UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Alexandre Rodrigues Monge

Um Modelo para Verificação de Qualidade e Normalização de Software

Dissertação submetida à Universidade Federal de Santa Catarina como parte dos requisitos para a obtenção do grau de Mestre em Ciência da Computação.

Dr. João Bosco da Mota Alves
Orientador

Florianópolis, 05/2001

Um Modelo para Verificação de Qualidade e Normalização de Software

Alexandre Rodrigues Monge

Esta Dissertação foi julgada adequada para a obtenção do título de Mestre em Ciência da Computação Área de Concentração Sistemas de Computação e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

Fernando A. Ostuni Gauthier, Dr.
Coordenador do CPGCC

João Bosco da Mota Alves, Dr.
Orientador

Banca Examinadora

João Arthur de Souza, Dr.

Luiz Fernando Jacinto Maia, Dr.

DEDICATÓRIA

Dedico meus esforços à aqueles que durante estes anos não me deixaram desanimar, e batalharam junto comigo. Entre eles destaco meus pais, minha namorada e a todos companheiros de sala de aula, que contribuíram para minha realização neste momento.

AGRADECIMENTOS

A Deus que durante toda a minha caminhada me deu forças para que eu pudesse lutar nas horas de dificuldade e me abençoou com mais esta vitória.

Aos meus pais, que sempre me apoiaram e deram forças para continuar buscando meus objetivos.

A todos os professores, que me oîereceram os conhecimento necessários para meu crescimento profissional., em especial ao professor João Bosco Alves que teve uma grande participação, orientando e fornecendo conselhos, para realização desta Dissertação.

E à UNIPAR: Universidade Paranaense e à UFSC: Universidade Federal de Santa Catarina.

Um Modelo para Verificação de Qualidade e Normalização de Software

Autor: Alexandre Rodrigues Monge

Orientador: Dr. João Bosco Alves

Resumo

A demanda por qualidade tem motivado a comunidade de software para o desenvolvimento de modelos para qualidade de software. Estes modelos estão orientados por duas visões: visão de processo e visão de produto. A visão de processo trata da avaliação e melhoria dos processos utilizados para o ciclo de vida do software. A visão de produto trata da avaliação de um produto de software, para verificação de sua qualidade. Esta dissertação introduz conceitos de qualidade de software e descreve os principais modelos e normas que tratam destas duas visões.

Abstract

The demand for quality has been motivating the software community for the development of models for quality software . These models are guided by two visions: process vision and product vision. The process vision treat the evaluation and improvement of the processes used for the cycle of life of the software. The product vision treat the evaluation of a software product, for verification of its quality. This dissertation introduces concepts of quality software and it describes the main models and norms that treated about this two visions.

Palavras-Chave

Qualidade de Software, Qualidade de Processo, Qualidade de Produto, NBR ISO 9001, NBR ISO 13596, ISO 9000-3, ISO/IEC 12207, Série ISO/IEC 14598, Série ISO/IEC 12119, ISO 9126, CMM, SPICE, Métricas de Software.

Key-words

Software Quality , Process Quality, Product Quality, NBR ISO 9001, NBR ISO 13596, ISO 9000-3, ISO/IEC 12207, Série ISO/IEC 14598, Série ISO/IEC 12119, ISO 9126, CMM, SPICE, Software Metrics.

LISTA DE FIGURAS

Figura 1 - : Visões da ISO 12207 19
Figura 2 - A Estrutura do CMM 21
Figura 3: Conjunto de atividades do processo de avaliação 61
Figura 4 - Característica de Qualidade 64
Figura 5 - Nível de Planejamento 65
Figura 6 - O Processo de Avaliação 74
Figura 7 - Elementos de um Sistema Especialista 119
Figura 8 - Base de Conhecimento do "QUALISOFT" 120
Figura 9: Funcionamento do Mecanismo de Inferência 121
Figura 10 - Engenharia do Conhecimento 122
Figura 11 - Regra de Produção sobre Legibilidade 125

LISTA DE SIGLAS

ISO	International Organization for Standardization
IEC	International Electrotechnical Comission,
NBR	Normas Brasileiras
IEEE	Instituto de Engenharia Elétrica e Eletrônica
ABNT	Associação Brasileira de Normas Técnicas
ANS	American National Standard
SEI	Instituto de Engenharia de Software
CMM	Capability Maturity Model
SPICE	Software Process Improvement and Capability dEterminatio
PSP	Personal Software Process
SE	Sistema Especialista
EC	Engenharia do Conhecimento

LISTA DE TABELAS

Tabela 1 – Principais Normais Nacionais e Internacionais..... 06
Tabela 2 – Conjunto de Normas ISO 9000..... 08
Tabela 3 – : Título das Normas ISO 9000..... 09
Tabela 4 – Sub Partes da ISO 9000..... 10
Tabela 5 – A Norma ISO 9004..... 10
Tabela 6 – Normas Integrantes da Série ISO 9000..... 11
Tabela 7 – Diretrizes da ISO 9000-3..... 12
Tabela 8 – Processos do Ciclo de Vida do Software..... 17
Tabela 9 – Os Níveis do CMM..... 24
Tabela 10 – As Áreas-chave de Processo do CMM..... 29
Tabela 11 – Objetivos das áreas-chave..... 30
Tabela 12 – Características Comuns e Práticas-base..... 33
Tabela 13 – Níveis do PSP..... 35
Tabela 14 – Categorias e Processos do SPICE..... 38
Tabela 15 – Níveis de Capacitação do SPICE..... 40
Tabela 16 – Os Manuais do SPICE..... 41
Tabela 17 – Variáveis Univaloradas existentes no sistema "QUALISOFT"..... 123
Tabela 18 - Variáveis Multivaloradas existentes no sistema "QUALISOFT"..... 123
Tabela 19 - Exemplo de perguntas existentes no "QUALISOFT"..... 126

SUMÁRIO

1) Introdução 01

2) Qualidade 04

3) Qualidade De Softwares 06
3.1) Qualidade Do Processo De Software 07
3.1.1) A Série ISO 9000 08
3.1.1.1) Sub Partes Da ISO 9000 10
3.1.1.2) O Padrão ISO 9000/3 11
3.1.2) ISO 12207 - Processos Do Ciclo De Vida Do Software 17
3.1.3) CMM - Capability Maturity Model 20
3.1.3.1) Definição Dos Níveis De Maturidade Do CMM 23
3.1.3.2) As Áreas-Chave De Processo Do CMM 28
3.1.3.3) Objetivos Das Áreas-Chave De Processo 29
3.1.3.4) Características Comuns e Práticas-Base 33
3.1.4) PSP - Personal Software Process 34
3.1.4.1) Níveis Do PSP 34
3.1.5) SPICE - Software Process Improvement And Capability Determination – ISO 15504 36
3.1.5.1) Categorias e Processos 37
3.1.5.2) Níveis De Capacitação 39
3.1.5.3) Os Nove Manuais Do SPICE 41
3.1.6) O Modelo Trillium - Modelo De Qualidade De Processo De Sofware 41
3.1.6.1) Formas De Utilização Do Modelo 42
3.1.6.2) Descrição Do Modelo 43
3.1.6.3) Níveis De Capacidade 44
3.1.6.4) As Práticas Trillium 45
3.2) Qualidade De Produtos De Software 47
3.2.1) ISO/IEC 9126 [NBR 13596] - Software Quality Characteristics And Metrics 48
3.2.1.1) Definições 48
3.2.1.2) Conceituação e Estrutura 50
3.2.1.3) Implementação da ISO 9126 51
3.2.1.4) Aplicacao da Norma 9126 55
3.2.2) ISO/IEC 14598 - Software Product Evaluation 56
3.2.2.1) Descrição Das Partes Da ISO/IEC 14598 57
3.2.2.2) O Processo De Avaliação 61
3.2.2.3) Processo De Avaliação De Produto De Software 62

3.2.3) ISO/IEC 12119 - Tecnologia Da Informação Pacotes De *Software* –Requisitos De Qualidade E Testes 75
3.2.3.1) Escopo 76
3.2.3.2) Definições 78
3.2.3.3) Requisitos De Qualidade 79
3.2.3.4) Descrição Do Produto 80
3.2.3.5) Documentação Do Usuário 85
3.2.3.6) Programas e Dados 87
3.2.3.7) Instruções Para Testes 90
3.2.3.8) Pré-Requisitos De Testes 91
3.2.3.9) Atividades De Testes 92
3.2.3.10) Registros De Teste 94
3.2.3.11) Relatório De Teste 94
3.2.3.12) Teste De Acompanhamento 96

4) Métricas De Qualidade De Software 97
4.1) Categorias De Métricas 99
4.2) Características Das Métricas 101
4.3) A Aplicação De Métricas 104
4.3.1) As Medições Por Estimativas 106
4.4) Validação De Métricas De Software 108

5) Técnicas Para Controle Da Qualidade 111
5.1) *Walkthrough e* Inspeções 112
5.2) Testes De Softwares 114
5.3) Método *Cleanroom* 115
5.4) Modelos De Confiabilidade 116

6) Especificações do Modelo Proposto 118
6.1) Estrutura do Modelo QUALISOFT 119
6.2) Regras de Produção 124
6.3) Tratamento de Incertezas 125

7) Conclusão 127
7.1) Objetivos 127
7.2) Limitações do QUALISOFT 128
7.3) Propostas Futuras 128

8) Referências Bibliográficas 129

1) Introdução

Atualmente, a questão da qualidade tornou-se uma necessidade para que as empresas possam vencer em um mercado global, hostil e agressivo. Mais do que nunca são exigidas das empresas provas formais dos processos de qualidade usados para projetar, desenvolver e suportar comercialmente os seus produtos.

Para realizar o acompanhamento da qualidade, são estabelecidos procedimentos, parâmetros e medidas as mais objetivas possíveis, através das quais se pode verificar se determinado produto está ou não dentro de limites aceitáveis. Para tanto, procura-se definir, para cada característica do produto, uma medida padrão composta de quantidade e unidade, de forma a tornar a avaliação a mais independente possível do avaliador.

No caso do software, entretanto, apesar de ser cada vez mais tratado como outro bem qualquer, raramente; é submetido a avaliações formais de qualidade. Na verdade, são poucas as ações tomadas nesse sentido. Normalmente, o assunto é apenas tratado superficialmente, ficando a qualidade do software dependente, basicamente, da habilidade e da opinião dos programadores e analistas.

Em sistemas mais complexos, a impossibilidade de testar o software por exaustão leva as empresas a lançar versões preliminares do software no mercado (conhecidas como versão Beta) para serem depuradas no campo por clientes específicos, acompanhados de perto pelos fornecedores do software.

Os erros encontrados nos testes "*Beta*" são corrigidos através do lançamento de sucessivas versões. O inconveniente da liberação de versões com número excessivo de erros, é levar lançamentos importantes à falta de credibilidade, acarretando sérios problemas na sua futura comercialização.

Por outro lado, softwares desenvolvidos a partir de requisitos estabelecidos pelo cliente, e que apresentam níveis satisfatórios de qualidade em seus atributos, normalmente são produtos mais confiáveis e com maior freqüência, encontram-se na lista dos vencedores.

Como conseqüência direta da pouca atenção dispensada à qualidade do software, principalmente nas etapas de projeto e desenvolvimento, tem-se que mais de 50% do tempo e custo empregados no desenvolvimento de um software estão concentrados em atividades de teste e manutenção. Portanto, uma atenção mais efetiva no início do projeto, principalmente na especificação correta de seus Requisitos, Objetivos de Qualidade e Critérios de Aprovação, pode significar a redução de tempo de desenvolvimento, menores custos e por conseqüência, maior lucratividade e competitividade.

O software é hoje o diferencial que determina a excelência, produtividade, desempenho, lucratividade e segurança. Sua versatilidade cria novas formas de convivência das pessoas, das organizações e da sociedade. Um número de aplicações crescente em atividade que dependem de softwares confiáveis e seguros mostra a importância da qualidade como responsabilidade social esperada dos profissionais da área de informática. Uma transferência eletrônica de valores, o diagnóstico de um exame médico, o controle do tráfego aéreo são exemplos de atividades que não admitem falhas.

É importante observar que, em função dessa responsabilidade social crescente atribuída ao software, é necessária a conscientização dos profissionais da área e da comunidade acadêmica para a busca contínua da melhoria da qualidade do software, mesmo daquele voltado a atividades não-críticas. Além disso, a recente abertura do mercado de informática no Brasil deu ao usuário médio acesso ao software importado, estabelecendo níveis de exigência mais altos na qualidade do software brasileiro.

Esta dissertação tem como objetivo elaborar com base nas normas e modelos existentes atualmente, um modelo de verificação da qualidade e normalização de softwares, para auxiliar as pessoas interessadas em adquirir um sistema que satisfaça realmente todas as suas necessidades. Desta maneira, almeja-se consciêntizar os desenvolvedores de softwares, a produzirem produtos com mais qualidade, pois só assim poderão conquistar o respeito do mercado atual.

2) Qualidade

Você sabe o que é qualidade? Existem diversas definições. Algumas pessoas, que tentaram uma definição simples chegaram a frases como:

"Qualidade é estar em conformidade com os requisitos dos clientes".
"Qualidade é antecipar e satisfazer os desejos dos clientes".
"Qualidade é escrever tudo o que se deve fazer e fazer tudo o que foi escrito".

Segunda a atual norma brasileira sobre o assunto (NBR ISO 8402), qualidade é:

"A totalidade das características de uma entidade que lhe confere a capacidade de satisfazer às necessidades explícitas e implícitas".

A entidade é o produto, que pode ser um bem ou um serviço. As necessidades explícitas são as próprias condições e objetivos propostos pelo produtor. As necessidades implícitas incluem as diferenças entre os usuários, a evolução no tempo, as implicações éticas, as questões de segurança e outras visões subjetivas.

Um aspecto interessante da qualidade, é que não basta que ela exista. Ela deve ser **reconhecida** pelo cliente. Por causa disso, é necessário que exista algum tipo de **certificação** oficial, emitida com base em um **padrão**.

Para que seja possível realizar uma **avaliação** e um **julgamento**, é necessário haver um padrão ou norma. Existem alguns organismos normalizadores reconhecidos mundialmente como:

ISO - International Organization for Standardization

- **IEEE** - Instituto de Engenharia Elétrica e Eletrônica
- **ABNT** - Associação Brasileira de Normas Técnicas

A Certificação em uma norma ou padrão é a emissão de um documento oficial indicando a conformidade com esta determinada norma ou padrão, mas antes da emissão do certificado, é preciso realizar todo um processo de avaliação e julgamento de acordo com uma determinada norma. Embora uma empresa possa auto-avaliar-se ou ser avaliada por seus próprios clientes, o termo *Certificação* costuma ser aplicado apenas quando efetuado por uma empresa independente e idônea, normalmente especializada neste tipo de trabalho. No Brasil, o INMETRO é o órgão do governo responsável pelo credenciamento destas instituições que realizam a certificação de sistemas de qualidade.

3) Qualidade de softwares

Atualmente, muitas instituições se preocupam em criar normas para permitir a correta avaliação de qualidade tanto de produtos de software quanto de processos de desenvolvimento de software. No tabela abaixo estão relacionadas as principais normais nacionais e internacionais nesta área:

Norma	Comentário
ISO 9126	Características da qualidade de produtos de software.
NBR 13596	Versão brasileira da ISO 9126.
ISO 14598	Guias para a avaliação de produtos de software, baseados na utilização prática da norma ISO 9126.
ISO 12119	Características de qualidade de pacotes de software (software de prateleira, vendido com um produto embalado).
IEEE P1061	Standard for Software Quality Metrics Methodology (produto de software)
ISO 12207	Software Life Cycle Process. Norma para a qualidade do processo de desenvolvimento de software.
NBR ISO 9001	Sistemas de qualidade - Modelo para garantia de qualidade em Projeto, Desenvolvimento, Instalação e Assistência Técnica (processo).
NBR ISO 9000-3	Gestão de qualidade e garantia de qualidade. Aplicação da norma ISO 9000 para o processo de desenvolvimento de software.
NBR ISO 10011	Auditoria de Sistemas de Qualidade (processo).
CMM	Capability Maturity Model. Modelo da SEI (Instituto de Engenharia de Software do Departamento de Defesa dos EEUU) para avaliação da qualidade do processo de desenvolvimento de software. Não é uma norma ISO, mas é muito bem aceita no mercado.
SPICE - ISO 15504	Projeto da ISO/IEC para avaliação de processo de desenvolvimento de software. Ainda não é uma norma oficial ISO, mas o processo está em andamento.

Tabela 1: Principais Normas Nacionais e Internacionais

3.1) Qualidade do Processo de Software

O estudo da Qualidade do Processo de Software é uma área ligada diretamente à Engenharia de Software. O estudo de um ajuda a entender e aprimorar o outro. Em ambas as disciplinas, estuda-se modelos do processo de desenvolvimento de software. Estes modelos são uma tentativa de explicar em detalhes **como** se desenvolve um software, quais são as etapas envolvidas. É necessário compreender cada pequena tarefa envolvida no desenvolvimento. Entre os estudos nesta área de maior importância, podemos citar:

- **ISO 9000-3** - Normas para aplicação da série ISO 9000 em processos de software;
- **ISO 12207** - Processos do Ciclo de Vida do Software;
- **CMM** - Capability Maturity Model;
- **PSP** - Personal Software Process;
- **ISO 15504** - ***SPICE*** - Software Process Improvement and Capability dEtermination;
- Modelo **Trillium**;

Dentre os trabalhos na área de Qualidade de Processo de Software, o único que realmente é norma oficial é o ISO 9000-3, que faz parte da série ISO 9000. Os demais modelos são normas não-oficiais criados por empresas e institutos ou então são normas em estágio de desenvolvimento.

3.1.1) A Série ISO 9000

Esta série é um conjunto de normas da ISO que define padrões para garantia e gerenciamento da qualidade. Veja algumas destas normas abaixo:

Norma	Trata de
ISSO 9001	Modelo para garantia da qualidade em projeto, desenvolvimento, produção, instalação e assistência técnica.
ISO 9002	Modelo para garantia da qualidade em produção e instalação
ISO 9003	Modelo para garantia da qualidade em inspeção e ensaios finais
ISO 9000-1	Diretrizes para escolher entre as normas ISO 9001, 9002 e 9003
ISO 9000-3	Orientação para a aplicação da ISO 9001 em Software

Tabela 2: Conjunto de Normas ISO 9000

As normas ISO série 9000 (ISO-9000) para a gestão e a garantia da qualidade representam o consenso de vários países para a normalização da qualidade. Elas orientam na busca da melhoria dos níveis da qualidade para produtos, serviços e relacionamento cliente/fornecedor, através de diretrizes para a implantação de sistemas da qualidade nas empresas.

A série 9000 especifica requisitos mínimos para que, através de ações gerenciais centradas na prevenção de problemas, as empresas possam assegurar a qualidade de seus produtos e serviços. Trata-se, em última instância, do emprego de normas básicas na busca de níveis de qualidade aceitáveis, tanto nos negócios locais como nos negócios internacionais.

A sigla ISO origina-se de *International Organization For Standardization.* Esse é o nome de um grupo internacional de normalização localizado em Genebra, Suíça. Essa organização foi fundada em 23 de Fevereiro de 1947 e não possui ligações com

órgãos governamentais.

Entidades de normalização de mais de noventa países fazem parte desse grupo, entre os quais o Brasil, através da ABNT - Associação Brasileira de Normas Técnicas.

A série 9000 foi elaborada em 1979 pelo ISO Technical Commitee 176 (ISO-TC-176), o primeiro dos comitês do grupo ISO a tratar dos assuntos da gestão e garantia da qualidade. A primeira versão, porém, somente foi publicada oficialmente em 1987.

A série é composta, basicamente, de cinco documentos numerados seqüencialmente: ISO-9000, ISO-9001, ISO-9002, ISO-9003, ISO-9004. Segue a titulação desses documentos e o correspondente documento oficializado pela ABNT (NBR-190XX):

Referência ISO/(ABNT)	Título do Documento
ISO-9000 (NBR-19000)	Normas de gestão da Qualidade e garantia da qualidade. Diretrizes para seleção (da Norma mais adequada ao caso da empresa) e uso.
ISO-9001 (NBR-19001)	Sistemas da Qualidade – Modelo para garantia da qualidade em Projeto Desenvolvimento, Produção, Instalação e Assistência Técnica.
ISO-9002 (NBR-19002)	Sistemas da Qualidade – Modelo para garantia da qualidade em Produção e Instalação.
ISO-9003 (NBR-19003)	Sistemas da Qualidade – Modelo para garantia da qualidade em Inspeção e Ensaios Finais.
ISO-9004 (NBR-19004)	Gestão da Qualidade e Elementos do sistema da Qualidade – Diretrizes.

Tabela 3: Título das Normas ISO 9000

3.1.1.1) Sub Partes da ISO 9000

ISO 9000	
Partes	**Título do Documento**
ISO 9000/1	Seleção e Uso.
ISO 9000/2	Diretrizes genéricas para a aplicação da ISO 9001, ISO 9002 e ISO 9003.
ISO 9000/3	Diretrizes para a aplicação da ISO-9001 em desenvolvimento, fornecimento e manutenção de software.
ISO 9000/4	Aplicações da gestão de dependência.

