0,82

Tabela IV.10: Área de interseção entre os critérios C_i e C_j do objetivo *Confiabilidade da Representação*

5.3 ***Construção da matriz de concordância de agregação***: calculados todos os graus de concordância entre cada par de critério C_i e C_j, foi construída a *matriz de concordância*, *MC*, *Tabela IV.11*.

Matriz de Concordância (MC)																			
C_i/C_j	C_1/C_i	C_2/C_i	C_3/C_i	C_4/C_i	C_5/C_i	C_6/C_i	C_7/C_i	C_8/C_i	C_9/C_i	C_{10}/C_i	C_{11}/C_i	C_{12}/C_i	C_{13}/C_i	C_{14}/C_i	C_{15}/C_i	C_{16}/C_i	C_{17}/C_i	C_{18}/C_i	C_{19}/C_i
C_i/C_1	1,00	0,73	0,52	0,40	0,67	0,95	0,23	0,10	0,70	0,45	0,83	0,90	0,55	0,88	0,48	0,39	0,50	0,97	0,52
C_i/C_2	0,73	1,00	0,39	0,53	0,50	0,69	0,30	0,14	0,91	0,60	0,62	0,76	0,73	0,65	0,36	0,29	0,38	0,71	0,71
C_i/C_3	0,52	0,39	1,00	0,22	0,75	0,55	0,12	0,04	0,39	0,26	0,61	0,45	0,31	0,58	0,90	0,70	0,95	0,54	0,29
C_i/C_4	0,40	0,53	0,22	1,00	0,28	0,39	0,55	0,28	0,57	0,90	0,34	0,43	0,73	0,36	0,20	0,16	0,21	0,39	0,74
C_i/C_5	0,67	0,50	0,75	0,28	1,00	0,71	0,16	0,06	0,49	0,32	0,80	0,59	0,39	0,75	0,68	0,54	0,71	0,69	0,37
C_i/C_6	0,95	0,69	0,55	0,39	0,71	1,00	0,22	0,09	0,67	0,43	0,88	0,85	0,53	0,93	0,50	0,41	0,52	0,97	0,50
C_i/C_7	0,23	0,30	0,12	0,55	0,16	0,22	1,00	0,51	0,32	0,50	0,19	0,24	0,41	0,20	0,11	0,08	0,11	0,22	0,41
C_i/C_8	0,10	0,14	0,04	0,28	0,06	0,09	0,51	1,00	0,15	0,26	0,08	0,10	0,20	0,08	0,03	0,02	0,03	0,09	0,20
C_i/C_9	0,70	0,91	0,39	0,57	0,49	0,67	0,32	0,15	1,00	0,63	0,60	0,70	0,77	0,63	0,36	0,30	0,38	0,68	0,75
C_i/C_{10}	0,45	0,60	0,26	0,90	0,32	0,43	0,50	0,26	0,63	1,00	0,39	0,48	0,81	0,41	0,24	0,19	0,25	0,44	0,81
C_i/C_{11}	0,83	0,62	0,61	0,34	0,80	0,88	0,19	0,08	0,60	0,39	1,00	0,74	0,47	0,94	0,55	0,45	0,58	0,86	0,45
C_i/C_{12}	0,90	0,76	0,45	0,43	0,59	0,85	0,24	0,10	0,70	0,48	0,74	1,00	0,58	0,79	0,41	0,32	0,43	0,88	0,56
C_i/C_{13}	0,55	0,73	0,31	0,73	0,39	0,53	0,41	0,20	0,77	0,81	0,47	0,58	1,00	0,49	0,28	0,23	0,30	0,54	0,94
C_i/C_{14}	0,88	0,65	0,58	0,36	0,75	0,93	0,20	0,08	0,63	0,41	0,94	0,79	0,49	1,00	0,53	0,43	0,56	0,90	0,47
C_i/C_{15}	0,48	0,36	0,90	0,20	0,68	0,50	0,11	0,03	0,36	0,24	0,55	0,41	0,28	0,53	1,00	0,77	0,94	0,49	0,26
C_i/C_{16}	0,39	0,29	0,70	0,16	0,54	0,41	0,08	0,02	0,30	0,19	0,45	0,32	0,23	0,43	0,77	1,00	0,73	0,40	0,21
C_i/C_{17}	0,50	0,38	0,95	0,21	0,71	0,52	0,11	0,03	0,38	0,25	0,58	0,43	0,30	0,56	0,94	0,73	1,00	0,51	0,27
C_i/C_{18}	0,97	0,71	0,54	0,39	0,69	0,97	0,22	0,09	0,68	0,44	0,86	0,88	0,54	0,90	0,49	0,40	0,51	1,00	0,51
C_i/C_{19}	0,52	0,71	0,29	0,74	0,37	0,50	0,41	0,20	0,75	0,81	0,45	0,56	0,94	0,47	0,26	0,21	0,27	0,51	1,00

Tabela IV.11: Matriz de Concordância entre os critérios C_i e C_j na avaliação dos atributos agregados: subfatores, fatores do objetivo *Confiabilidade da Representação*

Como todos os $C_{ij} \neq 0$, isto é, há interseção entre os critérios *i* e *j*, não é necessário proceder-se ao cálculo do *grau da não concordância*, $\overline{C}_{ij}$, entre esses critérios. A não ocorrência de $\overline{C}_{ij}$ pode ser um forte indício de que a árvore hierárquica dos atributos, que está sendo avaliada, esteja disposta coerentemente.

5.4 ***Construção da matriz do estado de agregação***: neste experimento, a *matriz do estado de agregação*, *MEA*, é a mesma da *Tabela V.8*, uma vez que não há $\overline{C}_{ij}$.

5.5 ***Cálculo do estado relativo de agregação***: como não há $\overline{C}_{ij}$, procede-se ao cálculo de ERA_i, apresentado na *Tabela IV.12*.

5.6 ***Cálculo do grau do estado relativo de agregação***: o grau do estado relativo de agregação, *GERA*, de cada atributo agregado é obtido pela *média ponderada* entre seus critérios constituintes, como mostra a *Tabela IV.12*.

SUBFATOR: *Correção no Uso do Método*							
Critérios	**ERA*i***	**GERA*i***	**CCA*i***	$f(x)_L = ax + b$		$f(x)_R = ax + b$	
				a_L	b_L	a_R	b_R
1	0,7004	0,2672	0,2672	1,00	-2,55	-2,31	9,20
2	0,7018	0,2678	0,2678	1,00	-2,40	-1,82	7,19
3	0,6073	0,2317	0,2317	1,00	-2,86	-7,28	29,13
4	0,6115	0,2333	0,2333	1,00	-2,06	-1,31	5,03
Total	**Total**	**Prod.**	**Total**				
4	2,6209	1,0000	1,0000				

Tabela IV.12: Estado relativo de agregação, grau do estado relativo de agregação, coeficiente de consenso do atributo, e os argumentos *a* e *b* das funções lineares dos critérios, que participaram do processo de agregação.

5.7 ***Cálculo do coeficiente de consenso do atributo***: o coeficiente de consenso do atributo, obtido para cada atributo (critério) *i* (CCA_i), que compõe o atributo que está sendo agregado, considerou apenas o valor de $GERA_i$, porque se está estabelecendo o *padrão de qualidade* para ERS. Neste caso, $W_i = 1$ e, portanto, $CCA_i = GERA_i$ (*item 5.6* anterior).

5.8 ***Avaliação do atributo agregado***: o resultado da avaliação do Subfator Correção do Uso do Método, também será um número *fuzzy* $\tilde{N}$, e segue os mesmos procedimentos de cálculo do *item 4.6*, sendo dado por:

- $\tilde{N} = (2,47; 3,47; 3,94)$

IV.3.1 Características do Modelo *Fuzzy* para Avaliação da Qualidade de Software

O Modelo *fuzzy* para avaliação da qualidade de software, proposto neste capítulo, possui algumas características relevantes:

i. **Preservação da concordância** (HSU *et al*., 1996, BARDOSSY *et al*., 1993): se todas as estimativas forem idênticas, a combinação resultante será a estimativa comum entre elas, isto é, se $\tilde{N}_i = \tilde{N}_j$ para todo *i, j*, então $\tilde{N} = \tilde{N}_i$.

Prova: tomando-se o *item 5.8* (*Etapa 5*)

$$\tilde{N} = \sum_{i=1}^{n} (CCA_i \bullet \tilde{N}_i)$$

$$\tilde{N} = \tilde{N}_i \bullet \sum_{i=1}^{n} (CCA_i)$$

Como $\sum_{i=1}^{n} (CCA_i) = 1$, então, $\tilde{N} = \tilde{N}_i$

ii. **Independência da ordem de combinação** (BARDOSSY *et al*., 1993): o resultado da agregação não depende da ordem da combinação das opiniões individuais. Se {σ(1), σ(2), ..., σ(*n*)} é uma permutação de {1, 2, ..., 3}, então

$$\tilde{N} = f(\tilde{N}_1, \tilde{N}_2, ..., \tilde{N}_n) = f(\tilde{N}_{\sigma(1)}, \tilde{N}_{\sigma(2)}, ..., \tilde{N}_{\sigma(n)})$$

iii. **Influência do grau de concordância e do peso do especialista**: se a estimativa de um especialista se distancia da estimativas dos demais (pequeno índice de concordância), e/ou o peso do perfil desse especialista é inferior aos dos demais, então sua estimativa terá uma importância reduzida, no processo de avaliação.

iv. **Preservação do número *fuzzy***: se as opiniões de todos os especialistas podem ser representadas por um número *fuzzy* triangular normal positivo, então a função de pertinência da combinação é também um número *fuzzy* triangular normal positivo.

Baseando-se nos resultados obtidos deste modelo *fuzzy* para avaliação da qualidade de software, pode-se obter índices de qualidade, na avaliação de novos produtos de software, de acordo com um padrão de qualidade estabelecido.

IV.4 Definição de Índices de Qualidade de Software

Se houver um *padrão de qualidade* (*PQ*), já definido por este modelo ou por um outro compatível, para o produto de software, que está sendo avaliado ou seu domínio de aplicação, pode-se comparar os resultados obtidos pela aplicação do modelo *fuzzy* proposto com esse padrão estabelecido. Com isto, obtém-se um *índice de qualidade*, que indicará se o produto de software avaliado está ou não dentro do padrão de qualidade desejado, e em que percentagem.

Neste contexto, pode-se determinar o ***índice de qualidade*** do produto avaliado, através dos seguintes procedimentos:

1. ***Redefinição da função característica do atributo de qualidade***: redefinir a função *fuzzy* de cada atributo de qualidade do tipo $\tilde{N}_i = (a_i, m_i, b_i)$, obtida na definição do *padrão de qualidade*, para a *função fuzzy do padrão de qualidade*, $\tilde{Q}_i = (a_i, \underline{m}_i, \overline{m}_i, b_i)$, isto é, um número *fuzzy* normal trapezoidal. O mapeamento de $\tilde{N}_i$ para $\tilde{Q}_i$ resultará, então, na função $\tilde{Q}_i = (a_i, m_i, b_n, b_n)$, onde n é o limite superior do *conjunto referencial* já definido. Esse mapeamento é possível, porque pode-se considerar que qualquer valor acima do padrão de qualidade (à direita da função característica), é também de qualidade e, portanto, plenamente aceitável.

2. ***Cálculo do índice de qualidade***: o *índice de qualidade*, q_k, de cada atributo k, que está sendo avaliado, será formalizado por:

$$q_k = \frac{\int_x (\min\{\mu_{\tilde{Q}_i}(x), \mu_{\tilde{N}_k}(x)\})dx}{\int_x (\mu_{\tilde{N}_k}(x))dx}$$

   - Uma vez que $q \in [0, 1]$, quando $q = 1$, isto significa que o atributo avaliado atinge totalmente o padrão de qualidade; se $q = 0$, então o atributo avaliado está totalmente fora do padrão de qualidade; se $0 \leq q \leq 1$, o atributo está dentro do padrão de qualidade, na proporção do valor de q, isto é, o atributo avaliado é q % do padrão de qualidade.

- No cálculo do *índice de qualidade* para *atributos agregados*, se algum de seus atributos constituintes tiver $q = 1$, considerar, neste caso, o valor do *padrão de qualidade* (*PQ*) desse atributo constituinte.

IV.5 Conclusão

Neste capítulo, foi definido um modelo *fuzzy* para avaliação da qualidade de software, a partir do Modelo Rocha. Com a extensão do Modelo Rocha, o ganho mais significativo foi realizar a agregação de atributos de qualidade de software, através de conceitos e propriedades da teoria *fuzzy*. Com isto, esse modelo foi provido de uma fundamentação matemática, para o tratamento de suas medidas subjetivas.

No capítulo a seguir, mostrar-se-á os resultados de uma experiência de uso do modelo *fuzzy* proposto, realizada com o objetivo de validá-lo.

Capítulo V

UMA EXPERIÊNCIA DE UTILIZAÇÃO DO MODELO *FUZZY*

No capítulo anterior, foi definido o *modelo fuzzy para avaliação da qualidade de software*, sendo exemplificada cada uma de suas cinco etapa. Neste capítulo, é descrita a experiência completa de sua utilização com especificações de requisitos de software (ERS), realizada com o objetivo de validá-lo. Um atributo de qualidade de software pode ser avaliado, segundo sua função característica, com é mostrado a seguir.

V.1 Avaliação dos Atributos de Qualidade

Cada atributo de qualidade de software avaliado pode ser representado por sua função de pertinência (característica) correspondente, como, por exemplo, o critério Correção da Notação, exemplificado no Capítulo anterior, e mostrado na *Figura V.1*.

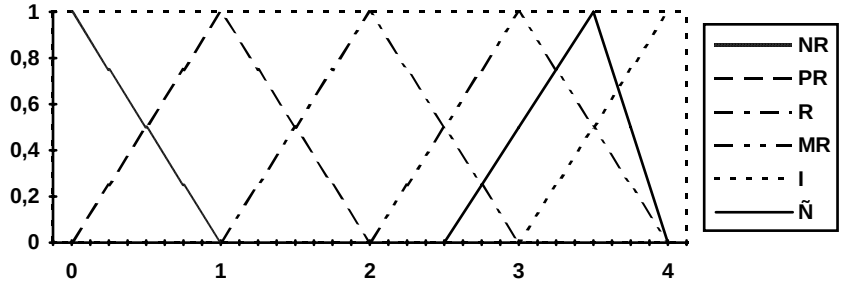


Figura V.1: Critério Correção da Notação

De acordo com a *Figura V.1*, pode-se obter as equações *Eq. V.1* do atributo de qualidade avaliado, que também é um número *fuzzy* do tipo $\tilde{N} = (a, m, b)$, através da seguinte fórmula:

_

$$f_{\tilde{N}}(x) = \begin{cases} \left(\frac{1}{m-a}\right)x - \left(\frac{1}{m-a}\right)a, & a \le x \le b \\ \left(\frac{1}{b-m}\right)b - \left(\frac{1}{b-m}\right)x, & m \le x \le b \end{cases} \qquad (Eq.\ V.1)$$

Assim sendo, o critério *Correção da Notação* tem a seguinte função de pertinência:

$$f_{\tilde{N}}(x) = \begin{cases} x - 1, & 2{,}55 \le x \le 3{,}55 \\ 9{,}068 - 0{,}44x, & 3{,}55 \le x \le 3{,}99 \end{cases}$$

O valor de m = 3,55 do critério avaliado, que corresponde ao *valor de qualidade desejável* desse critério (na matemática clássica), pode ser utilizado nas equações dos termos lingüísticos MR (*muito relevante*) e I (*imprescindível*), conforme *Seção IV.2*, uma vez que este valor pertence ao intervalo de abrangência de cada uma dessas funções: [3, 4]. Assim sendo:

- Para MR: $f(x) = 4 - x$; para $3 \le x \le 4$
  $= 4 - 3{,}55 = \underline{0{,}45}$
- Para I: $f(x) = x - 3$; para $3 \le x \le 4$
  $= 3{,}55 - 3 = \underline{0{,}55}$

Pode-se, então, fazer as seguintes deduções a respeito do critério *Correção da Notação*, para especificações de requisitos de software em geral:

i. Na linguagem matemática clássica, considerando-se a escala numérica de 0 a 4, previamente estabelecida, *Tabela IV.3*, o padrão de qualidade do critério avaliado encontra-se no intervalo de [2,55; 3,99] e o seu *valor de qualidade desejável* é 3,55.

ii. Na linguagem da teoria *fuzzy*, o padrão de qualidade do critério avaliado é o número *fuzzy* $\tilde{N}$ = (2,55; 3,55; 3,99), cuja função de pertinência está na *Figura V.1* e o seu *valor de qualidade desejável* é 45% *muito relevante* e 55% *imprescindível*.

A seguir, serão mostrados todos os experimentos realizados, através do modelo *fuzzy*, para ERS, com o objetivo de validá-lo.

V.2 Uma Experiência com o Modelo *Fuzzy*

Considerando-se as situações previstas para a utilização do modelo *fuzzy*, descritas na *Seção IV.3*, foram objeto do experimento as seguintes:

❶ ***Determinação do Padrão de Qualidade (PQ) de um produto de software***. Um conjunto de 16 especialistas, em engenharia de software, avaliou um conjunto de atributos de qualidade, identificados para especificações de requisitos de software (ERS). A definição desses atributos, cuja lista está no *Apêndice II*, e a pesquisa de campo correspondente foram realizadas por CLUNIE (1997). O objetivo era obter de cada especialista o grau de importância de cada um dos atributos avaliados, para que uma especificação de requisitos fosse considerada de qualidade, isto é, o *padrão de qualidade* que uma especificação de requisitos deveria apresentar.

❷ ***A avaliação da qualidade de um produto de software, apoiada em um PQ já previamente definido***. Um grupo de 3 especialistas avaliou a *modelagem de dados* do *módulo financeiro* do Sistema SIGAH-Multimídia, um sistema de informação hospitalar em desenvolvimento, na Unidade de Cardiologia e Cirurgia Cardiovascular do Hospital Universitário da UFBA / Fundação Bahiana de Cardiologia (UCCV/FBC), com a colaboração da área de Engenharia de Software da COPPE/UFRJ. Nesta avaliação, foi considerado um subconjunto dos atributos de qualidade de ERS, relacionados no *Apêndice IV*. Os resultados destes julgamento foram confrontados com o *PQ*, estabelecido para ERS, obtidos no item anterior. Desta forma, foram gerados *índices de qualidade*, para cada atributo avaliado, chegando-se à *medição da qualidade* do *modelo de dados*, um subproduto da ERS.

A seguir, serão descritos os resultados das duas situações expostas acima.

V.2.1 Determinação do Padrão de Qualidade para ERS

A determinação do ***padrão de qualidade*** (*PQ*) para especificações de requisitos de software segue as etapas do modelo *fuzzy* para avaliação da qualidade de software (MFAQS), *Seção IV.3*, segundo a notação utilizada no Modelo Rocha Estendido, *Seção IV.1*. Foram analisados todos os *critérios*, *subfatores* e *fatores* para cada um dos três *objetivos* de qualidade, apresentados no *Apêndice II*, para ERS em geral.

A *Tabela V.1* apresenta os resultados obtidos, através da aplicação do MFAQS, para o objetivo da *Confiabilidade da Representação* de especificações de requisitos de

software, incluindo os pesos correspondentes para cada um desses critérios na agregação de subfatores (W_{SUB}), de fatores (W_{FAT}) e do próprio objetivo de qualidade (W_{OBJ}).

–

ATRIBUTOS DE QUALIDADE DE SOFTWARE	*PQ* para ERS	W_{SUB}	W_{FAT}	W_{OBJ}
OBJETIVO:CONFIABILIDADE DA REPRESENTAÇÃO	$\tilde{N}$ (2,48; 3,48; 3,93)			
FATOR: COMUNICABILIDADE (COB)	$\tilde{N}$ (2,39; 3,39; 3,91)			
Subfator: Correção no Uso do Método (CUM)	$\tilde{N}$ (2,47; 3,47; 3,94)			
Correção da Notação (CNT)	$\tilde{N}$ (2,55; 3,55; 3,99)	0,2560	0,0760	0,0544
Correção Sintática (CST)	$\tilde{N}$ (2,40; 3,40; 3,95)	0,2449	0,0727	0,0520
Correção Semântica (CSM)	$\tilde{N}$ (2,86; 3,86; 4,00)	0,2784	0,0826	0,0591
Correção no Uso do Formato de Document. (CFD)	$\tilde{N}$ (2,06; 3,06; 3,82)	0,2207	0,0655	0,0469
Subfator: Uniformidade de Terminologia (UTE)	$\tilde{N}$ (2,66; 3,66; 4,00)			
Uniformidade de Termos (UTT)	$\tilde{N}$ (2,74; 3,74; 4,00)	0,5114	0,0800	0,0573
Uniformidade de Notação (UNT)	$\tilde{N}$ (2,57; 3,57; 4,00)	0,4886	0,0765	0,0547
Subfator:Uniformidade no Nível deAbstração(UNA)	$\tilde{N}$ (1,55; 2,55; 3,48)			
Uniformidade de Detalhes daDocumentação(UDD)	$\tilde{N}$ (1,72; 2,72; 3,65)	0,5336	0,0581	0,0416
Independência de Detalhes do Projeto (IDP)	$\tilde{N}$ (1,37; 2,37; 3,31)	0,4664	0,0508	0,0364
Subfator: Modularidade da Documentação (MDO)	$\tilde{N}$ (2,36; 3,36; 3,97)			
Coesão de Informações (COI)	$\tilde{N}$ (2,35; 3,35; 4,00)	0,3321	0,0717	0,0513
Acoplamento entre as Seções (ACS)	$\tilde{N}$ (2,10; 3,10; 3,91)	0,3071	0,0662	0,0474
Estrutura da Documentação (EDO)	$\tilde{N}$ (2,64; 3,64; 3,99)	0,3608	0,0778	0,0557
Subfator: Concisão (COC)	$\tilde{N}$ (2,55; 3,55; 3,86)			
Complementabilidade (COM)	$\tilde{N}$ (2,55; 3,55; 3,86)	1,0000	0,0759	0,0543
Subfator: Conformidade (COF)	$\tilde{N}$ (2,41; 3,41; 3,97)			
Aderência às Normas Org. Desenvolvedora (ANO)	$\tilde{N}$ (2,22; 3,22; 3,95)	0,4709	0,0688	0,0492
Aderência às Normas estab. p/ Contratante (ANC)	$\tilde{N}$ (2,61; 3,61; 3,99)	0,5291	0,0773	0,0553
FATOR: MANIPULABILIDADE (MAP)	$\tilde{N}$ (2,75; 3,75; 3,98)			
Subfator: Disponibilidade (DIS)	$\tilde{N}$ (2,95; 3,95; 4,00)			
Acessibilidade (ACE)	$\tilde{N}$ (2,90; 3,90; 4,00)	0,4940	0,2102	0,0598
Estar Atualizada (ETA)	$\tilde{N}$ (3,00; 4,00; 4,00)	0,5060	0,2153	0,0612
Subfator: Rastreabilidade (RAS)	$\tilde{N}$ (2,56; 3,56; 3,96)			
Organização da Documentação (ORD)	$\tilde{N}$ (2,88; 3,88; 4,00)	0,3638	0,2090	0,0595
Localizabilidade Interna (LOI)	$\tilde{N}$ (2,56; 3,56; 4,00)	0,3336	0,1917	0,0545
Localizabilidade Externa (LOE)	$\tilde{N}$ (2,23; 3,23; 3,87)	0,3025	0,1738	0,0494

Tabela V.1: Avaliação do objetivo Confiabilidade da Representação de ERS, através de números *fuzzy*.

Com base nos dados da tabela acima, foi elaborada a *Figura V.2*, que apresenta os resultados *defuzificados*, isto é, o valor de *m* do número *fuzzy* $\tilde{N}$ = (*a*, *m*, *b*) deste objetivo de qualidade, em ordem descendente do grau de relevância de seus critérios constituintes, para ERS em geral.

_

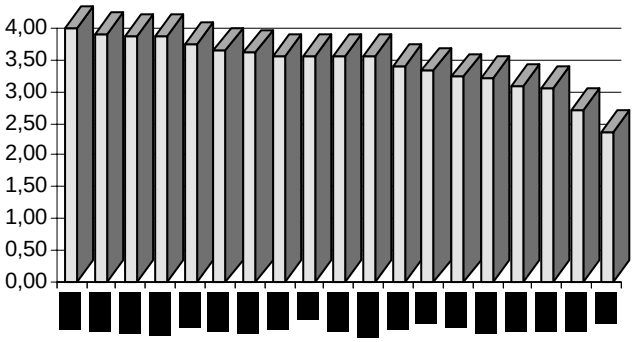


Figura V.2: Hierarquização dos critérios do objetivo Confiabilidade da Representação para ERS

Observa-se que 89% dos critérios desse objetivo estão entre MI (*muito importante*) e I (*imprescindível*), segundo o conjunto de termos lingüísticos da *Tabela IV.3*.

Como está evidenciado no gráfico acima, os critérios *Estar Atualizada* (ETA) e *Acessibilidade* (ACE) possuem os graus de relevância mais altos. Segundo CLUNIE (1997), esta tendência pode ser entendida pela preocupação de se ter uma especificação que possa ser facilmente manuseada por seus diversos usuários e na versão atualizada, ao longo do processo de desenvolvimento, devendo ainda atender às necessidades de manutenção.

Os critérios *Uniformidade de Detalhes da Documentação* (UDD) e *Independência de Detalhes de Projeto* (IDP) são os que apresentaram graus de relevância mais baixos. Isto pode ser explicado porque, na elaboração de uma especificação de requisitos de software, geralmente, muitas questões ainda não se encontram bem definidas, nem completamente entendidas. Assim sendo, neste contexto, esses dois critérios não se tornam tão importantes quanto os outros.

A *Tabela V.2* apresenta os dados obtidos, através da aplicação do MFAQS, para o objetivo da *Confiabilidade Conceitual* de especificações de requisitos de software, incluindo os pesos correspondentes a cada um desses critérios na agregação de subfatores (W_{SUB}), de fatores (W_{FAT}) e do próprio objetivo de qualidade (W_{OBJ}).

ATRIBUTOS DE QUALIDADE DE SOFTWARE	*PQ* para ERS	W_{SUB}	W_{FAT}	W_{OBJ}
OBJETIVO: CONFIABILIDADE CONCEITUAL	$\tilde{N}$ (2,49; 3,49; 3,90			
FATOR: FIDEDIGNIDADE (FID)	$\tilde{N}$ (2,69; 3,69; 3,97)			
Subfator: Consistência (CON)	$\tilde{N}$ (2,67; 3,67; 3,94)			
Consistência Interna (COI)	$\tilde{N}$ (2,89; 3,89; 4,00)	0,5303	0,2640	0,1257
Consistência Externa (COE)	$\tilde{N}$ (2,45; 3,45; 3,88)	0,4697	0,2338	0,1113
Subfator: Não Ambigüidade (NAB)	$\tilde{N}$ (2,70; 3,70; 3,99)			

–

		Ser Explícita EXP)	$\tilde{N}$ (2,81; 3,81; 3,99)	0,5143	0,2583	0,1230
		Precisão (PRC)	$\tilde{N}$ (2,60; 3,60; 3,99)	0,4857	0,2439	0,1161
FATOR: SUFICIÊNCIA (SUF)			$\tilde{N}$ (2,23; 3,23; 3,81)			
	Subfator: Necessidade (NEC)		$\tilde{N}$ (2,92; 3,92; 4,00)			
		Necessidade dos Requisitos (NRQ)	$\tilde{N}$ (2,92; 3,92; 4,00)	1,0000	0,2412	0,1264
	Subfator: Não Redundância (NRD)		$\tilde{N}$ (1,84; 2,84; 3,61)			
		Não Redundância de Informações (NRI)	$\tilde{N}$ (1,84; 2,84; 3,61)	1,0000	0,1746	0,0915
	Subfator: Completitude (COP)		$\tilde{N}$ (2,18; 3,18; 3,83)			
		Completitude Relação Roteiro def. p/ Org. (CRO)	$\tilde{N}$ (2,28; 3,28; 3,89)	0,3462	0,2022	0,1060
		Completitude c/Relação Método Desenv(CMM)	$\tilde{N}$ (1,63; 2,63; 3,58)	0,2775	0,1621	0,0849
		Completitude com Relação aos Requisitos(COR)	$\tilde{N}$ (2,57; 3,57; 4,00)	0,3763	0,2199	0,1152

Tabela V.2: Avaliação do Objetivo Confiabilidade Conceitual de ERS, através de números *fuzzy*.

Com base nos dados da tabela acima, a *Figura V.3* mostra, também, os resultados *defuzificados* em ordem descendente do grau de relevância dos critérios do objetivo confiabilidade conceitual, para ERS em geral.

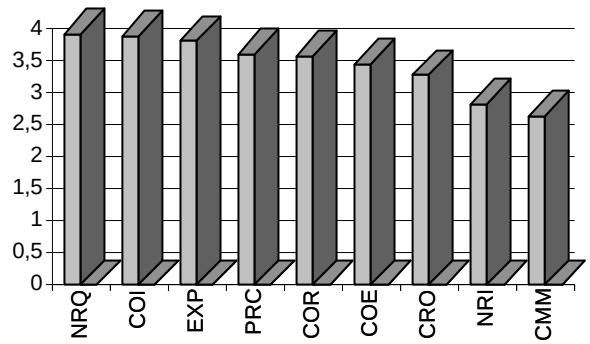


Figura V.3: Hierarquização dos critérios do objetivo Confiabilidade Conceitual para ERS

Um índice de 77% dos critérios pertencentes a este objetivo de qualidade, também, estão no intervalo de *muito importante* a *imprescindível*, enfatizando o quão é importante o entendimento, a consistência e a completitude conceitual da especificação. Se uma especificação não é precisa, e devidamente definida, por certo não atenderá, plenamente, às necessidades e reivindicações de seus usuários.

O critério mais relevante, neste grupo, é *Necessidade dos Requisitos* (NRQ). Assim sendo, é extremamente necessário que os requisitos considerados imprescindíveis ao produto sejam descritos, pois, sem eles, a especificação perde o sentido.

O critério, que vem logo a seguir em grau de importância, é a *Consistência Interna* (COI), uma vez que é de vital para a consecução acurada de uma especificação de requisitos, que não existam conflitos entre partes da mesma.

Já os critérios *N*ão *Redundância de Informações* (NRI) e *Completitude com Relação ao Método de Desenvolvimento* (CMM) foram consideram de menor monta. Isto pode ser atribuído ao fato de que, durante os procedimentos de elaboração de uma ERS, ainda não há consenso e um direcionamento bem definido sobre muitos aspectos conceituais. Portanto, algumas questões podem se repetir, até mesmo para serem melhor elucidadas. Por este mesmo motivo, pode-se não se ter ainda todas as informações requeridas por um método de desenvolvimento.

A *Tabela V.3* mostra os dados obtidos, através da aplicação do MFAQS, para o objetivo *Utilizabilidade* de especificações de requisitos de software, considerando, também, os pesos correspondentes a cada de seus critérios na agregação de subfatores (W_{SUB}), de fatores (W_{FAT}) e do próprio objetivo de qualidade (W_{OBJ}).

ATRIBUTOS DE QUALIDADE DE SOFTWARE	*PQ* para ERS	W_{SUB}	W_{FAT}	W_{OBJ}
OBJETIVO: UTILIZABILIDADE	- - -			
FATOR: AVALIABILIDADE (AVA)	$\tilde{N}$ (2,51; 3,51; 3,93)			
Subfator: Verificabilidade (VER)	- - -			
Subfator: Validabilidade (VAL)	- - -			
FATOR: MANUTENIBILIDADE (MAT)	$\tilde{N}$ (2,51; 3,51; 3,92)			
Subfator: Modificabilidade (MOD)	- - -			
Subfator: Evolutibilidade (EVO)	- - -			
FATOR: REUTILIZABILIDADE (REU)	$\tilde{N}$ (2,60; 3,60; 3,98)			
Subfator: Adaptabilidade (ADA)	- - -			
Subfator: Generalidade (GEN)	- - -			
FATOR: IMPLEMENTABILIDADE (IMP)	$\tilde{N}$ (2,74; 3,74; 3,97)			
Subfator: Viabilidade Econômica (VEC)	$\tilde{N}$ (2,82; 3,82; 4,00)			
Aceitabilidade de Custos (ACC)	$\tilde{N}$ (2,80; 3,80; 3,99)	0,3319	0,0792	
Relevância dos Benefícios (REB)	$\tilde{N}$ (2,90; 3,90; 4,00)	0,3408	0,0813	
Compatibilidade Custo/Benefício (CCB)	$\tilde{N}$ (2,75; 3,75; 4,00)	0,3273	0,0781	
Subfator: Viabilidade Financeira (VFI)	$\tilde{N}$ (2,96; 3,96; 4,00)			
Existência de Capital (ECT)	$\tilde{N}$ (2,99; 3,99; 4,00)	0,5039	0,0831	
Disponibilidade de Capital (DCT)	$\tilde{N}$ (2,93; 3,93; 4,00)	0,4961	0,0818	
Subfator: Viabilidade Tecnológica (VTE)	$\tilde{N}$ (2,89; 3,89; 4,00)			
Existência da Tecnologia (ETE)	$\tilde{N}$ (2,96; 3,96; 4,00)	0,5094	0,0825	
Disponibilidade da Tecnologia (DTE)	$\tilde{N}$ (2,82; 3,82; 4,00)	0,4906	0,0795	
Subfator: Viabilidade de Mão de Obra (VMO)	$\tilde{N}$ (2,66; 3,66; 3,96)			
Existência de Mão de Obra (EMO)	$\tilde{N}$ (2,63; 3,63; 3,99)	0,4961	0,0756	
Disponibilidade de Mão de Obra (DMO)	$\tilde{N}$ (2,69; 3,69; 3,94)	0,5039	0,0768	
Subfator: Viabilidade de Cronograma (VCR)	$\tilde{N}$ (2,71; 3,71; 4,00)			
Adequabilidade de Cronograma (ACR)	$\tilde{N}$ (2,60; 3,60; 4,00)	0,4845	0,0749	
Flexibilidade de Cronograma (FCR)	$\tilde{N}$ (2,83; 3,83; 4,00)	0,5155	0,0797	
Subfator: Viabilidade Social (VSO)	$\tilde{N}$ (2,06; 3,06; 3,79)			
Aceitabilidade da Engenharia Humana (AEH)	$\tilde{N}$ (1,91; 2,91; 3,69)	0,4743	0,0605	
Aceitabilidade dos Impactos Sociais (AIS)	$\tilde{N}$ (2,22; 3,22; 3,89)	0,5257	0,0671	

Tabela V.3: Avaliação do objetivo Utilizabilidade de ERS, através de números *fuzzy*.

A *Figura V.4* apresenta a hierarquização dos critérios que fazem parte apenas do fator *Implementabilidade*. Os outros fatores desse objetivo, segundo CLUNIE (1997) estão relacionados a critérios dos objetivos *Confiabilidade da Representação* e *Confiabilidade Conceitual*, que foram analisados anteriormente.

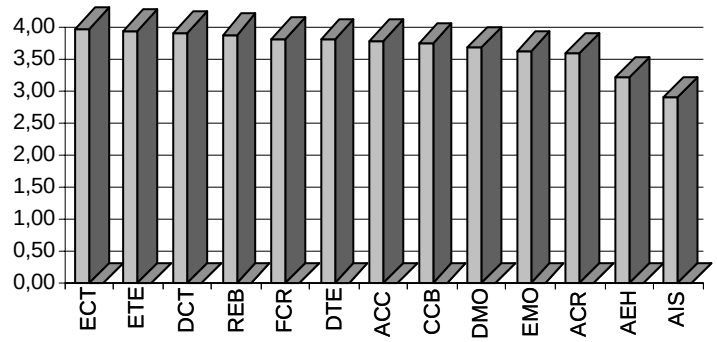


Figura V.4: Hierarquização dos critérios do objetivo Utilizabilidade para ERS.

Novamente, a grande maioria dos critérios desse objetivo de qualidade se inserem no intervalo *muito importante* e *imprescindível* (cerca de 92%). Os atributos de qualidade *Existência de Capital* (ECT) e *Existência de Tecnologia* (ETE) são os mais importantes neste grupo. Realmente, se não existirem recursos pecuniários e tecnológicos para a implementação de uma ERS, por certo ela não será viável.

Os critérios *Aceitabilidade dos Impactos Sociais* (AIS) e *Aceitabilidade da engenharia humana* (AEH) foram os que tiveram a menor relevância, entre os avaliadores. Como esta pesquisa de campo foi realizada com um grupo de especialistas, essencialmente técnico, provavelmente os possíveis impactos sociais decorrente do desenvolvimento de uma ERS foram, por eles, visualizados de forma secundária.

O critério AEH diz respeito ao grau de satisfação e ao desenvolvimento do potencial humano dos usuários. É provável que este baixo escore reflita ainda o tradicional conflito entre usuários e especialistas em computação (BELCHIOR, 1994, FREEDMAN, 1993). MUMFORD (1972) já alertava para a *excepcional falta de contato entre os especialistas de computador e os usuários finais, que possuem deficiências alarmantes não somente em consultas mas em relação à comunicação*.

A *Figura V.5* mostra a disposição dos subfatores de qualidade avaliados para ERS, também, em ordem decrescente do grau de importância. Os subfatores são atributos agregados de qualidade, compostos por um conjunto de critérios, segundo o Modelo Rocha estendido.

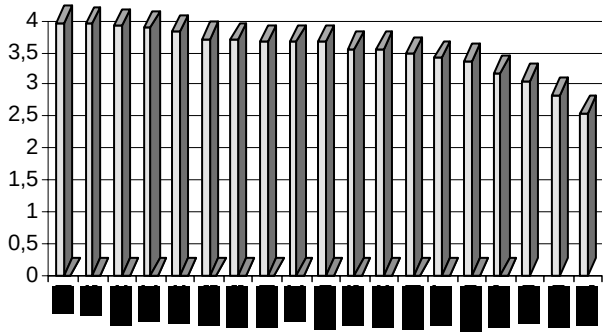


Figura V.5: Hierarquização dos subfatores de qualidade de ERS

Dentre os subfatores avaliados, a *Viabilidade Financeira* (VFI), a *Disponibilidade* (DIS) e *Necessidade* (NEC) foram, respectivamente, os mais relevantes, o que condiz com a avaliação dos critérios a eles pertinentes.

Sem sombra de dúvidas, o desenvolvimento de um ERS sem a existência e a disponibilidade de um suporte financeiro para este fim, dificilmente terá algum êxito. Uma vez atendido a este pre-requisito, uma ERS não atualizada e de difícil acesso impactará, diretamente e de forma negativa, no desenvolvimento do próprio produto.

Mesmo que uma ERS atenda convenientemente aos dois subfatores anteriores, mas seus requisitos tidos como imprescindíveis não forem estabelecidos devidamente, estará fadada ao insucesso.

Os subfatores de menor relevância, neste grupo, são, respectivamente, *Uniformidade de Abstração*, *Não Redundância* e *Viabilidade Social*, o que condiz com a avaliação de seus respectivos critérios de qualidade.

A *Figura V.6* apresenta a agregação dos subfatores de qualidade avaliados para ERS, em ordem decrescente de seu grau de relevância. Os fatores que mais se destacaram foram a *Manipulabilidade* (MAP), *Implementabilidade* (IMP) e *Fidedignidade* (FID).

–

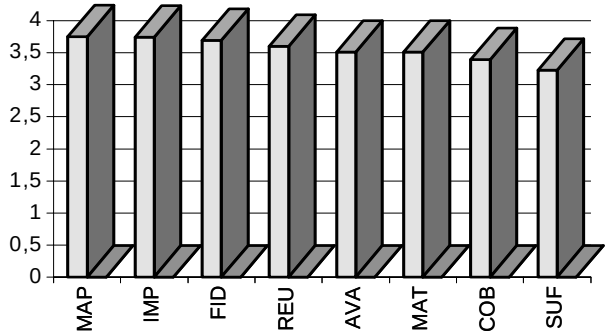


Figura V.6: Hierarquização dos fatores de qualidade de ERS

Portanto, deve-se procurar desenvolver especificações de requisitos que sejam facilmente manipuláveis, para diversas formas de uso, isto é, estejam disponíveis para seus usuários em sua versão mais atualizada e cujo conteúdo possa ser rastreado sem muito custo ou esforço, desde sua visão mais global até a mais detalhada e vice-versa.

O fator *Implementabilidade* (IMP) veio em segundo lugar em grau de importância, embora possua o subfator *Viabilidade Financeira* (VFI), avaliado como o mais relevante entre todos os subfatores. Esse fator, também, contém o subfator *Viabilidade Social* (VSO), avaliado com sendo de pouca relevância. Neste sentido pode-se fazer algumas ponderações:

i. O subfator *Viabilidade Social* deveria, realmente, compor o ramo da árvore hierárquica de qualidade do fator *Implementabilidade*?

ii. O processo de avaliação dos atributos agregados deveria simplesmente assumir o valor de seu atributo constituinte de maior importância, para ressaltar o grau de qualidade deste? Ou deveria assumir o grau de seu atributo de menor importância, considerado como o *elo mais frágil da corrente* (do grupo)?