Tabela 4: Sub Partes da ISO 9000

A seguir são as normas da ISO 9001 à ISO 9003, nas quais se especificam os requisitos do Sistema de Qualidade. A mesma deve ser usada quando existe um contrato entre duas partes, o qual requeira a demonstração da capacidade de um fornecedor em desenvolver e fornecer um produto, software em nosso caso, com uma qualidade garantida.

Outro componente importante do padrão, é o conjunto ISO-9004, o qual explica os requisitos para assegurar a qualidade, dando diretrizes acerca dos fatores técnicos, administrativos e de pessoal que afetem a qualidade dos produtos e/ou serviços, para todos os níveis do ciclo de qualidade, desde a detecção da necessidade ate a satisfação do cliente. Possui quatro partes:

ISO 9004	
Partes	**Título do Documento**
ISO 9004/1	Diretrizes Gerais.
ISO 9004/2	Diretrizes de Serviços.
ISO 9004/3	Diretrizes de Materiais Processados.
ISO 9004/4	Diretrizes de Gestão da Qualidade.

Tabela 5: A Norma ISO 9004

Como parte não integrante da série ISO 9000, porém sim, com um alto grado de relacionamento temos:

ISO 8402	Terminologia para a Gestão e Garantia da Qualidade.
ISO 2382	Vocabulário de Processamento de Dados.
ISO 10013	Desenvolvimento de Manuais de Qualidade.
ISO IEC-9126	Avaliação dos Produtos de Software. Características de Qualidade e Diretrizes para seu uso.

Tabela 6: Normas Integrantes da Série ISO 9000

3.1.1.2) O Padrão ISO 9000/3

Como se tem observado através dos 20 elementos padrão, acima definidos e explicados, muitos deles podem aplicar-se diretamente ao processo de desenvolvimento de software, mais infelizmente para alguns deles sua aplicação é um pouco forçada. Em reconhecimento à organização ISO, preparou a ISO 9000/3 Diretrizes para a aplicação da ISO 9001 para o desenvolvimento, fornecimento e manutenção de software. Não obstante, como seu mesmo nome o diz, são Diretrizes e devem ser reconhecidas como tais. Se alguma organização decide fazer as coisas de uma maneira diferente das sugestões providas da ISO 9000/3, ela deve ao menos estar segura, como a sua alternativa escolhida se dirige e satisfaz os requisitos dos padrões.

A ISO 9000/3 está estruturada e organizada, baseada sob a premissa que associado com cada um dos projetos de desenvolvimento de software um ciclo de vida, consistente de um conjunto de fases, ou segmentos definidos de trabalho. Esta não assume algum particular ciclo de vida, porem assume a existência de algum ciclo de vida com fases.

A ISO 9000/3 também supõe que o produto de software produzido é o resultado de algum acordo contratual entre um comprador e um fornecedor. A ISO 9000/3 consiste de 22 clausulas que não correspondem diretamente com as 20 clausulas de ISO 9001. As 22 clausulas de ISO 9000/3 estão agrupadas em 3 maiores seções.

Na tabela abaixo esta relacionada as diretrizes da ISO 9000/3

Grupo	Atividade
Estrutura do Sistema de Qualidade	Responsabilidade do fornecedor Responsabilidade do comprador Análise crítica conjunta.
Atividades do Ciclo de Vida	Análise crítica do contrato Especificação dos requisitos do comprador Planejamento do desenvolvimento Projeto e implementação Testes e validação Aceitação Cópia, entrega e instalação Manutenção.
Atividades de Apoio	Gerenciamento de configuração Controle de documentos Registros da qualidade Medição Regras, convenções Aquisição Produto de software incluído Treinamento.

Tabela 7: Diretrizes da ISO 9000/3

Além de diretrizes gerais, a ISO-9000/3 descreve as responsabilidades e ações relacionadas à qualidade que devem ser tomadas tanto pelo fornecedor (empresas de softwares) como pelo cliente, ao longo das diferentes fases do ciclo de vida do software.

Diferente da ISO-9001 as diretrizes da ISO-9000/3 prevêem atributos também para o cliente. É proposto que o cliente indique um representante para negociar com o fornecedor de software as questões contratuais, incluindo definições de requisitos, definições de critérios de aceitação e acordos de conclusão. Reuniões de revisão e acompanhamento entre fornecedor e cliente também são propostas nessa primeira parte da ISO-9000/3, cujas diretrizes se repetem para cada uma das fases do ciclo de desenvolvimento.

Em resumo, são quatro os pontos cobertos pelo capítulo da Norma que trata da Estrutura do Sistema da Qualidade:

- Responsabilidades gerenciais;
- Definição e documentação do sistema de qualidade;
- Procedimentos para auditoria interna do sistema de qualidade;
- Procedimentos para ações corretivas.

Independentes do modelo de ciclo de vida definido pela empresa, a Norma prevê que as atividades do ciclo podem ser agrupadas em nove grandes categorias:

a) *Analise Crítica de Contrato:* Cobre itens que devem constar nos contratos relativos a compra e venda de software. Quando se revisa um contrato deve-se tomar em consideração a identificação e cotas das possíveis contingências e perigos, a adequada proteção da propriedade da informação, a definição da responsabilidade do fornecedor com relação ao trabalho subcontratado, terminologia sobre os acordos para ambas partes e a capacidade do comprador para cumprir as obrigações contratuais;

b) *Especificação de Requisitos do cliente:* O fornecedor devera ter um completo, não ambíguo conjunto de requisitos funcionais que incluem todos os aspetos necessários para satisfazer as necessidades do comprador, serão testáveis, desenvolvidos em próxima cooperação com o comprador, estando sujeito ao controle de documentos ou a gerência de configuração, especificando todas as interfaces do produto. A especificação de requisitos deverá ser completa e aprovada antes de iniciar o desenvolvimento de atividades;

c) *Planejamento do Desenvolvimento:* Define o plano de desenvolvimento do software. O plano deve incluir: definição do projeto, organização do projeto, fases de desenvolvimento, gerência de projetos, métodos e ferramentas, cronogramas e planos de teste. Também deve incluir formas de controle de entradas e saídas para cada fase do ciclo de vida e um método de monitorar e verificar o progresso;

d) *Planejamento da Qualidade:* O fornecedor deverá preparar um plano de qualidade, o mesmo deve cobrir : objetivos de qualidade do produto de software, critérios de saída de cada fase, planejamento detalhado das atividades de verificação e validação, assim como responsabilidades especificas para atividades da qualidade;

e) *Projeto e Implementação*: Trata sobre as atividades de desenho e implementação que deverão de ser executadas de uma maneira disciplinada. Deve-se tomar em conta considerações para a identificação do desenho, metodologia de desenho, uso da experiência passada em desenho, processos subseqüentes (teste, manutenção e uso), regras de programação, metodologias de implementação, revisões, etc;

f) *Testes e Validação:* O fornecedor de software deve realizar aqueles testes que podem ser necessários em diferentes níveis ou partes do software durante o processo. Os testes devem levar em consideração fatores como ambiente, documentação, casos e dados de teste. A validação do sistema completo e os testes de campo devem também ser abordados pelo plano de testes;

g) *Aceitação:* Cobre os termos acordados previamente no contrato pelo cliente para a aceitação do software. Deve-se tomar em conta cronograma de tempo, procedimento para a avaliação, meio ambiente de software e hardware, critério de aceitação, entre outros;

h) *Reprodução,Expedição, Entrega e Instalação:* Trata do número de copias a ser entregue, tipo de meio físico utilizado, direitos autorais e licenças, custodia do maestro e copias de respaldo, incluído plano de recuperação de desastre, instalação, entre outros;

i) *Manutenção:* Os pedidos de manutenção pelo comprador deverá ser estipulado no contrato. Quando a manutenção é solicitada, o fornecedor devera estabelecer e documentar os procedimentos para a execução das atividades de manutenção. Os

itens a ser mantidos e seu período de tempo correspondente devera ser especificado no contrato;

As Atividades de Suporte do Sistema de Qualidade, é a terceira das três principais partes da ISO-9000/3. As atividades de suporte tratadas nesse grupo, não se encontram atreladas a uma determinada fase do ciclo de vida do software. Elas permeiam todas as fases.

As atividades de suporte compreendem nove itens, os quais devem ser desenvolvidos e implementados pelo fornecedor do software:

a) *Sistema de Gerência de Configuração:* Trata do controle e rastreabilidade do software e seus componentes de maneira que seja possível identificar cada versão destes, controlar a atualização simultânea, identificar, documentar, revisar e autorizar câmbios aos itens de software. Deve-se estabelecer procedimentos para registrar, gerênciar e reportar o estado dos itens do software;

b) *Controle de Documentos:* São procedimentos que determinam quais documentos devem ser controlados, as instruções para sua alteração e até critérios de segurança destes. Entre eles temos :

- Documentos que descrevem a qualidade do sistema;
- Documentos que descrevem os planos do projeto e seu progresso;
- Documentos que descrevem um software produto particular, incluindo as entradas/saídas na fase de desenvolvimento, os planos e resultados da verificação e validação, documentos para o comprador e usuário, e documentação de manutenção.

c) *Registros de Qualidade:* Indica a necessidade de procedimentos para identificar, coletar, classificar, armazenar, manter e recuperar registros de qualidade.

d) *Medição:* Procedimentos e técnicas para medição nos produtos, desde o desenvolvimento, até a expedição. A métrica deve ser comparável e deve-se considerar para seu uso que cumpra o seguinte :

- Os valores nos relatórios devem ser mostrados sobre uma base regular;
- Identificar nível de performance;
- Tomar ações corretivas;
- Estabelecer melhoras específicas as metas.

e) *Regras Práticas e Convenções:* O fornecedor deve de estabelecer e usar regras, práticas, e convenções para fazer o sistema de qualidade efetivo. Isto deve estar em concordância com o sistema de qualidade preconizado pelo ISO 9000/3;

f) *Ferramentas e Técnicas:* O fornecedor deve de identificar e usar as ferramentas e técnicas para fazer que o sistema de qualidade seja efetivo;

g) *Aquisição:* Deve de existir procedimentos para que a aquisição de todos os produtos e serviços necessários para compor o produto final estejam de acordo com os requisitos especificados, de maneira de assegurar sua qualidade; deve-se de validar o trabalho das sub-contratadas;

h) *Produto Incluído no Software:* São procedimentos e cuidados relativos ao uso de software do próprio comprador ou de terceiros, que serão parte integrante do software a desenvolver. No caso que fossem componentes de terceiros se deve de considerar como será a manutenção do mesmo;

i) *Treinamento:* Procedimentos para identificar necessidades de treinamento interno, visando a qualificação do fornecedor e do cliente. O mesmo pode incluir treinamento em ferramentas especificas, técnicas, metodologias e recursos de cômputo usados durante o desenvolvimento;

3.1.2) ISO 12207 - Processos do Ciclo de Vida do Software

Este padrão formaliza a arquitetura do ciclo de vida do software, que é um assunto básico em Engenharia de Software e também em qualquer estudo sobre Qualidade do Processo de Software. Esta norma possui mais de 60 páginas e detalha os diversos processos envolvidos no ciclo de vida do software. Estes processos estão divididos em três classes: Processos Fundamentais, Processos de Apoio e Processos Organizacionais. Veja a lista completa dos processos na tabela abaixo:

Processos Fundamentais	Início e execução do desenvolvimento, operação ou manutenção do software durante o seu ciclo de vida.
Aquisição	Atividades de quem um software. Inclui: definição da necessidade de adquirir um software (produto ou serviço), pedido de proposta, seleção de fornecedor, gerência da aquisição e aceitação do software.
Fornecimento	Atividades do fornecedor de software. Inclui preparar uma proposta, assinatura de contrato, determinação recursos necessários, planos de projeto e entrega do software.
Desenvolvimento	Atividades do desenvolvedor de software. Inclui: análise de requisitos, projeto, codificação, integração, testes, instalação e aceitação do software.
Operação	Atividades do operador do software. Inclui: operação do software e suporte operacional aos usuários.
Manutenção	Atividades de quem faz a manutenção do software.
Processos de Apoio	Auxiliam um outro processo.
Documentação	Registro de informações produzidas por um processo ou atividade. Inclui planejamento, projeto, desenvolvimento, produção, edição, distribuição e manutenção dos documentos necessários a gerentes, engenheiros e usuários do software.
Gerência de Configuração	Identificação e controle dos itens do software. Inclui: controle de armazenamento, liberações, manipulação, distribuição e modificação de cada um dos itens que compõem o software.
Garantia da Qualidade	Garante que os processos e produtos de software estejam em conformidade com os requisitos e os planos estabelecidos.
Verificação	Determina se os produtos de software de uma atividade atendem completamente aos requisitos ou condições impostas a eles.

Validação	Determina se os requisitos e o produto final (sistema ou software) atendem ao uso específico proposto.
Revisão Conjunta	Define as atividades para avaliar a situação e produtos de uma atividade de um projeto, se apropriado.
Auditoria	Determina adequação aos requisitos, planos e contrato, quando apropriado.
Resolução de Problemas	Análisar e resolução dos problemas de qualquer natureza ou fonte, descobertos durante a execução do desenvolvimento, operação, manutenção ou outros processos.
Processos Organizacionais	Implementam uma estrutura constituída de processos de ciclo de vida e pessoal associados, melhorando continuamente a estrutura e os processos.
Gerência	Gerenciamento de processos.
Infra-estrutura	Fornecimento de recursos para outros processos. Inclui: hardware, software, ferramentas, técnicas, padrões de desenvolvimento, operação ou manutenção.
Melhoria	Atividades para estabelecer, avaliar, medir, controlar e melhorar um processo de ciclo de vida de software.
Treinamento	Atividades para prover e manter pessoal treinado.

Tabela 8: Processos do Ciclo de Vida do Software

A norma detalha cada um dos processos acima. Ela define ainda como eles podem ser usados de diferentes maneiras por diferentes organizações (ou parte destas), representando diversos pontos de vista para esta utilização. Cada uma destas visões representa a forma como uma organização emprega estes processos, agrupando-os de acordo com suas necessidades e objetivos.

As Visões têm o objetivo de organizar melhor a estrutura de uma empresa, para definir suas gerências e atividades alocadas às suas equipes. Existem cinco visões diferentes: contrato, gerenciamento, operação, engenharia e apoio. Veja na figura abaixo como estas visões se relacionam aos processos.

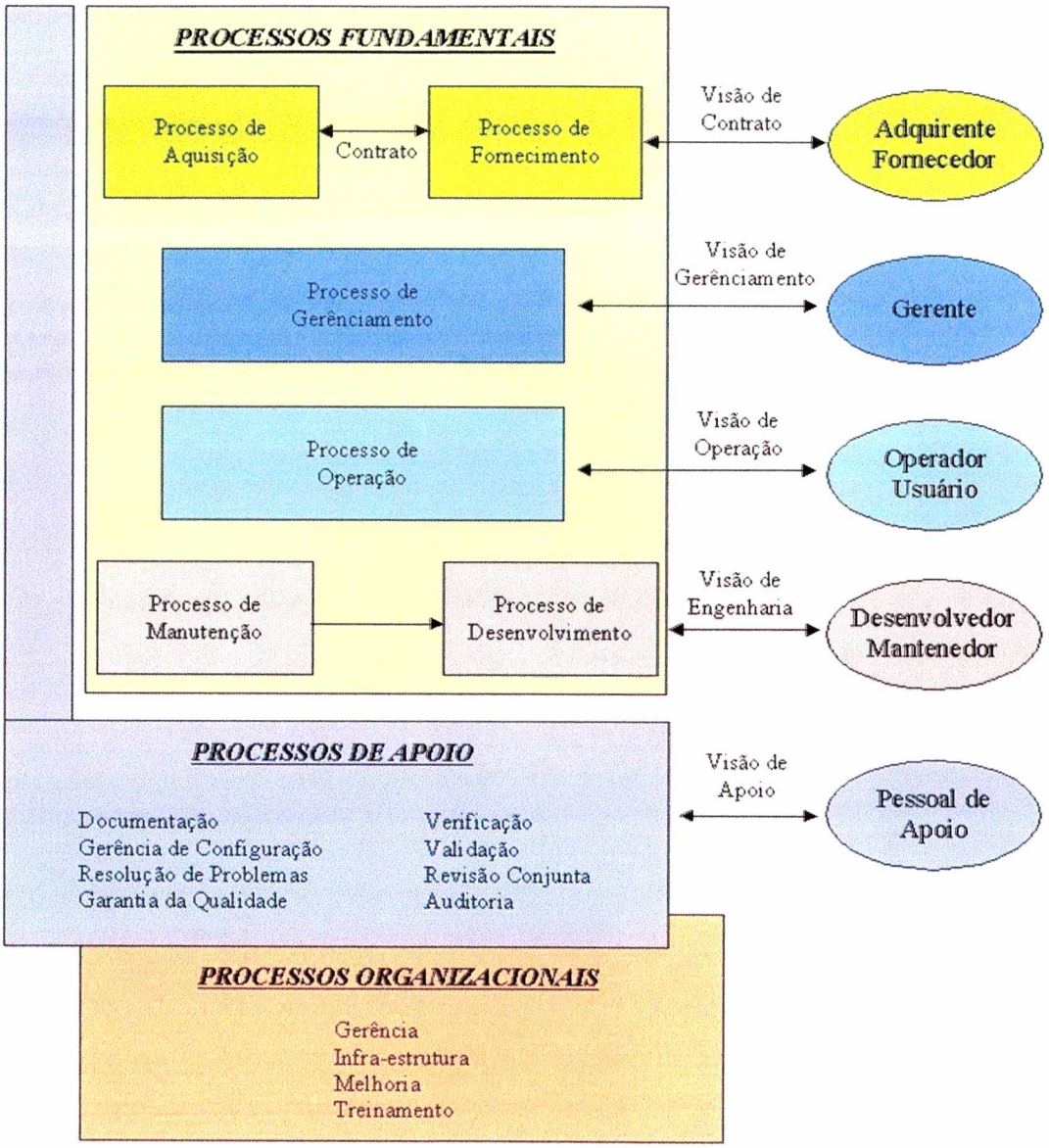


Figura 1: Visões da ISO 12207

A ISO/IEC 12207 é a primeira norma internacional que descreve em detalhes os processos, atividades e tarefas que envolvem o fornecimento, desenvolvimento, operação e manutenção de produtos de software. A principal finalidade desta norma é servir de referência para os demais padrões que venham a surgir. Lançada em agosto de 1995, ela é citada em quase todos os trabalhos relacionados à Engenharia de Software desde então, inclusive aqueles relativos à qualidade.

3.1.3) CMM - Capability Maturity Model

O CMM"Modelo de Maturidade da Capacidade" foi criado por Watts Humphrey da SEI (Instituto de Engenharia de Software) em 1988, para avaliar e melhorar a capacitação de empresas que produzem software. O projeto CMM foi apoiado pelo Departamento de Defesa do Governo dos Estados Unidos, que é um grande consumidor de software e precisava de um modelo formal que permitisse selecionar os seus fornecedores de software de forma adequada. Embora não seja uma norma emitida por uma instituição internacional (como a ISO ou o IEEE), esta norma tem tido uma grande aceitação mundial, até mesmo fora do mercado americano. O CMM também é chamado de SW-CMM (Software CMM).

Basicamente, o CMM pode ser utilizado para:

- Melhoramento do processo de desenvolvimento de software, no qual uma organização planeja, desenvolve e implementa mudanças ao seu processo de criação;
- Avaliação do processo de desenvolvimento de software, onde uma equipe treinada de profissionais determina o estado atual da organização no processo, determina as prioridades das dificuldades relacionadas ao desenvolvimento que a empresa enfrenta;
- Avaliação da capacidade dos desenvolvedores, no qual uma equipe de profissionais treinados identifica quais as pessoas qualificadas a trabalhar no desenvolvimento do produto ou monitorar os níveis do processo no qual a empresa se encontra.

O CMM (Modelo de Maturidade da Capacidade), compõe-se de cinco níveis de maturidade. Com a exceção do Nível 1, cada nível de maturidade compõe-se de algumas

áreas-chave de processo. Cada área-chave de processo organiza-se em cinco seções denominadas características comuns. As características comuns especificam as práticas-base que, quando realizadas simultaneamente, alcançam os objetivos da área-chave de processo. Ilustra-se a estrutura do CMM na Figura abaixo.