O processo de avaliação dos atributos agregados, utilizado pelo MFAQS, leva em consideração todos os graus de importância de cada atributo, que compõe o ramo da árvore hierárquica, que está sendo agregado. O resultado desta agregação tende sempre (pela estrutura do próprio modelo - quinta etapa, capítulo IV) para o maior grau de concordância entre os atributos da composição, como também considera o peso relativo de cada um deles, segundo o padrão de qualidade apurado, se for o caso.

V.2.2 Avaliação de uma ERS

–

Na seção anterior, foi estabelecido um *padrão de qualidade* (*PQ*) para ERS em geral, através do MFAQS proposto. Através deste padrão, pode obter-se *índices de qualidade* (*Seção IV.4*), para a avaliação de especificações, o que indicará se ela está ou não dentro do *PQ* estabelecido.

Foi avaliada a *Modelagem de dados*, subproduto da *Especificação de Requisitos do Módulo Financeiro* do sistema SIGAH-Multimídia, da UCCV/FBC.

A avaliação realizada por 3 especialistas, que não participaram de sua confecção, baseados em uma cópia da mesma, fornecida a cada um deles. A técnica de avaliação utilizada foi *inspeção individual*.

O perfil de cada especialista avaliador está na *Tabela V.4*, que retrata a apuração dos resultado do *QIPE (Questionário de Identificação do Perfil do Especialista*).

Apuração dos Resultados do *QIPE*									
E_i / item	i_1	i_2	i_3	i_4	i_5	i_6	i_7	*tQIPE*	p_{Ei}
1	2,80	1,30	2,40	0,30	0,70	0,80	5,90	4,177	0,260
2	2,70	6,00	2,50	0,70	1,00	0,80	9,00	5,990	0,373
3	3,10	3,00	3,90	0,70	0,90	0,80	9,20	5,900	0,367
Total								**Total**	**Total**
3								16,067	1,000

Tabela V.4.: Resultados do QIPE para ERS real

O conjunto de termos lingüísticos utilizados nesta avaliação está descrito no questionário CAERS, *Apêndice IV*, e possui uma relação direta com os termos lingüísticos, estabelecidos no levantamento do *padrão de qualidade* (*PQ*), com é mostrado na *Tabela V.5* abaixo.

Grau de importância	**Termo Lingüístico do *PQ***	**Termo Lingüístico da CAERS**	**Número *fuzzy* normal**
0,0	Nenhuma Relevância	Total Ausência	$\tilde{\tilde{N}}_1$ = (0,0; **0,0**; 1,0)
1,0	Pouca Relevância	Baixa Presença	$\tilde{\tilde{N}}_2$ = (0,0; **1,0**; 2,0)
2,0	Relevante	Moderada Presença	$\tilde{\tilde{N}}_3$ = (1,0; **2,0**; 3,0)
3,0	Muito Relevante	Alta Presença	$\tilde{\tilde{N}}_4$ = (2,0; **3,0**; 4,0)
4,0	Imprescindível	Total presença	$\tilde{\tilde{N}}_5$ = (3,0; **4,0**; 4,0)

Tabela V.5: Relação entre os números *fuzzy* normais do *PQ* e da CAERS

Na avaliação, foi considerado um subconjunto dos atributos de qualidade de ERS, pertinente ao subproduto sob avaliação. Este subconjunto, pode ser identificado nas *Tabelas V.6* e *V.7*.

Os resultados dessa avaliação e seus respectivos *índices de qualidade* (*IQ*), também são exibidos nas *Tabelas V.6* e *V.7*.

ATRIBUTOS DE QUALIDADE DE SOFTWARE	*PQ* para ERS	ERS	*IQ*
OBJETIVO:CONFIABILIDADE DA REPRESENTAÇÃO	$\tilde{N}$ (2,48; 3,48; 3,93)	- - -	
FATOR: COMUNICABILIDADE (COB)	$\tilde{N}$ (2,39; 3,39; 3,91)	- - -	
Subfator: Correção no Uso do Método (CUM)	$\tilde{N}$ (2,47; 3,47; 3,94)	$\tilde{N}$ (3,00; 4,00; 4,00)	1,000
Correção da Notação (CNT)	$\tilde{N}$ (2,55; 3,55; 3,99)	$\tilde{N}$ (3,00; 3,00; 4,00)	1,000
Correção Sintática (CST)	$\tilde{N}$ (2,40; 3,40; 3,95)	$\tilde{N}$ (3,00; 3,00; 4,00)	1,000
Correção Semântica (CSM)	$\tilde{N}$ (2,86; 3,86; 4,00)	$\tilde{N}$ (3,00; 4,00; 4,00)	1,000
Correção no Uso do Formato de Document. (CFD)	$\tilde{N}$ (2,06; 3,06; 3,82)	$\tilde{N}$ (3,00; 4,00; 4,00)	1,000
Subfator: Uniformidade de Terminologia (UTE)	$\tilde{N}$ (2,66; 3,66; 4,00)	$\tilde{N}$ (2,17; 3,17; 3,75)	0,477
Uniformidade de Termos (UTT)	$\tilde{N}$ (2,74; 3,74; 4,00)	$\tilde{N}$ (2,17; 3,17; 3,51)	0,328
Uniformidade de Notação (UNT)	$\tilde{N}$ (2,57; 3,57; 4,00)	$\tilde{N}$ (2,17; 3,17; 4,00)	0,607
Subfator:Uniformidade no Nível deAbstração(UNA)	$\tilde{N}$ (1,55; 2,55; 3,48)	$\tilde{N}$ (2,47; 3,47; 4,00)	1,000
Uniformidade de Detalhes daDocumentação(UDD)	$\tilde{N}$ (1,72; 2,72; 3,65)	$\tilde{N}$ (2,00; 3,00; 4,00)	1,000
Independência de Detalhes do Projeto (IDP)	$\tilde{N}$ (1,37; 2,37; 3,31)	$\tilde{N}$ (3,00; 4,00; 4,00)	1,000
Subfator: Modularidade da Documentação (MDO)	$\tilde{N}$ (2,36; 3,36; 3,97)	- - -	
Coesão de Informações (COI)	$\tilde{N}$ (2,35; 3,35; 4,00)	- - -	
Acoplamento entre as Seções (ACS)	$\tilde{N}$ (2,10; 3,10; 3,91)	- - -	
Estrutura da Documentação (EDO)	$\tilde{N}$ (2,64; 3,64; 3,99)	- - -	
Subfator: Concisão (COC)	$\tilde{N}$ (2,55; 3,55; 3,86)	$\tilde{N}$ (1,65; 2,65; 3,27)	0,199
Complementabilidade (COM)	$\tilde{N}$ (2,55; 3,55; 3,86)	$\tilde{N}$ (1,65; 2,65; 3,27)	0,199
Subfator: Conformidade (COF)	$\tilde{N}$ (2,41; 3,41; 3,97)	- - -	
Aderência às Normas Org. Desenvolvedora (ANO)	$\tilde{N}$ (2,22; 3,22; 3,95)	- - -	
Aderência às Normas estab. p/ Contratante (ANC)	$\tilde{N}$ (2,61; 3,61; 3,99)	- - -	
FATOR: MANIPULABILIDADE (MAP)	$\tilde{N}$ (2,75; 3,75; 3,98)	- - -	
Subfator: Disponibilidade (DIS)	$\tilde{N}$ (2,95; 3,95; 4,00)	$\tilde{N}$ (2,48; 3,48; 3,93)	
Acessibilidade (ACE)	$\tilde{N}$ (2,90; 3,90; 4,00)	$\tilde{N}$ (2,32; 3,32; 3,66)	0,000
Estar Atualizada (ETA)	$\tilde{N}$ (3,00; 4,00; 4,00)	$\tilde{N}$ (1,04; 2,04; 2,70)	0,000
Subfator: Rastreabilidade (RAS)	$\tilde{N}$ (2,56; 3,56; 3,96)	- - -	
Organização da Documentação (ORD)	$\tilde{N}$ (2,88; 3,88; 4,00)	- - -	
Localizabilidade Interna (LOI)	$\tilde{N}$ (2,56; 3,56; 4,00)	$\tilde{N}$ (0,00; 1,00; 1,00)	0,000
Localizabilidade Externa (LOE)	$\tilde{N}$ (2,23; 3,23; 3,87)	- - -	

Tabela V.6: Resultados da avaliação para o objetivo Confiabilidade da Representação

ATRIBUTOS DE QUALIDADE DE SOFTWARE	*PQ* para ERS	ERS	*IQ*
OBJETIVO: CONFIABILIDADE CONCEITUAL	$\tilde{N}$ (2,49; 3,49; 3,90	- - -	
FATOR: FIDEDIGNIDADE (FID)	$\tilde{N}$ (2,69; 3,69; 3,97)	- - -	
Subfator: Consistência (CON)	$\tilde{N}$ (2,67; 3,67; 3,94)	$\tilde{N}$ (2,90; 3,90; 4,00	0,961
Consistência Interna (COI)	$\tilde{N}$ (2,89; 3,89; 4,00)	$\tilde{N}$ (2,82; 3,82; 4,00	0,876
Consistência Externa (COE)	$\tilde{N}$ (2,45; 3,45; 3,88)	$\tilde{N}$ (3,00; 4,00; 4,00	1,000
Subfator: Não Ambigüidade (NAB)	$\tilde{N}$ (2,70; 3,70; 3,99)	$\tilde{N}$ (2,71; 3,71; 4,00	0,871
Ser Explícita EXP)	$\tilde{N}$ (2,81; 3,81; 3,99)	$\tilde{N}$ (3,00; 4,00; 4,00	1,000
Precisão (PRC)	$\tilde{N}$ (2,60; 3,60; 3,99)	$\tilde{N}$ (2,40; 3,40; 4,00	0,770
FATOR: SUFICIÊNCIA (SUF)	$\tilde{N}$ (2,23; 3,23; 3,81)	$\tilde{N}$ (2,88; 3,88; 4,00	1,000
Subfator: Necessidade (NEC)	$\tilde{N}$ (2,92; 3,92; 4,00)	$\tilde{N}$ (3,00; 4,00; 4,00	1,000
Necessidade dos Requisitos (NRQ)	$\tilde{N}$ (2,92; 3,92; 4,00)	$\tilde{N}$ (3,00; 4,00; 4,00	1,000
Subfator: Não Redundância (NRD)	$\tilde{N}$ (1,84; 2,84; 3,61)	$\tilde{N}$ (2,82; 3,82; 4,00	1,000
Não Redundância de Informações (NRI)	$\tilde{N}$ (1,84; 2,84; 3,61)	$\tilde{N}$ (2,82; 3,82; 4,00	1,000
Subfator: Completitude (COP)	$\tilde{N}$ (2,18; 3,18; 3,83)	- - -	
Completitude Relação Roteiro def. p/ Org. (CRO)	$\tilde{N}$ (2,28; 3,28; 3,89)	- - -	
Completitude c/Relação Método	$\tilde{N}$ (1,63; 2,63; 3,58)	$\tilde{N}$ (2,00; 3,00; 4,00	1,000

		Desenv(CMM)			
		Completitude com Relação aos Requisitos(COR)	$\tilde{N}$ (2,57; 3,57; 4,00)	- - -	

Tabela V.7: Resultados da avaliação para o objetivo Confiabilidade Conceitual

A partir dos resultados apresentados nas duas tabelas anteriores, pode-se inferir que o produto avaliado está dentro do *padrão de qualidade* (*PQ*) para ERS, segundo o MFAQS. Neste caso, dos atributos avaliados pelos especialistas, 58% estão com *PQ* = 1, isto é, com qualidade ideal, 26% dos critérios estão dentro do *PQ*, e apenas 16% dos estão abaixo do padrão de qualidade.

Os atributos que não atingiram a qualidade ideal (q = 1), mas que estão dentro do padrão de qualidade, em termos percentuais, são: (*i*) *Uniformidade de Termos* - 32,8%; (*ii*) *Uniformidade de Notação* - 60,7%; (*iii*) *Complementabilidade* - 19,9%; (*iv*) *Consistência Interna* - 87,6%; e (*v*) *Precisão* - 77,0%.

V.3 Conclusão

Neste capítulo, foi apresentado um experimento para validar o MFAQS, baseado em uma pesquisa de campo realizada, para levantar o padrão de qualidade de especificações de requisitos de software. Baseado no padrão de qualidade obtido, foi também avaliada uma ERS real, obtendo-se os índices de qualidade para os atributos avaliados.

Neste contexto, o MFAQS atendeu, convenientemente, a seus objetivos nesta experiência: (*i*) estabeleceu um padrão de qualidade para o produto de software considerado, e (*ii*) avaliou um produto de software (similar) real, com base nesse padrão estabelecido.

Capítulo VI

CONCLUSÃO

Neste trabalho, foram discutidos vários tópicos referentes a qualidade de processos e de produtos de software, em conformidade com o padrão ISO/IEC, juntamente com alguns modelos de avaliação mais representativos, tendo sido discutido, também, a temática de medição do software (*Capítulo II*).

Discorreu-se ainda sobre vários conceitos e propriedades da teoria dos conjuntos *fuzzy*, que vem sendo o elo de ligação entre muitos modelos imprecisos (subjetivos) do mundo real e a sua representação matemática (*Capítulo III*), para que pudessem dar suporte ao modelo *fuzzy* para avaliação da qualidade de software, proposto a partir da extensão do *Modelo Rocha* (Rocha, 1983) (*Capítulo IV*).

Após a definição do modelo *fuzzy*, foi realizada uma experiência de sua utilização (*Capítulo V*), objetivando-se validar o modelo e fornecer um exemplo de sua utilização.

Este trabalho teve as seguintes contribuições:

i. A extensão do Modelo Rocha para suportar os conceitos da teoria *fuzzy*, através do *modelo fuzzy para avaliação da qualidade de software*, fornecendo, assim, uma base matemática sólida e um mecanismo capaz de interpretar, na linguagem do desenvolvedor e/ou usuário, os resultados da medição do software.

ii. A utilização do peso do especialista em qualidade de software, através da elaboração do *Questionário de Identificação do Perfil do Especialista*, como também a definição de uma metodologia de apuração de seus resultados. Estes dados poderão, também, ser usados para avaliar e selecionar, antecipadamente, o grupo de especialistas, que participarão da avaliação do produto de software.

iii. A aplicação de funções de pertinência (números *fuzzy* normais triangulares), para representar um atributo de software, em qualquer ramo da árvore hierárquica de qualidade a que pertença.

iv. A agregação de atributos de qualidade de software, nos vários níveis de sua árvore hierárquica de qualidade, considerando-se, além do grau de importância, o peso de cada atributo agregante (primitivo), obtido no levantamento de seu padrão de qualidade (se tiver sido estabelecido). Com este procedimento, visa-se evitar (ou minimizar) possíveis distorções na apuração dos resultados da avaliação, isto é, atributos que tenham sido avaliados com valores muitos diferentes (para mais ou para menos) daqueles estabelecidos como padrão de qualidade.

v. Aplicação do conceito de similaridade (usado em tomada de decisão), para a avaliações da qualidade de software, na apuração das estimativas dos especialistas. Assim sendo, o resultado final da avaliação tende para onde houve maior grau de consenso, não gerando um resultado de tendência central. Quando, por exemplo, um especialista divergir totalmente dos outros especialistas, isto é, seu estado de concordância em relação a todos os outros for nulo, seu julgamento é automaticamente desprezado, pelo próprio modelo.

vi. A extensão do conceito do *grau de concordância nulo*, para o *grau de não concordância*, no intervalo [-1, 0], para o processo de agregação de atributos de qualidade de software. Isto porque todos os atributos de qualidade, que compõem um atributo agregado, independentemente de seu grau de importância, influencia o ramo hierárquico ao qual pertence, e não deve ser simplesmente desprezado, como ocorre em (*v*). O gr*au de concordância e grau de não concordância* passaram a se chamar, então, *estado de concordância*. Com isto, quando algum atributo agregante tiver seu *estado de concordância* não positivo, passa a ser considerado, também, no processo de agregação. Nesta situação, pode-se ponderar se, de fato, esse atributo deveria pertencer ou não àquele ramo hierárquico, que está sendo avaliado, e tomar a decisão, se for o caso, de rearranjar o atributo em sua árvore hierárquica de qualidade.

vii. Definição de procedimentos para cálculo de *índices de qualidade*, baseados em um padrão de qualidade já tenha sido estabelecido, mediante o modelo proposto.

VI.1 Perspectivas Futuras

Várias são as perspectivas para dar continuidade a este trabalho, que podem ser sugeridas:

- Automação do *modelo fuzzy proposto para avaliação da qualidade de* software, de maneira que possa ser facilmente utilizado como uma ferramenta de avaliação.
- Utilização deste modelo na avaliação da qualidade de outros produtos de software ou de domínios de aplicação, para melhor ser validado e refinado.
- Continuar investindo na aplicação da teoria dos conjuntos *fuzzy* no processo de avaliação da qualidade software, por esta ser uma teoria robusta, fundamentada axiomaticamente, e que manuseia, convenientemente, conceitos subjetivos e vagos, normalmente encontrados ao longo do ciclo de vida de um produto de software.

A experiência do uso da teoria *fuzzy* na avaliação da qualidade de software, especialmente em especificações de requisitos, foi enriquecedora, motivando-nos cada vez mais a pesquisar outras áreas afins, que possam fornecer subsídios e contribuir para a consolidação e maturidade da engenharia de software.

REFERÊNCIAS BIBLIOGRÁFICAS

ACKERMAN, A. F. *et al*., 1989, *Software Inspections: An Effective Verification Process*, IEEE Software.

ALMEIDA, M. S., MORAES, L. F. R., 1995, *Qualidade de Vida no Trabalho dos Profissionais de Informática e Cultura da Qualidade nas Empresas de Informática*, Workshop de Qualidade, IX SBES, Recife.

ALSINA, C., 1985, *On a family of connectives for fuzzy sets*, Fuzzy Sets and Systems 16, 231-235, in (ZIMMERMMAN, 1991).

AMBRIOLA, V. *et al*., 1994, *Applying a metric framework to the software process: an experiment*, European Workshop Software Process Technology 94, in (MARIANO, 1996).

ANDRADE, C. J., 1991, *ANAFOR: um analisador de Programas FORTRAN*, Tese de mestrado, COPPE/UFRJ, Rio de Janeiro.

ANGER, F. D. *et al*., 1994, *Temporal Complexity and Software Faults*, Proceeding of IEEE International Symposium on Software Reliability Engineering 94, IEEE Computer Society, Los Alamitos, CA, in (MUNSON, 1995).

ANSI/IEEE Std-730, 1993, *Standard for Software Quality Assurance Plans (ANSI)*, IEEE Standards Collection, Software Engineering.

ARTHUR, L. J., 1993, *Improving Software Quality - An Insider's Guide to TQM*, Wiley Professional Computing.

ASANOME, C. R.,1995, *Modelagem da Confiabilidade de Software*, Publicações Técnicas COPPE/UFRJ, ES-336/95.

AYTON, P., 1993, *On the Competence and Incompetence of Expert*, Expertise and Decision Support, Wright and F. Bolger, eds., Plenum Press, in (STRIGINI, 1996).

BACH, J., 1994, *The Imaturity of the CMM*, American Programmer, New York, September, in (BASTOS, 1996).

BACHE, R., BAZZANA, G., 1994, *Software Metrics for Product Assessment*, Mc Graw Hill Book Company, in (MARIANO, 1996).

BAHIA, A. S., 1992, *Avaliação da Complexidade de Software Científico*, Tese de mestrado, COPPE/UFRJ, Rio de Janeiro.

BALDWIN, J. F., 1979, *A new approach to approximate reasoning using a fuzzy logic*, Fuzzy Sets and Systems, 2,309-325, in (ZIMMERMANN, 1991).

BANDEMER, H., KRAUT, A., 1990, *A case study on modelling impreciseness and vagueness of observations to evaluate a functional relationship*, in W. H. Janko, Ed., Progress in Fuzzy Sets and Systems (Kluwer Academic, Netherlands), in (RÖMER, 1995).

BAKER, A. *et al*., 1990, *A philosofhy for software measurement*, Journal of System Software, vol. 12, July, in (FENTON, 1994).

BARDOSSY, A., DUCKSTEIN, L., BOGARDI, I., 1993, *Combination of fuzzy number representing expert opinions*, Fuzzy Sets and Systems 57, 173-181.

BASILI, V., ROMBACH, H. D., 1987, *Tailoring the Software Process to Project Goals and Environments*, Department of Computer Science, University of Maryland, ACM, 1987, in (ROSENBERG, 1996).

BASILI, V. R., MUSA, J. D., 1991, *The Future Engineering of Software: A Management Perspective*, IEEE Computer, September.

BASILI, V. *et al*., 1994, *The Goal Question Metric Approach*, University of Maryland, ftp.cs.umd.edu/pub/sel/papers/gqm.ps.

BASILI, V. R., BRIAND, L., Melo, W., 1995, *A Validation of Object-Oriented Design Metrics*, Technical Report CS-TR-3443, University of Maryland, Dep. of Computer Science, April, in (CLUNIE, 1997).

BASTOS, A. R. G., 1996, *Um estudo sobre adoção de um modelo que visa promover melhorias no processo de desenvolvimento de software*: *O caso do CMM em uma empresa do setor financeiro no Brasil*, Tese de Mestrado, COPPE/UFRJ, Rio de Janeiro.

BAZZANA, G. *et al*., 1993, *ISO 9126 and ISO 9000: Friends or Foes?*, Software Engineering Standards Symposium, Brighton, UK.

BELASCO, J. A., STAYER, R. C., 1994, *O vôo do búfalo: decolando para a excelência, aprendendo a deixar os empregados assumirem a direção*, Editora Campus, Rio de Janeiro, in (ALMEIDA *et al.*, 1995).

BELCHIOR, A. D., 1992a, *Características de Qualidade de Programas*, Relatório Técnico ES-265/92, COPPE/UFRJ, Rio de Janeiro.

BELCHIOR, A. D., 1992b, *Controle da Qualidade de Software Financeiro*, Tese de Mestrado, COPPE/UFRJ, Rio de Janeiro.

BELCHIOR, A. D., GAIO, F. J., 1994, *A Tecnologia de Informação Apoiando os Novos Desafios Empresariais: Um Estudo de Casos*, VII Congreso Latino-Ibero Americano de Investigacion de Operaciones e Ingenieria de Sistemas (CLAIO), XVII Taller de Ingenieria de Sistema, Santiago-Chile.

BELCHIOR, A. D., GAIO, F. J., 1995a, *Aspectos não Tecnológicas Envolvidos no Desenvolvimento de Software: Um Estudo de Casos*, Anais do IX SBES, Recife.

BELCHIOR, A. D., XEXÉO, G. B., 1995b, *Um Modelo Fuzzy para Qualidade de Software*, Relatório Técnicos ES-344/95 da COPPE/UFRJ.

BELCHIOR, A. D., XEXÉO, G., ROCHA, A. R. C., 1995c, *Uma Proposta Fuzzy* para *Medição de Atributos de Qualidade de Software*, Anais do IX SBES, Recife.

BELCHIOR, A. D., XEXÉO, G. B., ROCHA, A. R. C., 1996a, , *Aplicação da Teoria Fuzzy em Requisitos de Qualidade de Software*, XX II Conferencia Latinoamericana de Informática (CLEI), Bogotá de Santa Fé, Colômbia.

BELCHIOR, A. D., XEXÉO, G. B., ROCHA, A. R. C., 1996b, , *Agregação de Atributos de Qualidade de Software usando-se a Teoria Fuzzy*, Workshp de Qualidade e Produtividade de Software, X Simpósio Brasileiro de Engenharia de Software (SBES), São Carlos - SP.

BELCHIOR, A. D., XEXÉO, G. B., ROCHA, A. R. C., 1996c, *Evaluating Software Quality Requirements using Fuzzy Theory*, International Conference on Information Systems Analysis and Synthesis (ISAS), Orlando, USA.

BELL Canada Inc., 1994, *Trillium: Model for Telecom Product Development and Suport Process Capability*, release 3.0, December, in (OLIVEIRA, 1995c).

BELLMANN, R. E., 1970, *Decision making in a fuzzy enviroment*, Management Sci. 17, in (FELIX, 1994).

BELLMANN, R. E., GIERTZ, M., 1973, *On the analytic formalism of theory of fuzzy set*, Information Science 5, 149-157, in (SDORRA, 1993).

BERGAMO Filho, V., 1991, *Gerência econômica da qualidade através do TQC: controle total da qualidade*, Editora Makron, McGraw Hill, São Paulo.

BERTOLINO, A., STRIGINI, L., 1996, *On the use of testability measures for dependability assessment*, IEEE Transaction on Software Engineering, vol. 22, nº 2, February, in (MARIANO, 1996).

BIEMANN, J. M. and OTT, L. M., 1994, *Measuring functional cohesion*, IEEE Transaction on Software Engineering, vol. 20, nº 8, August, in (MARIANO, 1996).

BLASCHECK, J. R., 1995, *Planejamento de Projetos*, Tese de Doutorado, COPPE/UFRJ, junho.

BOEGH, J. *et al*., 1992, *Guide to software product evaluation - The evaluators guide*, Draft technical report, ISO/IEC/JTC1/SC7, October, in (BOEGH, 1993).

BOEGH, J. *et al*., 1993, *A Practitioners Guide to Evaluation of Software*, IEEE Computer, August.

BOEHM, B., 1987, *Industrial software metrics top ten list*, IEEE Software, September.

BOEHM, B., HOH, I., 1996, *Identifying Quality-Requirement Conflits*, IEEE Software, March.

BOLOIX, G., *et al*., 1995, *A Software System Evaluation Framework*, IEEE Software, December.

BOSC, P. *et al*., 1995, *Quantified Statements in a Flexible Relational Query Language*, Proceedings of the 1995 ACM Symposium on Applied Computing, Nashville, February.

BROCKLEHURST, P. Y *et al*., 1990 *Recalibrating software reliability models*, IEEE Transaction Software Engineering, vol. 16, nº 4, April, in (FENTON, 1994).

BROEK, P. M., BERG, K. G., 1995, *Generalised approach to software structure metrics*, Software Engineering Journal, vol. 10, nº 2, March, in (MARIANO, 1996).

BUCKLEY, J. J., HAYASHI, Y., 1994, *Fuzzy neural networks: A survey*, Fuzzy Sets and Systems 66, 1-13.

CALDIERA, G., BASILI, V. R., 1991, *Identifying and Qualifying Reusable Software Components*, IEEE Computer, February.

CAMPOS, V. F., 1990, *Gerência da Qualidade Total: Estratégia para Aumentar a Competitividade da Empresa Brasileira*, Fundação Christiano Ottoni, Escola de Engenharia da UFMG, Belo Horizonte.

CAMPOS, F. C., 1994a, *Hipermídia na Educação: Paradigmas e Avaliação da Qualidade*, Tese de mestrado, COPPE/UFRJ, Rio de Janeiro, agosto.

CAMPOS, G. H. B., 1994b, *Metodologia para avaliação da qualidade de software educacional: Diretrizes para desenvolvedores e usuários*, Tese de Doutorado, COPPE/ UFRJ, Rio de Janeiro, novembro.

CARCHIOLO, V., MALGERI, M., 1995, *A fuzzy approach to co-design system partitioning*, Proceedings of the 1995 ACM Symposium on Applied Computing, Nashville, February.

CARD, D. N., GLASS, R. L., 1990, *Measuring Software Desing Quality*, Prentice-Hall, July.

CARLSSON, C., FULLÉR, R., 1996, *Fuzzy multiple criteria decision making*: *Recent developments*, Fuzzy Sets and Systems 78, 139-153.

CARVALHO, D., 1994, *Requisitos de Qualidade para o Software Médico*, Relatório Técnico, Fundação Bahiana de Cardiologia.

CHAIM, M. L., 1991, *POKE-TOOL - Uma Ferramenta para Suporte ao Teste Estrutural de Programas Baseado em Análise de Fluxo de Dados*, Tese de Mestrado, DCA/FEE/UNICAMP, Abril, in (VILLAS, 1995).

CHAN, C. B. I., 1991, *Case Observations and Lessons*, in The Knowledge Engineering Review, volume 6:2, 91.

CHEN, S. J., HWANG, C. L., 1992, *Fuzzy Multiple Attribute Decision-making, Methods and Applications*, Lecture Notes in Economics and Mathematical Systems, Vol. 375, Springer, Heidelberg, in (CARLSSON, 1996).

CHEN, C. T., HSY, H. M., 1993, *A study of fuzzy TOPSIS model*, Proc. of the Chinese Institute of Industrial Engineers National Conference, in (HSU, 1996).

CHEN, J. E., OTTO, K. N., 1995, *Constructing membership function using interpolation and measurement theory*, Fuzzy Sets and Systems 73, 313-327.

CHIUEH, T., 1992, *Optimization of Fuzzy Logic Inference Architecture*, Computer, May.

CHUNG, L. *et al*., 1995, *Dealing with Non-Functional Requirements: Three Experimental Studies of a Process-Oriented Approach*, 17th International Conference on Software Engineering, Seattle, EUA, in (WEBER, 1997).

CLUNIE, C. E., 1987, *Verificação e Validação de Software na fase de Especificação de Requisitos*, Tese de Mestrado, COPPE/UFRJ.

CLUNIE, C. E., 1997, *Avaliação da Qualidade de Especificações Orientadas a Objeto*, Tese de Doutorado, COPPE/UFRJ.

COBB, R. H. *et al*., 1990, *Engineering software under statistical quality control*, IEEE Software, 7, 6, November, in (COLLINS *et al.*, 1994).

COELHO, A. C. B., 1997 *Atributos de Qualidade para uma Abordagem Sociotécnica na Aquisição de Software*, Tese de Mestrado, COPPE/UFRJ, Rio de Janeiro, abril.

COLEMAN, D. *et al*., 1994, *Using metrics to evaluate software systems maintainability*, IEEE Computer, August.

COLLINS, W. R. *et al*., 1994, *How Good is Good enough?*, Communication of the ACM, January.

COMMERLATO, C. A., 1994, *Uma Ferramenta para Seleção de Módulos Reutilizáveis*, Tese de Mestrado, COPPE/UFRJ, Rio de Janeiro, julho.

CONTE, S. D. *et al*., 1986, *Software Engineering Metrics and Models*, The Benjamim/Cummings Pulishing Company, Inc.

COSENZA, C. A. N., 1981, *Modelo para Localização Industrial*, Tese de Doutorado.

COWARD, P. D., 1988, *A Review of Software Testing*, Information and Software Tecnology, vol. 30, nº 3, April 1988, in (OLIVEIRA, 1995b).

COX, E. D., 1995, *Fuzzy logic for business and industry*, Rockland, Massachusetts.

CROSBY, P. B., 1990, *Qualidade - falando Sério*, Editora McGraw Hill, tradução de José Carlos Barbosa dos Santos, São Paulo.

DARKIN. K., 1996, *Keys to Engineering a Workable Contract*, IEEE Software, January.

DALE, C. J. *et al*., 1992, *Software productivity metrics: Who needs them?*, Information and Software Technology, vol. 34 (11), November, in (CAMPOS, 1995b).

DAVENPORT, THOMAS H. *et al*., 1990 *The New Industrial Engineering Information Technology and Business Process Redesign*, Sloan Management Review, Summer.

DAVIS, C. J. *et al*., 1993, *Industrial Acceptance of Software Quality Assurance Standards*, IEEE Computer, August.

DAY, W. H. E., 1988, *Consensus methods as tools for data analysis*, in H.H. Bock, Ed., Classification and Related Methods for Data Analysis (Elservier, Amsterdam), 317-324, in (KUNCHEVA, 1996).

DELAMARO, M. E., 1993, *Proteum - Um Ambiente de Teste Baseado na Análise de Mutantes*, Tese de Mestrado, ICMSC-UDP, Outubro, in (VILLAS, 1995).

DeMARCO, T., LISTER, T., 1987, *Pleopleware: Productive Projects and Teams*, New York: Dorset House Publishing, in (WEINBERG, 1993).

DEMING, W. E., 1986, *The Deming Management Method*, New York: Dodd, Mead & Co., in (WEINBERG, 1993).

DEMING, W. E., 1989, *Out of Crisis*, MIT Center for Advanced Engineering Study, MIT Press, Cambridge, Mass.

DRIANKOV, D. *et al*., 1993, *An Introduction to Fuzzy Logic Control*, Springer, Berlin, in (YAGER, 1994).

DROMEY, R. G., 1995, *A Model for Software Product Quality*, IEEE Transactions on Software Engineering, vol. 21, nº 2, February.

DROMEY, R. G., 1996, *Cornering the Chimera*, IEEE Software, January.

DRUCKER, P. F., 1990, *The Comming of the new Organizations*, Harvard Bussines Review.

DUBOIS, D., PRADE, H., 1980, *Fuzzy Sets and Systems*: *Theory and Applications*, Academic Press, New York.

DUBOIS, D., PRADE, H., 1982, *A class of fuzzy measures based on triangular norms*, Inter. J. Gen. Syst. 8, 43-61, in (ZIMMERMANN, 1991).

DUBOIS, D., PRADE, H., 1984, *Criteria aggregation and ranking of alternatives in the framework of fuzzy set theory*, Studies in Management Sci, 20, in (FELIX, 1994).

DUBOIS, D., 1985, *A review of fuzzy set aggregation connectives*, Information Science 36 85-121.

DUBOIS, D., PRADE, H., 1988, *Processing of imprecision and uncertainty in expert system reasoning models*, In C.J. Ernst, 67-88, in (ZIMMERMANN, 1991).

DUBOIS, D., PRADE, H., 1989, *Fuzzy sets, probability and measurement*, European J. Oper. Res. 40, in (TURKEN, 1991).

DUBOIS, D., PRADE, H., 1991, *Fuzzy sets in approximate reasoning, Part 1: Inference with possibility distributions*, Fuzzy Sets and Systems, IFSA, Special Memorial Volume: 25 years of fuzzy sets, North-Holland - Amsterdam.

DUNN, R. H., 1990, *Software Quality, Concepts and Plans*, Prentice-Hall, Englewood Cliffs, NJ, in (ZAGE, 1995).

DRUCKER, PETER F., 1990, *The Comming of the new Organizations*, Harvard Bussines Review.

DYER, M., 1992, *The Cleanroom Approach to Quality Software Development*, John Wiley & Sons, Inc. New York.

ENGEMANN, K. J., YAGER, R. R., 1997, *Risk Management with Imprecise Information*, in Fuzzy information engineering: a guide tour of applications, edited by Didier Dubois, Henri Prade, and Ronald R. Yager, Jonh Wiley & Sons, Inc., USA.

FAGAN, M. E., 1976, *Design and Code Inspection to Reduce Error in Program Development*, IBM System Journal 15.

FAGAN, M., 1986, *Advances in Software Inspections*, IEEE Transactions on Software Engineering, vol. SE-12, nº 7, July, in (HUMPHREY, 1995).

FARBEY, B., 1990, *Software quality metrics: considerations about requirements and requirement specifications*, Information and Software Technology, vol. 32, nº 1, January/February.

FEDRIZZI, M., KACPRZYK, J., 1993, *Consensus degrees under fuzzy majorities and preferences using OWA* (*Ordered Weighted Average*) *operators, Proc. 5th IFSA World Congress, Seoul, Korea*, in (KUNCHEVA, 1996).

FELIX, R., 1994, *Relationships between goals in multiple attribute decision making*, Fuzzy Sets and Systems 67, 47-52.

FENTON, N. E., 1990, *Software Metrics: theory, tools and validation*, Chapman and Hall, in (HUMPHREY, 1995).

FENTON, N. E., 1991, *Software Metrics: A Rigorous Approach*, Chapman and Hall.

FENTON, N., 1994, *Software Measurement: A Necessary Scientific Basis*, IEEE Transaction on Software Engineering, vol. 20, nº 3, March.

FENTON, N., 1996, *Improve Quality?*, IEEE Software, January.

FENTON, N. E., PFLEEGER, S. L., 1997, *Software Metrics: A Rigorous & Practical Approach*, Second Edition, PWS Publishing Company.

FICKAS, S., HELM, B. R., 1992, *Knowledge Representation and in the Design of Composite Systems*, IEEE Transaction on Software Engineering,vol.18, nº 6, June.

FILEV, D. P., 1993, *An adaptive approach to defuzzificaton based on level sets*, Fuzzy Sets and Systems 54, 355-360.

FINKELSTEIN, L., 1984, *A review of the fundamental concepts of measurement*, Measurement, vol. 2, nº 1, in (FENTON, 1994).

FIORINI, S. T., 1995, *TQM, CMM e Ambientes de Desenvolvimento de Software - uma idéia*, Workshop de Qualidade de Software, IX SBES, Recife.

FODOR, J. C., ROUBENS, M., 1994, *Fuzzy Preference Modelling and Multicriteria Decision Support*, Kluwer, Dordrecht.

FREEDMAN, D. P., WEINBERG, G. M., 1984, *Reviews, Walkthroughs and Inspections*, IEEE Transaction of Software Engineering, January.

FREEDMAN, A. L., 1993, *Computer Systens Development: History, Organization and Implementation*, John Wiley & Sons Ltda, England.

FRENCH, S., 1986, *Decision Theory: An Introduction to the Mathematics of Rationality*, John Wiley and Sons, New York, in (SDORRA, 1993).

FRENCH, S., 1989, *Fuzzy sets: the unanswered questions*, Report 89.6, School of Computer Studies Research Report Series, University of Leeds, in (SDORRA, 1993).

FUHRMANN, G., 1990, *Note on the Generality of Fuzzy Sets*, Information Sciences 2, 143-152.

GALE, J. L. *et al*., 1990, *Implementing the Defect Prevention Process in the MVS Interactive Programming Organization*, IBM Systems Journal, vol. 29, nº 1, in (HUMPHREY, 1995).

GARCIA, J., PAZOS, J., RÍOS, J., 1992, *Methodology of linguistics evaluation in risk situations using fuzzy techniques*, Fuzzy Sets and Systems 48, 185-194.

GARVIN, D. A., 1988, *Managing Quality: The Strategic and Competitive Edge*, Free Press, New York, in (SIMMONS, 1996).

GEGOV. A. E., 1995, *Hierarchical fuzzy control of multivariable systems*, Fuzzy Sets and Systems 72, 299-310.

GENEST. C., 1984, *Um conflict between two axioms for combining subjective distributions*, J. R. Statist. Soc., in (BARBOSSY, 1993).

GENTLEMAN, W. M., 1996, *Is software quality is a perception, how do we measure it?*, The 6th I.C. on Software Quality, Ottawa, October.

GILARDONI, G. L., CLAYTON, M.K., 1993, *On reaching a consensus using DeGroot's iterative pooling*, Ann. Statist, 21, in (KUNCHEVA, 1993).

GILB, T., 1996, *Level 6: Why We Can't Get There from Here*, IEEE Software, January.

GILES, R., 1976, *Lukasiewicz logic and fuzzy theory*, Inter. J. Man-Mach, Stud. 8, 313-327, in (ZIMMERMANN, 1991).

GILES, R., 1988, *The concept of grade of membership*, Fuzzy Sets and Systems 25, 297-323, in (SDORRA, 93).

GRABISCH, M., 1995, *Fuzzy integral in multicriteria decision making*, Fuzzy Sets and Systems 69, 279-298.

GRADY, R. B., 1992, *Practical Software Metrics for Project Management and Process Improvement*, Prentice-Hall, Engelewooe Cliffs, NJ.

GRADY, R. B., 1994, *Successfully applying software metrics*, IEEE Computer, September.

GRAHAM, B. P., NEWELL, R. B., 1988, *Fuzzy identification and control of a liquid level rig.*, Fuzzy Sets and Systems, 26(3), 47-65, in (KLIR *et al*., 1995a).

GUPTA, M. M., 1992, *Fuzzy logic and neural network*, Proc. 2nd Internat. Conf. on Fuzzy Logic and Neural Networks, Iizuka, Japan, 157-160, in (BUCKLEY, 1994).

GÜRMAN, N. T., 1995, *Generation and Improvement of Fuzzy Classifier with Incremental Learning using Fuzzy RuleNet*, Proceedings of the 1995 ACM Symposium on Applied Computing, Nashville, February.

HAASE, V. *et al*., 1994, *Bootstrap: Fine-tuning Process Assessment*, IEEE Software.

HABRIAS, H., 1995, *Mesure du logiciel*, Editions Teknea, in (MARIANO, 1996).

HAILSTONE, R., 1991, *Quality management and software engineering*, Software Quality and Reliability, Tools and methods, ed. Darrel Ince, Chapman & Hall, London, in (GENTLEMAN, 1996).

HAMMER, Michael, 1990, *Reengineering Work: Don't Automate, Obliterate*, Harvard Bussines Review.

HAPKE, M. *et al*., 1994, *Fuzzy project scheduling system for software development*, Fuzzy Sets and Systems 67, 101-117.

HAUSEN, HANS-Ludwig *et al*., 1993, *Guides to Software Evaluation*, Arbeitspapiere der GMD 746, April.

HEFLEY, W. E. *et al*., 1995, *The People Capability Maturity Model: Status and Progress*, 5th International Conference on Software Quality (5ICSC), Austin, EUA, October, in (WEBER, 1997).

HERTZ, J., KROGH, A., PALMER, R., 1991, *Introduction to the Theory of Neural Computation*, Addison-Wesley, Reading, Mass., in (MUNAKATA, 1994).