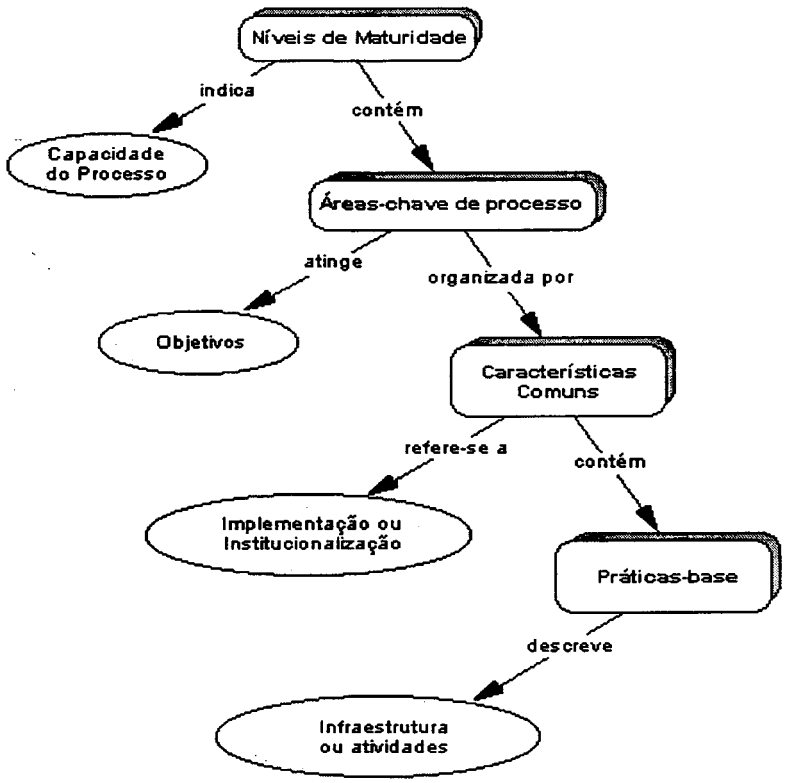


Figura 2: A Estrutura do CMM

Os componentes do CMM incluem:

a) *Níveis de maturidade*

Um nível de maturidade descreve um processo evolutivo bem-definido para alcançar um processo de software maduro. Os cinco níveis de maturidade proporcionam a estrutura de mais alto nível do CMM.

b) *Capacidade do processo*

A capacidade do processo de software descreve o intervalo de resultados esperados que podem ser alcançados seguindo um processo de software. A capacidade de processo de software de uma organização provê estimativas de resultados para o próximo projeto de software da organização empreendedora.

c) *Áreas-chave de processo*

Cada nível de maturidade está composto de áreas-chave de processo. Cada área-chave de processo identifica um grupo de atividades relacionadas que, quando executadas simultaneamente, alcançam objetivos considerados importantes para estabelecer capacidade de processo àquele nível de maturidade. As áreas-chave de processo foram definidas para pertencerem a um único nível de maturidade. Planejamento de Projeto de Software exemplifica uma das áreas-chave de processo para o Nível 2.

d) *Objetivos*

Os objetivos resumem as práticas-base de uma área-chave de processo e podem ser usados para determinar se uma organização ou projeto implementou a área-chave de processo efetivamente. Os objetivos significam o âmbito, os limites, e a intenção de cada área-chave de processo.

e) *Características comuns*

As práticas-base são divididas em cinco seções: Compromisso de realizar,

Capacidade de realizar, Atividades realizadas, Medições e Análises e Implementação com Verificação. As características comuns são atributos que indicam se a implementação de uma área-chave de processo é efetivo, repetível e duradouro.

A característica comum, "Atividades Realizadas", descreve atividades de implementação. As outras quatro características comuns descrevem os fatores de institucionalização, que faz um processo parte da cultura organizacional.

e) *Práticas-base*

Cada área-chave de processo estrutura-se em termos de práticas-base que, quando implementadas, ajudam satisfazer os objetivos daquela área-chave. As práticas-base descrevem a infra-estrutura e as atividades que mais contribuem para implementação efetiva e institucionalização da área-chave de processo.

3.1.3.1) Definição dos Níveis de Maturidade do CMM

O CMM classifica as organizações em cinco níveis distintos, cada um com suas características próprias. No nível 1, o das organizações mais imaturas, não há nenhuma metodologia implementada e tudo ocorre de forma desorganizada. No nível 5, o das organizações mais maduras, cada detalhe do processo de desenvolvimento está definido, quantificado e acompanhado e a organização consegue até absorver mudanças no processo sem prejudicar o desenvolvimento. Veja a tabela abaixo:

Nível CMM	Descrição
1) Inicial	O processo de desenvolvimento é desorganizado e até caótico. Poucos processos são definidos e o sucesso depende de esforços individuais e heróicos.
2) Repetível	Os processos básicos de gerenciamento de projetos estão estabelecidos e permitem acompanhar custo, cronograma e funcionalidade. É possível repetir o sucesso de um processo utilizado anteriormente em outros projetos similares.
3) Definido	Tanto as atividades de gerenciamento quanto de engenharia do processo de desenvolvimento de software estão documentadas, padronizadas e integradas em um padrão de desenvolvimento da organização. Todos os projetos utilizam uma versão aprovada e adaptada do processo padrão de desenvolvimento de software da organização.
4) Gerenciado	São coletadas medidas detalhadas da qualidade do produto e processo de desenvolvimento de software. Tanto o produto quanto o processo de desenvolvimento de software são entendidos e controlados quantitativamente.
5) Otimizado	O melhoramento contínuo do processo é conseguido através de um "feedback" quantitativo dos processos e pelo uso pioneiro de ideais e tecnologias inovadoras.

Tabela 9: Os Níveis do CMM

a) Nível 1 - O Nível Inicial

No nível inicial, a organização normalmente não provê um ambiente estável para desenvolver e manter o software. Quando numa organização há carência de práticas administrativas, os benefícios das práticas de uma boa engenharia de software são minados pelos planejamentos ineficazes e compromissos não organizados.

Durante uma crise, os projetos normalmente abandonam procedimentos planejados e passam para a codificação e testes. O sucesso depende de um gerente excepcional e de um time de software efetivo. Ocasionalmente, gerentes de software capazes e fortes podem resistir às pressões de tomar atalhos no processo de software; mas quando eles deixam o projeto, a estabilidade parte com eles. Nem sequer um forte processo de engenharia pode superar a instabilidade criada pela ausência de práticas administrativas.

A capacidade de processo de software de organizações no Nível 1 são impraticáveis porque o processo de software modifica-se constantemente de acordo com o progresso do trabalho. Tempo, orçamentos, funcionalidade e qualidade do produto são geralmente impossíveis de prever. O desempenho depende da capacidade individual e varia com as habilidades natas, conhecimentos e motivações.

b) Nível 2 - O Nível Repetível

No Nível Repetível, políticas para administrar um projeto de software e procedimentos para implementar essas políticas são estabelecidas. Planejamento e administração de novos projetos são baseados em experiências com projetos semelhantes. Um objetivo em alcançar o Nível 2 institucionaliza-se na administração efetiva para projetos de software, que permite repetir práticas bem sucedidas desenvolvidas em projetos anteriores, embora os processos específicos implementados pelos projetos possam ser diferenciados. Um processo efetivo pode ser caracterizado como prático, documentado, treinado, medido, e capaz de ser melhorado.

Organizações no Nível 2 instalaram controles básicos de gerenciamento de software. Compromissos de projeto realistas estão baseados nos resultados observados em projetos prévios e nas exigências do projeto atual. Os gerentes de software traçam, para o projeto, custos, tempo e funcionalidade; problemas são identificados quando surgem. Exigências de softwares e produtos de trabalho desenvolvidos para satisfaze-los, são fundamentadas e controla-se a integridade. São definidos padrões de projeto de software e as organizações garantem seguirem-nos fielmente. A capacidade de processo de software de organizações no Nível 2 podem ser resumidas como disciplinadas porque o planejamento e trajetória do projeto de software são estáveis e sucessos anteriores podem ser repetidos.

c) Nível 3 - O Nível Definido

No Nível Definido, a engenharia e a manutenção do software formam um processo padrão documentado, padronizado e integrado em um todo coerente. Referencia-se o processo padrão ao longo do CMM, como o processo padrão de software da organização (organization's standard software process). Processos estabelecidos no Nível 3 são usados ou modificados para ajudar os gerentes de software e o pessoal técnico a trabalhar melhor. A organização melhora suas práticas de engenharia de software quando os processos são padronizados. Um grupo responsabiliza-se pelas atividades de processo de software da organização, por exemplo, um grupo de processos relacionados à engenharia de software. Implementa-se um largo programa de treinamento para assegurar que o pessoal técnico e gerentes tenham o conhecimento e as habilidades exigidas para cumprir as suas funções.

Projetos modificam o processo padrão de software da organização para desenvolver o próprio processo de software definido, o que caracteriza o projeto. Um processo de software definido contém um coerente e integrado conjunto de bem definidos processos de engenharia de software e de gerenciamento. Um processo bem definido pode ser caracterizado com critérios, entradas, padrões e procedimentos para execução do trabalho e verificação de processos de gerenciamento. Devido ao processo de software ser bem definido, o gerenciamento tem uma boa perspicácia em progresso técnico em todos os projetos.

A capacidade de processo de software do Nível 3 pode ser resumida como padrão e consistente porque tanto o gerenciamento das atividades quanto à engenharia de software são estáveis e repetíveis. Como o custo, tempo e funcionalidade dos produtos estão sob controle, a qualidade de software pode ser mantida. Esta capacidade de processo está baseada em um conhecimento pleno das atividades da organização.

d) Nível 4 - O Nível Gerenciado

No Nível Gerenciado, a organização estabelece metas de qualidade quantitativas para produtos de software e processos. Produtividade e qualidade são medidas nas atividades importantes do processo de software em todos os projetos como parte de um programa de medidas. Utiliza-se um banco de dados para coletar e analisar os dados disponíveis dos processos de software definidos. Processos de software são instrumentados com medidas bem definidas e consistentes. Estas medidas estabelecem uma função quantitativa para avaliar os processos de software dos projetos e produtos.

Projetos alcançam o controle sob os produtos e processos estreitando a variação no desempenho do processo, para caírem dentro de limites quantitativos aceitáveis. Podem ser distinguidas variações significantes em desempenho de processo de variação, particularmente dentro de linhas de produto estabelecidas. Os riscos envolvidos promovendo a curva de aprendizagem de um domínio de aplicação novo são conhecidos e cuidadosamente administrados.

A capacidade de processo de software de organizações no Nível 4 podem ser resumidas como previsível porque mede-se o processo e opera-se dentro de limites mensuráveis. Este nível de capacidade de processo permite uma organização predizer tendências de processo e qualidade de produto dentro dos saltos quantitativos destes limites. Quando excede esses limites, ações de correção são executadas. Produtos de software de alta qualidade e previsíveis.

e) Nível 5 - O Nível Otimizado

No Nível Otimizado, a organização inteira volta-se para a melhoria do processo contínuo. A organização tem como identificar fraquezas e fortalecer o processo, com a meta de prevenir a ocorrência de defeitos. Dados na efetividade do processo de software são usados para executar análises de benefício de custo de novas tecnologias e propor mudanças para o processo de software da organização. Identificam-se inovações que

exploram as melhores práticas de engenharia software, difundindo-as para o resto da organização.

Times de projeto de software no Nível 5 analisam defeitos para determinar as causas. Avaliam-se os processos de software para prevenir a ocorrência de tipos conhecidos de defeitos, e lições aprendidas são disseminadas para outros projetos.

As organizações no Nível 5 podem ser caracterizadas por estarem em processo de melhoramento contínuo, pois no Nível 5 as organizações esforçam-se para melhorar a capacidade de processo e o desempenho do trabalho dos projetos. Melhorias acontecem por avanços com incremento no processo existente e por inovações que usam novas tecnologias e métodos.

3.1.3.2) As Áreas-chave de Processo do CMM

Exceto no Nível 1, todos os níveis são detalhados em áreas-chave de processo. Cada área-chave identifica um conjunto de atividades relacionadas que, quando realizadas simultaneamente, atingem objetivos importantes. Estas áreas são exatamente aquilo no que a organização deve focar para melhorar o seu processo de desenvolvimento de software. Para que uma empresa possa se qualificar em um determinado nível de maturidade CMM, deve estar realizando os processos relacionados às áreas-chave daquele determinado nível. Todas as áreas-chave estão citadas na tabela a seguir:

Nível CMM	Foco	Áreas-chave de processo
1) Inicial	Pessoas competentes.	
2) Repetível	Processos de gerenciamento de projetos	• Gerenciamento de requisitos; • Planejamento do projeto; • Visão geral e acompanhamento do projeto; • Gerenciamento de sub-contratados; • Garantia da qualidade do software; • Gerenciamento de configuração.
3) Definido	Processos de engenharia e apoio	• Foco do processo organizacional; • Definição do processo organizacional; • Programa de treinamento; • Gerenciamento de software integrado; • Engenharia de produto de software; • Coordenação intergrupos; • Revisão conjunta.
4) Gerenciado	Qualidade do produto e do processo	• Gerenciamento quantitativo dos processos; • Gerenciamento da qualidade de software.
5) Otimizado	Melhoramento contínuo do processo	• Prevenção de defeitos • Gerenciamento de mudanças tecnológicas; • Gerenciamento de mudanças no processo.

Tabela 10: As Áreas-chave de Processo do CMM

3.1.3.3) Objetivos das áreas-chave de processo

O modelo CMM define um conjunto de dois a quatro objetivos para cada área-chave. Estes objetivos definem aquilo que deve ser alcançado no caso dos processos desta área-chave serem realmente realizados. Veja na tabela abaixo a lista destes objetivos.

Nível CMM	Áreas-chave de processo	Objetivos
1) Inicial		
2) Repetível	Gerenciamento de requisitos	Os requisitos do sistema definidos para o software são controlados de forma a estabelecer um perfil mínimo a ser utilizado pela engenharia de software e pela administração Os planos, produtos e atividades do software são sempre consistentes com os requisitos de sistema definidos para o software.
	Planejamento do projeto	Estimativas relativas ao software são documentadas para uso no planejamento e acompanhamento do projeto do software. As atividades de projeto de software e compromissos assumidos são planejados e documentados. Grupos e pessoas afetadas concordam com seus compromissos relacionados ao projeto do software.
	Visão geral e acompanhamento do projeto	Resultados reais são acompanhados de acordo com o planejamento do software Quando os resultados apresentam um significativo desvio do planejamento do software, são tomadas ações corretivas que são acompanhadas até o final do projeto Mudanças nos compromissos assumidos são feitas em comum acordo com os grupos e indivíduos afetados.
	Gerenciamento de subcontratados	O contratante seleciona subcontratos qualificados O contratante e os subcontratatos estão de acordo no que diz respeito aos compromissos assumidos um com o outro. O contratante e os sub-contratados mantêm uma comunicação constante. O contratante acompanha os resultados reais do subcontratado de acordo com os compromissos assumidos.
	Garantia da qualidade do software	As atividades de garantia de qualidade de software são planejadas A conformidade dos produtos de software e atividades com os padrões, procedimentos e requisitos é verificada objetivamente. Os grupos e indivíduos afetados são informados das atividades de garantia de qualidade de software e de seus resultados. Questões relacionadas à não conformidade que não são resolvidas dentro do projeto de software são encaminhadas à gerência geral.

Nível CMM	Áreas-chave de processo	Objetivos
	Gerenciamento de configuração	As atividades de gerenciamento de configuração são planejadas. Os produtos de trabalho de software são identificados, controlados e estão disponíveis. Mudanças nos produtos de trabalho identificados são controladas. Os grupos e pessoas afetadas são informados da situação atual e projetadas dos produtos de trabalho de software.
3) Definido	Foco do processo organizacional	São coordenadas atividades de desenvolvimento e melhoramento do processo de software em toda a organização Os pontos fortes e fracos do processo de desenvolvimento de software utilizado são identificados, de acordo com um padrão de processo. São planejadas atividades de desenvolvimento e melhoramento do processo, a nível de organização.
	Definição do processo organizacional	O processo padrão de desenvolvimento de software da organização é desenvolvido e mantido. A informação relacionada ao uso do processo padrão de desenvolvimento de software é coletada, revisada e disponibilizada.
	Programa de treinamento	As atividades de treinamento são planejadas É fornecido treinamento para o desenvolvimento de habilidades e conhecimentos necessários para realizar o gerenciamento do software e as funções técnicas. As pessoas no grupo de engenharia de software e outros grupos relacionados a software recebem o treinamento necessário para realizar as suas funções.
	Gerenciamento de software integrado	O processo de software definido para o projeto é uma versão adaptada do processo padrão de desenvolvimento de software da organização O projeto é planejado e gerenciado de acordo com o processo de desenvolvimento de software definido para o projeto.
	Engenharia de produto de software	As atividades de engenharia de software são definidas, integradas e consistentemente realizadas para produzir o software. Os produtos de trabalho do software são mantidos consistentes entre si.

Nível CMM	Áreas-chave de processo	Objetivos
	Coordenação intergrupos	Todos os grupos de trabalho afetados concordam com os requisitos dos clientes. Todos os grupos de trabalho afetados concordam com os acordos entre os grupos de engenharia Os grupos de engenharia identificam, acompanham e resolvem todas as questões intergrupos.
	Revisão conjunta	Atividades de revisão conjunta são planejadas Defeitos nos produtos de trabalho são identificados e removidos.
4) Gerenciado	Gerenciamento quantitativo dos processos	As atividades de gerenciamento quantitativo dos processos são planejadas. A performance do processo de desenvolvimento de software definido para o projeto é controlada quantitativamente. A capacidade do processo desenvolvimento de software padrão da organização é conhecida em termos quantitativos.
	Gerenciamento da qualidade de software	As atividades de gerenciamento da qualidade de software do projeto são planejadas. Objetivos mensuráveis da qualidade do produto de software e suas prioridades são definidas. O progresso real em direção à realização dos objetivos de qualidade para os produtos de software é quantificado e gerenciado.
5) Otimizado	Prevenção de defeitos	As atividades de prevenção de defeitos são planejadas As causas comuns de defeitos são procuradas e identificadas. As causas comuns de defeitos são priorizadas e sistematicamente eliminadas.
	Gerenciamento de mudanças tecnológicas	A incorporação de mudanças tecnológicas é planejada. Novas tecnologias são avaliadas para determinar seu efeito na qualidade e na produtividade. Novas tecnologias adequadas são incorporadas na prática normal de toda a organização.
	Gerenciamento de mudanças no processo	O melhoramento contínuo do processo é planejado. Toda a organização participa das atividades de melhoramento do processo de software. O padrão de processo de software da organização e os processos de software de cada projeto definido são melhorados continuamente.

Tabela 11: Objetivos das áreas-chave

3.1.3.4) Características Comuns e Práticas-base

As características comuns são itens a serem observados para verificar a implementação e institucionalização de cada área-chave de processo. Elas podem indicar se a área-chave de processo é eficiente, repetível e duradoura. São cinco as características comuns no modelo CMM e cada uma possui suas práticas-base a serem realizadas.

Característica Comum	Descrição	Práticas-base relacionadas a
Compromisso de realizar	Atitudes a serem tomadas pela organização para garantir que o processo se estabeleça e seja duradouro.	Estabelecimento de políticas e apadrinhamento de um gerente experiente.
Capacidade de realizar	Pré-requisitos que devem existir no projeto ou na organização para implementar o processo de forma competente.	Alocação de recursos, definição da estrutura organizacional e de treinamento.
Atividades realizadas	Papéis e os procedimentos necessários para implementar uma área-chave de processo.	Estabelecimento de planos e procedimentos, realização do trabalho, acompanhamento do trabalho e tomada de ações corretivas, se necessário.
Medições e análise	Necessidade de medir o processo e analisar as medições.	Realização de medições para determinar o estado e a efetividade das atividades realizadas.
Implementação com Verificação	Passos para garantir que as atividades são realizadas de acordo com o processo estabelecido.	Revisão, auditoria e garantia de qualidade.

Tabela 12: Características Comuns e Práticas-base

As práticas-chave descrevem as atividades que contribuem para atingir os objetivos de cada área-chave do processo. Em geral são descritas com frases simples, seguidas de descrições detalhadas (chamadas de sub-práticas) que podem até incluir exemplos. As práticas-base devem descrever "o que" deve ser feito e não "como" os

objetivos devem ser atingidos. O modelo CMM inclui um extenso documento em separado, chamado "Práticas-base para o CMM", que lista todas as práticas-chave e sub-práticas para cada uma das áreas-chave de processo.

3.1.4) PSP - Personal Software Process

O Modelo CMM é muito interessante, mas aplica-se mais a grandes empresas de software. O pessoal do Software Engineering Institute (SEI) acabou percebendo que havia a necessidade de definir um modelo mais simples, voltado para pequenas empresas ou até para um único indivíduo. Foi daí que surgiu o PSP, que significa "Processo Pessoal de Software".

3.1.4.1) Níveis do PSP

Assim como o CMM, no modelo PSP, existem diversos níveis com características próprias. O modelo PSP possui os seguintes níveis:

Nível	Nome	Atividades
PSP0 PSP0.1	Medição Pessoal	Registro de tempo; Registro de defeitos; Padrão de tipos de defeitos; Padrão de codificação; Medida de tamanho; Proposta de melhoramento do processo.
PSP1 PSP1.1	Planejamento Pessoal	Estimativa de tamanho; Relatório de testes; Planejamento de tarefas; Cronogramas.
PSP2 PSP2.1	Qualidade Pessoal	Revisões de código; Revisões de projeto; Padrões de projeto.
PSP3	Processo Cíclico Pessoal	Desenvolvimento cíclico

Tabela 13: Níveis do PSP

No nível de **Medição Pessoal**, você aprende a registrar o tempo gasto em cada etapa do ciclo do desenvolvimento, registrando ainda os defeitos encontrados. Isto é conseguido através do uso de formulários adequados. O nível PSP0.1 inclui o uso de um padrão de codificação, de medidas padronizadas e do formulário de proposta de melhoramento do processo.

No nível de **Planejamento Pessoal**, você aprende a planejar. A idéia geral é obter a capacidade de estimar quanto tempo levará para realizar uma tarefa baseado nas medições feitas em tarefas semelhantes anteriormente. Neste nível aprende-se a assumir compromissos que podem realmente ser cumpridos. O nível PSP1.1 inclui o planejamento de tarefas e a elaboração de cronogramas.