HSU, H. M., CHEN, C. T., 1996, *Aggregation of fuzzy opinions under group decision making*, Fuzzy Sets and Systems 79, 279-285.

HULL, L. G *et al*., 1991, *Expert System Development Methodology and Management*, Proceedings of the IEEE/ACM International Conference on Development and Managing Expert System Programs, Washington, D.C.

HUMPHREY, W. S., 1989, *Managing the Software Process*, Addison-Wesley, Mass., in (MASHAYEKHI, 1993).

HUMPHREY, W. S. *et al*., 1991 *Software Process Improvement at Hughes Aircraft*, IEEE Software, July.

HUMPHREY, W. S., 1995, *A discipline for Software Engineering*, Addison-Wesley Publishing Company.

HWANG, C., YOON, K., 1981, *Multiple Attribute Decision Making: Methods and Applications*, Springer-Verlag, New York, in (KLIR *et al*., 1995a).

IBRAHIM, A., AYYUB, B. M., 1992, *Multi-criteria ranking of components according to their priority for inspection*, Fuzzy Sets and Systems 48, 1-14.

IEEE, 1987, Draft Version 13b, *Standard for a Software Quality Metrics Methodology*, IEEE Quality Metrics Standard Committee: P1061, July.

IEEE, 1988, Standard for a Software Quality Metrics Methodology.

IEEE, 1990, Standard Glossary of Software Engineering Terminology (ANSI).

INCE, D., 1990, *Software Metrics: Introduction*, Information and Software Technology, vol. 32, May.

ISHIKAWA, A *et al*., 1993, *The max-min Delphi method and fuzzy Delphi method via fuzzy integration*, Fuzzy Sets and Systems 55, 241-253.

ISO, 1987, ISO 9000, *Quality management and quality assurance standards*, Internacional Standards Organisation, in (BOEGH, 1993).

ISO, 1990, ISO 9000-3, *Quality Management and Quality Assurance Standards - Part 3: Guidelines for the Application of ISO9001 to the Development, Supply and Maintenance of Software*, ISO, September.

ISO, 1991, ISO/IEC 9126, *Information Technology - Software Product Evaluation*, Quality characteristics and guidelines for their use.

ISO, 1994, ISO DIS 8402, *Quality Vocabulary*, in (TSUKUMO, 1996).

ISO, 1995, ISO/IEC 14598, *International Standard, Information Technology - Software product evaluation - Parts 1 to 6*, (Working Draft), in (TSUKUMO, 1996).

JACK, R., 1993, *Applying ISO 9001 to Small Scale Software Development Project*, IEEE Computer, August.

JONES, C., 1991, *Applied Software Measurement: Assuring Productivity and Quality*, McGraw Hill, Software Engineering Series, New York.

JONES, D. W., 1995a, *Repeatable Software Development, Part I: Quality Assurance*, Software Development Magazine, May.

JONES, WEINBER, G. M., 1995b, *Assessment and control of software risk*, (book), USA.

JOYCE, D. T., 1994, *Examining the Potencial of Fuzzy Software Requirements Specifications*, Information Sciences 2, 85-102.

JURAN, J. M *et al*., 1988, *Juran's Quality Control Handbook*, Fourth Edition, New York, McGrawHill Book Company, in (HUMPHREY, 1995).

KACPRZYK, J. *et al*., 1992, *Group decision making and consensus under fuzzy preference and fuzzy majority*, Fuzzy Sets and Systems 49, 21-31.

KANDEL, A. *et al*., 1993, *Fuzzy Control Systems*, CRC Press, Boca Raton, Fl, (YAGER, 1994).

KANTROWITZ, M. *et al*., 1996, *Answers to Questions about Fuzzy Logic and Fuzzy Expert Systems*, http://www-cgi.cs.cmu.edu/afs/cs/project/ai-repository/ai/areas/fuzzy/faq/fuzzy.faq, (last modified: 25/07/06), in 11/12/96.

KARISSON, E. A., 1995, *Software reuse: a holistic Approach*, John Wiley & Sons.

KASABOV, N. K. *et al*., 1996, *Foundations of Neural Networks, Fuzzy Systems and Knowledge Engineering*; Massachusetts Institute of Technology.

KAUFMANN, A., GUPTA, M. M., 1991, *Introduction to Fuzzy Arithmetic*: *Theory and Applications*, Van Nostrand Reinhold, New York.

KHOSHGOFTAAR, T. M. *et al*., 1996, *Reuse history in software quality models*, IEEE Software, in (PONNAMBALAM, 1996).

KICKERT, W. J. M., MANDANI, E. H., 1978, *Analysis of a fuzzy logic controller*, Fuzzy Sets and Systems, 1(1), 29-44, in (KLIR *et al*., 1995a).

KIRKHAM, D. M. *et al*., 1996, *Experiences in Analyzing Software Inspection Data*, The 6th I.C. on Software Quality, Ottawa, October.

KITCHENHAM, B. A *et al*., 1986, *Software Metrics and Integrated Project Support Environments*, Software Engineering Journal, January, in (MÖLLER, 1993).

KITCHENHAM, B., 1990a, *Editorial: Software reliability and metrics*, IEE Software Engineering Journal., January.

KITCHENHAM, B. *et al*., 1990b, *An evaluation of some design metrics*, IEE Software Engineering Journal., January.

KITCHENHAM, B. A. *et al*., 1990c, *Cost modelling and estimation*, Software Reliability Handbook, P. Rook Ed., New York: Elsevier Applied Science, in (FENTON, 1994).

KITCHENHAM, B. *et al*., 1995, *Towards a Framework for Software Measurement Validation*, IEEE Transaction on Software Engineering, vol. 21, nº 12, December.

KITCHENHAM, B. *et al*., 1996, *Software Quality: The Elusive Target*, IEEE Software, January.

KLEMENT, E. P., SCHWYHLA, W., 1982, *Correspondence between fuzzy measures and classical measures*, FSS 7, 57-70, in (ZIMMERMANN, 1991).

KLIR, G. J., FOLGER, T. A., 1988, *Fuzzy Sets, Uncertainty and Information*, Prentice Hall, New Jersey.

KLIR, G. J., YUAN, B., 1995a, *Fuzzy Sets and Fuzzy Logic: Theory and Applications*, Prentice Hall, New Jersey.

KLIR, G. J., 1995b, *Principles of uncertainly: What ar they? Why do we need them*?, Fuzzy Sets and Systems 74, 15-311.

KNIGHT, J. C., MYERS, E. A., 1993, *An Improved Inspection technique*, Communications of the ACM, November.

KOSKO, B.; 1995, *Neural Networks ans Fuzzy Systems: A dynamical systems approach to machine intelligence*; Prentice Hall, Englewood Cliffs, NJ.

KRISHNAPURAM, R., LEE, J., 1992, *Fuzzy connective-based hierarchical aggregation networks for decision making*, Fuzzy Sets and Systems 46, 21-31, in (KUNCHEVA, 1996).

KUNCHEVA, L. I., 1995, *Using degree of consensus in two-level fuzzy pattern recognition*, European J. Oper. Res. 80, 365-370, in (KUNCHEVA, 1996).

KUNCHEVA, L. I., KRISHNAPURAM, R., 1996, *A fuzzy consensus aggregation operator*, Fuzzy Sets and Systems 79, 347-356.

KUMAR, S. R. *et al*., 1992, *Methodology Engineering: A proposal for situation specific methodology construction*, Challenges and Strategies for Research in Systems Development, John Wiley & Sons, Washington, in (ROSSI, 1996).

KUMAR, S. R., GHOSHAL, J., 1996, *Approximate reasoning approach to pattern recognition*, Fuzzy Sets and Systems 77, 125-150.

KUVAJA, P. *et al*., 1994, *Software Process Assessment & Improvement - The Bootstrap Approach*, Blackwell Publishers, Oxford, in (OLIVEIRA, 1995c).

LASEK, M., 1992, *Hierarchical structures of fuzzy ratings in the analysis of strategic goal of enterprises*, Fuzzy Sets and Systems 50, 127-134.

LEE, C. C., 1990, *Fuzzy logic in control systems: fuzzy logic controller - parte I e II*, IEEE Trans. Systems Man Cybernet. 20, 2, 404-418, 419-435.

LEE, K. C., 1994, *Fuzzy post adjustment mechanisms to improve the quality of knowledge-based solutions*, Fuzzy Sets and Systems 68, 67-81.

LEE, H. M., 1996a, *Applying fuzzy set theory to evaluate the rate of aggregative risk in software development*, Fuzzy Sets and Systems 79, 323-336.

LEE, H. M., 1996b, *Group decison making using fuzzy theory for evaluating the rate of aggregative risk in software development*, Fuzzy Sets and Systems 80, 261-271.

LESMO, L., SAITTA, L., TORASSA, P., 1982, *Learning of fuzzy production rules for medical diagnosis*, in (ZIMMERMANN, 1991).

LEVENDEL, Y., 1990, *Reliability Analysis of Large Software Systems: Defect Data Modeling*, IEEE Transaction on Software Engineering, vol. 16, nº 2, February, in (HUMPHREY, 1995).

LÉVY, P., 1993, *As Tecnologias da Inteligência - O Futuro do Pensamento na Era da Informática*, Editora 34, Rio de Janeiro, in (COELHO, 1995).

LIGGESMEYER, P., 1995, *A set of complexit measures for guiding the software teste process*, Software Quality Journal, vol. 4, in (MARIANO, 1996).

LINGER, R. C., 1994, *Cleanroom Process Model*, IEEE Software, March.

LIOU, T. S., JIUN, M., WANG, J., 1992, *Fuzzy weighted average: An improved algorithm*, Fuzzy Sets and Systems 49, 307-315.

LUCENA, C. J. P., 1991, *Tecnologias de Software: Tendências e Interesses para o Brasil*, Seminário Avançado de Software, CITS, Curitiba, Outubro, in (WEBER 97).

LUHANDJULA, M. K., 1989, *Fuzzy optimization*: *an appraisal*, Fuzzy Sets and Systems 30, 139-153.

MABUCHI, S., 1993, *A proposal for defuzzification strategy by the concept of sensitivity*, Fuzzy Sets and Systems 55, 1-14.

McCABE, T. J., 1976, *A Complexity Measure*, IEEE Transactions on Software Engineering; vol. SE 2, December.

MAININI, M. T., BILLOT, L., 1990, *PERFIDE: an environment for evaluation and monitoring of software reliability metrics during the test phase*, IEE Software Engineering Journal, January.

MALDONADO, J. C. *et al*., 1995, *Análise de Mutantes: Uma Avaliação Empíritca do Axioma de Antiextensionalidade e da Propriedade de Equivalência*, Workshop de Qualidade, IX SBES, Recife.

MANDEVILLE, W. A, 1990, *Software Cost of Quality*, IEEE Journal on Select Areas in Communications, vol. 8, nº 2, February, in (HUMPHREY, 1995).

MARIANO, G., 1996, *Metrics in software engineering*, http://estas1.inrets.fr:8001/Public/Mariano.Georges/DundeeB/node3.html, (also node 4,5,6,7), in 21/01/97.

MASHAYEKHI, 1993, V. *et al*., *Distributed, Collaborative Software Inspection*, IEEE Software, September.

MAYS, R. G. *et al*., 1990, *Experiences with Defect Prevention*, IBM Systems Journal, vol. 29, nº 1, in (HUMPHREY, 1995).

MELTON, A., 1996, *Software Measurement*, International Thomson Computer Press, in (MARIANO, 1996).

MICH, L., FEDRIZZI, M., GAIO, L., 1993, *Approximate reasoning in the modeling of consensus in group decisions*, Proc. FLAI' 93, Lecture Note in Artificial Intelligence, Vol. 695, 91-102, in (KUNCHEVA, 1996).

MILLER, S. E. *et al*., 1993, *Quality Standards: The Role of Software Process Assessments*, IEEE Computer, August.

MILLET, P. B. *et al*., 1993, *Os princípios da qualidade total aplicados à informática*, Rio de Janeiro, LTC.

MIYOSHI, T., AZUME, M., 1993, *An Empirical Study of Evaluating Software Development Environment Quality*, IEEE Transactions on Software Engineering, vol. 19, no. 5, May.

MÖLLER, K. H., 1993, *Software Metrics: a practitioner's guide to improved product development*, Chapman & Hall Computing.

MOONEY, J. D., 1993, *Issues in the Specification and Measurements of Software Portability*, http:www.cs.wvu.edu/~jdm/research/portability/reports/TR_93-6.html# RT FToC17, in 24/10/96.

MOOR, J., 1985, *What is computer ethics?*, Metaphilosophy 16, 4, October 1985, in (COLLINS *et al.*, 1994).

MUMFORD, E., 1972, *Job Satisfactin: A Stude of Computer Specialist*, Longman, London, in (FREEDMAN, 1993).

MUNAKATA, J., JANI, Y., 1994, *Fuzzy Systems: An Overview*, Communications of the ACM, vol. 37, no 3, March.

MUNSON, J. C., 1995, *Software measurement: Problens and practice*, Annals of Software Engineering 1, 255-285.

MUSA, J. D., IANINO, A., OKUMOTO, K., 1987, *Software Reliability - Measurement, Prediction, Application*, McGraw-Hill, Inc., Singapore, in (ASANOME, 1995).

NAKAMURA, K., 1986, *Preference relation on a set of fuzzy utilities*, Fuzzy Sets and Systems 20, 279-285.

NEGOITA, C. V.; 1985, *Expert Systems and Fuzzy Systems*, Menlo Park, Reading, London, Amsterdam, in (ZIMMERMANN, 1991).

NICULESCU, S. P., VIERTL, R., 1992, *Bernoulli's Law of Large Numbers for vague data*, Fuzzy Sets and Systems 50, 167-173.

NOVÁK, V., 1992, *Fuzzy logic as a basis of approximate reasoning*, in Fuzzy Logic for the Management of Uncertainty, edited by Lotfi Zadeh and Janusz Kacprzyk, John Wiley & Sons, USA.

OLIVÉ, A., SANCHO, M. R., 1996, *Validating Conceptual Specifications through Model Execution*, Information Systems, Vol. 21, no.2 (167-186).

OLIVEIRA, J. V., 1995a, *A set-theoretical defuzzification method*, Fuzzy Sets and Systems 76, 63-71.

OLIVEIRA, K. M., 1995b, *Avaliação da Qualidade de Sistemas Especialistas*, Tese de mestrado, COPPE/UFRJ, Rio de Janeiro, março.

OLIVEIRA, A. M. *et al*., 1995c, *Avaliação de Processos de Software: Modelos e o TAQS-PROC*, Workshop de Qualidade de Software, IX SBES, Recife.

OMAN, P. *et al*., 1994, *Construction and testing of polynomials predicting software software maintainability*, Journal of System and Software, 24(3), in (PONNAMBALAM, 1996).

PAGE-JONES, M., 1988, *Projeto Estruturado de Sistemas*, McGraw-Hill, São Paulo.

PAL, S. K. *et al*., 1992, *Fuzzy geometry in image analysis*, Fuzzy Sets and Systems 48, 23-40, in (SEONG, 1995).

PALERMO, S., ROCHA, A. R. C., 1989, *An experience on Evaluating Software Quality for High energy Physics*, Computer Physics Communications.

PASSOS, 1991, M. C. J. F., *Uma Ferramenta para Construção e Avaliação de Gráficos de Estrutura*, Tese de mestrado, COPPE/UFRJ, Rio de Janeiro, maio.

PASSOS, M. C. J. F., ROCHA, A. C. R., WERNECK, V. M. B., 1996, *Uma Experiência em Aquisição de Modelagem do Conhecimento com o Método Kads-Estendido*; Workshop sobre Sistemas Baseados em Conhecimento, Vitória.

PEDRYCZ, W., 1990, *Relevancy of Fuzzy Models*, Informations Sciences 52, 285-302.

PETERS, T., 1993, *Rompendo as barreiras da administração para enfrentar a nova realidade*, Editora Harbra, São Paulo, in (ALMEIDA *et al.*, 1995).

PFEFFER, J., 1994, *Vantagens competitiva através de pessoas*, Editora Makron Bolks, São Paulo, in (ALMEIDA *et al.*, 1995).

PFLEEGER, S. L., 1991, *Software Engineering: The Production of Quality Softwre*, Second Ed., Macmillan, New York.

PONNAMBALAM, K., 1996, *Analyzing Sensitivities of Software Qualities to Various Metrics*, The 6th I.C. on Software Quality, Ottawa, October.

PRATHER, R. E., 1996, *Convexity and independence in software metric theory*, Software Engineering Journal, July.

PRESEDO, J. *et al*., 1996, *Fuzzy modelling of the expert's knowledge in ECG-based ischaemia detection*, Fuzzy Sets and Systems 77, 63-75.

PRESSMAN, R., 1992, *Software Engineering - A Practioner's Approach*, Third Edition, McGraw Hill International Edition.

RAFFY, J. L., 1995, *Mesures de complexité de spécifications écrites en Estelle*, CFIP'95, May, in (MARIANO, 1996).

RIBEIRO, R. A. *et al*., 1995, *Uncertainty in decision making: an abductive perspective*, Decision Support Systems, 13, in (RIBEIRO, 1996).

RIBEIRO, R. A., 1996, *Fuzzy multiple attribute decision making: A review and new preference elicitation techniques*, Fuzzy Sets and Systems 78, 155-181.

RICHARDS, B. L., 1988, *When Facts Get Fuzzy*, Byte, April.

ROBINS, Edward, 1995, The weighted average and its limitations in decision supports, Available: http://www.arlingsoft.com/wt_avr.htm.

ROCHA, A. R. C., 1983, *Um Modelo para Avaliação da Qualidade de Especificações*. Tese de Doutorado, PUC-RJ, Rio de Janeiro.

ROCHA, A. R. C., 1987, *Análise e Projeto Estruturado de Sistemas*, Editora Campus, Rio de Janeiro.

ROCHA, A. R. C. *et al*., 1994, *Uma Experiência na Definição do Processo de Desenvolvimento e Avaliação de Software segundo as Norma ISO*, Anais do Workshop de Qualidade de Software, Curitiba.

ROCHA, A. R. C. *et al*., 1996, *Processo de Desenvolvimento de Software Baseado em Reutilização*, Relatório Técnico do Projeto MEMPHIS, 1/96, COPPE/UFRJ.

ROCHA, A. R. C., 1997, *Tedências da Qualidade e Produtividade em Software*, in (WEBER, 1997).

ROMBACH, H. D., 1990, *Design measurement: Some lessons learned*, IEEE Software, March, in (CAMPOS, 1995b).

RÖMER, C., KANDEL, A., 1995, *Statistical tests for fuzzy data*, Fuzzy Sets Systems 72, 1-26.

ROSENBERG, L. *et al*., 1996, *Develping An Effective Metrics Program*, European Space Agency Software Assurance Symposium, Netherlands, March, in http:/satc.gsfc. nasa.gov/paper/esa_met.html, in 09/11/96.

ROSSI, M., BRINKKEMPER, S., 1996, *Complexity Metrics for Systems Development Methods and Techniques*, Information Systems, Vol. 21, nº 2 (209-227).

ROTHMAN, P., 1989, *Selecting an Uncertainty Management System*, AI Expert, July.

ROUBENS, M., 1990, *Inequality constraints between fuzzy number and their use in mathematical programming*, in (SLOWINSKI, 1990).

RUONING, X., XIAOYAN, Z., 1992, *Extensions of the analytic hierachy process n fuzzy environment*, Fuzzy Sets and Systems 52, 251-257.

SAADE, J. J., 1996, *Mapping convex and normal fuzzy sets*, Fuzzy Sets and Systems 81, 251-256.

SAATY, T. L., 1980, *The Analytic Hierarchy Process*, McGraw-Hill, New York.

SAIEDIAN, H., KUZARA, R., 1995, *SEI Capability Maturity Models Impact on Contractors*, IEEE Computer, January.

SAKAWA, M., SAWADA, K., 1994, *An interactive fuzzy satisficing method for large-scale multiobjective linear programming problems with block angular structure*, Fuzzy Sets and Systems 67, 5-17.

SANDERS, J., CURRAN, E., 1994, *Software Quality*, Dublin: ACM Press Books.

SANDRI, S. A., 1997, *Elicitation, Pooling, and Assessment of Expert Opinion in the Possibilistic Framework*, in Fuzzy information engineering: a guide tour of applications, edited by Didier Dubois, Henri Prade, and Ronald R. Yager, Jonh Wiley & Sons, Inc., USA.

SASAKI, M., 1993, *Fuzzy functions, Bull Sect.*, Fuzzy Sets and Systems 60, 336, North-Holland, in (BUCKLEY, 1994).

SCALET, D., 1995, *Avaliação da Qualidade do Produto de Sotfware*, Workshop da Qualidade e Produtividade em Software e IX SBES/SBC, Recife, Outubro.

SCHACH, S. R., 1990, *Software Engineering*, Aksen Associates Inc., Homewood, IL, in (GENTLEMAN, 1996).

SCHMAUCH, C. H., 1994, *ISO-9000 for Software Developer*.

SCHMUCKER, K. J., 1984, *Fuzzy Sets, Natural Language Science Press*, Rockville, MD, in (KLIR *et. al*., 1995a).

SCHNEIDEWIND, N. F., 1992, *Methodology for validating software metrics*, IEEE Transaction Software Engineering, vol. 18, nº 5, May, in (FENTON, 1994).

SCHNEIDEWIND, N. F., 1993, *Report on the IEEE standard for a software quality metrics methodology*, IEEE, Conference on Software Measurements, Montreal, Canada, September.

SCHNEIDEWIND, N. F., 1995, *Software metrics validation*: *Space Shuttle flight software example*, Annals of Software Engineering 1, 287-309

SCHNEIDEWIND, N. F., 1996, *Do Standards*, IEEE Software, January.

SCHWARE, R., 1992, *Software Entry Strategies for Developing Countries: A Walking on Two Legs Proposition*, World Development, vol. 20, no. 2.

SCHWARTZ, D. G., KLIR, G. J., 1992, *Fuzzy logic flowers in Japan*, IEEE Spectrum, July, in (JOYCE, 1994).

SDORRA, P. B., *et al*., 1993, *A measure-theoretic axiomatization of fuzzy sets*, Fuzzy Sets and Systems 60, 295-307.

SEFCIK, J. G., 1994, *Critical Sucess Factors for Implementing Software Quality Plans*, ACM SIGSOFT, Software Engineering Notes, vol. 19 nº 1, January.

SEI, 1995, Software Engineering Institute, *The Capability Maturity Model*, Reading, Massachussetts, Addison-Wesley Co.

SEONG, K. A., 1995, *α-Kernel problem with fuzzy visibility*, Fuzzy Sets and Systems 73, 377-388.

SHAW, M., 1981, *When is "Good" Enough? Evaluating and Selecting Software Metrics*, in Software Metrics: An Analysis and Evaluation, Perlis, A. *et al*., The MIT Press Cambridge.

SHEPPERD, M., 1989, *Metrics, Outlier Analysis and the Software Design Process*, Information and Software Technology, 31, 2, 91-98, in (YU, 1995).

SHEPPERD, M., 1990, *Design metrics: an empirical analysis*, IEE Software Engineering Journal, January.

SHEPPERD, M., 1992, *Products, progress and metrics*, Information and Software Tecnology, vol. 34 nº 10, October, in (OLIVEIRA, 1995b).

SIMMONS, P., 1996, *Quality Outcomes: Determining Business Value*, IEEE Sofware, January.

SIMONELLI, M. R., 1996, *On fuzzy interactive knowledge*, Fuzzy Sets and Systems 80, 159-165.

SLOWINSKI, R *et al*., 1990, *Stochastic Versus Fuzzy Approaches to Multiobjetive Mathematical Programming under Uncertainty*, Kluwer Academic Publishers, Dordrecht, in (HAPKE, 1994).

SOARES, M. T. D. *et al*., 1994, *Total Quality Strategies in Industry: the Experience of Two Multinacionals in Brazil*, Quality Management Journal, nº 1, Milwawkee, ASQC, Quality Press, in (FIORINI, 1995).

SONG, Q., BORTOLAN, G., 1994, *Some properties of defuzzification neural networks*, Fuzzy Sets and Systems 61, 83-89.

SRI, 1994, Software Research Inc., *STW/Coverage Tool Suite for C*, User Manual, Release 2, May, in (VILLAS, 1995).

STAA, A. v., 1987, *Engenharia de Programa*, Livros Técnicos e Científicos Editora, 2ª edição, Rio de Janeiro.

STARK, G. *et al*., 1994, *Using metrics in management decision making*, IEEE Computer, September.

STRIGINI, L., 1996, *Limiting the Dangers of Intuitive Decision Making*, IEEE Software, January.

SUGENO, M., 1985, *Industrial application of fuzzy control*, North Holland, in (CARCHIOLO, 1995).

SUZUKI, H., 1993, *Fuzzy sets and membership functions*, Fuzzy Sets and Systems 58, 123-132.

TANINO, T., 1984, *Fuzzy preference orderings in group decision making*, Fuzzy Sets and Systems 12, 117-131.

TAUSWORTHE, R. C., 1995, *Software quality management through process and product modeling*, Annals of Software Engineering 1, 119-139.

TERVONEN, I., 1996, *Support for Quality-Based Design and Inspection*, IEEE Software, January.

TINNIRELLO, P. C., 1995, *Handbook of Application Development, Second Edition*, Auerbach Publication, Boston, in (GENTLEMAN, 1996).

THOLE, U., ZIMMERMANN, H. J., Zysno, P., 1979, *On the suitability of minimum and product operator for the intersection of fuzzy sets*, Fuzzy Sets and Systems 2, 167-180.

TRILLIUM, 1997, *Trillium Model Overview*, http://ricis.cl.uh.du/process_maturity/ trillium /t3modc1.html, in 21/01/97.

TROSTER, J., TIAN, J., 1995, *Measurement and defect modeling for a legacy software system*, Annals of Software Engineering 1, 95-118.

TSUKUMO, A. N. *et al*., 1995, *ISO/IEC 9126: An Experiment of Application on Brazilian Software Products*, 2nd International Software Engineering Standards Symposium (ISESS'95), Montreal, Quebec, Canada, August.

TSUKUMO, A. N *et al*., 1996a, *Avaliação incremental de Qualidade de Produto de Software baseada na ISO/IEC 9126* (*NBR 13596*), SBES'96 - Workshop de Qualidade, São Carlos.

TSUKUMO, A. N *et al*., 1996b, *Modelos de Processos de Software: Visão Global e Análise Comparativa*, Anais do VII CITS - Conferência Internacional de Tecnologia de Software: Qualidade de Software, Curitiba.

TURBAN, E., 1988, *Decision Support and Expert Systems*, Macmillan, New York, in (RIBEIRO, 1996).

TURKSEN, I. B., 1991, *Measurement of membership functions and their acquisition*, Fuzzy Sets and Systems, IFSA, Special Memorial Volume: 25 years of fuzzy sets, North-Holland - Amsterdam.

TVERSKY, A., KAHNEMANN, D., 1990, *Judgement under uncertainty: heuristics and biases*, in Shafer and Pearl, Eds. Reading in Uncertain Reasoning, in (RIBEIRO, 1996).

VALLE, C., 1997, *Sistemas de Acesso Público para a Educação de Pacientes*, Tese de Mestrado, COPPE/UFRJ, Rio de Janeiro, Maio.

VILLAS-Boas, A. *et al*., 1995, *Estudo Comparativo de Critérios de Teste de Software para o Estabelecimento de Estratégias de Aplicação*, Workshop de Qualidade, IX SBES, Recife.

VIOT, G., 1993, *Fuzzy Logic in C*, Dr. Dobb's Journal, February.

VOTTA Jr., L. G., 1993, *Does Every Inspection Need a Meeting?*, ACM SIGSOFT, Software Engineering.

WATTS, R., 1987, *Measuring software quality*, NCC Publications, in (PONNAMBALAM, 1996).

WEBER, K. C., DeLUCA, J. C. M., ROCHA, A. R. C., 1997, *Qualidade e Produtividade em Software: Termo de Referência do Subprograma Setorial da Qualidade e Produtividade em Software, do Programa Brasileiro da Qualidade e Produtividade -PBQP*, 2ª edição, Makron Books, São Paulo.

WEINGBERG, G. M., 1993, *Quality Software Management*, Vol. 2, First-Order Measurement, Dorset House Publishing, N.Y.

WERNER, C. M. L., 1996, *MEMPHIS: Um Ambiente para Desenvolvimento de Software Baseado em Reutilização - Definição da Arquitetura*, Relatório Tecnico COPPE/UFRJ, 3/96.

WERNERS, B., 1984, *Interaktive Entscheidungsunterstützung values*, Fuzzy Sets and Systems 23, 131-147, in (ZIMMERMANN, 1991).

WHITTY, R. W., 1990, *Structural metrics for Z specifications*, Proceedings of the 4th annual Z User Meeting, in (MARIANO, 1996).

XAVIER, A. C. R., 1992, *Reflexões sobre a qualidade da educação e a Gestão da qualidade total nas escolas*, Tecnologia Educacional, vol 20 (101), Rio de Janeiro, jul/ago, in (CAMPOS 95b).

XU, R. N., Zhai, X. Y., 1992, *Extensions of the analytic hierarchy process in fuzzy environment*, Fuzzy Sets and Systems 52, 117-131.

YAGER, R. R., 1978, *Fuzzy decision making including unequal objectives*, Fuzzy Sets and Systems 1, 87-95, in (RIBEIRO, 1996),

YAGER, R. R., 1980, *On a general class of fuzzy connectives*, FSS 4, 235-242, in (ZIMMERMANN, 1991).

YAGER R. R., 1983a, *Quantifiers in the formulation of multiple objective decision functions*, Informations Science 31, 107-138.

YAGER R. R., 1983b, *Quantifiers propositions in a linguistic logic*, Internat. J. Man-Machine Stud. 19, 195-227, in (ZACPRZYK, 1992).

YAGER, R. R., 1988, *On ordered weighted averagingh aggregation operators in multicriteria decision making*, IEEE Trans. on Systems, Man and Cybernectics, 18 (1) 183-190, in (KLIR, 1995a).

YAGER R. R., 1991, *Connectives and quantifiers in fuzzy sets*, Fuzzy Sets and Systems, IFSA, Special Memorial Volume: 25 years of fuzzy sets, North-Holland - Amsterdam.

YAGER R. R., FILEV, D., 1993a, *On general class of fuzzy connectives*, Fuzzy Sets and Systems 55, 255-271.

YAGER R. R., 1993b, *Toward a General Theory of Information Aggregation*, Information Sciences 68, 191-206.

YAGER R. R., 1994, *Aggregation operators and fuzzy systems modeling*, Fuzzy Sets and Systems 67, 129-145.

YAGER, R. R., RYBALOV, A., 1996, *Uninorm aggregation operators*, Fuzzy Sets and Systems 80, 111-120.

YOURDON, E., 1989, *Structured Walkthroughs*, Prentice Hall, Englewood Cliffs, N.J.

YOURDON, E., 1995, *When Good Enough Software is Best*, IEEE Software, May.

YU, C., 1993, *Correlation of fuzzy numbers*, Fuzzy Sets and Systems 55, 303-307.

YU, X., LAMB, D. A., 1995, *Metrics applicable to software design*, Annals of Software Engineering 1, 23-41.

ZADEH, L. A., 1965, *Fuzzy sets*, Information and Control 8, 338-353.

ZADEH, L. A., 1973a, *Outline of a new approach to the analysis of complex systems and decision process*, IEEE Trans. on Systems, Man, Cybernectics, vol SMCL, no. 1, January.

ZADEH, L. A., 1973b, *The concept of a linguistic variable*, ERL-M Memo, Berkeley, October, in (JOYCE, 1994).

ZADEH, L. A., 1977, *A theory of approximate reasoning*, Memorandoum no. UCB/ERLM 77/58, in (TURKEN, 1991).

ZADEH, L. A., 1978, *Fuzzy sets as a basis for the theory of possibiliy*, Fuzzy Sets and Systems 1.

ZADEH, L. A., 1983, *A computacional approach to fuzzy quantifiers in natural languages*, Computer Mathematics with Applications, 9, 149-184.

ZADEH, L. A., 1988, *Fuzzy logic*, IEEE Transaction Comput. 35, in (TURKEN, 1991).

ZADEH, L. A., 1990, *The birth and evolution of fuzzy logic*, in I.B. Turksen, Ed., Proceeding of NAFIP'90 (June, 6-8).

ZAGE, W. M., ZAGE, D. M., 1995, *Avoiding metric monsters: A design metrics approach*, Annals of Software Engineering 1, 43-55.

ZELENY, M., 1992, *An essay into a philosophy of MCDM: a way of thinking or another algorithm*?, Comput. Oper. Res. 19, 563-566, in (CARLSSON, 1996).

ZIMMERMANN, H. J., 1988, *Fuzzy sets theory - and inference mechanism*, Mitra, 727-741, in (ZIMMERMANN, 1991).

ZIMMERMANN, H. J., 1991, *Fuzzy Set Theory and Its Applications*, Kluwer Boston, 2nd revised edition.

ZIMMERMANN, H. J., 1997, *Operators in Models of Decision Making*, in Fuzzy information engineering: a guide tour of applications, edited by Didier Dubois, Henri Prade, and Ronald R. Yager, Jonh Wiley & Sons, Inc., USA.

ZOVÁCS, Z. L., 1996, *Redes Neurais Artificiais: fundamentos e aplicações*, Editora Acadêmica.

ZUSE, H., 1990, *Software Complexit: Measures and Methods*, Amsterdam: de Gruyter, in (FENTON, 1994).

ZWICK, R. *et al*., 1987, *Measures of similarity among fuzzy concepts*: *A comparative analysis*, Internat, J. Approximate Reasoning 1, in (HSU, 1996).

Questionário de Identificação do Perfil do Especialista

I. Questionário de Identificação do Perfil do Especialista

O *Questionário de Identificação do Perfil do Especialista* (*QIPE*) pode e deve ser adaptado de acordo com as necessidades de quem o está aplicando, bem como o estabelecimento dos pesos para cada um de seus *subitens*, para a obtenção de resultados convenientes. Neste trabalho, utilizou-se um *QIPE*, que contém apenas 7 *itens* (questões) e cada item possui vários *subitens*, como é mostrado na *Seção I.1*.

Sugere-se que a *apuração dos resultados* do *QIPE* siga a escala de pesos apresentada na *Seção I.2*, e que o *escore total* de cada item seja normalizado, quando necessário. ROBINS (1995) adverte que a definição de escores é árdua, pois estes freqüentemente levam a incertezas nos valores finais da avaliação.

Neste contexto, os resultados do *QIPE* serão obtidos como se segue:

i. A apuração de cada *item* do *QIPE* é feita pelo somatório de seus *subitens* selecionados pelo especialista, E_i, que o avaliou, de acordo com escores, previamente definidos para cada um desses *subitens* (*Seção I.2*).

ii. Se for especificado pelo especialista, um valor "*Maior que 7*" para qualquer um dos *subitens* (quantificáveis) acima descritos, acrescentar ao escore correspondente, para cada acréscimo de 7 (isto é, > 14, > 21, ...), o seu fator base, ou seja, o índice que quantifica a opção "*Entre 1 e 2*". Por exemplo, alguém que tenha marcado no *item 2* do *QIPE*, no *subitem* sistemas de *Grande Porte*, simplesmente a opção "*Maior que 7*", receberá o escore igual a 3 . Mas, se tiver, por exemplo, tiver indicado a quantidade 18, para sistemas de *Grande Porte* que tenha desenvolvido, seu escore será 4, isto é, 3 + 1 (>14); se tiver indicado 25, seria 3 + 2 (> 21), e assim por diante. Note-se que, neste caso, o escore 1 corresponde a seu fator base, isto é, o escore pertencente à opção "*Entre 1 e 2*" deste *subitem* exemplificado.

iii. Cada *item* do *QIPE*, quando necessário, deve ser normalizado, isto é, deve ter seu valor, no máximo, igual a 1. Para isto, quando um determinado *item*, considerando-se este mesmo *item* para todos os especialistas participantes do processo de avaliação, possui um *escore-total*, et_i, maior que 1 (um), calcula-se o *escore-normalizado*, en_i, desse *item* em particular, para cada E_i, onde o símbolo $\lceil\ \rceil$ representa o maior valor do *item*, que está sendo apurado. Neste

caso, o *escore-final* do *item* avaliado é *escore-normalizado*. Formalizando-se:

$$en_i = et_i / \lceil et_i \rceil$$

iv. Finalmente, procede-se ao somatório de todos os *escores-finais* dos *itens* (questões) do *QIPE*, obtendo-se, assim, o *total do QIPE*, *tQIPE*, por especialista. Neste trabalho, como se tem apenas 7 *itens* definidos no *QIPE*, então, $tQIPEi \leq 7$.

A seguir, os escores são definidos para cada *item* e *subitem* do *QIPE*, como também as regras para a apuração de seus resultados.

I.1 *QIPE*

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO

COPPE/UFRJ - Programa de Engenharia de Sistemas e Computação

Linha de Pesquisa: Engenharia de Software

Questionário de Identificação do Perfil do Especialista

Produto de Software: _____ *Instituição*:_____

Avaliador: _____ *Fone/fax*: _____

1) *Marque sua experiência ou as atividades (cargos) que já exerceu ou exerce, na área de computação*:

Gerente de projeto Professor (universitário) de informática (ou de cursos de informática de mais de 40 horas)

Analista de sistemas Projetista

Programador Usuário Outra _____

2) *Já participou do desenvolvimento de <u>quantos</u> sistemas de computação*?

Grande Porte:	Nenhum	Entre 1 e 2	Entre 3 e 7	Maior que 7 ____
Médio Porte:	Nenhum	Entre 1 e 2	Entre 3 e 7	Maior que 7 ____
Pequeno Porte:	Nenhum	Entre 1 e 2	Entre 3 e 7	Maior que 7 ____

3) *Se for o caso, em que fases do ciclo de vida de um produto de software já participou*?

Especificação de Requisitos Projeto Teste

Codificação Manutenção Outra(especificar) _____

4) *Quantos produtos de software* (*especificações, projetos, sistemas, etc.*) *similares ao que está sendo avaliado já desenvolveu*?

Nenhum Entre 1 e 2 Entre 3 e 7 Maior que 7 ____

5) *Como você classificaria seu entendimento em relação ao produto de software em questão (conceituação, objetivos e viabilidade do sistema)*:

Excelente Alto Bom Médio Baixo Nenhum

6) *Marque a opção que melhor caracteriza seu grau de escolaridade*?

Segundo Grau Terceiro Grau (área) _____

Especialização (a nível de pós-graduação) _____

Mestrado (área) _____ Doutorado (área) _____

7) *Marque os subitens (e a quantidade a eles referentes), que dizem respeito a seu grau de treinamento em informática:*

Cursos (até 8 hs): Nenhum Entre 1 e 2 Entre 3 e 7 Maior que 7 __

Cursos (até 40 hs): Nenhum Entre 1 e 2 Entre 3 e 7 Maior que 7 __

Cursos (mais de 40 hs): Nenhum Entre 1 e 2 Entre 3 e 7 Maior que 7 __

Simpósios/Congressos: Nenhum Entre 1 e 2 Entre 3 e 7 Maior que 7 __

Publicações de artigos nacionais: Nenhum Entre 1 e 2 Entre 3 e 7 >7__

Publicações de artigos internacionais: Nenhum Entre 1 e 2 Entre 3 e 7 >7__

I.2 Definição dos Escores do *QIPE*

1) *Escore-total* do item*:* _____ *Escore-final* do item*:* _____

Gerente de projeto 1 Professor (universitário) de informática 0,9

(ou de cursos de informática de mais de 40 horas)

Analista de Sistemas 0,8 Projetista 0,7

Programador 0,6 Usuário 0,5

Outros (considerar o escore, se for pertinente à informática) 0,3

2) *Escore-total* do item*:* _____ *Escore-final* do item*:* _____

Grande Porte: Nenhum 0 Entre 1 e 2 1 Entre 3 e 7 2 Maior que 7 3 .

Médio Porte: Nenhum 0 Entre 1 e 2 0,7 Entre 3 e 7 1,4 Maior que 7 2,1 .

Pequeno Porte: Nenhum 0 Entre 1 e 2 0,3 Entre 3 e 7 0,6 Maior que 7 0,9 .

3) *Escore-total* do item*:* _____ *Escore-final* do item*:* _____

Especificação de Requisitos 1 Projeto 0,8 Teste 0,7

Codificação 0,6 Manutenção 0,5 Outra (se pertinente) 0,3

4) *Escore-total* do item*:* _____ *Escore-final* do item*:* _____

Nenhum 0 Entre 1 e 2 0,3 Entre 3 e 7 0,7 Maior que 7 1 .

5) *Escore-total* do item*:* _____ *Escore-final* do item*:* _____

Excelente 1 Alto 0,9 Bom 0,7 Médio 0,5 Baixo 0,3 Nenhum 0

6) *Escore-total* do item*:* _____ *Escore-final* do item*:* _____

Segundo Grau 0,2 Terceiro Grau 0,6 (área de informática)

0,4 (outras áreas)

Especialização (a nível de pós-graduação) 0,7 (área de informática)

0,5 (outras áreas)

Mestrado 0,8 (área de informática) Doutorado 1 (área de informática)

0,6 (outras áreas) 0,8 (outras áreas)

7) *Escore-total* do item*:* _____ *Escore-final* do item*:* _____

Cursos (até 8 hs): Nenhum 0 Entre 1 e 2 0,3 Entre 3 e 7 0,6 Maior que 7 0,9.