No nível de **Qualidade Pessoal** você aprende a lidar com seus erros. Deve-se ter uma idéia precisa de quantos erros são cometidos (em média) em cada fase do ciclo de desenvolvimento. O modelo PSP mostra que a forma mais adequada para tratar erros é evitá-los desde a sua origem. Você deve utilizar os dados sobre defeitos já coletados para criar uma lista de verificação (checklist) a ser utilizada em suas revisões de projeto e de código. O nível PSP2.1 inclui a criação de padrões de projeto, bem como métodos de análise e prevenção de defeitos.

O nível de **Processo Cíclico Pessoal** é a última etapa do PSP. Neste nível, o PSP sai do desenvolvimento de pequenos programas para tratar do desenvolvimento de projetos maiores, embora ainda em nível pessoal. A idéia é dividir os grandes projetos em pequenos projetos que possam ser tratados no PSP2. Neste caso, o desenvolvimento acontece em passos incrementais.

O treinamento do PSP é realizado através de 10 exercícios de desenvolvimento de programas. Além servirem como exemplos de desenvolvimento, os exercícios propostos pelo treinamento do PSP são pequenos utilitários que ajudam você a aplicar o PSP, pois permitem medir o número de linhas e objetos nos seus programas, calcular desvio padrão, prever intervalos etc.

Uma descrição completa deste modelo e do treinamento proposto pode ser encontrada no livro "Introduction to the Personal Software Process", publicado em 1996 por Watts Humphery, o criador do PSP.

3.1.5) SPICE - Software Process Improvement and Capability dEtermination - ISO 15504

O SPICE é uma norma em elaboração conjunta pela ISO e pelo IEC. Ela constitui-se de um padrão para a avaliação do processo de software, visando determinar a capacitação de uma organização. A norma visa ainda orientar a organização para uma melhoria contínua do processo. Ela cobre todos os aspectos da Qualidade do Processo de Software e está sendo elaborada num esforço conjunto de cinco centros técnicos espalhados pelo mundo (EUA, Canadá/América Latina, Europa, Pacífico Norte e

Pacífico Sul).

Um grupo de estudos da ABNT está participando do processo de desenvolvimento, além de trabalhar na tradução das versões preliminares da norma para o português. Tenho a honra de participar como membro colaborador da comissão SPICE da ABNT.

O SPICE inclui um modelo de referência, que serve de base para o processo de avaliação. Este modelo é um conjunto padronizado de processos fundamentais, que orientam para uma boa engenharia de software. Este modelo é dividido em cinco grandes categorias de processo: Cliente-Fornecedor, Engenharia, Suporte, Gerência e Organização. Cada uma destas categorias é detalhada em processos mais específicos. Tudo isso é descrito em detalhes pela norma.

Além dos processos, o SPICE define também os 6 níveis de capacitação de cada processo, que pode ser incompleto, executado, gerenciado, estabelecido, previsível e otimizado. O resultado de uma avaliação, portanto, um perfil da instituição em forma de matriz, onde temos os processos nas linhas e os níveis nas colunas.

3.1.5.1) Categorias e Processos

Uma das contribuições do modelo SPICE é definir em seu modelo de referência todos os processos envolvidos no desenvolvimento de software, agrupados em categorias. Observe no quadro abaixo a estrutura completa das categorias, dos processos de cada categoria:

Processo	Descrição
CUS – Cliente-Fornecedor - Processos que empatam diretamente os produtos e serviços de software no fornecedor para o cliente.	
CUS.1	Adquirir Software
CUS.2	Gerenciar necessidades do Cliente
CUS.3	Fornecer Software
CUS.4	Operar Software
CUS.5	Prover Serviço ao Cliente
ENG – Engenharia - Processos que especificam, implementam ou mantém um sistema ou produto de software e sua documentação	
ENG.1	Desenvolver requisitos e o projeto do sistema
ENG.2	Desenvolver requisitos de software
ENG.3	Desenvolver o projeto do software
ENG.4	Implementar o projeto do software
ENG.5	Integrar e testar o software
ENG.6	Integrar e testar o sistema
ENG.7	Manter o sistema e o software
SUP – Suporte - Processos que podem ser empregados por qualquer um dos outros processos	
SUP.1	Desenvolver a documentação
SUP.2	Desempenhar a gerência de configuração
SUP.3	Executar a garantia da qualidade
SUP.4	Executar a verificação dos produtos de trabalho
SUP.5	Executar a validação dos produtos de trabalho
SUP.6	Executar revisões conjuntas
SUP.7	Executar auditorias
SUP.8	Executar resolução de problemas
MAN – Gerência - Processos que contém práticas de natureza genérica que podem ser usadas por quem gerencia projetos ou processos dentro de um ciclo de vida de software	

MAN.1	Gerenciar o projeto
MAN.2	Gerenciar a qualidade
MAN.3	Gerenciar riscos
MAN.4	Gerenciar subcontratantes
ORG – Organização - Processos que estabelecem os objetivos de negócios da organização	
ORG.1	Construir o negócio
ORG.2	Definir o processo
ORG.3	Melhorar o processo
ORG.4	Prover recursos de treinamento
ORG.5	Prover infra-estrutura organizacional

Tabela 14: Categorias e Processos do SPICE

A norma define detalhes de cada um dos processos mencionados acima. Para cada um deles existe uma definição mais detalhada, uma lista dos resultados da sua implementação bem sucedida e uma descrição detalhada de cada uma das práticas básicas.

3.1.5.2) Níveis de Capacitação

O SPICE, entretanto, não se limita a listar categorias e processos. Seu principal objetivo, na realidade, é avaliar a capacitação da organização em cada processo e permitir a sua melhoria. O modelo de referência do SPICE inclui seis níveis de capacitação. Cada um dos processos mencionados acima deve ser classificado nestes níveis. Os níveis são descritos a seguir:

Nível	Nome	Descrição
0	Incompleto	Há uma falha geral em realizar o objetivo do processo. Não existem produtos de trabalho nem saídas do processo facilmente identificáveis.
1	Realizado	O objetivo do processo em geral é atingido, embora não necessariamente de forma planejada e controlada. Há um consenso na organização de que as ações devem ser realizadas e quando são necessárias. Existem produtos de trabalho para o processo e eles são utilizados para atestar o atendimento dos objetivos.
2	Gerenciado	O processo produz os produtos de trabalho com qualidade aceitável e dentro do prazo. Isto é feito de forma planejada e controlada. Os produtos de trabalho estão de acordo com padrões e requisitos.
3	Estabelecido	O processo é realizado e gerenciado usando um processo definido, baseado em princípios de Engenharia de Software. As pessoas que implementam o processo usam processos aprovados, que são versões adaptadas do processo padrão documentado.
4	Predizível	O processo é realizado de forma consistente, dentro dos limites de controle, para atingir os objetivos. Medidas da realização do processo são coletadas e analisadas. Isto leva a um entendimento quantitativo da capacitação do processo a uma habilidade de predizer a realização.
5	Otimizado	A realização do processo é otimizada para atender as necessidades atuais e futuras do negócio. O processo atinge seus objetivos de negócio e consegue ser repetido. São estabelecidos objetivos quantitativos de eficácia e eficiência para o processo, segundo os objetivos da organização. A monitoração constante do processo segundo estes objetivos é conseguida obtendo feedback quantitativo e o melhoramento é conseguido pela análise dos resultados. A otimização do processo envolve o uso piloto de idéias e tecnologias inovadoras, além da mudança de processos ineficientes para atingir os objetivos definidos.

Tabela 15: Níveis de Capacitação do SPICE

3.1.5.3) Os nove manuais do SPICE

Esta norma se constituirá de um conjunto de 9 manuais, totalizando quase 400 páginas, conforme o detalhamento a seguir, baseado na atual versão preliminar.

Parte	Páginas	Descrição
1	16	Guia de Introdução e Conceitos.
2	38	Modelo de referência para processos e capacidade de processos.
3	7	Realizando uma avaliação.
4	36	Guia para realização de uma avaliação.
5	145	Um modelo de avaliação e guia de indicadores.
6	31	Guia para qualificação de avaliadores.
7	47	Guia para uso no melhoramento de processos.
8	25	Guia para uso na determinação da capacidade do processo de fornecedor.
9	9	Vocabulário.

Tabela 16: Os Manuais do SPICE

3.1.6)O Modelo Trillium – Modelo de Qualidade de Processo de Software.

Trillium é um modelo de desenvolvido para controle de processos de produtos de telecomunicações é também baseado nas condições de maturidade de um processo produtivo. No conceito deste modelo, o processo é focalizado na tentativa de se conseguir um produto (software) de qualidade mantendo a qualidade em todo seu processo de desenvolvimento.

Desenvolvido pela Bell Canada, em cooperação com a Northern Telecom, o Trillium é um modelo de qualidade que descreve um modelo de desenvolvimento de produtos de telecomunicações e de suporte à capacidade de processo. O Trillium pode ser enquadrado em um programa de Melhoria do Processo de Software.

É fortemente baseado no CMM, versão 1.1, tendo recebido influências de outros padrões:

- ISO 9001 e ISO 9000/3;
- Regulamentos internos da Bell;
- Critérios do Prêmio Nacional de Qualidade Malcom Baldrige;
- Padrões de Engenharia de Software da IEEE Software e IEC 300;
- Referências técnicas e profissionais.

3.1.6.1) Formas de Utilização do Modelo

Trillium e utilizado para avaliar o desenvolvimento de produtos da organização e no suporte ao processo de capacidade através do uso de melhores práticas. Usado também como um mecanismo de auto-avaliação, auxiliando a identificar oportunidades de melhoria em uma organização de desenvolvimento de produto e aplicado em negociações pré-contratuais, assistindo na seleção de um fornecedor.

- Para o cliente a alta capacidade significa:
  - Desenvolvimento de uma organização é mais sensível ao cliente e à demanda de mercado;
  - Custo do ciclo-de-vida é minimizado;

- Satisfação do usuário final é maximizada.

➢ Para a organização a alta capacidade significa:

- Redução do custo de desenvolvimento e manutenção, com o encurtamento do ciclo-de-vida e dos intervalos de desenvolvimento;
- Aumento na habilidade para alcançar o conteúdo e assegurar o planejamento, devido à efetividade na análise de riscos e na estimativa de esforços;
- Aumento na habilidade para encontrar um projeto quantificável e objetivos de qualidade em todos os estágios do processo de desenvolvimento.

3.1.6.2) Descrição do Modelo

Trillium possui 5 níveis, definidos de forma ligeiramente diferente em relação ao CMM:

➢ Desestruturado: como no CMM, este é um Processo de Software *ad hoc* e caótico;

➢ Repetível e Orientado a Projeto: o sucesso do projeto é alcançado através de um forte gerenciamento de projeto, planejamento e controle;

➢ Definido e Orientado a Processo: processos são definidos e utilizados em nível organizacional;

➢ Gerenciado e Integrado: medidas e análises são utilizadas como o mecanismo-chave para a melhoria do processo;

➢ Plenamente Integrado: metodologias formais são extensivamente utilizadas. Dados de medidas para a melhoria do processo são utilizados e efetivos.

As 8 Áreas de Capacidades Trillium são:

- Qualidade de Processo Organizacional;
- Desenvolvimento e Gerenciamento de Recurso Humano;
- Processo;
- Gerenciamento;
- Qualidade;
- Práticas de Desenvolvimento de Sistemas;
- Ambiente de Desenvolvimento;
- Suporte ao Cliente.

3.1.6.3) Níveis de Capacidade

Para atingir um nível Trillium, uma organização deve satisfazer um mínimo de 90% dos critérios em cada uma das 8 Áreas de Capacidade naquele nível.

Os níveis 3, 4 e 5 exigem o cumprimento de todos os níveis Trillium mais baixos (isto é, níveis não podem ser ignorados). Cada Área de Capacidade incorpora um ou mais roteiros.

Um roteiro é um conjunto de práticas relacionadas que enfoca uma área ou necessidade organizacional, ou um elemento específico dentro de um processo de desenvolvimento de produto.

Cada roteiro representa uma capacidade significativa para uma organização de desenvolvimento de software.

3.1.6.4) As Práticas Trillium

O conjunto de práticas no modelo Trillium é construído utilizando-se o seguinte algoritmo:

- Práticas são tomadas do SEI CMM Versão 1.1;
- Cláusulas ISO 9000 e ISO 9000-3 são mapeadas para este conjunto e, onde possível, são modificadas para integrar estes requisitos;
- Todas as cláusulas ISO 9000 e ISO 9000-3 remanescentes são adicionadas;
- As cláusulas dos padrões Bellcore são mapeadas para as práticas geradas pelos passos 1, 2, e 3. Quando possível, práticas são modificadas para integrar estes requisitos;
- Todas as cláusulas dos padrões Bellcore remanescentes (isto é, aquelas que não puderam ser mapeadas) são adicionadas ao conjunto de práticas;
- mesmo processo é repetido com porções relevantes do Malcolm Baldrige National Quality Award Criteria;
- Práticas da IEC 300 são adicionadas;
- Referências aos padrões IEEE relevantes são adicionadas

Quando as práticas forem extraídas do CMM, ou outros padrões, elas seguem as seguintes transformações, se aplicável:

- A prática é generalizada removendo-se as referências à palavra "software", ou trocando-as por "produto e serviços" ou "sistemas";
- A prática é generalizada removendo-se as referências ao termo "desenvolvimento", ou trocando-as por "desenvolvimento e suporte";
- As referências "grupo" ou outras unidades organizacionais específicas são

trocadas por "função";

- Referências indiretas a documentos específicos são trocadas por referências a um processo (por exemplo, "plano de qualidade" por "planejamento de qualidade"), ou "documentação" ou "informação".

As práticas são atribuídas a um dado nível baseando-se nas seguintes diretivas gerais:

- Práticas que são consideradas fundamentais para a conclusão bem sucedida de um projeto de desenvolvimento são atribuídas ao nível 2.
- Práticas que são consideradas pertinentes ao amplo escopo da organização ou fundamentais para o melhoramento contínuo do processo de desenvolvimento são atribuídas ao nível 3.
- Práticas que lidem com tecnologia CASE ou caracterizem avançado processo de maturidade (por exemplo, gerenciamento de mudanças, integração de prevenção de defeitos, controle estatístico de processos ou métricas avançadas) são geralmente atribuídas ao nível 4.
- O nível 5 lida tipicamente com tecnologia avançada quando aplicada à automação de processos, metodologias formais e utilização estratégica de repositórios da organização.

Existem 3 Estilos de texto para práticas:

- *Estilo 1: Alguma coisa é feita (informalmente)*

  Este estilo é usado para declarar que:

  - Alguma coisa está no lugar ou existe;
  - Uma atividade é realizada informalmente na organização e/ou ele é realizado sobre um projeto através da base do projeto.

- *Estilo 2: Alguma coisa é feita de acordo com um procedimento documentado*

  Este estilo especifica que:

  - Uma atividade é realizada;
  - Existe um procedimento escrito explicando como realizar a atividade;
  - O procedimento é compreendido pelos praticantes;
  - O procedimento é consistentemente usado pelos praticantes.

- *Estilo 3: Alguma coisa é feita formalmente*

  Este estilo inclui todos os requisitos do segundo estilo e acrescentam as seguintes atividades:

  - o conteúdo de produto (s) intermediário é revisado;
  - a aderência ao procedimento escrito é revisada;
  - registros da revisão(s) são mantidos;
  - todos os itens de ação sobre o relatório revisado são rastreados até a conclusão;
  - a prática é documentada e uniformemente aplicada através da organização.

3.1) Qualidade de Produtos de Software

Quando se pensa em qualidade de um "produto físico", é fácil imaginar padrões de comparação, provavelmente ligado às dimensões do produto ou alguma outra característica física. Quando se trata de software, como podemos definir exatamente o que é a qualidade?

Parece difícil, mas felizmente, a ISO (Organização Internacional de Padrões), publicou uma norma que representa a atual padronização mundial para a qualidade de produtos de software. Esta norma chama-se ISO/IEC 9126 e foi publicada em 1991. Ela é uma das mais antigas da área de qualidade de software e já possui sua tradução para o Brasil, publicada em agosto de 1996 como NBR 13596.

3.2.1) ISO/IEC 9126 [NBR 13596] - Software quality characteristics and metrics

3.2.1.1) Definições

As seguintes definições aplicam-se aos propósitos desta Norma:

a)*Características de qualidade de software:* Um conjunto de atributos de um produto de software, através do qual sua qualidade é descrita e avaliada. Uma característica de qualidade pode ser detalhada em múltiplos níveis de sub-características.

b)*Critério de julgamento de qualidade de software:* O conjunto de regras e condições definidas e documentadas que são usadas para decidir se a qualidade total de um produto de software específico é aceitável ou não. A qualidade é representada pelo conjunto de níveis de pontuação associados ao produto de software;

c)*Firmware:* Hardware que contém os dados e um programa de computador que não podem ser alterados no ambiente do usuário. Os dados e o programa contidos no

firmware são classificados como software; os circuitos que contêm os dados e o programa de computador são classificados como hardware;

d)*Julgamento:* Ação de aplicar critérios de julgamento específicos e documentados a um produto, pacote ou módulo de software específico, com o propósito de determinar sua aceitação ou liberação para o uso;

e)*Medição:* Ação de aplicar uma métrica de qualidade de software a um produto de software específico;

f)*Métrica de qualidade de software:* Um método e uma escala quantitativa que podem ser usados para determinar o valor que uma particularidade (feature) recebe em um produto de software específico;

g)*Nível de desempenho:* O grau em que as necessidades são satisfeitas, representadas por um conjunto específico de valores para as características de qualidade;

h)*Nível de pontuação:* Uma faixa de valores numa escala para permitir que o software seja classificado (pontuado) de acordo com as necessidades explícitas ou implícitas;

i)*Particularidades (features):* São propriedades identificadas de um produto de software que podem ser relacionadas às características de qualidade;

j)*Pontuação:* Ação de mapear o valor medido ao nível de pontuação apropriado. Usado para determinar o nível de pontuação obtido pelo software em uma característica de qualidade específica;

k)*Produto de software:* Uma entidade de software disponível para liberação a um usuário;

l)*Qualidade:* A totalidade das características de uma entidade que lhe confere a capacidade de satisfazer as necessidades explícitas e implícitas. (NBR ISO 8402/1993);

m)*Software:* Programas, procedimentos, regras e qualquer documentação associada, pertinente à operação de um sistema computacional.

3.2.1.2) Conceituação e Estrutura

Essa Norma apresenta seis características básicas através das quais podem ser estabelecidas métricas para a qualidade do software, Essas características fornecem também uma base para posterior refinamento e descrição da qualidade desejada dos produtos de software. Cada característica possui sub-características associadas.

Dessa forma, a ISO/IEC-9126 é aplicada na definição dos requisitos do software e, após a conclusão, na sua avaliação. As características tratadas nessa Norma são as seguintes:

A norma ISO/IEC 9126 ou NBR 13596 lista um conjunto de características que devem ser verificadas em um software para que ele seja considerado um "software de qualidade". São seis grandes grupos de características, cada uma é dividida em algumas sub-características.

Os nomes dados pelo ISO/IEC para as características e sub-características são complexos, entretanto, uma pessoa que trabalha com software não terá dificuldade em entendê-las.

As características definidas pela norma ISO/IEC 9126 ou NBR 13596 são as seguintes:

- Funcionalidade: Satisfazem necessidades explícitas ou implícitas.
- Confiabilidade: Capacidade do software em manter seu nível de desempenho, sob condições estabelecidas, por um período de tempo.
- Usabilidade: Esforço necessário para utilizar o software e para o julgamento individual deste uso, por um determinado conjunto de usuários.
- Eficiência: Relacionamento entre o nível de desempenho do software e a quantidade de recursos usados, sob condições estabelecidas.
- Manutenibilidade: Esforço necessário para a realização de modificações específicas no software.
- Portabilidade: Habilidade de um software ser transferido de um ambiente para outro.

3.2.1.3)Implementação da ISO 9126

A implementação da norma ISO 9126 ou NBR 13596 - Tecnologia de Informação: Avaliação de Produto de Software - Características de Qualidade e Diretrizes para seu Uso. Definem seis características de qualidade de produto de software, como citado anteriormente no tópico anterior, que são subdivididas em diversas sub-características. A seguir, é apresentado a descrição destas características e sub-características.

a) Funcionalidade: Conjunto de atributos que evidenciam a existência de um conjunto de funções e suas propriedades especificadas. Adequação Atributos do

software que evidenciam a presença de um conjunto de funções e sua apropriação para as tarefas especificadas.

- Sub-caracteríscas:
  - Acurácia - atributos do software que evidenciam a geração de resultados ou efeitos corretos ou conforme acordados.
  - Interoperabilidade - atributos do software que evidenciam sua capacidade de interagir com sistemas especificados.
  - Conformidade - atributos do software que fazem com que ele esteja de acordo com as normas, convenções ou regulamentações previstas em leis e descrições similares, relacionadas à aplicação.
  - Segurança de acesso - atributos do software que evidenciam sua capacidade de evitar o acesso não autorizado, acidental ou deliberado, a programas e dados.

b) Confiabilidade: Conjunto de atributos que evidenciam a capacidade do software de manter seu nível de desempenho sob condições estabelecidas durante um período de tempo estabelecido.

- Sub-caracteríscas:
  - Maturidade - atributos do software que evidenciam a freqüência de falhas por defeitos do software.
  - Tolerância a falhas - atributos do software que evidenciam sua capacidade em manter um nível de desempenho especificado nos casos de falhas no software ou de violação nas interfaces especificadas.
  - Recuperabilidade - atributos do software que evidenciam a sua capacidade de restabelecer seu nível de desempenho e recuperar os dados diretamente afetados, em caso de falha, e o tempo de esforço para tal.

c) Usabilidade (Capacidade Para Uso): Conjunto de atributos que evidenciam o esforço necessário para poder-se utilizar o software, bem como o julgamento individual deste uso, por um conjunto implícito ou explícito de usuários.