Cursos (até 40 hs): Nenhum 0 Entre 1 e 2 0,6 Entre 3 e 7 1,2 Maior que 7 1,8 .

Cursos (mais de 40 hs): Nenhum 0 Entre 1 e 2 0,8 Entre 3 e 7 1,6 Maior que 7 2,4.

Simpósios/Congressos: Nenhum 0 Entre 1 e 2 0,5 Entre 3 e 7 1,0 Maior que 7 1,5.

Publicações de artigos nacionais: Nenhum 0 Entre 1 e 2 0,8 Entre 3 e 7 1,6 > 7 2,4.

Publicações de artigos internacionais: Nenhum 0 Entre 1e2 1 Entre 3 e 7 2 >7 3.

TOTAL DE ESCORES DO QIPE (tQIPE): _____

Atributos de Qualidade de Especificações de Requisitos de Software

II. Atributos de Qualidade para ERS

Os produtos de software são desenvolvidos para atenderem a determinadas necessidades dos usuários. Depois de serem colocados em operação, espera-se que tenham uma vida útil longa e produtiva. Para que isto se concretize, devem ser atingidos os seguintes objetivos de qualidade: *confiabilidade da representação*, *confiabilidade conceitual* e *utilizabilidade* (ROCHA, 1987). Neste apêndice, as tabelas apresentam o conjunto de atributos de qualidade para especificações de requisitos de software (ERS) em geral, definido em CLUNIE (1997), por objetivo de qualidade, segundo o *Modelo Rocha*.

O objetivo ***Confiabilidade da Representação*** refere-se às características que tornam a especificação confiável para seus usuários, considerando-se aspectos referentes à sua forma, e que tornam possível a sua compreensão e manipulação, tendo-se em conta que a especificação evolui ao longo do desenvolvimento, sendo modificada durante a vida útil do produto ao serem feitas manutenções. Este objetivo se realiza através dos fatores de qualidade: ***Comunicabilidade*** e ***Manipulabilidade***.

OBJETIVO: CONFIABILIDADE DA REPRESENTAÇÃO	
COMUNICABILIDADE	Conjunto de atributos de qualidade que avaliam a capacidade da especificação de comunicar seu conteúdo.
Correção no Uso do Método	Conjunto de atributos de qualidade que avaliam a correção da especificação do ponto de vista do uso do método de desenvolvimento.
Correção da Notação	Característica que avalia se a notação do método foi usada de forma correta.
Correção Sintática	Característica que avalia se o conjunto de regras sintáticas definidas no método foi usado de forma correta.
Correção Semântica	Característica que avalia se o conjunto de regras semânticas definidas no método foi usado de forma correta.
Correção no Uso do Formato de Documentação	Característica que avalia se o formato de documentação definido no método foi usado de forma correta.
Uniformidade de Terminologia	Conjunto de atributos de qualidade que avaliam se a especificação foi escrita com uniformidade de termos e notação.
Uniformidade de Termos	Característica que avalia se os termos técnicos foram utilizados de forma uniforme ao longo do texto e de acordo a definições preestabelecidas.
Uniformidade de Notação	Característica que avalia se a notação foi utilizada de forma uniforme ao longo do texto.
Uniformidade no Nível de Abstração	Conjunto de atributos de qualidade que avaliam se a especificação possui um nível de detalhe uniforme, considerando-se um determinado estágio do desenvolvimento.

Legenda

Fatores de qualidade Critérios de qualidade

Subfatores de qualidade

Tabela AII.1 - Características de qualidade relacionadas ao objetivo *Confiabilidade da Representação* para ERS (CLUNIE, 1997)

OBJETIVO: CONFIABILIDADE DA REPRESENTAÇÃO (cont.)	
Uniformidade de Detalhes da Documentação	Característica que avalia se todos os aspectos estão descritos na especificação com o mesmo nível de detalhamento, considerando-se um determinado estágio de desenvolvimento.
Independência de Detalhes do Projeto	Característica que avalia na especificação de requisitos a existência de restrições com relação à escolha de alternativas de solução próprias da fase de projeto.
Modularidade da Documentação	Conjunto de atributos de qualidade que avaliam se as partes da especificação podem ser entendidas e modificadas de forma independente.
Coesão de Informações	Característica que avalia o grau de associação das informações de um capítulo e, dentro deste, seções.
Acoplamento entre as Seções	Característica que avalia o grau de interdependência entre os módulos - *capítulos ou seções* - .
Estrutura da Documentação	Característica que avalia se a estrutura, composta de capítulos e, dentre destes, seções, está organizada numa seqüência lógica.
Concisão	Conjunto de atributos de qualidade que avaliam se a especificação contém um volume mínimo de texto por ter-se maximizado o volume de informação por unidade de texto.
Complementabilidade	Característica que avalia se a especificação faz uso de documentos auxiliares - *referências, glossários, dicionários de dados* - que facilitam seu entendimento.
Conformidade	Conjunto de atributos de qualidade que avaliam se a especificação foi gerada considerando as normas estabelecidas para o desenvolvimento do produto.
Aderência a Normas da Organização Desenvolvedora	Característica que avalia se a especificação foi gerada obedecendo as normas estabelecidas pela organização desenvolvedora.
Aderência a Normas estabelecidas pelo Contratante	Característica que avalia se a especificação foi gerada considerando as normas estabelecidas pelo contratante.
MANIPULABILIDADE	Conjunto de atributos de qualidade que avaliam a facilidade de manipulação da especificação para diversas formas de uso.
Disponibilidade	Conjunto de atributos de qualidade que avaliam a facilidade de acesso de uma especificação para seus usuários autorizados, na sua versão mais atualizada.
Acessibilidade	Característica que avalia se qualquer usuário autorizado pode facilmente consultar a especificação e/ou obter uma cópia da mesma.
Estar Atualizada	Característica que avalia se a especificação reflete a informação mais recente.
Rastreabilidade	Conjunto de atributos de qualidade que avaliam a facilidade de se percorrer as especificações de requisitos e de projeto, identificando a agregação de detalhes a um determinado aspecto, desde sua visão mais global até a mais detalhada e vice-versa.
Organização da Documentação	Característica que avalia se o conjunto de especificações está organizado, de forma a facilitar sua manipulação.
Localizabilidade Interna	Característica que avalia se existem facilidades para se localizar todos os elementos, dentro de uma especificação, relacionados com determinado aspecto ou assunto.
Localizabilidade Externa	Característica que avalia se existem facilidades para localizar todas as especificações e demais documentos relacionados a um determinado aspecto ou assunto.

Legenda

Fatores de qualidade — Critérios de qualidade

Subfatores de qualidade

Tabela AII.1 - Características de qualidade relacionadas ao objetivo *Confiabilidade da Representação* para ERS (continuação)

–

O objetivo ***Confiabilidade Conceitual*** refere-se às características que avaliam se um especificação é confiável para seus usuários, segundo o seu conteúdo, satisfazendo os requisitos que motivaram a sua construção. Este objetivo se realiza através dos fatores de qualidade: ***Fidedignidade*** e ***Suficiência***.

OBJETIVO: CONFIABILIDADE CONCEITUAL	
FIDEDIGNIDADE	Conjunto de atributos de qualidade que avaliam se a especificação representa o que é entendido como sendo as necessidades e expectativas dos usuários do produto.
Consistência	Conjunto de atributos de qualidade que avaliam se a especificação está isenta de contradições entre os aspectos especificados.
Consistência Interna	Característica que avalia se existem conflitos entre aspectos especificados na mesma especificação.
Consistência Externa	Característica que avalia se existem conflitos entre aspectos especificados em outras especificações ou entidades externas.
Não Ambigüidade	Conjunto de atributos de qualidade que avaliam seu o conteúdo da especificação está expresso, de forma a evitar a possibilidade de diferentes interpretações para qualquer aspecto ou assunto.
Ser Explícita	Característica que avalia se na especificação existem condições, hipóteses e/ou restrições definidas por contexto.
Precisão	Característica que avalia se os aspectos especificados estão descritos de forma precisa e, sempre que possível quantificada.
SUFICIÊNCIA	Conjunto de atributos de qualidade que avaliam se estão presentes na especificação todos os aspectos necessários e somente estes.
Necessidade	Conjunto de atributos de qualidade que avaliam se todos os aspectos considerados na especificação são imprescindíveis.
Necessidade dos Requisitos	Característica que avalia se na especificação estão descritos os requisitos considerados imprescindíveis.
Não Redundância	Conjunto de atributos de qualidade que avaliam se existem aspectos repetitivos na especificação.
Não Redundância de Informações	Característica que avalia se um mesmo aspecto é descrito em mais de um lugar da especificação.
Completitude	Conjunto de atributos de qualidade que avaliam se todos os aspectos que devem ser especificados estão presentes na especificação.
Completitude com Relação ao Roteiro definido pela Organização	Característica que avalia se o roteiro definido pela organização desenvolvedora foi totalmente coberto pela especificação.
Completitude com Relação ao Método de Desenvolvimento	Característica que avalia se foram utilizados todos os recursos previstos no método de desenvolvimento.
Completitude com Relação aos Requisitos	Característica que avalia se na especificação estão definidos todos os requisitos estabelecidos para o produto.

Legenda

Fatores de qualidade Critérios de qualidade

Subfatores de qualidade

Tabela AII.2 - Características de qualidade relacionadas ao objetivo *Confiabilidade Conceitual* para ERS (CLUNIE, 1997)

O objetivo ***Utilizabilidade*** refere-se às características de qualidade que tornam possível a utilização da especificação, sob as mais diversas formas e propósitos, durante o processo de desenvolvimento, avaliação, manutenção e implementação. Este objetivo se realiza através dos fatores de qualidade: ***Avaliabilidade, Manutenibilidade, Reutilizabilidade*** e ***Implementabilidade***.

OBJETIVO: UTILIZABILIDADE			
AVALIABILIDADE			Conjunto de atributos de qualidade que avaliam a capacidade da especificação poder ser avaliada com relação à sua forma e conteúdo.
	Verificabilidade		Conjunto de atributos de qualidade que avaliam a facilidade de se avaliar uma especificação segundo à sua forma.
	Validabilidade		Conjunto de atributos de qualidade que avaliam a especificação com relação ao seu conteúdo.
MANUTENIBILIDADE			Conjunto de atributos de qualidade que avaliam a capacidade da especificação poder ser facilmente modificada e detalhada.
	Modificabilidade		Conjunto de atributos de qualidade que avaliam a capacidade da especificação poder ser alterada com facilidade sem com isso perder a sua qualidade.
	Evolutibilidade		Conjunto de atributos de qualidade que avaliam a facilidade de se poder introduzir novos requisitos ou realizar refinamentos em especificações sem que esta perca a qualidade.
REUTILIZABILIDADE			Conjunto de atributos de qualidade que avaliam se a especificação de tem os seus componentes organizados e desenvolvidos de maneira a permitir sua reutilização parcial ou total em outras aplicações similares.
	Adaptabilidade		Conjunto de atributos de qualidade que avaliam a capacidade da especificação poder ser adaptada para corresponder a outra aplicação similar.
	Generalidade		Conjunto de atributos de qualidade que avaliam se a especificação tem os seus componentes desenvolvidos de forma a poderem ser utilizados em outros contextos.
IMPLEMENTABILIDADE			Conjunto de atributos de qualidade que avaliam se uma especificação poder ser implementada, tendo em conta aspectos econômicos, financeiros, tecnológicos, de mão de obra, de cronograma e sociais.
	Viabilidade Econômica		Conjunto de atributos de qualidade que avaliam a compatibilidade entre o custo estimado para o desenvolvimento e operação do software e os benefícios esperados com sua utilização.
		Aceitabilidade de Custos	Característica que avalia se as estimativas de custos, para o desenvolvimento e/ou produção do software, são aceitas por usuários e desenvolvedores.
		Relevância dos Benefícios	Característica que avalia se as estimativas de benefícios tangíveis e intangíveis são aceitas como relevantes, por usuários e desenvolvedores.
		Compatibilidade Custo/Benefício	Característica que avalia se os custos estimados para o desenvolvimento e operação são compatíveis com os benefícios esperados com sua utilização.

Legenda

Fatores de qualidade — Critérios de qualidade

Subfatores de qualidade

Tabela AII.3 - Características de qualidade relacionadas ao objetivo *Utilizabilidade* para ERS (CLUNIE, 1997)

OBJETIVO: UTILIZABILIDADE (cont.)	
Viabilidade Financeira	Conjunto de atributos de qualidade que avaliam a existência e à disponibilidade de capital necessário para conduzir o desenvolvimento do produto especificado.
Existência de Capital	Característica que avalia se a organização possui capital suficiente para custear o desenvolvimento.
Disponibilidade de Capital	Característica que avalia se a organização é capaz de tornar disponível o capital necessário para o desenvolvimento.
Viabilidade Tecnológica	Conjunto de atributos de qualidade que avaliam a existência e à disponibilidade da tecnologia necessária para conduzir o desenvolvimento do produto especificado.
Existência da Tecnologia	Característica que avalia se existe o nível de tecnologia necessário para conduzir o desenvolvimento.
Disponibilidade da Tecnologia	Característica que avalia se a equipe encarregada do desenvolvimento tem disponível a tecnologia necessária para conduzir o desenvolvimento.
Viabilidade de Mão de Obra	Conjunto de atributos de qualidade que avaliam a existência e à disponibilidade da mão de obra necessária para conduzir o desenvolvimento do produto especificado.
Existência de Mão de Obra	Característica que avalia se existe na instalação a mão de obra necessária para o desenvolvimento.
Disponibilidade de Mão de Obra	Característica que avalia se estão disponíveis os recursos humanos com o conhecimento e experiência necessários para realizar o desenvolvimento e operação do software.
Viabilidade de Cronograma	Conjunto de atributos de qualidade que avaliam se o software pode ser construído dentro do limite de tempo estabelecido.
Adequabilidade de Cronograma	Característica que avalia se o software pode ser construído no tempo previsto pelo cronograma, considerando possíveis ocorrências de imprevistos e sem descuidar da qualidade definida para o produto.
Flexibilidade de Cronograma	Característica que avalia se o cronograma aceito para o desenvolvimento pode atender, na medida do possível, fatores tais como introdução de atividades não projetadas, contingências etc.
Viabilidade Social	Conjunto de atributos de qualidade que avaliam as implicações que o software que vai ser construído terá sobre o grupo social ao qual este deverá servir, assim como sobre qualquer outro grupo social externo.
Aceitabilidade da Engenharia Humana	Característica que avalia se o software que vai ser construído leva em consideração o grau de satisfação e o desenvolvimento do potencial humano previsto para os usuários.
Aceitabilidade dos Impactos Sociais	Característica que avalia se o software que vai ser construído leva em consideração seus impactos sobre o sistema social ao qual deverá servir.

Legenda

Fatores de qualidade Critérios de qualidade

Subfatores de qualidade

Tabela AII.3 - Características de qualidade relacionadas ao objetivo *Utilizabilidade* para ERS (continuação)

Os critérios relacionados aos subfatores: (*i*) *verificabilidade* (VER) e *validabilidade* (VAL) do fator *avaliabilidade*, (*ii*) *modificabilidade* (MOD) e *evolutibilidade* (EVO) do

fator *manutenibilidade*, (*iii*) *adaptabilidade* (ADA) e *generalidade* (GEN) do fator *reutilizabilidade*, deste objetivo de qualidade, estão indicados nas Tabelas AII.4 e AII.5.

		UTILIZABILIDADE					
		VER	VAL	MOD	EVO	ADA	GEN
	COMUNICABILIDADE						
C	**Correção no Uso do Método**						
O	Correção da Notação	x		x	x	x	
N	Correção Sintática	x		x	x	x	
F	Correção Semântica	x		x	x	x	
I	Correção no Uso do Formato de Documentação	x		x	x	x	
A	**Uniformidade de Terminologia**						
B	Uniformidade de Termos	x	x	x	x		x
I	Uniformidade de Notação	x	x	x	x		x
L	**Uniformidade no Nível Abstração**						
I	Uniformidade de Detalhes da Documentação	x	x	x	x	x	
D	Independência de Detalhes de Projeto			x	x		x
A	**Modularidade da Documentação**						
D	Coesão das Informações	x		x	x	x	
E	Acoplamento entre as Seções	x		x	x	x	
	Estrutura da Documentação	x		x	x	x	
R	**Correção da Arquitetura**						
E	**Concisão**						
P	Complementabilidade	x					
R	**Conformidade**						
E	Aderência a Normas da Organização Desenvolvedora	x				x	
S	Aderência a Normas estabelecidas pelo Contratante	x				x	
E	**MANIPULABILIDADE**						
N	**Disponibilidade**						
T	Acessibilidade	x		x	x	x	
A	Estar Atualizada	x		x	x	x	
Ç	**Rastreabilidade**						
Ã	Organização da Documentação	x		x	x	x	
O	Localizabilidade Interna	x		x	x	x	
	Localizabilidade Externa	x		x	x	x	

Tabela AII.4 - Características de qualidade do objetivo Confiabilidade da Representação, relacionadas ao objetivo *Utilizabilidade* para ERS (CLUNIE, 1997)

		UTILIZABILIDADE					
		VER	VAL	MOD	EVO	ADA	GEN
	FIDEDIGNIDADE						
C	**Consistência**						
O	Consistência Interna		x	x	x		
N	Consistência Externa		x	x	x		
F	**Não Ambigüidade**						
	Ser Explícita		x	x	x		
C	Precisão		x	x	x		
O	**SUFICIÊNCIA**						
N	**Necessidade**						
C	Necessidade dos Requisitos		x				
E	**Não Redundância**						
I	Não Redundância de Informações		x	x	x	x	
T	**Completitude**						
U	Completitude com Relação ao Método de Desenvolvimento		x				

A	Completitude com Relação ao Roteiro Definido pela Organização		**x**				
L	Completitude com Relação aos Requisitos		**x**				

Tabela AII.5 - Características de qualidade do objetivo Confiabilidade Conceitual, relacionadas ao objetivo *Utilizabilidade* para ERS (CLUNIE, 1997)

Instrumento para Hierarquizar Critérios de Qualidade para Especificações

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
COPPE - PROGRAMA DE ENGENHARIA DE SISTEMAS E COMPUTAÇÃO
Linha de Pesquisa - Engenharia de Software

Instrumento para Hierarquizar Critérios de Qualidade para Especificações de Requisitos de Software (CLUNIE, 1997)

I. Objetivo

Determinar o grau de importância de um conjunto de atributos de qualidade para especificações de requisitos de software em geral.

II. Instruções

Atribua valores de 0 a 4, segundo a escala apresentada na Tabela AIII.1, aos critérios de qualidade definidos para ***Especificações de Requisitos de Software*** de acordo com o grau de importância na sua opinião e/ou experiência.

Grau de Importância	Explicação
0	Indica que a característica que está sendo apresentada não tem nenhuma importância.
1	Indica que a característica que está sendo apresentada tem pouca importância.
2	Indica que a característica que está sendo apresentada tem importância em algumas circunstâncias mas nem sempre.
3	Indica que a característica que está sendo apresentada é muito importante.
4	Indica de maneira absoluta que não há dúvida que a característica que está sendo apresentada é imprescindível.

Tabela AIII.1 - Escala de Valores

III. Identificação do Avaliador

Nome do Avaliador: _____

Instituição: _____

IV. Hierarquização dos Critérios de Qualidade de Especificações em Geral

OBJETIVO: CONFIABILIDADE DA REPRESENTAÇÃO	
Critérios	**Importância**
1. Correção da Notação	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
2. Correção Sintática	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
3. Correção Semântica	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
4. Correção no Uso do Formato de Documentação	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
5. Uniformidade de Termos	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
6. Uniformidade de Notação	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
7. Uniformidade de Detalhes da Documentação	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
8. Independência de Detalhes de Projeto	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
9. Coesão das Informações	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
10. Acoplamento entre Seções	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
11. Estrutura da Documentação	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
12. Complementabilidade	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
13. Aderência às Normas da Organização Desenvolvedora	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
14. Aderência às Normas Estabelecidas pelo Contratante	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
15. Acessibilidade	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
16. Estar Atualizada	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
17. Organização da Documentação	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
18. Localizabilidade Interna	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
19. Localizabilidade Externa	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒

OBJETIVO: CONFIABILIDADE CONCEITUAL	
Critérios	**Importância**
1. Consistência Interna	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
2. Consistência Externa	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
3. Ser Explícita	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
4. Precisão	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
5. Necessidade dos Requisitos	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
6. Não Redundância das Informações	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
7. Completitude com Relação ao Roteiro definido pela Organização Desenvolvedora	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
8. Completitude com Relação ao Método de Desenvolvimento	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
9. Completitude com Relação aos Requisitos	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒

OBJETIVO: UTILIZABILIDADE	
Critérios	**Importância**
1. Aceitabilidade de Custos	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
2. Relevância dos Benefícios	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
3. Compatibilidade Custo/Benefício	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
4. Existência de Capital	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
5. Disponibilidade de Capital	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
6. Existência de Tecnologia	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
7. Disponibilidade de Tecnologia	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
8. Existência de Mão de Obra	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
9. Disponibilidade de Mão de Obra	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
10. Adequabilidade de Cronograma	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
11. Flexibilidade de Cronograma	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
12. Aceitabilidade dos Impactos Sociais	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒
13. Aceitabilidade da Engenharia Humana	0 1 2 3 4 ❒ ❒ ❒ ❒ ❒

Formulário de Avaliação da Qualidade de Especificações de Requisitos de Software

_

Critérios de Avaliação de Especificações de Requisitos de Software (*CAERS*)

I. Instruções

Assinale de acordo com a escala abaixo, *Tabela AIV.1*, o valor que lhe parece melhor representar o grau em que cada critério foi atingido (adaptado de CLUNIE, (1997)).