- Sub-caracteríscas:
  - Inteligibilidade - atributos do software que evidenciam o esforço do usuário para reconhecer o conceito lógico e sua aplicabilidade.
  - Apreensibilidade - atributos do software que evidenciam o esforço do usuário para apreender sua aplicação.
  - Operacionalidade - atributos do software que evidenciam o esforço do usuário para a sua operação e controle da sua operação.

d) Eficiência: Conjunto de atributos que evidenciam o relacionamento entre o nível de desempenho do software e a quantidade de recursos usados, sob condições estabelecidas.

- Sub-caracteríscas:
  - Comportamento em relação ao tempo - atributos do software que evidenciam seu tempo de resposta, tempo de processamento e velocidade na execução de suas funções.
  - Comportamento em relação aos recursos - atributos do software que evidenciam a quantidade de recursos usados e a duração de seu uso na execução de suas funções.

e) Portabilidade: Conjunto de atributos que evidenciam a capacidade do software em ser transferido de um ambiente para outro.

- Sub-caracteríscas:
  - Adaptabilidade - atributos do software que evidenciam sua capacidade de ser adaptado a ambientes diferentes especificados, sem a necessidade de aplicação de outras ações ou meios além daqueles fornecidos para esta finalidade pelo software considerado.

- Capacidade para ser instalado - atributos do software que evidenciam o esforço necessário para sua instalação num ambiente especificado.
- Conformidade - atributos do software que o tornam consoante com padrões ou convenções relacionados à portabilidade.
- Capacidade para substituir - atributos do software que evidenciam sua capacidade e esforço necessário para substituir um outro software, no ambiente estabelecido para este outro software. Conjunto de atributos que evidenciam o esforço necessário para fazer modificações especificadas no software.
- Analisabilidade - atributos do software que evidenciam o esforço necessário para diagnosticar deficiências ou causas de falhas, ou para identificar partes a serem modificadas.

f) Manutenibilidade: Conjunto de atributos que evidenciam a capacidade do software em ser transferido de um ambiente para outro.

➢ Sub-caracteríscas:

- Modificabilidade - atributos do software que evidenciam o esforço necessário para modificá-lo, remover seus defeitos ou adaptá-lo a mudanças ambientais.
- Estabilidade Atributos do software que evidenciam o risco de efeitos inesperados ocasionados por modificações.
- Testabilidade Atributos do software que evidenciam o esforço necessário para validar o software modificado.

3.2.1.3) **Aplicação da Norma 9126**

Estas seis características de qualidade, definidas pela norma ISO/IEC 9126 ou NBR 13596 devem ser avaliadas em um produto de software, qualquer que seja a sua forma de aquisição ou de desenvolvimento.

No caso de aquisição, a norma atua como um guia na elaboração do objeto técnico, definindo o que deve ser avaliado (as características e sub-características de qualidade). A aplicação da norma é adaptável ao produto que se deseja adquirir, ou seja, podemos definir quais características e sub-características são mais determinantes para o produto de software em questão. Por exemplo, para um determinado produto, as questões de segurança de acesso e de recuperabilidade podem ser mais importantes do que para outro. Assim, é possível definir uma pontuação para as características e sub-características de qualidade de acordo com o que se espera do produto de software desejado, de forma objetiva e clara, com a atribuição de pesos para as características mais importantes para o produto desejado.

Entretanto, apesar da norma definir cada característica e sub-característica de qualidade, ela não define como medi-las. Há a necessidade, então, de um trabalho de definição das métricas a serem aplicadas em cada uma das sub-características de qualidade.

No caso do desenvolvimento de um produto de software, pode ser aplicada durante a realização dos testes de aceitação do produto ou de subprodutos do desenvolvimento, quase da mesma forma que a descrita acima. A diferença existente é que não há a necessidade de se definir uma fórmula de pontuação geral do produto, mas sim a definição de níveis mínimos de pontuação aceitáveis, abaixo dos quais o produto

ou subproduto não será aceito. Também deve ser definida a sistemática a ser aplicada durante os testes: definição dos avaliadores, perfil dos avaliadores, critérios para consolidação da pontuação em caso de divergências entre os avaliadores, etc.

3.2.2) ISO/IEC 14598 - Software Product Evaluation

A ISO/IEC 14598, pode ser usada para suporte de atividades como:

- Especificação de requisitos de qualidade e usabilidade e confronto com estes requisitos;
- Incorporação da usabilidade dentro de um sistema de qualidade;
- Guia para o processo de projeto orientado ao usuário.

Objetivo: visão geral dos processos de avaliação de produto de *software* e guia e requisitos para avaliação.

Essa norma é composta de um grupo de guias que servem de apoio ao planejamento e ao controle de uma avaliação da qualidade de produtos de software. Esse conjunto de guias é composto de seis partes, sendo:

- 14598-1 - Parte 1: Visão Geral - visão geral da estrutura da norma e do processo de avaliação;
- 14598-2 - Parte 2: Planejamento e Gerenciamento - composta das atividades de planejamento e gerenciamento do processo de avaliação;
- 14598-3 - Parte 3: Processo para Desenvolvedores - processo do desenvolvedor: composta das atividades de avaliações que deverão ser feitas durante um processo de desenvolvimento de softwares (quando for o caso de desenvolvimento de um software);

- 14598-4 - Parte 4: Processo para Compradores - processo do comprador: composta das atividades de avaliações que deverão ser feitas durante um processo de compra de softwares (quando for o caso de compra de um software);
- 14598-5 - Parte 5: Processo Para Avaliadores - processo do avaliador: ciclo de vida de avaliação e suas atividades;
- 14598-6 - Parte 6: Documentação de Módulos de Avaliação - módulos de avaliação: pacotes de métodos e dispositivos de apoio aos processos de avaliação do desenvolvedor, do comprador e do avaliador.

3.2.2.1) DESCRIÇÃO DAS PARTES DA ISO/IEC 14598

Atualmente este padrão está sob desenvolvimento e apenas às partes de 1 a 5 foram publicadas como "Draft International Standard (DIS)", enquanto as outras ainda estão no status de "Committee Draft (CD)".

a) *ISO/IEC 14598-1 - General Overview (Visão Geral)*

- Visão Geral

  Documento introdutório da série. Provê uma visão geral das outras partes e explica a relação entre a avaliação do produto de software e o modelo de qualidade definido na norma ISO/IEC 9126. Adicionalmente, descreve o processo de avaliação nos seguintes passos: Estabelecer os requisitos da avaliação, especificar, projetar e executar a avaliação.

- Público Alvo

  Pessoas envolvidas em avaliação de produtos de *software* desenvolvedores, compradores e avaliadores independentes.

- Conteúdo

- Visão dos guias da série
- Visão geral de um processo de avaliação
- Definição de termos técnicos

b) ISO/IEC 14598-2 - Planning and Management (Planejamento e Gerenciamento)

➢ *Visão Geral*

Requisitos e Guias para Gerenciamento de avaliação e de tecnologias necessárias para a avaliação de produtos de *software*.

➢ Público Alvo

- Responsáveis por: gerenciamento da aplicação da tecnologia, suporte à avaliação de produtos de *software*, gerenciamento de organizações de desenvolvimento de *software*.
- Pessoas que exerçam função associada à garantia da qualidade

➢ Conteúdo

•Guia para gerenciamento da avaliação

c) ISO/IEC 14598-3 - Process for Developers (Processo para Desenvolvedores)

➢ Visão Geral

- Provê requisitos e recomendações para avaliação do produto de software quando esta é conduzida em paralelo com o desenvolvimento e é feita pelo desenvolvedor. O foco é o uso de indicadores de predição da qualidade do produto final através da media de produtos desenvolvidos durante o ciclo de vida.
- Avaliação durante o desenvolvimento
- Aplicado em todas as fases do ciclo de vida

➢ Público Alvo

•Gerente do projeto

•Desenvolvedor de *software* e equipe de manutenção

•Organização da garantia da qualidade, de auditoria e controle

- Conteúdo

•Critério para seleção de indicadores de qualidade

•Guia para melhorar o processo de medida

•Guia de como conduzir o processo de avaliação

d)ISO/IEC 14598-4 - Process for Acquirers (Processo para Compradores)

- Visão Geral
  - Provê requisitos e recomendações para avaliação do produto de software para ser conduzida por consumidores durante o planejamento da compra ou reuso de um produto de software existente ou pré-desenvolvido.
  - Seleção e aquisição de:
    - Produtos de prateleira;
    - Produtos específicos para o cliente;
    - Modificação de *software* já existente.
- Público Alvo:

•Gerentes de Projetos e Engenheiros de Sistemas

•Equipe de Desenvolvimento e Manutenção de *Software*

- Conteúdo

•Requisitos e Guias para seleção e aceitação de produto

•Processos sistemáticos para avaliação e aquisição de produtos de prateleira, produtos específicos e modificações a *software* já existentes

e)ISO/IEC 14598-5 - Process for Evaluators (Processo para Avaliadores)

- Visão Geral

•Requisitos e recomendações para a execução de avaliação de produtos de *software*.

- Público Alvo

- Avaliadores de laboratório de teste
- Fornecedores, Compradores, Avaliadores independentes
- Órgãos Certificadores

➢ Conteúdo

- Define atividades para analisar requisitos da avaliação, especificar, projetar e executar ações de avaliação e concluir a avaliação de qualquer tipo de produto de *software*.

f) ISO/IEC 14598-6 - Documentation of Evaluation Modules (Documentação de Módulos de Avaliação)

➢ Visão Geral

- Define a documentação de "Módulos de Avaliação" que são pacotes de métodos, técnicas, ferramentas e instruções para auxiliar na avaliação de um atributo de qualidade. Estes módulos representam a especificação do modelo de qualidade correspondente às medidas internas ou externas que serão aplicadas em uma avaliação particular.
- Um conjunto de Módulos de Avaliação forma uma Biblioteca.

➢ Público Alvo

- Peritos em tecnologia de avaliação como laboratórios de teste e institutos de pesquisa.

➢ Conteúdo

- Estrutura e conteúdo da documentação para ser usada na descrição, desenvolvimento e validação de Módulos de Avaliação.

3.2.2.2) O Processo de avaliação

O processo de avaliação consiste de um conjunto de atividades, que são conduzidas pelo requisitante da avaliação e pelo avaliador cooperativamente. As atividades do processo usam como base os dados fornecidos pelo requisitante, pelo avaliador, ou produzidos por outras atividades do conjunto. Elas produzem dados que são os resultados do processo de avaliação e que também podem ser utilizados por outras atividades do processo.

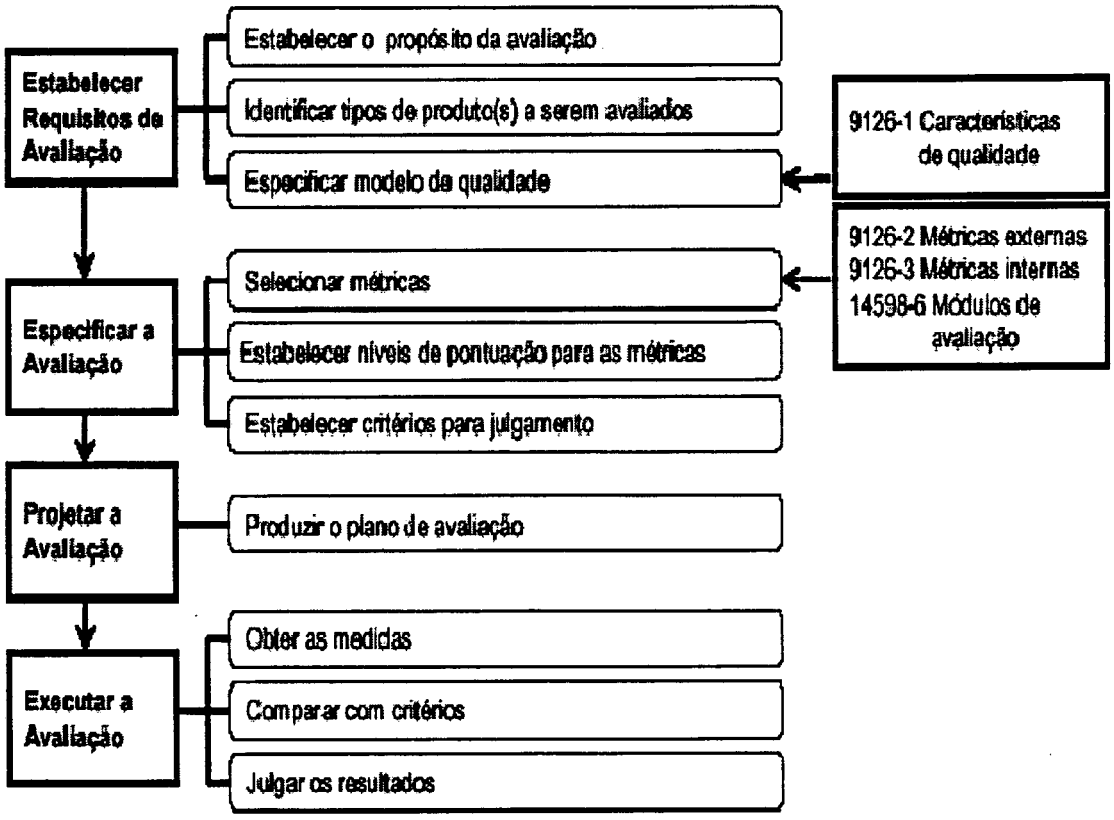


Figura 3: Conjunto de atividades do processo de avaliação

3.2.2.3) Processo de Avaliação de Produto de Software

A) Estabelecer requisitos de avaliação

➢ Estabelecer o propósito da avaliação

- Geral - O propósito da avaliação de qualidade de software é apoiar diretamente o desenvolvimento e a aquisição de software que atenda às necessidades dos usuários e do cliente. O objetivo final é assegurar que o produto forneça a qualidade requerida - que ele atenda às necessidades explicitas e implícitas dos usuários do software, ou mantenedores de software.
- Aquisição - Na aquisição de um produto de software sob encomenda, convém que o adquirente estabeleça requisitos de qualidade externa, especifique os requisitos para o fornecedor, e avalie compras potenciais em relação a estes requisitos antes da aquisição. Quando um produto está sendo desenvolvido, o objetivo da especificação dos requisitos de qualidade é assegurar que o produto atenda às necessidades explícitas e implícitas do usuário. Na compra de um produto de software, a avaliação pode ser utilizada para comparar produtos alternativos e para assegurar que o produto selecionado atende aos requisitos de qualidade.

- Fornecimento - O fornecedor pode utilizar os resultados da avaliação de produto de software para assegurar que os produtos atendam aos critérios de qualidade requeridos, que podem ter sido definidos pelo adquirente, ou por comparação com outros produtos.

- Desenvolvimento - Convém que a avaliação de software seja utilizada

para prever e verificar a qualidade durante o desenvolvimento. Os resultados da avaliação podem ser utilizados para obter "feedback" sobre o quanto os diferentes processos de desenvolvimento, métodos de projeto ou ferramentas CASE, podem ser utilizados para atender aos requisitos de qualidade.

- Operação - A organização que opera um sistema de software pode utilizar a avaliação para validar que os requisitos de qualidade são atendidos sob diferentes condições de operação, e para fornecer "feedback" aos responsáveis pela manutenção sobre a necessidade de qualquer alteração.

- Manutenção - A organização que mantém o sistema de software pode utilizar a avaliação para validar se os requisitos de qualidade ainda são atendidos, e se os requisitos para manutenibilidade e portabilidade são atingidos.

➢ Identificar os tipos de produto(s) a serem avaliados

O tipo de produto de software, quer seja, um dos produtos intermediários ou final, a ser avaliado, dependerá do estágio no ciclo-de-vida e do propósito da avaliação.

➢ Especificar o modelo de qualidade

A primeira etapa na avaliação de software é selecionar as características de qualidade relevantes, utilizando um modelo de qualidade que desdobre a qualidade de software em diferentes características. Os modelos de qualidade para avaliação de software geralmente representam a totalidade dos atributos de qualidade de software classificados em uma árvore hierárquica de características e sub-características. O nível mais alto desta estrutura é composto pelas características de qualidade e o nível mais baixo é composto pelos atributos de qualidade do software. Exemplo:

Característica de Qualidade: Usabilidade			
Subcaracterística: Operacionalidade			
Id.	Descrição do Requisito	Prioridade	Alocado (S/N)
U01	O produto de software deverá possibilitar a customização das funções pelo próprio usuário	Essencial	Sim
Observação: somente poderão ser customizadas as funções às quais o usuário tem acesso autorizado. A prioridade de atendimento ao requisito é "Essencial" devido ao número elevado de usuários que executam somente algumas funções do sistema.			
Subcaracterística: Apreensibilidade			
Id.	Descrição do Requisito	Prioridade	Alocado (S/N)
U02	O produto de software deverá apresentar *help* de contexto para campos.	Essencial	Sim
Observação:			

Figura 4: Característica de Qualidade

b) Especificar a Avaliação

- Selecionar métricas

É importante que as medições de um produto de software possam ser feitas fáceis e economicamente e que as medidas resultantes sejam fáceis de usar.

A forma pela qual as características de qualidade têm sido definidas não permite sua medição direta. É necessário estabelecer métricas que se correlacionem às características do produto de software. Todo atributo interno quantificável do software e todo atributo externo quantificável do software interagindo com seu ambiente e que se correlacione com uma característica, pode ser definido como uma métrica.

Norma 14598-3 - O desenvolvedor deve definir as condições sob as quais as medições devem ser executadas. Isto significa que se deve

identificar outros atributos cujos valores influem nas medições e definir os valores desses mesmos atributos.

- Estabelecer níveis de pontuação para as métricas

As particularidades quantificáveis podem ser medidas quantitativamente usando-se métricas de qualidade. O resultado, isto é, o valor medido, é mapeado numa escala. Este valor, por si só, não mostra o nível de satisfação. Para isso, a escala precisa ser dividida em faixas correspondentes aos diversos graus de satisfação dos requisitos. São exemplos:

- dividir a escala em duas categorias: satisfatória e insatisfatória;
- dividir a escala em quatro categorias delimitadas por: o pior caso, o nível atual

Para um produto existente ou alternativo, e o nível planejado. Exemplo:

Figura 5: Nível de Planejamento

- Estabelecer critérios para julgamento

Para julgar a qualidade do produto, o resultado da avaliação de cada característica precisa ser sintetizado. Convém que o avaliador prepare um procedimento para isto, com critérios diferentes para características de qualidade diferentes, onde cada característica poderá estar representada em termos de suas sub-características ou de uma combinação ponderada de sub-características. O procedimento normalmente incluirá outros aspectos como tempo e custo, os quais contribuem para o julgamento de um produto de software num ambiente particular.

c) Projetar a avaliação

- Produzir o plano de avaliação

O Plano de Avaliação descreve os métodos de avaliação e o cronograma das ações do avaliador. Convém que ele esteja consistente com o Plano de Medições.

Norma 14598-3 - O desenvolvedor de software deve especificar ações (procedimentos) para coleta dos dados a serem executadas para obter valores reais para cada métrica externa ou interna. Isto inclui a especificação de cronogramas, responsabilidades, e o uso de ferramentas de coleta de dados e de análise. Se for necessário treinamento especial para o pessoal, isto também deve ser planejado.

O desenvolvedor deve também definir o grau de precisão da medição. Todo modelo estatístico aplicado deve ser especificado, incluindo requisitos de entrada de dados, estratégias de amostragem, etc.

O desenvolvedor deve definir ações em caso de eventualidades, como avaliações extras, caso os resultados das medições sejam não conclusivos ou alarmantes.

d) Executar a avaliação

- Tomar as medidas

Para medição, as métricas selecionadas são aplicadas ao produto de software. Como resultado obtém-se os valores nas escalas das métricas.

Norma 14598-3 - O desenvolvedor de software deve coletar os valores reais das medidas para os atributos internos definidos de acordo com as ações para coleta de dados definidos. Se os requisitos de qualidade forem modificados, o desenvolvedor deve reconsiderar es especificações da avaliação e o projeto da avaliação.

O desenvolvedor deve tomar as medidas necessárias para assegurar a qualidade dos dados coletados. As ações devem incluir, quando apropriado, validação de ferramentas automáticas para coleta de dados e utilização de pessoas para conferência dos dados.

- Comparar com critérios

Na etapa de pontuação, o valor medido é comparado com critérios predeterminados.

- Julgar os resultados

O julgamento é a etapa final do processo de avaliação do software, onde um conjunto de níveis pontuados são resumidos. O resultado é uma declaração de quanto o produto de software atende aos requisitos de qualidade. Então a qualidade resumida é comparada com outros aspectos como tempo e custo. Finalmente uma decisão gerencial será tomada baseada nos critérios gerenciais. O resultado é uma decisão gerencial quanto à aceitação ou rejeição, ou quanto à liberação ou não do produto de software.

Os resultados da avaliação são importantes para decisões sobre os próximos passos no ciclo-de-vida de desenvolvimento do software. Por exemplo, os requisitos devem ser alterados ou são necessários mais recursos

para o processo de desenvolvimento?

Norma 14598-3 - Convém que o desenvolvedor de software utilize os valores obtidos para os indicadores definidos para estimar a qualidade do produto final, levando-se em conta. Para tanto, a experiência da organização do desenvolvedor em projetos anteriores com requisitos de qualidade similares.

Também convém que o desenvolvedor utilize os valores obtidos para monitorar tendências de forma a identificar riscos de desenvolvimento, tomando ações de contingências quando necessário.

O desenvolvedor deve fazer um julgamento dos resultados da avaliação e para tanto convém que os valores obtidos sejam resumidos e comparados com outros valores, como tempo e custo. De maneira a sustentar uma decisão sobre o resultado do desenvolvimento (por exemplo, melhorando o produto, revendo os requisitos etc.).

Por fim, o desenvolvedor deve rever os resultados da avaliação e a validade do processo de avaliação, dos indicadores e das métricas aplicadas. Convém que a retroalimentação sobre esta revisão seja utilizada de maneira a melhorar o processo de avaliação e os módulos de avaliação. Quando for necessário melhorar os módulos de avaliação, convém que seja incluída a coleta de dados sobre indicadores extras, de maneira a validá-los para uso posterior .

e) Escolhendo Métricas

A precisão de uma avaliação de qualidade depende em grande parte das métricas escolhidas. Para aumentar a confiabilidade dos resultados são apresentadas a seguir algumas características que as métricas deveriam apresentar. Tais características estão de acordo com os requisitos de avaliação enumerados na 14598-1(itens 3 e 8.1.2). 14598-5(item 4.3) e 9126-1 (item 6.4), além de incluir outros fatores significativos para

o resultado final. São elas:

- Significância: os resultados da medição devem agregar informação sobre o comportamento do software ou suas características de qualidade.

- Custo e complexidade: a aplicação da métrica deve ser econômica e tecnicamente viável dentro do processo de avaliação.

- Repetibilidade: o uso da métrica i) no mesmo produto; ii) com a mesma especificação de avaliação; iii) com os mesmos avaliadores, usuários-teste e ambiente: deveria produzir resultados aceitos como idênticos.

- Reproducibilidade: o uso da métrica
  - No mesmo produto;
  - Com a mesma especificação de avaliação;
  - Com diferentes avaliadores. usuários-teste e ambiente: deveria produzir resultados aceitos como idênticos.

- Validade: deve ser possível demonstrar a corretude e precisão ou a margem de erro dos resultados da medição. E importante considerar neste requisito todos os fatores que podem influir no resultado, como:
  - Características de hardware;
  - Características de software;
  - Avaliadores;
  - Modelos matemáticos.

- Objetividade: os resultados da medição devem ser objetivos, e não podem sofrer influência de opinião e sentimentos do avaliador, usuários de teste, etc. Quando isto não for possível, a métrica deve oferecer algum tratamento (p.ex., estatístico) de correção.

- Imparcialidade: a medição não deve ser tendenciosa. Este requisito pode ser descumprido de várias formas:
  - Publicação parcial de resultados;
  - Ambiente tendencioso.

f) Processo para Adquirentes

A Norma 14598-4 está dividida em dois processos distintos: um para aquisição de produtos de software de prateleira e outro para aquisição de produtos de software sob encomenda ou modificações em produtos de software existentes. O Processo para Adquirentes é resultado da combinação do processo genérico de avaliação e do processo de aquisição definido pela NBR 12207 - Processo do Ciclo de Vida do Software.

O propósito da avaliação no processo de aquisição pode ser a comparação de produtos alternativos ou garantir que um produto desenvolvido ou modificado sob encomenda atenda aos requisitos inicialmente especificados.

- Processo de Aquisição

O processo de aquisição definido pela NBR 12207 é constituído das seguintes atividades:

- Iniciação - O adquirente inicia o processo de aquisição a partir de uma necessidade em adquirir, desenvolver ou melhorar um sistema, produto ou serviço de software. Nesta atividade são identificados e analisados os requisitos de sistema, os quais podem ser: de negócio, organizacionais e de usuário, segurança entre outros.

- Preparação do pedido de proposta - Nesta atividade é realizada a

documentação dos requisitos de aquisição. Além disto são determinados os pontos de controle nos quais o progresso do fornecimento deverá ser revisado e auditado.

- Preparação e atualização do controle - Nesta atividade realiza-se a seleção de fornecedores, baseando-se na avaliação das propostas dos fornecedores, bem como suas capacidades. Um contrato com o fornecedor deve ser preparado e negociado, incluindo custo e cronograma de execução.

- Monitoração do fornecedor - A monitoração do fornecedor consiste em atividades de avaliação aplicadas durante a execução do contrato levando à aceitação e entrega do produto de software.

- Aceitação e conclusão - A aceitação e conclusão são atividades executadas durante a aceitação do produto e entrega do produto de software final, obedecendo aos critérios de aceitação previamente definidos durante a atividade de iniciação.

g) Processo para o Avaliador

Na Norma 14598-5 - Processo para o avaliador fornece requisitos e recomendações para implementação prática de avaliação de produto de software. Ela segue o processo de avaliação como definido na 14598-1 e detalha as fases deste processo para a função de avaliação.

O processo de avaliação proposto na norma pode ser usado para avaliar produtos já existentes ou produtos em desenvolvimento. Podem ser utilizadas por avaliadores de laboratório, fornecedores de softwares, adquirentes de usuários e entidades certificadoras, cada um com objetivos específicos.

É importante ressaltar que o objetivo principal de avaliação de produto de software é fornecer resultados quantitativos sobre a qualidade de produto de software que sejam compreensíveis, aceitáveis e confiáveis por quaisquer das partes interessadas. Assim, a norma fornece orientações para manter o nível de objetividade da avaliação o mais alto possível em todas as circunstâncias.

- Partes envolvidas na avaliação:

  - Potenciais requisitantes de avaliações podem ser:
    - desenvolvedores de software;
    - fornecedores de software;
    - adquirentes de software;
    - usuários de software;
    - Integradores de sistemas atuando como adquirentes de software.

  - Potenciais avaliadores podem ser:
    - laboratórios de testes de terceira-parte;
    - equipes de testes integrantes de organizações de produção ou distribuição de software;
    - equipes de testes integrantes de organizações compradoras ou usuárias de software;
    - equipes de testes integrantes de organizações integradoras de sistemas;
    - organizações que fazem comparações entre produtos.

h) Características do processo de avaliação

O objetivo principal do processo de avaliação descrito nesta parte da ISO/IEC 14598, é promover as seguintes características desejáveis em processos de avaliações:

- repetibilidade: convém que avaliações repetidas do mesmo produto, com a

mesma Especificação de Avaliação, pelo mesmo avaliador, produzam resultados que possam ser aceitos como sendo idênticos;

- reprodutibilidade: convém que avaliações do mesmo produto, com a mesma especificação de avaliação, por avaliadores diferentes, produzam resultados que possam ser aceitos como sendo idênticos;
- imparcialidade: convém que a avaliação não seja tendenciosa com relação a algum resultado particular;
- objetividade: convém que os resultados das avaliações sejam baseados em fatos, isto é, desprovidos de sentimentos ou opiniões do avaliador.

i) Entradas do Processo

- Providas pelo que requer a avaliação:
  - Uma versão inicial dos requisitos de avaliação;
  - Uma descrição do produto de software a ser avaliado, de modo a identificar componentes relevantes à avaliação.

- Providas pelo avaliador:
  - Especificações de avaliação pré-definidas;
  - Métodos de avaliação;
  - Ferramentas de avaliação.

j) Saídas do Processo

- Requisitos de avaliação: descreve os objetivos da avaliação, o nível de criticidade requerido para o produto.

- Especificação da avaliação: descreve todas as análises e medidas a serem

realizadas e todos os componentes do produto a serem levados em consideração.

- Plano de avaliação: descreve procedimentos operacionais necessários à implementação da Especificação da avaliação (todos os métodos e ferramentas são descritos).

- Registros de Avaliação: contabilização de ações realizadas pelo avaliador durante a execução do Plano de avaliação.

- Relatório de Avaliação: reunião de todos os documentos gerados anteriormente e toda informação adicional necessária para repetir/reproduzir a avaliação. Deve ser entregue para a parte requerente da avaliação.

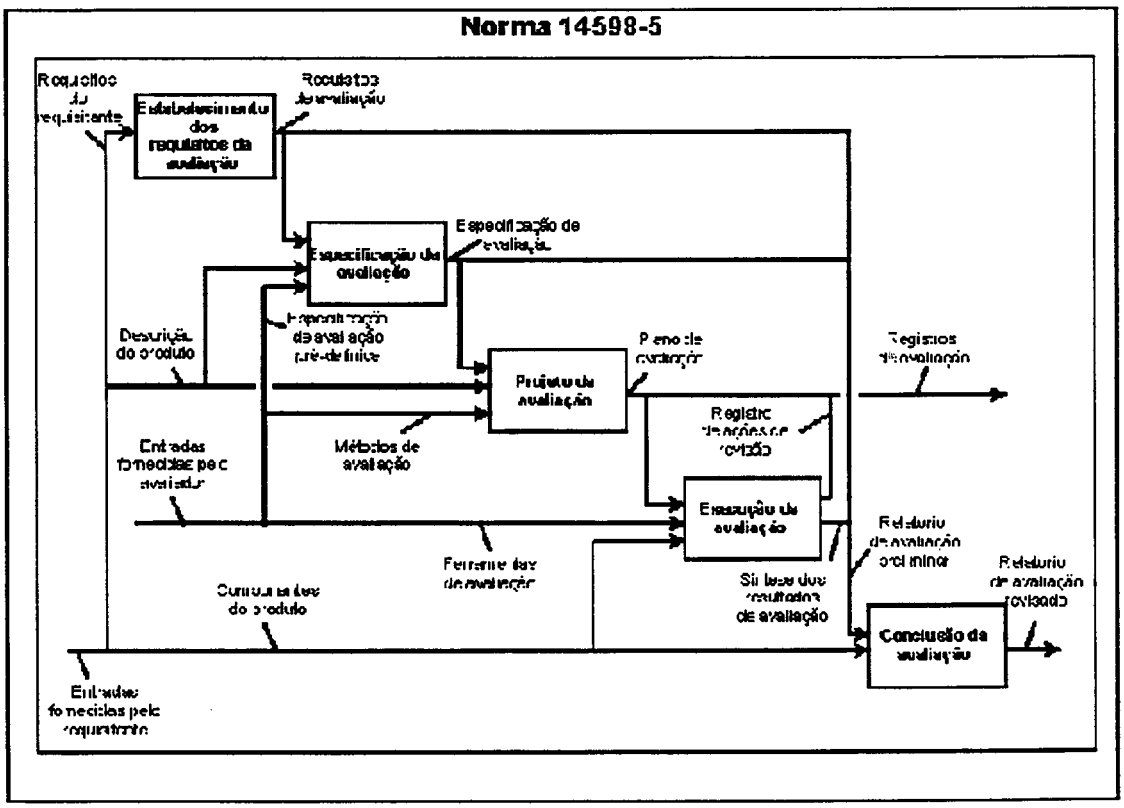


Figura 6: O Processo de Avaliação

3.2.3) ISO/IEC 12119 - Tecnologia da informação. Pacotes de Software –Requisitos de qualidade e testes

Essa Norma é aplicável a pacotes de software. São exemplos: processadores de texto, planilhas eletrônicas, bancos de dados, software gráficos, programas para funções técnicas ou científicas e programas utilitários, estabelecendo requisitos e instruções a respeito de como testá-los em relação aos requisitos estabelecidos.

Trata de todos os componentes do produto disponíveis aos usuários (documentação, manual de instruções e guia para instalação), ou seja define pacotes de *software* como o conjunto completo e documentado de programas fornecidos a diversos usuários para uma aplicação ou função genérica.

A norma inclui detalhes que devem estar presentes no produto, tais como:

- Documentação do usuário de fácil compreensão;
- Um sumário e um índice remissivo na documentação do usuário;
- Presença de um Manual de instalação com instruções detalhadas;
- Especificação de valores limites para todos os dados de entrada, que deverão ser testados;
- Operação normal mesmo quando os dados informados estão fora dos limites especificados;
- Consistência de vocabulário entre as mensagens e a documentação;
- Função de auxílio (help) com recursos de hipertexto;
- Mensagens de erro com informações necessárias para a solução da situação de erro;
- Diferenciação dos tipos de mensagem: confirmação, consulta, advertência e erro;

- Clareza nos formatos das telas de entrada e relatórios;
- Capacidade de reverter funções de efeito drástico;
- Alertas claros para as conseqüências de uma determinada confirmação;
- Identificação dos arquivos utilizados pelo programa;
- Identificação da função do programa que está sendo executada no momento.

A norma ISO 12119 foi publicada em 1994 e trata da avaliação de pacotes de software, também conhecidos como "software de prateleira". Além de estabelecer os requisitos de qualidade para este tipo de software, ela também destaca a necessidade de instruções para teste deste pacote.

3.2.3.1) Escopo

Esta Norma Internacional é aplicável a pacotes de *software*. São exemplos: processadores de textos, planilhas eletrônicas, programas de banco de dados, pacotes gráficos, programas para funções científicas ou técnicas e programas utilitários.

Ela estabelece:

- Requisitos para pacotes de *software* (requisitos de qualidade);
- Instruções de como testar um pacote de *software* de acordo com estes requisitos (instruções para testes, em particular para testes de terceira parte).

Ela trata somente de como pacotes de *software* são ofertados e entregues. Ela não abrange seus processos de produção (nem tão pouco atividades e produtos intermediários, como por exemplo especificações). O sistema de qualidade de um

fornecedor está fora do escopo desta Norma Internacional.

Como possíveis usuários desta Norma Internacional incluem-se:

a) produtores quando
   - especificam requisitos para um pacote de *software*;
   - projetam um documento para descrever produtos;
   - avaliam seus próprios produtos;
   - emitem declarações de conformidade [Guia ISO/IEC 22];
   - solicitam certificados ou marcas de.conformidade [Guia ISO/IEC 23];

b) entidades de certificação que desejem estabelecer um esquema de certificação de terceira parte (internacional, regional ou nacional) [Guias ISO/IEC 16, 28 e 44];

c) laboratórios de testes que deverão seguir as instruções para teste quando realizarem avaliações para emissão de um certificado ou um selo de conformidade [Guia ISO/IEC 25];

d) entidades de credenciamento que credenciem entidades de certificação e laboratórios de testes [Guias ISO/IEC 40 e 58];

e) auditores quando avaliam a competência de laboratórios de teste;

f) compradores que podem;
   - comparar os requisitos do seu produto com os descritos aqui;
   - comparar os requisitos para uma determinada tarefa com as informações contidas nas descrições de produtos existentes;
   - procurar por produtos certificados;

- checar se os requisitos foram atingidos;

g) usuários que podem beneficiar-se de melhores produtos.

3.2.3.2) Definições

a) **função**: A implementação de um algoritmo dentro de um programa com o qual o usuário ou o programa pode realizar parte ou toda uma tarefa de trabalho.

b) **documento de requisitos**: Um documento contendo qualquer combinação de recomendações, requisitos ou regulamentos a ser atendida por um pacote de *software*.

c) **descrição do produto**: Um documento declarando propriedades de um pacote de *software*, com o objetivo principal de auxiliar compradores em potencial na avaliação da adequação do produto antes de sua aquisição.

d) **documentação do usuário**: O conjunto completo de documentos disponíveis em formato impresso ou em outro formato, fornecido para o uso do produto, também sendo parte integrante do mesmo.

e) **documentação do pacote**: A descrição do produto e a documentação do usuário.

f) **caso de teste**: Uma instrução documentada para o avaliador que especifica como uma função ou combinação de funções devem ou deveriam ser testadas. Um caso de teste inclui informações detalhadas dos seguintes pontos:

- O objetivo do teste;
- As funções a serem testadas;
- O ambiente de testes e outras condições (detalhes de configuração e trabalho preparatório);
- Os dados de teste;
- Os procedimentos;
- O comportamento esperado do sistema.

g) **manutenção**: Parte da manutenção de sistema (veja A.5.2) que preocupa-se com modificações de um pacote de *software*.

3.2.3.3)Requisitos de qualidade

As subcláusulas 3.1 a 3.3 contêm:

- O requisito de que cada pacote de *software* tenha uma descrição do produto e uma documentação do usuário;
- Os requisitos para a descrição do produto. Em particular, existe um requisito de que esta descrição deve conter informações específicas e todas as suas declarações devem ser testáveis e corretas;

- requisitos para a documentação do usuário;
- requisitos para os programas e dados, se existirem, inclusos no pacote de *software*.

Um pacote de *software* está em conformidade com esta Norma Internacional se ele cumpre com todos os requisitos apresentados de 3.1 a 3.3. As recomendações (indicadas pelo uso do termo " é importante que" são opcionais.

3.2.3.4)Descrição do produto

Cada pacote de *software* deve ter uma descrição do produto.

A descrição do produto define o produto. Ela faz parte da documentação do pacote do produto. Ela provê informações sobre a documentação do usuário, sobre os programas e, se houver, sobre os dados.

Os principais objetivos da descrição do produto são:

- ajudar o usuário ou comprador em potencial na avaliação da adequação do produto a sua realidade. Ela também fornece informações comerciais;
- servir como base para testes (ver cláusula 4).

A descrição do produto deve estar disponível para qualquer pessoa nele interessada.

a) Requisitos gerais sobre o conteúdo - É importante que a descrição do produto seja

suficientemente compreensível, completa e que facilite uma visão geral, auxiliando compradores em potencial na avaliação da adequação do produto às suas necessidades antes de adquirí-lo. Deve estar livre de inconsistências internas. É importante que cada termo tenha o mesmo significado em todo lugar. As declarações da descrição do produto devem ser testáveis e corretas.

b) Identificações e indicações

- Identificação da descrição do produto - A descrição do produto deve possuir uma identificação única no documento. Pode ser chamada de outras maneiras além de "descrição do produto", por exemplo: "Descrição Funcional", "Informação do Produto", "*Folder* do Produto".
- Identificação do produto - A descrição do produto deve identificar o produto. A identificação do produto deve ter, pelo menos o nome do produto e uma versão ou data. Se houver duas ou mais variações mencionadas na descrição do produto, então cada variação deve ter, pelo menos o nome do produto, o nome da variação e uma versão ou data.
- Fornecedor - A descrição do produto deve conter o nome e o endereço de pelo menos um fornecedor.
- Tarefa de trabalho - A descrição do produto deve identificar as possíveis tarefas de trabalho que podem ser realizadas com o produto.
- Conformidade com documentos de requisitos - A descrição do produto pode referenciar-se aos documentos de requisitos com os quais o produto estiver em conformidade, neste caso as edições relevantes devem ser identificadas.
- Configuração necessária - A configuração necessária (*hardware*, *software* e suas configurações) para colocar o produto em uso deve ser especificada, incluindo nomes dos fabricantes e identificadores de tipos de todas as partes, por exemplo:
    - unidade de processamento, incluindo coprocessadores;
    - tamanho da memória principal;
    - tipos e tamanhos dos periféricos de armazenamento;
    - cartões de extensão;

- equipamento de I/O;
- ambiente de rede;
- *software* de sistema e outro *software*.

Diferentes configurações podem ser especificadas, por exemplo, para diferentes tarefas de trabalho podem ser especificadas diferentes valores limite ou diferentes requisitos de eficiência.

A expressão " (ou qualquer outro..., se compatível)" pode aparecer na descrição do produto se, previamente, um determinado produto de *hardware* ou *software* tiver sido identificado.

A expressão " ou uma versão atualizada se compatível" pode aparecer se, previamente, uma versão do produto tiver sido identificada. A expressão " a partir da versão X à, pelo menos, versão Y" pode aparecer; " a partir da versão X" não deve aparecer.

- Interfaces com outros produtos - Se a descrição do produto faz referência a interfaces com outros produtos, estas interfaces ou produtos devem ser identificados.
- Itens a serem entregues - Cada componente físico do produto fornecido deve ser identificado, em particular todos os documentos impressos e todas as mídias de dados. Os formatos de todos os programas fornecidos devem ser declarados, por exemplo, programas-fonte, módulos-objeto ou módulos de carga.
- Instalação - Deve ser informado se a instalação do produto pode ou não ser conduzida pelo usuário.
- Suporte - Deve ser informado se o suporte para a operação deste produto é ou não oferecido.
- Manutenção - Deve ser informado se a manutenção do produto é ou não oferecida. Se for, deve ser informado o que especificamente está incluso nesta manutenção.

c) Declarações sobre funcionalidade

- Visão Geral das funções - A descrição do produto deve fornecer uma visão geral das funções disponíveis ao usuário, os dados necessários e facilidades oferecidas. Deve ser claramente declarado para cada função mencionada (especialmente para uma opção ou variação) se esta faz parte:
  - do produto;
  - de uma extensão do produto inteiramente apresentada na descrição do produto;
  - de uma extensão do produto referenciada na descrição do produto;
  - de um suplemento sem garantia.
- Valores-limite - Se o uso do produto está limitado por valores específicos, estes devem ser fornecidos. Exemplos:
  - valores mínimos e máximos;
  - tamanhos de chaves;
  - número máximo de registros em um arquivo;
  - número máximo de critérios de busca;
  - tamanho mínimo de amostra.