ESCALA	EQUIVALÊNCIA	INTERPRETAÇÃO
0	*Total Ausência*	Indica de maneira absoluta que o critério está ausente ou não possui nenhuma importância.
1	*Baixa Presença*	Indica um baixo grau de presença do critério, seja por deficiência ou pouca importância do mesmo.
2	*Moderada Presença*	Indica um grau de presença moderada (aceitável) do critério.
3	*Alta Presença*	Indica um alto grau de presença do critério, mas não de forma plena.
4	*Total Presença*	Indica que não há dúvidas de que o critério está totalmente presente.

Tabela AIV.1 - Graus de Presença do Critério

Nome do Avaliador: _____

Fone: _____ **Instituição:** _____

_

Especificações de Requisitos de Software
OBJETIVO: Confiabilidade da Representação

1. Correção da Notação	**Técnica de Avaliação:** Inspeção Individual
Definição: característica que avalia se a notação do método foi usada de forma correta.	
Processo de Avaliação: • A especificação está escrita utilizando de forma correta a notação pré-definida no método de especificação?	**Valores** 0 1 2 3 4 □ □ □ □ □
Comentários:	

2. Correção Sintática	**Técnica de Avaliação:** Inspeção Individual
Definição: característica que avalia se o conjunto de regras sintáticas, pré-definidas no método de especificação, foi usado de forma correta.	
Processo de Avaliação: • A especificação está escrita utilizando de forma correta o conjunto de regras sintáticas pré-definidas no método de especificação?	**Valores** 0 1 2 3 4 □ □ □ □ □
Comentários:	

3. Correção Semântica	**Técnica de Avaliação:** Inspeção Individual
Definição: característica que avalia se o conjunto de regras semânticas pré-definidas no método foi usado de forma correta.	
Processo de Avaliação: • A especificação está escrita utilizando de forma correta o conjunto de regras semânticas pré-definidas no método de especificação?	**Valores** 0 1 2 3 4 □ □ □ □ □
Comentários:	

4. Correção no Uso do Formato de Documentação	**Técnica de Avaliação:** Inspeção Individual
Definição: característica que avalia se o formato de documentação definido no método foi usado de forma correta.	
Processo de Avaliação: • A especificação está escrita utilizando de forma correta o formato de documentação definido no método?	**Valores** 0 1 2 3 4 □ □ □ □ □
Comentários:	

_

5. Uniformidade de Termos	**Técnica de Avaliação:** Inspeção Individual
Definição: característica que avalia se os termos técnicos foram utilizados de forma uniforme ao longo do texto e de acordo com as definições pré-estabelecidas.	
Processo de Avaliação: • Os termos técnicos são utilizados de forma uniforme ao longo da especificação e obedecem a definições pré-estabelecidas?	**Valores** 0 1 2 3 4 □ □ □ □ □
Comentários:	

6. Uniformidade de Notação	**Técnica de Avaliação:** Inspeção Individual
Definição: característica que avalia se a notação foi utilizada de forma uniforme ao longo do texto.	
Processo de Avaliação: • A notação foi utilizada de forma uniforme ao longo da especificação e obedece a definições pré-estabelecidas?	**Valores** 0 1 2 3 4 □ □ □ □ □
Comentários:	

7. Uniformidade de Detalhes da Documentação	**Técnica de Avaliação:** Reunião de Inspeção / Inspeção Individual
Definição: característica que avalia se todos os aspectos estão descritos na especificação com o mesmo nível de detalhamento, considerando-se um determinado estágio de desenvolvimento.	
Processo de Avaliação: • Os aspectos descritos na especificação apresentam o mesmo nível de detalhamento, considerando-se o estágio de desenvolvimento?	**Valores** 0 1 2 3 4 □ □ □ □ □
Comentários:	

8. Independência de Detalhes de Projeto	**Técnica de Avaliação:** Reunião de Inspeção / Inspeção Individual
Definição: característica que avalia na especificação de requisitos a existência de restrições com relação à escolha de alternativas de solução próprias da fase de projeto.	
Processo de Avaliação: • A especificação de requisitos não apresenta descritas, restrições com relação à escolha de alternativas próprias da fase de projeto e/ou implementação?	**Valores** 0 1 2 3 4 □ □ □ □ □
Comentários:	

_

9. Coesão de Informações	**Técnica de Avaliação:** Reunião de Inspeção / Inspeção Individual
Definição: característica que avalia o grau de associação das informações de um capítulo e, dentro deste, seções.	
Processo de Avaliação: • A informação que apresenta o módulo - *capítulo ou seção* - descreve aspectos relacionados ao capítulo ou seção que está sendo descrito?	**Valores** 0 1 2 3 4 □ □ □ □ □
Comentários:	

10. Acoplamento entre Seções	**Técnica de Avaliação:** Inspeção Individual
Definição: característica que avalia o grau de interdependência entre os módulos - *capítulos ou seções* - da especificação.	
Processo de Avaliação: • O conteúdo de um módulo - *capítulo ou seção* - faz referência ao conteúdo de outro módulo, apenas como complemento, sem modificá-lo?	**Valores** 0 1 2 3 4 □ □ □ □ □
Comentários:	

11. Estrutura da Documentação	**Técnica de Avaliação:** Inspeção Individual
Definição: característica que avalia se a estrutura, composta de capítulos e, dentre seções, está organizada numa seqüência lógica.	
Processo de Avaliação: • A especificação tem seus capítulos e seções organizadas numa seqüência lógica?	**Valores** 0 1 2 3 4 □ □ □ □ □
Comentários:	

12. Complementabilidade	**Técnica de Avaliação:** Inspeção Individual
Definição: característica que avalia se a especificação faz uso de documentos auxiliares - *referências, glossários, dicionários de dados* - que facilitam seu entendimento.	
Processo de Avaliação:	**Valores**
1. Existe um glossário com definições de termos técnicos, simbologia e notação utilizados na especificação e que não são de uso comum?	0 1 2 3 4 □ □ □ □ □
2. Existe um dicionário de dados centralizando as informações?	0 1 2 3 4 □ □ □ □ □
3. Existe referência a documentos prévios e/ou complementares necessários ao entendimento da especificação?	0 1 2 3 4 □ □ □ □ □
Comentários:	

_

13. Aderência às Normas da Organização Desenvolvedora	**Técnica de Avaliação:** Inspeção Individual
Definição: característica que avalia se a especificação foi gerada considerando as normas estabelecidas pela organização desenvolvedora.	
Processo de Avaliação: • A especificação foi gerada considerando as normas estabelecidas pela organização desenvolvedora?	**Valores** 0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
Comentários:	

14. Aderência às Normas estabelecidas pelo Contratante	**Técnica de Avaliação:** Inspeção Individual
Definição: característica que avalia se a especificação foi gerada considerando as normas estabelecidas pelo contratante.	
Processo de Avaliação: • A especificação foi gerada considerando as normas estabelecidas pelo contratante?	**Valores** 0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
Comentários:	

15. Acessibilidade	**Técnica de Avaliação:** Inspeção Individual
Definição: característica que avalia se qualquer usuário autorizado pode facilmente consultar a especificação e/ou obter uma cópia da mesma.	
Processo de Avaliação: 1. A especificação é possível de ser acessada por todos os seus usuários autorizados?	**Valores** 0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
2. A especificação pode ser facilmente reproduzida?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
3. Existe uma cópia de segurança da especificação fora do local de trabalho?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
Comentários:	

16. Estar Atualizada	**Técnica de Avaliação:** Inspeção Individual
Definição: característica que avalia se a especificação reflete a informação mais recente.	
Processo de Avaliação: 1. O conteúdo da especificação corresponde à informação mais recente, fruto da última manutenção?	**Valores** 0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
2. As atualizações realizadas estão todas datadas?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
3. As atualizações realizadas estão claramente indicadas?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
Comentários:	

_

17. Organização da Documentação	**Técnica de Avaliação:** Inspeção Individual
Definição: característica que avalia se o conjunto de especificações está organizado de forma a facilitar sua manipulação.	
Processo de Avaliação: • O conjunto de especificações é facilmente manuseável, em virtude da organização de sua documentação?	**Valores** 0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
Comentários:	

18. Localizabilidade Interna	**Técnica de Avaliação:** Inspeção Individual
Definição: característica que avalia se existem facilidades para se localizar todos os elementos, dentro de uma especificação, relacionados com determinado aspecto ou assunto.	
Processo de Avaliação: 1. A especificação apresenta um sumário?	**Valores** 0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
2. Existem índices remissivos?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
3. Existem referência cruzadas explícitas?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
Comentários:	

19. Localizabilidade Externa	**Técnica de Avaliação:** Inspeção Individual
Definição: característica que avalia se existem facilidades para se localizar todas as especificações e demais documentos relacionados a um determinado aspecto assunto.	
Processo de Avaliação: 1. Existem sumários que organizam os documentos?	**Valores** 0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
2. Existem índices remissivos?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
3. Existem referência cruzadas explícitas?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
Comentários:	

_

Especificações de Requisitos de Software
OBJETIVO: Confiabilidade Conceitual

1. Consistência Interna	**Técnica de Avaliação:** Reunião de Inspeção / Inspeção Individual
Definição: característica que avalia a existência de conflitos entre aspectos especificados na mesma especificação.	
Processo de Avaliação:	**Valores**
1. A especificação não apresenta termos diferentes que possuem o mesmo significado e que são utilizados para descrever um mesmo objeto, que é tratado em contextos e lugares diferentes?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
2. A especificação não apresenta contradições entre características específicas do produto?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
3. A especificação não apresenta conflitos lógicos ou temporais entre requisitos do produto que dependem do tempo?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
4. A especificação não apresenta conflitos na descrição de aspectos de comportamento do produto?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
Comentários:	

2. Consistência Externa	**Técnica de Avaliação:** Reunião de Inspeção / Inspeção Individual
Definição: característica que avalia a existência de conflitos entre aspectos especificados em outras especificações ou entidades externas.	
Processo de Avaliação:	**Valores**
• A especificação não apresenta aspectos conflitantes com relação a outras especificações ou entidades externas?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
Comentários:	

3. Ser Explícita	**Técnica de Avaliação:** Reunião de Inspeção/ Inspeção Individual
Definição: característica que avalia se na especificação existem condições, hipóteses e/ou restrições definidas por contexto.	
Processo de Avaliação:	**Valores**
• A especificação não apresenta aspectos definidos por contexto?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
Comentários:	

_

4. Precisão	**Técnica de Avaliação:** Reunião de Inspeção/ Inspeção Individual
Definição: característica que avalia se os aspectos especificados estão descritos de forma precisa e, sempre que possível, quantificada.	
Processo de Avaliação:	**Valores**
1. Os termos de significado múltiplo, apresentam uma definição precisa?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
2. Os requisitos do produto estão descritos de forma possível de serem validados?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
Comentários:	

5. Necessidade dos Requisitos	**Técnica de Avaliação:** Reunião de Inspeção/ Inspeção Individual
Definição: característica que avalia se na especificação estão descritos os requisitos considerados imprescindíveis.	
Processo de Avaliação:	**Valores**
• Os aspectos descritos na especificação são considerados imprescindíveis?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
Comentários:	

6. Não Redundância de Informações	**Técnica de Avaliação:** Reunião de Inspeção / Inspeção Individual
Definição: característica que avalia se um mesmo aspecto é descrito em mais de um lugar da especificação.	
Processo de Avaliação:	**Valores**
• A especificação não apresenta aspectos redundantes?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
Comentários:	

7. Completitude com Relação ao Roteiro Definido pela Organização Desenvolvedora	**Técnica de Avaliação:** Inspeção Individual
Definição: característica que avalia se o roteiro definido pela organização desenvolvedora foi totalmente coberto pela especificação.	
Processo de Avaliação:	**Valores**
• O roteiro definido pela organização desenvolvedora foi totalmente coberto pela especificação?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
Comentários:	

_

8. Completitude com Relação ao Método de Desenvolvimento	**Técnica de Avaliação:** Inspeção Individual
Definição: característica que avalia se foram utilizados todos os recursos previstos no método de desenvolvimento.	
Processo de Avaliação:	**Valores**
• Todos os recursos que auxiliam na produção da documentação, que o método que está sendo aplicado fornece, são utilizados?	0 1 2 3 4 □ □ □ □ □
Comentários:	

9. Completitude com Relação aos Requisitos	**Técnica de Avaliação:** Reunião de Inspeção/ Inspeção Individual
Definição: característica que avalia se na especificação estão definidos todos os requisitos do produto.	
Processo de Avaliação:	**Valores**
1. Todas as funções a serem desempenhadas pelo software estão definidas e modeladas?	0 1 2 3 4 □ □ □ □ □
2. Todas as necessidades de informação estão identificadas?	0 1 2 3 4 □ □ □ □ □
3. Todas as interfaces com o usuário estão identificadas?	0 1 2 3 4 □ □ □ □ □
4. Todas as interfaces com o ambiente de hardware estão identificadas?	0 1 2 3 4 □ □ □ □ □
5. Todas as interfaces com outros software estão identificadas?	0 1 2 3 4 □ □ □ □ □
6. Todas as possíveis respostas que o sistema deve fornecer para todas as possíveis classes de entrada constatadas, válidas ou inválidas, estão identificadas?	0 1 2 3 4 □ □ □ □ □
7. Todas as características relativas ao desempenho do software estão identificadas?	0 1 2 3 4 □ □ □ □ □
8. Todas as características de qualidade exigidas pelo usuário estão identificadas?	0 1 2 3 4 □ □ □ □ □
9. Todas as características de qualidade inerentes ao software são identificadas?	0 1 2 3 4 □ □ □ □ □
10. Todas as categorias possíveis de tratamento de erros e exceções, por falhas de hardware e software estão identificadas?	0 1 2 3 4 □ □ □ □ □
Comentários:	

_

Especificações de Requisitos de Software
OBJETIVO: Utilizabilidade

1. Aceitabilidade de Custos	**Produto:** Especificação de Requisitos
Definição: Característica que avalia se as estimativas de custos, para o desenvolvimento e/ou produção do software, são aceitas por usuários e desenvolvedores.	
Processo de Avaliação:	**Valores**
1. O custo estimado para o desenvolvimento do software é aceitável?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
2. O custo estimado para operação do software é aceitável?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
Comentários:	

2. Relevância de Benefícios	**Produto:** Especificação de Requisitos
Definição: Característica que avalia se as estimativas de benefícios tangíveis e intangíveis são aceitas como relevantes, por usuários e desenvolvedores.	
Processo de Avaliação:	**Valores**
1. As estimativas de benefícios tangíveis são aceitáveis?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
2. As estimativas de benefícios intangíveis (atribuindo valores) são aceitáveis?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
Comentários:	

3. Compatibilidade Custo/Benefício	**Produto:** Especificação de Requisitos
Definição: Característica que avalia se os custos estimados para o desenvolvimento e operação do produto são compatíveis com os benefícios esperados com sua utilização.	
Processo de Avaliação:	**Valores**
• A relação custo/benefício é aceitável?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
Comentários:	

4. Existência de Capital	**Produto:** Especificação de Requisitos
Definição: Característica que avalia se a organização possui capital suficiente para custear o desenvolvimento.	
Processo de Avaliação:	**Valores**
• A empresa possui capital suficiente para custear o desenvolvimento?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐

Comentários:

5. Disponibilidade de Capital	**Produto:** Especificação de Requisitos
Definição: Característica que avalia se a organização é capaz de tornar disponível o capital necessário para o desenvolvimento.	
Processo de Avaliação: 1. Existem recursos financeiros disponíveis para o desenvolvimento?	**Valores** 0 1 2 3 4 □ □ □ □ □
2. Caso não estejam disponíveis os recursos, existe a possibilidade de serem obtidos?	0 1 2 3 4 □ □ □ □ □
Comentários:	

6. Existência de Tecnologia	**Produto:** Especificação de Requisitos
Definição: Característica que avalia se existe o nível de tecnologia necessário para conduzir o desenvolvimento.	
Processo de Avaliação: • Existe a tecnologia necessária para desenvolver o software especificado?	**Valores** 0 1 2 3 4 □ □ □ □ □
Comentários:	

7. Disponibilidade de Tecnologia	**Produto:** Especificação de Requisitos
Definição: Característica que avalia se a equipe encarregada do desenvolvimento tem disponível a tecnologia necessária para conduzir o desenvolvimento.	
Processo de Avaliação: 1. A equipe encarregada pelo desenvolvimento dispõe da tecnologia que precisa em termos de hardware?	**Valores** 0 1 2 3 4 □ □ □ □ □
2. Caso a tecnologia de hardware não esteja disponível, ela pode ser adquirida?	0 1 2 3 4 □ □ □ □ □
3. A equipe encarregada pelo desenvolvimento dispõe da tecnologia que precisa em termos de software?	0 1 2 3 4 □ □ □ □ □
4. Caso a tecnologia de software não esteja disponível, ela pode ser adquirida ou desenvolvida?	0 1 2 3 4 □ □ □ □ □
Comentários:	

8. Existência de Mão de Obra	**Produto:** Especificação de Requisitos
Definição: Característica que avalia se existe na instalação a mão de obra necessária para o desenvolvimento.	
Processo de Avaliação: • Existe a mão de obra para construir o software?	**Valores** 0 1 2 3 4 □ □ □ □ □

Comentários:

9. Disponibilidade de Mão de Obra	**Produto:** Especificação de Requisitos
Definição: Característica que avalia se estão disponíveis os recursos humanos com o conhecimento e experiência necessários para realizar o desenvolvimento e operação do software.	
Processo de Avaliação:	**Valores**
1. A organização dispõe de mão de obra para o desenvolvimento e operação do software em termos de conhecimento e domínio da tecnologia?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
2. Caso não disponha de mão de obra, para o desenvolvimento e operação é possível adquiri-la por meio de treinamento?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
3. Caso não disponha de mão de obra, para o desenvolvimento e operação é possível adquiri-la por meio de contratação de pessoal?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
Comentários:	

10. Adequabilidade de Cronograma	**Produto:** Especificação de Requisitos
Definição: Característica que avalia se o software pode ser construído no tempo previsto pelo cronograma, considerando possíveis ocorrências de imprevistos e sem descuidar da qualidade definida para o produto.	
Processo de Avaliação:	**Valores**
• O software é possível de ser construído no tempo previsto pelo cronograma?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
Comentários:	

11. Flexibilidade de Cronograma	**Produto:** Especificação de Requisitos
Definição: Característica que avalia se o cronograma aceito para o desenvolvimento pode atender, na medida do possível, fatores tais como introdução de atividades não projetadas, contingências etc..	
Processo de Avaliação:	**Valores**
• Existe flexibilidade de cronograma?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
Comentários:	

12. Aceitabilidade de Engenharia Humana	**Produto:** Especificação de Requisitos
Definição: Característica que avalia se o software que vai ser construído leva em consideração o grau de satisfação e o desenvolvimento do potencial humano previsto para os usuários.	
Processo de Avaliação:	**Valores**
1. Poderá o software especificado prover uma forma satisfatória para que o usuário possa desempenhar a sua função operacional?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
2. Poderá o software especificado satisfazer as necessidades humanas	0 1 2 3 4

em seus diversos níveis?	☐ ☐ ☐ ☐ ☐
3. Poderá o software especificado ajudar ao desenvolvimento das capacidades humanas?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
Comentários:	

13. Aceitabilidade dos Impactos Sociais	**Produto:** Especificação de Requisitos
Definição: característica que avalia se o software que vai ser construído leva em consideração seus impactos sobre o sistema social ao qual deverá servir.	
Processo de Avaliação: • O software especificado leva em consideração seus impactos sobre o sistema social ao qual deverá servir?	0 1 2 3 4 ☐ ☐ ☐ ☐ ☐
Comentários:	