  No caso de não ser possível informar valores-limite fixos (quando, por exemplo, eles dependem do tipo de aplicação ou da entrada de dados), então as limitações devem ser declaradas. Combinações possíveis de valores podem ser fornecidas e referências a informações mais detalhadas na documentação do usuário devem ser feitas.
- Segurança - A descrição do produto deve incluir informação dos meios, se fornecido, de prevenção de acesso não autorizado (acidental ou deliberado) a programas e dados.

d) Declarações sobre confiabilidade - descrição do produto deve incluir informações sobre procedimentos para realizar cópia de segurança (*backup*) dos dados. Propriedades adicionais do produto que asseguram a capacidade funcional do mesmo deveriam ser descritas. Exemplos:

- Checagem de entrada plausível;
- proteção contra sérias conseqüências advindas de um erro do usuário;
- recuperação de erros.

e) Declarações sobre usabilidade

- Interface com o usuário - O tipo de interface com o usuário deve ser especificado, por exemplo, linha de comando, menu, janelas, tecla de função, função de ajuda.
- Conhecimento Necessário - No caso de ser necessário um conhecimento específico para a utilização do produto, este deve ser especificado. São exemplos:
  - conhecimento de uma área técnica;
  - conhecimento de um sistema operacional;
  - conhecimento obtido através de treinamento especial;
  - conhecimento de um outro idioma diferente daquele em que foi escrita a descrição do produto.

Todos os idiomas utilizados na documentação do usuário e na interface com o usuário (incluindo mensagens de erro e dados visíveis) devem ser descritos no pacote de *software* e em todos os outros produtos mencionados na descrição do produto.

- Adaptação às necessidades do usuário - Se o produto pode ser adaptado pelo usuário, então as ferramentas para esta adaptação e as condições para seu uso devem ser identificadas. São exemplos:
  - mudança de parâmetros;
  - mudança de algoritmos para computação;
  - atribuição de teclas de função.
- Proteção contra infração de diretos autorais - Se a proteção técnica contra infração de diretos autorais pode dificultar a usabilidade, então esta proteção deve ser especificada. São exemplos:
  - proteção técnica contra cópia;

- datas de expiração programadas para uso;
- lembretes interativos para pagamento de cópias.

- Eficiência de uso e satisfação de usuário - A descrição do produto pode incluir dados sobre a eficiência de uso e satisfação de usuário.

f) Declarações sobre eficiência - A descrição do produto pode incluir dados sobre o comportamento do produto em relação ao tempo, tais como tempos de resposta, taxas de performance para dadas funções sobre condições específicas (por exemplo, as configurações do sistema e perfil de carga do sistema).

g) Declarações sobre manutenibilidade - A descrição do produto pode conter declarações sobre manutenibilidade.

h) Declarações sobre portabilidade - A descrição do produto pode conter declarações sobre portabilidade.

3.2.3.5)Documentação do Usuário

a) Completitude - A documentação do usuário deve conter as informações necessárias para o uso do produto. Todas as funções especificadas na descrição do produto e todas as funções possíveis de serem chamadas pelo usuário no programa devem ser completamente descritas na documentação do usuário. Todos os valores-limite dados na descrição do produto devem ser repetidos na documentação do usuário. Se a instalação puder ser realizada pelo usuário, a documentação do usuário deve incluir um manual de instalação contendo todas as informações necessárias (veja 3.3.1 a). É

importante que o manual de instalação especifique os tamanhos mínimo e máximo dos arquivos quando instalados. Se a manutenção puder ser realizada pelo usuário, a documentação do usuário deve incluir um manual de manutenção do programa contendo toda a informação necessária para o tipo de manutenção em questão.

b) Corretude - Todas as informações na documentação do usuário devem ser corretas. Além disso, sua apresentação deve ser livre de ambigüidades e erros.

c) Consistência - Os documentos da documentação do usuário devem ser livres de contradições internas, entre si e em relação à descrição do produto. É importante que cada termo tenha o mesmo significado em todo lugar.

d) Inteligibilidade - É importante que documentação do usuário seja compreensível pela classe de usuários que normalmente desenvolve as tarefas de trabalho especificadas. Esta compreensão pode ser facilitada, por exemplo, através do uso de uma seleção apropriada de termos, gráficos, explicações detalhadas e referência à fontes de informação úteis.

e) Facilidade de Visão Geral - É importante que a documentação do usuário seja apresentada de uma forma que facilite uma visão geral, de maneira que os relacionamentos entre os documentos sejam reconhecíveis. É importante que cada documento tenha uma tabela de conteúdos e um índice.

Se um documento não é fornecido no formato impresso, é importante que o procedimento para impressão seja indicado.

3.2.3.6) Programas e dados

a) Funcionalidade

- Instalação - Se a instalação puder ser realizada pelo usuário, deve ser possível instalar os programas com sucesso seguindo as instruções contidas no manual de instalação. Cada configuração necessária mencionada na descrição do produto deve ser suficiente para a instalação dos programas. Após a instalação deve ser possível reconhecer se os programas funcionam ou não, usando, por exemplo, casos de teste fornecidos ou auto-teste com mensagens correspondentes.
- Presença de funções - Todas as funções mencionadas na documentação do usuário devem ser executáveis na forma descrita na documentação do usuário com as facilidades, propriedades e dados correspondentes, e dentro dos valores-limite fornecidos.
- Corretude - Os programas e dados devem corresponder a todas as declarações mencionadas na descrição do produto e na documentação do usuário. As funções devem ser executadas de uma maneira correta para as tarefas de trabalho a que se propõem. Em particular, programas e dados devem cumprir com todos os requisitos em qualquer documento de requisitos referenciado pela descrição do produto.
- Consistência - Os programas e dados devem ser livres de contradições internas, com a descrição do produto e com a documentação do usuário. É importante que cada termo tenha o mesmo significado em todo lugar. O controle da operação do programa pelo usuário e o comportamento do programa (por exemplo, mensagens, formatos de telas de entrada e relatórios) devem

ser uniformemente estruturados.

b) Confiabilidade – O sistema, compreendendo *hardware*, *software* requerido e aqueles programas que pertencem ao produto, não deve falhar de tal forma que o usuário não possa controlá-lo, e não deve também corromper ou perder dados.. Este requisito deve ser atendido até mesmo no caso de:

- a capacidade ser explorada até os limites especificados;
- haver tentativas de explorar a capacidade além dos limites especificados;
- uma entrada incorreta ser feita pelo usuário ou ser proveniente de outros programas listados na descrição do produto;
- instruções explícitas na documentação do usuário serem violadas.

Somente aquelas possibilidades para interrupção do *hardware* e do sistema operacional que não podem ser controladas por nenhum programa (por exemplo, a tecla ou combinação de teclas para realizar a função *reset* do sistema operacional) são excluídas.

Os programas devem reconhecer violações de condições sintáticas para entrada. No caso de um programa reconhecer uma entrada como incorreta ou como indefinida, ele não deve processá-la como uma entrada permitida.

c) Usabilidade - Em relação à usabilidade, as partes de um acordo baseado nesta Norma Internacional são estimulados a investigar a possibilidade de aplicação das edições mais recentes das normas das séries ISO 9421.

- Inteligibilidade - As perguntas, mensagens e resultados dos programas devem ser compreensíveis, por exemplo:
  - através de uma seleção adequada de termos;
  - através de representações gráficas;
  - através do fornecimento de informações complementares;
  - através de explicações de uma função de ajuda.

  Mensagens de erro devem oferecer informações detalhadas explicando a

causa ou a correção dos erros correspondentes (através, por exemplo, de uma referência a um item na documentação do usuário).

- Facilidade de visão geral - Cada mídia de dados deve apresentar a identificação do produto e, se existir mais de uma mídia, um número ou texto para diferenciá-las deve existir. Para o usuário, quando trabalhando com os programas, deve ser sempre possível saber qual função está sendo executada. Os programas devem fornecer informações ao usuário de uma forma facilmente visível e de fácil leitura. O usuário deve ser guiado por codificação e agrupamento de informações apropriados. É importante que os programas alertem o usuário onde for necessário. É importante que mensagens dos programas sejam projetadas de forma que o usuário possa facilmente diferenciá-las pelo tipo. São exemplos:
  - notificação;
  - solicitações de programas;
  - advertências;
  - mensagens de erro.

  É importante que formatos de telas de entrada, relatórios e outras entradas e saídas sejam projetadas para serem claras e para facilitar uma visão geral. Possibilidades para isso incluem:
  - alinhamento de campos alfanuméricos pela esquerda;
  - alinhamento de campos numéricos pela direita;
  - em tabelas, posicionamento de pontos decimais ou vírgulas na mesma linha vertical;
  - identificação de limites de campos ;
  - identificação de campos obrigatórios;
  - destaque para falhas de entradas identificadas;
  - captura da atenção do usuário para uma mudança na tela através de um sinal visual ou sonoro.
- Operabilidade - A execução de funções que tenham sérias conseqüências devem ser reversíveis, ou os programas devem dar uma clara advertência

das conseqüências e solicitar confirmação antes de executar o comando. Em particular, apagar e sobrescrever dados, como também interrupções de uma operação de processamento muito demorado, têm sérias conseqüências. Se um texto de documentação é disponibilizado de forma *online*, é importante que seja possível ao usuário acessar subcláusulas do texto de uma maneira direta,por exemplo, selecionando a partir de uma tabela de conteúdos e por uma função de pesquisa baseada em palavras-chave.

d) Eficiência - Nada é requerido. Porém, declarações sobre eficiência na descrição do produto devem estar em conformidade com o mesmo.

e) Manutenibilidade – Nada é requerido. Porém, declarações sobre manutenibilidade na descrição do produto devem estar em conformidade com o mesmo.

f) Portabilidade - Nada é requerido. Porém, declarações sobre portabilidade na descrição do produto devem estar em conformidade com o mesmo.

3.2.3.7)Instruções para testes

As instruções de 4.1 a 4.5 especificam como um produto deve ser testado de acordo com os requisitos de qualidade. Elas incluem tanto testes das propriedades requeridas de todos os produtos em conformidade como testes das propriedades prometidas pela descrição do produto. Elas incluem tanto testes através de inspeção de

documentos como testes de caixa-preta de programas e dados.

Estas instruções descrevem testes funcionais (testes de caixa-preta). Testes estruturais não são incluídos porque eles iriam requerer a disponibilidade do código fonte.

O produto é testado somente dentro da configuração necessária especificada. A avaliação ergonômica no local de trabalho do computador não é considerada nesta Norma Internacional.

3.2.3.8)Pré-requisitos de testes

a) *Presença de itens do produto* - Para testar um pacote de *software*, todos os itens a serem entregues (ver 3.1.2 h) bem como os documentos de requisitos identificados na descrição do produto (ver 3.1.2 e) devem estar presentes.

b) *Presença dos componentes de sistema* - Para testar um pacote de *software*, é necessário que os requisitos de *hardware* e *software* relacionados na descrição do produtor estejam disponíveis.

c) *Treinamento* - Se um treinamento é mencionado na descrição do produto, o examinador deve Ter acesso ao material de treinamento e ao programa de treinamento.

3.2.3.9)Atividades de testes

A descrição do produto, a documentação do usuário, os programas e quaisquer dados entregues como partes do pacote de *Software*:

- devem ser testados em relação aos requisitos da cláusula 3;
- deveriam ser testados em relação às recomendações da cláusula 3.

Os objetivos do teste devem ser derivados, incluindo todos, dos requisitos da cláusula 3 (completitude, consistência etc.).

Se outros produtos são mencionados na descrição do produto sob teste, estes outros produtos necessitam ser testados apenas para as exigências feitas nesta descrição.

Detalhes da descrição do produto, da documentação do usuário, de funções ou de dados do produto não necessitam ser testados se, de acordo com a decisão do examinador:

- eles tem influência desprezível na identificação da adequação do produto para a execução das tarefas de trabalho a que se propõe;
- em princípio, eles podem ser testados mas não com custo justificável.

Aqueles detalhes que não foram testados deverão ser mencionados nos registros de teste e no relatório de teste. As razões pelas quais eles não foram testados deverão ser documentadas nos registros de teste.

Os programas devem ser testados em todos os ambientes de *hardware* e *software* mencionados na descrição do produto.

Se existem muitas variações do programa, cada uma deve ser testada. Funções

que, de acordo com a descrição do produto e a documentação do usuário, são idênticas em um conjunto de variações, podem ser testadas em uma única delas.

Os programas e os dados fornecidos com eles devem ser testados usando casos de teste construídos com base na descrição do produto e na documentação do usuário. Materiais adicionais (por exemplo, programas-fonte) não necessitam ser considerados, a menos que o teste de declarações da descrição do produto ou da documentação do usuário tornem isto necessário. Os casos de teste devem ser construídos metódica e sistematicamente.

Se exemplos são dados na documentação do usuário, estes devem ser usados como casos de teste, mas os testes não devem ser restritos a estes exemplos.

Os casos de testes disponibilizados pelo fornecedor do pacote de *software* podem ser usados, mas os testes não devem restringir-se apenas a esses casos de teste:

a) Instalação - Se, de acordo com a descrição do produto, a instalação puder ser executada pelo usuário, deve ser testado se programas podem ser instalados e testados corretamente a partir das explicações contidas no manual de instalação. Além disso, deve ser assegurado que o ambiente de *hardware* e *software* dos programas instalados correspondem às declarações da descrição do produto para tal sistema computadorizado.

b) Execução do programa - Os casos de teste devem cobrir todas as funções apresentadas na descrição do produto e na documentação do usuário e devem levar em consideração combinações de funções que são representativas para a tarefa de trabalho. Os programas devem ser testados para todos os valores-limite (de acordo com a descrição do produto e a documentação do usuário) nas configurações onde estes valores se aplicam. Seqüências de entrada ou de comando que são explicitamente desaprovadas ou declaradas como proibidas na documentação do usuário devem ser usadas nos testes.

3.2.3.10)Registros de teste

Os registros para cada teste devem conter informações suficientes que permitam a repetição do teste [Guia 25 ISO/IEC]. Devem incluir:

a) Um plano ou especificação de teste contendo casos de testes (cada caso de teste declarando seus objetivos, veja 2.6);
b) Todos os resultados associados com os casos de teste, incluindo todas as falhas ocorridas durante o teste;
c) A identificação do pessoal envolvido no teste.

3.2.3.11) Relatório de teste

Os objetos e os resultados de teste (como os relatados nos registros de teste) devem ser resumidos em um relatório de teste. Tal relatório deve ter a seguinte estrutura:

a) Identificação do produto;
b) Sistema computadorizado usado para executar o teste (*hardware*, *software* e sua configuração);
c) Documentos utilizados (com suas identificações);
d) Resultados dos testes da descrição do produto, da documentação do usuário e dos programas e dados;

e) Lista das não-conformidades com os requisitos;
f) Uma lista das não-conformidades com as recomendações, ou uma lista das recomendações que não foram seguidas, ou uma declaração de que o produto não foi testado em relação à conformidade com as recomendações;
g) Data do término dos testes.

O Resultados do teste deve conter uma declaração correspondente para cada cláusula de 3.1 até 4.2.

Além da declaração de que o produto não foi testado em relação à conformidade com as recomendações, o capítulo 6 do relatório de teste pode fornecer uma lista de não-conformidades observadas em relação às recomendações.

A identificação do relatório de teste (laboratório de teste, identificação do produto, data do relatório de teste) e o número total de páginas devem aparecer em cada página do relatório de teste.

Cada relatório de teste deve incluir:

- Uma declaração de que os resultados de teste relacionam-se somente aos itens testados;
- Uma declaração de que o relatório de teste não deve ser reproduzido parcialmente sem a aprovação escrita do laboratório de testes [Guia 25 da ISO/IEC].

3.2.3.12)Teste de acompanhamento

Quando um produto, que já tenha sido testado, é testado novamente (levando em consideração o teste anterior), então:

a) Todas as partes modificadas nos documentos, funções e dados devem ser testadas como se fossem novos produtos;
b) Todas as partes não-modificadas que podem ter sido influenciadas pelas partes alteradas ou por mudanças em requisitos de *hardware* e *software* (de acordo com o *know-how* do examinador), devem ser testadas como se fosse um novo produto;
c) Todas as outras partes devem, no mínimo, ser testadas através de exemplos.

4) Métricas de Qualidade de Software

Muitos trabalhos têm usado a teoria das medições para métricas de software, e aplicado métricas para controlar e estimar desenvolvimento formal. 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.

Já não é mais aceitável conceber projetos de engenharia de software, que lidem com alvos firmados em ambigüidades e abstrações. "*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.

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*.

Uma definição simples para métricas é encontrada nos livros de matemática:

- Métricas são funções que estabelecem uma medida.

Um modelo estrutural para medição de software permite descrever os objetos envolvidos na medição e seus relacionamentos como:

- *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.

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.

4.1)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*.

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 (PRESSMAN,

1995). As *métricas dinâmicas* possuem uma dimensão temporal. As *métricas estáticas* permanecem invariáveis a despeito do tempo.

As *métricas preditivas* ou *métricas a priori,* podem ser obtidas ou geradas previamente, para realizar prognósticos do valor de uma propriedade do sistema, somente se tornará diretamente observável em um estágio posterior a seu desenvolvimento (FENTON *et al*, 1997). 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 *et al*, 1997).

As *métricas explanatórias*, também conhecidas como *métricas de resultado* , métricas *descritivas* ou *métricas a posteriori* 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, em um período de tempo, pode indicar a confiabilidade do software.

Existem, também, as métricas de produto e as métricas de processo. As *métricas de produto* indicam, objetivamente, características mensuráveis do produto de software tal como tamanho, complexidade, acoplamento.

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.

A qualidade de software é geralmente expressa em função de diversas métricas de software, onde diferentes índices medem diferentes aspectos de qualidade. Na prática, com o processo de medição, deseja-se saber se um bom software é um bom negócio. A seguir, apresentar-se-á as principais características das métricas de qualidade.

4.2)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, 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.
- *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.

Na avaliação do uso de métricas de software, é importante considerar a qualidade das próprias métricas, 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.

4.3) 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)

- *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:

- 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.

A utilização de métricas de qualidade, também, oferece algumas limitações :

- *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 . Neste sentido 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 á organização uma linha básica, confrontada por práticas aceitas como adequadas pela indústria de software.

4.3.1)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.

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:

- *Usar métodos quantitativos, sempre que possíve*l;
- *Projetar experimentos cujos resultados dependam, mais de fatos concretos do que de influências individuais*;
- *Explorar como o uso da teoria pode ser refutado por experimento*s;
- *Verificar a consistência das conjecturas sobre si mesmas e com fatos conhecido*s.

Psicólogos, reportando-se a julgamentos intuitivos, têm observado que:

- *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 especialist*a.

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:

- Fazerem prognósticos freqüentemente;
- Receberem rápidos *'feedback'* sobre seus sucessos ou erros;
- Serem treinados em questões específicas;

- Executarem medições.

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

- *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.

4.4)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.

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.

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.

Para decidir se uma medição é válida, necessita-se assegurar a:

- *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 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:

- *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.

5) 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.

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:

- *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. A seguir, são apresentadas algumas técnicas de controle de qualidade.

5.1) *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, 1995).

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.

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 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. As inspeções suportam a análise de dados, durante o ciclo de desenvolvimento de um produto de software, desempenhando um importante papel na

redução de falhas estruturais não esperadas.

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.

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 tomo 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) desenvolveu 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.

As reuniões são consideradas como um ponto central nas inspeções, abaixo estam algumas razões importantes para as reuniões de inspeção, dispostas em ordem decrescente de freqüência:

- *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.
- *Formação*: os revisores de menor experiência aprendem com os revisores de maior experiência, ao longo do processo de inspeção.
- *Prazo final de planejamento*: a inspeção cria um evento planejado, para que os participantes possam trabalhar convenientemente.
- *Incentivo*: os participantes da revisão publicam suas contribuições e ganham a estima de seus examinadores e, portanto, esforçam-se por melhorar a si mesmos.
- *Requisitos*: o documento do processo de inspeção especifica que a reunião deve trazer coletados os comentários dos revisores.

5.2)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, 1995).

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.

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.

A *técnica baseada em erros* emprega informações de erros padrões cometidos, no processo de desenvolvimento de software, para derivar requisitos de teste.

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.*

5.3)**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.

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.

5.4) 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.

Os modelos de confiabilidade de software conhecidos podem ser classificados em:

- *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:
  - o tempo entre falhas é independente;
  - os testes são representativos do perfil operacional utilizado;
  - novas faltas ou falhas não são introduzidas durante a correção do programa;
  - 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.

6) Especificações do Modelo Proposto

Utilizando as normas e modelos relacionados à qualidade de softwares descritos nesta dissertação, foi criado um modelo para verificação de qualidade e normalização de software chamado "QUALISOFT", que é um Sistema Especialista que reuni de forma aplicável, um conjunto de características que devem ser verificadas em um software para que ele seja considerado um "Software com Qualidade".

Em síntese o QUALISOFT é um programa, que procuram representar o conhecimento do especialista através de regras de produção (se...então...). Esse conhecimento armazenado é acionado por máquinas de inferência, cujo objetivo é deduzir algum novo conhecimento das regras embutidas. Esse novo conhecimento poderá ser, por exemplo, a solução de um problema. O QUALISOFT trabalha com uma técnica chamada de TRATAMENTO DOS FATORES DE CERTEZA. Essa técnica consistia em atribuir uma espécie de "peso" a uma afirmação como um grau de certeza sobre a afirmação e, também, a uma não-afirmação. Esses pesos passariam por uma série de compilações. Ao final, um resultado poderia ter sua resposta modificada, mesmo apontando estar correta, pois poderia não estar de acordo, conforme os pesos atribuídos durante o processo.

O QUALISOFT como todos os Sistemas Especialistas, foi projetado e desenvolvido para atender a uma aplicação determinada e limitada do conhecimento humano. É capaz de emitir uma decisão, apoiado em conhecimento justificado, a partir de uma base de informações, tal qual um especialista de determinada área do conhecimento humano.

6.1) Estrutura do Modelo QUALISOFT

Existem várias arquiteturas de Sistemas Especialistas sendo usadas. Dentre elas a mais simples de compreender e a mais difundida compõem-se de 3 elementos básicos.

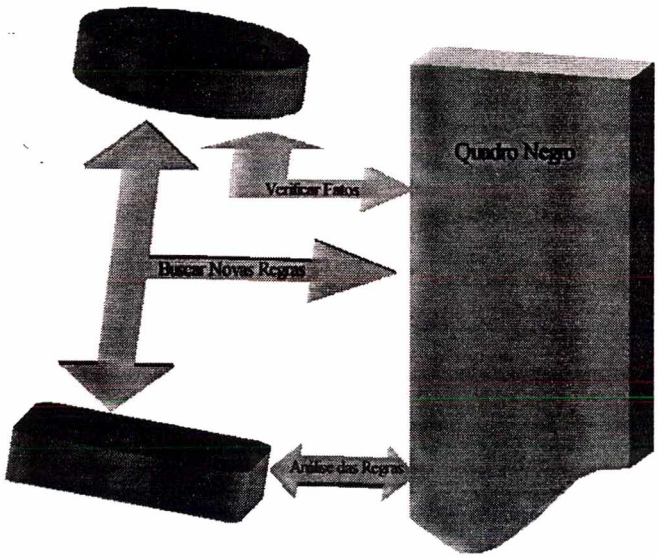


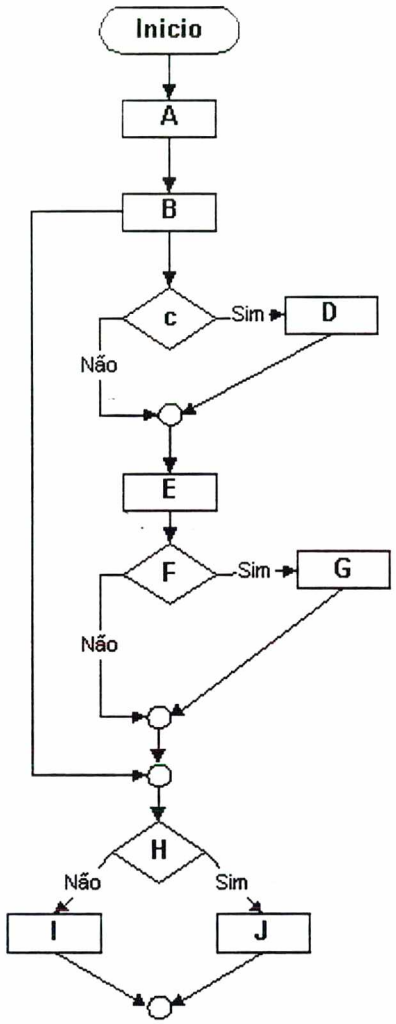
Figura 7: Elementos de um Sistema Especialista

- *Base de Conhecimento* - É um elemento permanente, mas específico de um sistema especialista. É onde estão armazenadas as informações de um sistema especialista, ou seja os fatos e as regras.

Figura 8: Base de Conhecimento do "QUALISOFT"

- *Quadro-Negro* - É responsável pela comunicação das informações entre os sistemas especialistas. O quadro-negro é um lugar dentro da memória do computador no qual as informações armazenadas em um sistema especialista são "afixadas" para que qualquer outro sistema especialista possa usá-lo se precisar das informações lá contidas.

- *Mecanismo de Inferência* - É um elemento permanente, que pode inclusive ser reutilizado por vários sistemas especialistas. É a parte responsável pela busca das regras da base de conhecimento para serem avaliadas, direcionando o processo de inferência. Basicamente o mecanismo de inferência busca as regras na base de conhecimento, essas regras serão colocadas no quadro-negro, sendo que as regras já existentes só serão avaliadas depois das mais recentes.



A - Verificar a meta

B - Enquanto hover referências às regras a serem avaliadas faça

C - Meta ajusta com a conclusão

D - Empilha regra na área de trabalho

E - Avaliar se leva a condição de Loop

F - Leva à Loop ?

G - Desempilha a regra

H - Empilhou alguma regra ?

I - Não ha regras para avaliar essa meta

J - Chama organizador

Figura 9: Funcionamento do Mecanismo de Inferência

Para criar o sistema "QUALISOFT", foi utilizada uma ferramenta computacional chamada "EXPERT SINTA", que permitem que o criador do sistema especialista preocupe-se somente com a representação do conhecimento do especialistas, deixando para a shell a tarefa de interpretar o conhecimento representado e executá-lo em uma máquina, além de permitir seu uso por qualquer pessoa sem conhecimentos de informática. O Expert Sinta é uma ferramenta gratuita, que foi desenvolvida na Universidade Federal do Ceará, é implementada na linguagem de programação orientada a objetos Borland Delphi, dando um suporte visual de fácil operação que utiliza técnicas de Inteligência Artificial para geração automática de sistemas especialistas e utiliza também, um modelo de representação do conhecimento

baseado em regras de produção e probabilidades, tendo como objetivo principal simplificar o trabalho de implementação de sistemas especialistas através do uso de uma máquina de inferência compartilhada, da construção automática de telas e menus, do tratamento probabilístico das regras de produção e da utilização de explicações sensíveis ao contexto da base de conhecimento modelada.

O processo de construção de um SE é geralmente chamado de Engenharia do Conhecimento (EC). Tipicamente envolve uma forma especial de interação entre o construtor do SE, chamado Engenheiro do Conhecimento, e um ou mais especialistas em alguma área. O engenheiro do conhecimento extrai dos especialistas seus procedimentos, estratégias e regras práticas para solução de problemas e constrói este conhecimento em um SE, como mostra a figura 10. O resultado é um programa que soluciona problemas a maneira dos especialistas humanos.

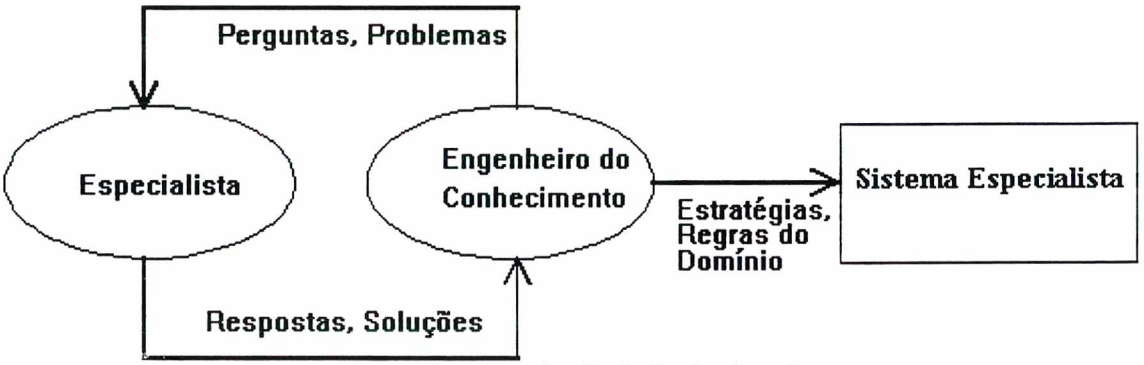


Figura 10: Engenharia do Conhecimento

Após a etapa de engenharia do conhecimento vem a etapa de criação das variáveis, pois é necessário que todas as variáveis utilizadas, bem como seus respectivos valores, sejam criados. Através desse mecanismo, a base fica organizada, fácil de manter e as regras podem ser criadas visualmente.

Existem Variáveis Univaloradas, variáveis que podem assumir apenas um valor durante a consulta, e as Variáveis Multivaloradas, variáveis que podem assumir mais de um valor durante a consulta.

Algumas das Variáveis Univaloradas e Multivaloradas existentes no QUALISOFT estão relacionadas nas tabelas abaixo:

Variáveis Univaloradas	Valor	Motivo	Descrição
Áreas Livres	Sim ou Não	Dados apresentados muitos próximos são difíceis de localizar e de ler	Característica de Legibilidade
Rótulos Identificativos	Sim ou Não	É necessário definir rótulos significativos para os dados, de modo a auxiliar a compreensão da tela.	Característica de Presteza
Tecla TAB	Sim ou Não	A movimentação do cursor entre os campos de dados deve ser facilitada.	Característica de Ações Mínimas
Mensagens de processamento	Sim ou Não	O usuário deve ser Informado sobre o resultado de processamentos longos.	Característica de Feedback

Tabela 17: Variáveis Univaloradas existentes no sistema "QUALISOFT"

Variáveis Multivaloradas	Valor	Motivo	Descrição
Legibilidade	Boa Regular Imprópria	Dados necessários para avaliar a qualidade da Característica de Usabilidade	Sub-característica de Usabilidade
Presteza	Boa Regular Imprópria	Dados necessários para avaliar a qualidade da Característica de Usabilidade	Sub-característica de Usabilidade
Ações Mínimas	Boa Regular Imprópria	Dados necessários para avaliar a qualidade da Característica de Usabilidade	Sub-característica de Usabilidade
Feedback	Boa Regular Imprópria	Dados necessários para avaliar a qualidade da Característica de Usabilidade	Sub-característica de Usabilidade

Tabela 18: Variáveis Multivaloradas existentes no sistema "QUALISOFT"

A criação da base de conhecimento é um dos pontos críticos na elaboração de um SE. Existem diversas formas para a representação do conhecimento por parte do projetista do conhecimento, as mais utilizadas são as chamadas regras de produção: são regras no formato *SE* - *ENTÃO*, permitindo-se o uso dos conectivos lógicos (E, OU, NÃO, e outros desejados), além do tratamento de incertezas, garantindo maior legibilidade da base de conhecimentos. Como pode ser analisado no exemplo a seguir.

6.2) Regras de Produção

As principais vantagens das utilização de regras de produção são:

- Modularidade: cada regra, por si mesma, pode ser considerada como uma peça de conhecimento independente;
- Facilidade de edição (uma conseqüência da modularidade): novas regras podem ser acrescentadas e antigas podem ser modificadas com relativa independência;
- Transparência do sistema: garante maior legibilidade da base de conhecimentos.

A modularidade de um sistema baseado nessa arquitetura permite a construção passo-a-passo da base de conhecimentos, ou seja, é possível realizar vários testes com apenas um subconjunto de regras concluído. Obviamente, sabe-se que menos regras implicam geralmente em um menor número de casos abrangidos.

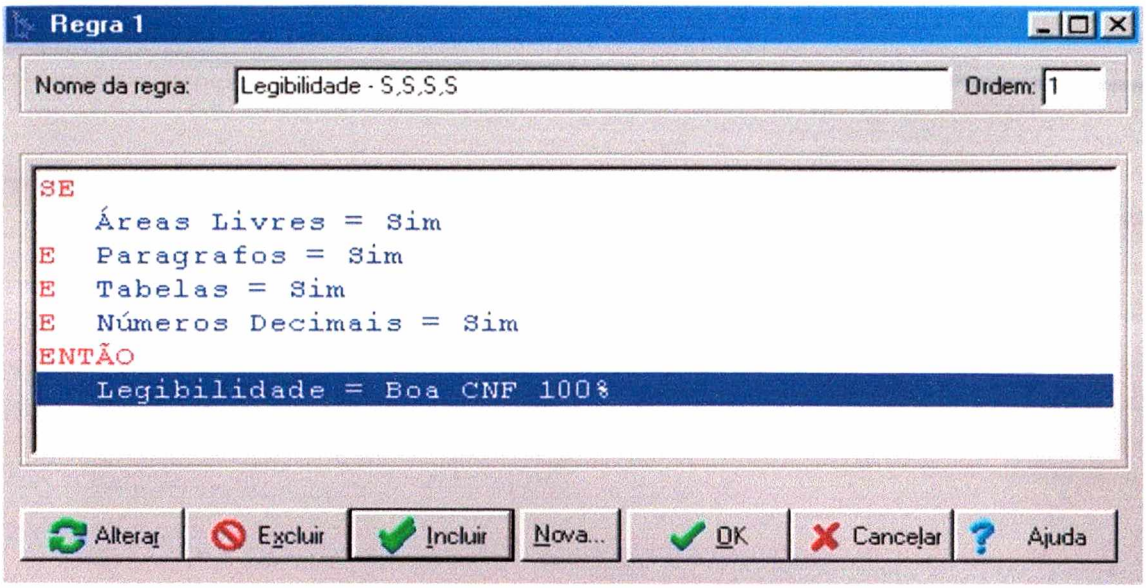


Figura 11: Regra de Produção sobre Legibilidade

6.3) Tratamento de Incertezas

Ao se examinar uma das conclusões da regra dada como exemplo, verifica-se a presença de um grau de confiança na decisão na qual a Legibilidade é boa. A dificuldade em representar a confiabilidade das informações segundo [Bratko, 1990] são:

- Especialistas humanos não se sentem confortáveis em pensar em termos de probabilidade. Suas estimativas não precisam corresponder àquelas definidas matematicamente;
- Tratamentos rigorosamente matemáticos de probabilidade utilizam informações nem sempre disponíveis ou simplificações que não são claramente justificáveis em aplicações práticas.

O QUALISOFT utiliza algumas regras de cálculo de incertezas que utilizam, inclusive, o fator de confiança do usuário em relação às respostas dadas.

A etapa de implementação do sistema, consiste em criar uma interface entre os usuários com o sistema, isto é feito através de perguntas. Deve-se criar preguntas para todas as variável existentes em cada uma das regras de produção. Na tabela abaixo estão relacionadas algumas das perguntas existentes no sistema QUALISOFT.

Variáveis	Perguntas
Botões Defauld	Os grupos de botões de comando possuem sempre um botão definido como default ?
Estado das Impressões	O sistema fornece informações sobre o estado das impressões ?
Mensagens de Processamento	O sistema apresenta uma mensagem informando sobre o sucesso ou fracasso de um processamento demorado ?
Tabelas	Nas tabelas, linhas em branco são empregadas para separar grupos ?
Tecla TAB	O usuário dispões de um modo simples e rápido para a navegação entre os campos de um formulário ?
Formato Particular	Caso o dado a entrar possua um formato particular, esse formato encontra-se descrito na tela ?

Tabela 19: Exemplo de perguntas existentes no "QUALISOFT".

Os usuários interessados em avaliar a qualidade de um software, poderão responder a estas perguntas, levando em conta o grau de confiança das respostas dadas de acordo com seus conhecimentos e assim realizar a avaliação deste software em questão.

7) 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 e, conseqüentemente, garantir a qualidade do produto que hoje dependente, basicamente, da habilidade e da opinião dos programadores e analistas. Para solucionar o problema citado, foi desenvolvido através de pesquisas das normas e modelos que estão relacionadas à Qualidade de Softwares, o sistema denominado " QUALISOFT ", que é um Modelo para Verificação de Qualidade e Normalização de Software, ou seja, um sistema especialista utilizando o Expert SINTA como ferramenta, que auxilia as empresas na escolha de sistemas com qualidade, que realmente irão suprir sua necessidades.

7.1) Objetivos

Com o " QUALISOFT " almeja-se atingir dois objetivos :

- Conscientizar os usuários que eles podem e devem avaliar o software oferecido pelos desenvolvedores, antes de efetuar a compra do mesmo. Podendo, com base na qualidade de cada um dos produtos a serem adquiridos, escolher o software

mais adequado para a empresa, oferecendo uma garantia maior de segurança, comodidade e funcionalidade.

- Induzir a conscientização dos desenvolvedores de softwares, a produzirem produtos com qualidade, pois só assim poderão conquistar o respeito do mercado atual.

7.2) Limitações do QUALISOFT

O sistema QUALISOFT possui apenas 16 perguntas com 145 regras de produção, as quais servem para avaliar a característica de Usabilidade dos softwares, pois como o sistema foi criado inicialmente com o intuito de demostração perante a banca examinadora, uma avaliação completa das características descritas na ISO 9126 tornaria a apresentação muito extensa.

7.3)Propostas Futuras

Com a aprovação da banca examinadora, a base de conhecimentos do QUALISOFT, será ampliada para que ele possa avaliar um software em sua totalidade, ou seja, avaliar todas as características descritas na ISO 9126. E posteriormente o sistema QUALISOFT disponível para download em uma home page na Internet, para que todas as empresas interessadas, possam realizar suas avaliações.

8) Referência: Bibliograficas

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **Tecnologia de Informação:** Avaliação de Produto de Software -Características de qualidade e diretrizes para o seu uso:13596:1996. Rio de Janeiro, 1996.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **Tecnologia de Informação:** Avaliação de Produto de Software – Processos do ciclo de vida do software:12207:1998. Rio de Janeiro, 1998.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **Tecnologia de informação**: Pacotes de software - Teste e requisitos de qualidade:12119:1998. Rio de Janeiro, 1998.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **Sistemas de qualidade**: Modelo para garantia da qualidade em projetos/desenvolvimento, produção, instalação e assistência técnica: 9001:1990. Rio de Janeiro, 1990.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **Normas de gestão da qualidade e garantia da qualidade - Parte 3:** Diretrizes para aplicação da NBR 19001 ao desenvolvimento, fornecimento e manutenção de software: 9003:1993. Rio de Janeiro, 1993.

INSTITUTO DE SOFTWARE DO CEARÁ. Grupo de Estudos em Qualidade de Software. **Tradução Livre da Norma Internacional ISO/IEC 12119**: Nov/1998. Disponível em: <http://www.insoft/softex.br/home/GEQS/index.html>. Acesso em 31 Mar.2001.

KOSCIANSKI, André; VILLAS-BOAS, André; RÊGO, Claudete. **Guia Para Utilização Das Normas Sobre Avaliação De Qualidade De Produto De Software**: ISO/IEC 9126 e ISO/IEC 14598. Mai/1999.Disponível em: <http://www.abnt_sw@pr.gov.br>. Acesso em 15 Mar.2001.

MARIANO, G. **Metrics in software engineering**. Jan/1997. Disponível em: < http://estas1.inrets.fr:8001/Public/Mariano.Georges/DundeeB/node3.html> Acesso em 15 Fev.2001.

AUGUSTO NETO, Álvaro. **Uma Estratégia Para Gerência Da Qualidade E Produtividade No Desenvolvimento De Software**. 1997.Dissertação (Mestrado) – Programa de Pós Graduação em Informática, Instituto Tecnológico de Aeronáutica, São José dos Campos, 1997.

CAMPOS, G. H. B. **Metodologia para avaliação da qualidade de software educacional**: Diretrizes para desenvolvedores e usuários.1994.193f. Tese (Doutorado) – Universidade Federal do Rio de Janeiro, Rio de Janeiro,1994.cap.3.

MATIAS, Marcio. **Checklist**: uma Ferramenta de suporte à avaliação ergonômica de interfaces.1995. Dissertação (Mestrado) - Pós Graduação em Engenharia da Produção, Universidade Federal de Santa Catarina, Florianópolis, 1995.

PRESSMAN, Roger . **Engenharia de Software**. São Paulo: Makron Books, 1995. 1056p.

FENTON, N. E., PFLEEGER, S. L. **Software Metrics:** A Rigorous & Practical Approach, 2 ed. PWS Publishing Company. 638p.

HERBERT, Juliana Silva; PRICE, Ana Maria Alencar. Métodos para Avaliação da Qualidade de Software. In: XI JORNADA DE ATUALIZAÇÃO EM INFORMÁTICA, .27., 1995, Porto Alegre. **Anais...** Porto Alegre, 1995. p.11-29.

WEINBERG, Gerald M. **Software Com Qualidade: Pensando e Idealizando Sistemas**. 2.ed. São Paulo: Makron Books, 1993. p.37-48.

Moller, Karl-Heinrich, and Daniel Paulish. **Software Metrics**: A Practitioner's Guide to Improved Product Development: IEEE Computer Society,1993. 257p.

DOMICIANO, Marco Antonio Pizani. **ISO/IEC 14598**: Software Product Evaluation. Setembro/2000. 52 Slides, color.

BELCHIOR, Arnaldo Dias; ROCHA, Ana Regina Cavalcanti da. Características De Qualidade De Programas. **Publicações Técnicas COPPE/UFRJ**, Rio de Janeiro, p.2.., 25 Jun.1992.

JÚNIOR, José Barreto- "Qualidade de Software", url: <http://www.barreto.com.br/qualidade > Acesso em: 10 Fev.2001.

[ABNT] Associação Brasileira de Normas Técnicas, url:< http://www.abnt.org.br/ > .Acesso em: 31 Mar.2001.

[GEQS] Grupo de Estudos em Qualidade de Software - Instituto de software do Ceara url: < http://www.insoft/softex.br/home/GEOS/index.html >Acesso em: 31 Mar. 2001.

[ISO] International Organization for Standardization, url: <http://www.iso.ch/ > Acesso em: 10 Fev. 2001.

[CITS] - Conferência Internacional de Tecnologia de Software url: <http://www.ic.cti.br/taqs > Acesso em: 10 Fev. 2001.

[IEEE] - Instituto de Engenharia Elétrica e Eletrônica, url: <http://www.ieee.org/> Acesso em: 10 Fev. 2001.